Generation of Continuous Hybrid Zig-zag and Contour Paths For 3D Printing

Gorka Gomez
Industry and Advanced Manufacturing, Vicomtech, Donostia-San Sebasti ´an, Gipuzkoa, Spain
Camilo Cortés
eHealth and Biomedical Applications, Vicomtech, Donostia-San Sebasti ´an, Gipuzkoa, Spain
Carles Creus
Industry and Advanced Manufacturing, Vicomtech, Donostia-San Sebasti ´an, Gipuzkoa, Spain
Maialen Zelaia Amilibia (mzelaia@vicomtech.org)
eHealth and Biomedical Applications, Vicomtech, Donostia-San Sebasti ´an, Gipuzkoa, Spain
https://orcid.org/0000-0002-7414-2111
Aitor Moreno
Industry and Advanced Manufacturing, Vicomtech, Donostia-San Sebasti ´an, Gipuzkoa, Spain

Research Article

Keywords: Additive Manufacturing, Continuous path, Polygon decomposition

Posted Date: July 23rd, 2021

DOI: https://doi.org/10.21203/rs.3.rs-608881/v1

The International Journal of Advanced Manufacturing Technology manuscript No.
(will be inserted by the editor)

Generation of continuous hybrid zig-zag and contour paths for 3D printing

Gorka Gomez · **Camilo Cortés** · **Carles Creus** · **Maialen Zelaia Amilibia** · **Aitor Moreno**

Received: date / Accepted: date

Abstract The generation of the printing paths is a decisive step in additive manufacturing (AM). There is a variety of patterns that offer different characteristics, but those that are strictly continuous become especially relevant in certain types of AM by extrusion, with materials like bioinks, carbon or clays, since they do not allow the retraction of the material and travelling movements result in the generation of artefacts. In this work, we present (1) a method that generates continuous paths to fill 2D polygons with a hybrid zig-zag and contour pattern with any direction and line separation, which is based on the extension of an algorithm that decomposes the 2D area to be filled into convex areas, (2) a method to join the subpolygon trajectories such that a continuous path that fills the whole polygon is obtained, and (3) a publicly available dataset of 2D polygons that are relevant to test the performance of the algorithms.

Keywords Additive Manufacturing · Continuous path · Polygon decomposition

G. Gomez
Industry and Advanced Manufacturing, Vicomtech, Donostia-San Sebastián, Gipuzkoa, Spain
E-mail: ggomez@vicomtech.org

C. Cortés
eHealth and Biomedical Applications, Vicomtech, Donostia-San Sebastián, Gipuzkoa, Spain
E-mail: ccortes@vicomtech.org
Biodonostia Health Research Institute, Donostia-San Sebastián, Gipuzkoa, Spain

C. Creus
Industry and Advanced Manufacturing, Vicomtech, Donostia-San Sebastián, Gipuzkoa, Spain
E-mail: ccreus@vicomtech.org

M. Zelaia Amilibia
eHealth and Biomedical Applications, Vicomtech, Donostia-San Sebastián, Gipuzkoa, Spain
E-mail: mzelaia@vicomtech.org
Biodonostia Health Research Institute, Donostia-San Sebastián, Gipuzkoa, Spain

A. Moreno
Industry and Advanced Manufacturing, Vicomtech, Donostia-San Sebastián, Gipuzkoa, Spain
E-mail: amoreno@vicomtech.org

1 Introduction

AM is a manufacturing method that describes a group of processes to produce objects by depositing materials layer by layer [13]. Given its ability to obtain near-net shape products with very little material waste, this method is revolutionizing the manufacturing industry [11]. Some of the advantages of AM are its high resolution and the possibility to decide the pattern to fill the areas that form the 2D layers of the model. There is a great variety of patterns, which can be grouped according to their shape and/or continuity. The fact of having a strictly continuous trajectory becomes very relevant in certain cases. In wire and arc AM one of the requirements is to minimize the number of tool path-passes, and following continuous paths a single pass per polygon is obtained [5]. In solid freeform fabrication based on welding, one of the most important requirements is that the path is strictly continuous [7]. In fused deposition modelling, recent studies show that the use of continuous carbon fibre-reinforced thermoplastics improves the mechanical properties of the model compared to different approaches like the use of additive of short fibres [8, 12]. Yao et al. [14] employed a continuous toolpath planning strategy for fused filament fabrication that not only reduced the post-processing time but also supported material recycling and reuse. When it comes to scaffold fabrication, it is very important to minimize the number of start and stop points, as they cause material agglomeration, which does not satisfy the Tissue Engineering goals [4]. Finally, there are materials, such as clays [9] or bioinks, which do not have the ability to retract

and the dripping produced by travelling movements could produce undesired material overlap. This fact becomes especially relevant in bioprinting, where respecting the internal geometry with a high fidelity is crucial to guarantee cell survival and proliferation [10]. However, it cannot be guaranteed that non-convex polygons, i.e. those with at least one interior angle greater than 180°, can be continuously filled with any separation between parallel lines and orientation. Therefore, if the area to be filled is concave, it will be necessary to break it down into convex areas. In this manuscript, we present a methodology for the generation of continuous infill paths that follow a hybrid zig-zag and contour pattern with configurable separation between parallel lines and direction. In this way, the print head avoids travelling movements within a closed polygon with or without holes. To do this, an algorithm that decomposes concave polygons into convex subpolygons has been extended. Then we present an algorithm that fills each of the convex areas with the desired pattern and joins each of them into a single continuous path. Finally, we provide a publicly available dataset of 2D polygons that present challenges to the convex decomposition and path joining algorithms, such that other researchers can evaluate their own methods and compare with our results. The outline of this article is organized as follows: section 2 shows a summary of the most relevant literature. Section 3 explains the method followed for path planning. Section 4 shows and discusses the results obtained by this method. Finally, section 5 summarizes the conclusions drawn about the proposed method.

2 Literature review

2.1 Generation of continuous path

Hergel et al. [9] propose a method to generate continuous paths without transfer movements to avoid artefacts when printing ceramics, but that prioritizes to be self-supporting rather than having a concrete internal structure. Zhao and Guo [15] reviewed the most typical patterns used in conventional AM, namely raster, zig-zag, contour, grid, spiral, Hilbert filling curve and honeycomb. Among the mentioned patterns only the zig-zag, the spiral, the Hilbert filling curve and the honeycomb can be obtained with continuous paths. Zhao et al. [16] propose a variation of the conventional spiral, resulting in a single connected path of Fermat spirals, but which doesn't allow the direction of the infill to be changed. Bertoldi et al. [3] agree that when the head deposits long straight lines, the quality of union can be very poor, so they proposed using the Hilbert curve. However, the printing time is between 2 and 4 times longer than other patterns. Hexagonal honeycombs offer excellent mechanical strength at low density, but like the Hilbert curve, the printing time increases considerably compared to other patterns [6]. Ding et al. [5] present a variation of contour and zig-zag pattern, obtaining a continuous path that acquires the precision of the contour pattern and the simplicity of the zig-zag. Besides, zig-zag pattern works well for filling with high density, the computational cost to generate it is relatively short and its internal structure is very regular, which makes it suitable for being applied in fields that require fidelity in internal structures. The main disadvantage of this pattern, in contrast, is that continuity cannot be guaranteed for every direction and polygon combination. In fact, for many concave polygons it will be impossible to obtain a continuous path if they are not processed previously. Section 2.2 discusses different methods to break down concave polygons into convex areas.

2.2 Polygon decomposition

Whenever we want to fill in a non-convex polygon with a continuous zig-zag pattern we will proceed to break it down into convex parts. There are different methods to decompose concave polygons into convex subpolygons. De Berg et al. [2] collect several methods of triangular decomposition which would result in multiple convex triangular areas, but some triangle angles could be too sharp, potentially leading to overlapping material. Decomposing a polygon into trapezoids is a valid option that was catalogued by Asano [1] as an NP-complete problem with a $\mathcal{O}(n \log n)$ time approximation algorithm if the polygon contains holes. Ding et al. [5] proposed a method to decompose a concave polygon into convex subpolygons following a divide and conquer strategy. This method defines as notches all the vertices that join edges that form internal angles greater than 180 degrees, and tries to eliminate them by 3 different routines depending on the number of notches and vertices that lie between the extensions of the two edges that form each notch and their distance to the bisector line. Given its relative simplicity and its compatibility with the zig-zag pattern, we extend their formulation.

2.3 Limitations of the literature

The method proposed by Ding et al. [5] does not include the following aspects in relation to the generation of continuous paths with hybrid zig-zag and contour pattern:

- The method to obtain convex polygons does not mention the possibility of two extension lines intersecting in different polygons, and therefore does not state how these cases should be managed.
- The method to obtain convex polygons may achieve a suboptimal solution when a hole is contained in the area covered by the extension lines and the polygon in which the node to be removed is located.

– The method to obtain convex polygons may achieve a suboptimal solution because only contemplates joining the reference notch with the notch/vertex that is closest to the bisector.
– The method to obtain a continuous trajectory does not cover how to join the paths of different subpolygons. In addition, the literature does not include a 2D polygon dataset with its coordinates that would allow to check and compare the performance of convex polygon decomposition and the joining of the paths of the decomposed subpolygons.

3 Methodology

The infill generation produces the paths that will be printed in the internal area of the polygons of each layer that are obtained from the slicing of the model. It means that only geometrical aspects are considered but not the process parameters (deposition, speed, acceleration, etc.). In this work, we implemented an algorithm that aims to fill in the polygons that form a 2D layer following a continuous hybrid zig-zag and contour pattern with any direction and separation between parallel lines. Figure 1 shows a polygon of a 2D layer and a possible infill.

Given a 2D polygon $P \subset \mathbb{R}^2$ that may contain holes and that is closed and bounded, the goal is to obtain a sequence of points $P_1, P_2, \ldots, P_k \in \mathbb{R}^2$, that defines a closed and continuous polyline, without self-intersections, and moreover such polyline forms a hybrid zig-zag pattern which fills the area described by the input polygon P.

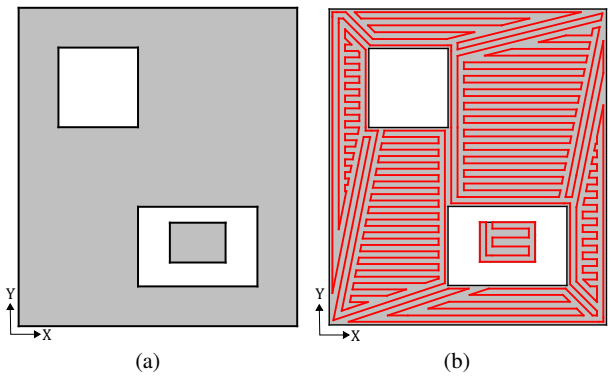


Fig. 1 Definition of the problem to be solved in this work. **a)** Polygon P to be filled, **b)** Polygon P filled with a possible hybrid zig-zag and contour pattern

The scope of the methods proposed in this work is shown resumed in Figure 2.

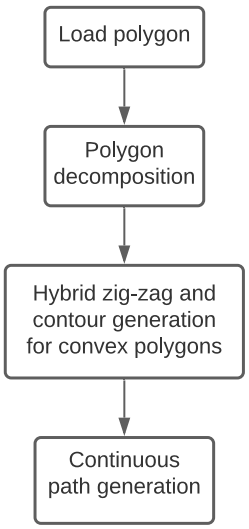


Fig. 2 Flowchart of the main steps of the method described in this work

3.1 2D layer layout

For each 2D layer there is a hierarchy of polygons in the form of a tree, like the one shown on the left of Figure 3. Even-level polygons are polygons that are going to be filled, and their vertices are numbered counter-clockwise, whereas odd-level polygons are holes that do not have to be filled and their vertices are numbered clockwise. It is assumed that these structures are already defined and will be input parameters for our algorithm. To simplify the problem, the algorithm restructures the 2D layer, so that all the even-depth polygons greater than 1 are set as level 0 polygons and their holes as level 1 polygons. By eliminating the nests, a new tree with maximum depth 2 (see the right side of Figure 3) is obtained and each sibling polygon is tackled as if they were independent problems.

3.2 Polygon decomposition

The first step is to check if the polygon is convex or not, for which we check if there are internal angles greater than 180 degrees, as Ding et al. [5] do. If it is convex, the decomposition is unnecessary and we can proceed to generate the infill (Section 3.3). Otherwise, we will need to carry out the decomposition, for which we present an extension of the method of Ding et al. [5] that handles cases that were not considered in their formulation. On the one hand, the method of Ding et al. [5] does not consider the possibility that the extensions of the edges that form a notch intersect two different polygons, which is illustrated in Figure 4(a). It turns out that in these cases it is not possible to determine

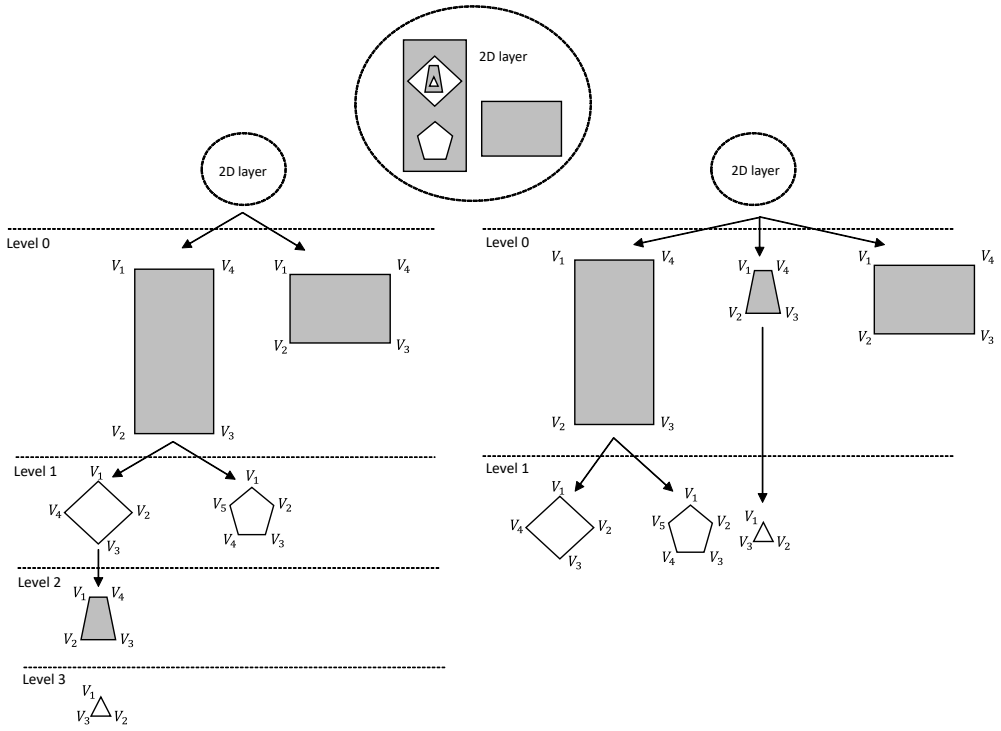


Fig. 3 Example of the hierarchy of polygons and vertex numbering that can be found in a 2D layer

how many vertices and notches remain between both since they are two different contours. In our formulation, we have catalogued those notches as unmanageable and we will try to eliminate them once some partition line has been generated and the vertices have been reorganized.

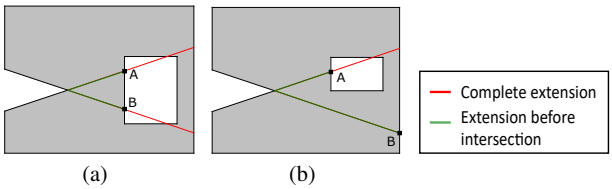


Fig. 4 First limitation of the method proposed by Ding et al. [5] to decompose polygons into convex areas. **a)** Example where both extension lines intersect the same polygon, **b)** example where the extension lines intersect two different polygons

To generate the partition lines the method of Ding et al. [5] follows 3 different routines according to the number of notches and vertices along the edges from point A to B. However, this way of counting notches and vertices presents two limitations when a hole is completely within the area covered by the extension lines and the contour of the polygon they intersect. The first limitation, which is shown in Figure 5(a), can occur when there are notches along the geometry edges between points A and B. In those cases, according to the method of Ding et al. [5] the partition line should be the one that joins the reference notch with the closest notch to the bisector line of the reference notch, thus eliminating two notches with a single partition line. However, if the hole is along the path of the partition line, the second notch that was expected to be removed cannot be removed. The second limitation is derived from the first one, and is illustrated in Figure 5(b). As we said, only the union with the notches and vertices that are along the geometry edges between points A and B is considered. However, if the notches of the hole were also taken into account, we would avoid the previous limitation and would give the possibility of eliminating the reference notch and another one of the hole with a single partition line.

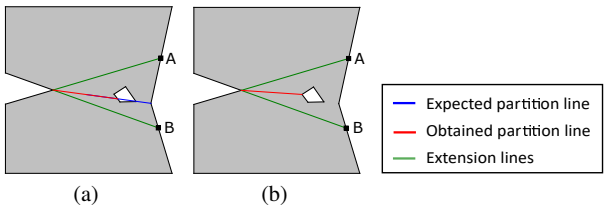


Fig. 5 Second limitation of the method proposed by Ding et al. [5] to decompose polygons into convex areas. **a)** Partition line generated following the method of Ding et al. [5], **b)** Partition line generated following the method proposed in this work

In case there is more than one notches/vertices between points A and B, according to the method of Ding et al. [5], the partition line should be formed with the notch/vertex that

is closer to the bisector line of the reference notch. However, we have found cases in which the decomposition results in more subpolygons than the necessary to have only convex areas. With the aim of obtaining better results in such cases, we also consider the option of creating the partition line with the closest notch/vertex to the reference notch. The number of subpolygons obtained with this method is always equal or smaller than following the method proposed by Ding et al. [5], but shapes can vary. In order to obtain the subpolygons with less sharp turns we look at the angles of the ends of each of the possible partition lines:

1. Get the interior angles in both ends of the possible partition line (see the red and blue angles in Figure 6) and add them.
2. Get the exterior angles in both ends of the possible partition line (see the angles plotted in black in Figure 6) and add them.
3. Keep the result of the smallest summation.
4. Repeat steps 1-3 for the other possible partition line.
5. Compare the results of each possible partition line that have been stored. The definitive partition line will be the one with the greatest result.

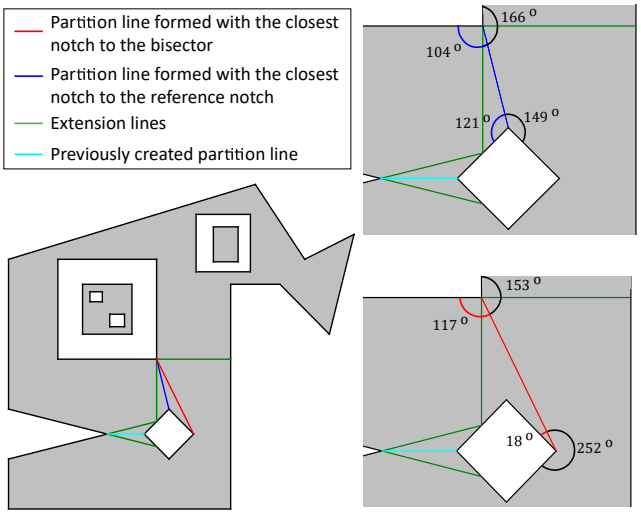


Fig. 6 Evaluation made to choose the partition line that gives better results. The selected partition line is the blue one

Figure 7 presents the results obtained following the decomposition method proposed by Ding et al. [5] and the method proposed in this work.

Once the decomposition is finished, it is necessary to identify those edges shared by two subpolygons, and in case they have different lengths, add the necessary vertices to the one with the greatest length, so that the edge they finally share has the same vertices (see Figure 8). Although this may mean having collinear segments in the same subpolygon, it will simplify the union of their zig-zags.

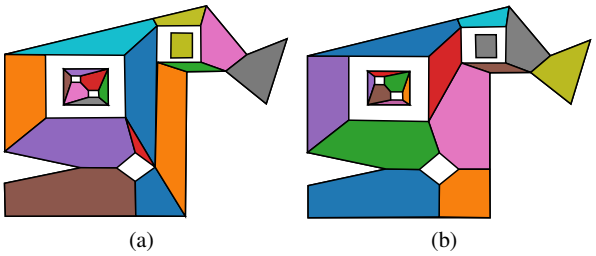


Fig. 7 Comparison of the convex subpolygons obtained following the method proposed by Ding et al. [5] and the method proposed in this work. **a)** Polygon decomposition achieved following the method proposed by Ding et al. [5], where 19 convex subpolygons are obtained **b)** Polygon decomposition achieved following the method proposed in this work, where 18 convex subpolygons are obtained.

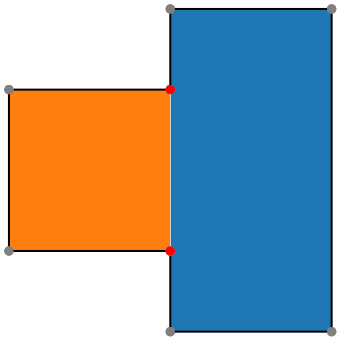


Fig. 8 Proposed vertices for convex subpolygons. Red dots, which were not originally part of the blue subpolygon, have been added to ease the generation of the continuous pattern

3.3 Path generation for convex polygons

The steps to generate the path with the desired orientation and distances between lines for convex polygons are described below:

1. Transform the polygon to a coordinate system in which the main lines of the zig-zag are parallel to the horizontal axis, which simplifies the code.
2. If the polygon to be filled is the result of the decomposition of a non-convex polygon, apply an offset of a distance D equal to one of the zig-zag separations. This will prevent it from overlapping with the polygon next to it, as well as giving enough space for the union between them. The resultant polygon, which will be called *unionPolygon*, is shown in green in Figure 9(a). If *unionPolygon* cannot be generated, no infill pattern can be generated. In case the polygon was originally convex, *unionPolygon* is the polygon itself.
3. Identify the left chain of *unionPolygon*, which is defined as the polyline that goes from the lowest to the highest vertex of the polygon by the left side of it. The left chain of *unionPolygon* is plotted in dark blue in Figure 9(b).
4. Offset the left chain the same distance as in step 2 and form a new polygon using the remaining edges of *union-*

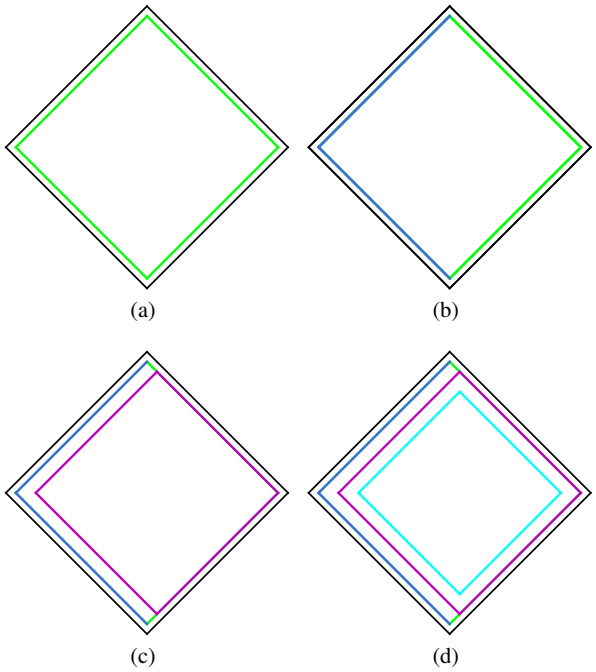


Fig. 9 Auxiliary polygons used to generate the infill pattern. **a)** *unionPolygon* **b)** *leftChain* of *unionPolygon* in blue **c)** *contourPolygon* in violet **d)** *zigzagPolygon* in cyan

Polygon. The resultant polygon, called *contourPolygon*, is used together with *unionPolygon* to generate the contour of the path, and it is graphed in violet in Figure 9(c).

5. Offset *contourPolygon* the same distance as in step 2. The resultant polygon, plotted in cyan in Figure 9(d), is called *zigzagPolygon* and it is used to create the zig-zag of the path. If *zigzagPolygon* cannot be generated (result of too wide line separations), the resultant infill pattern will be the contour of *unionPolygon*.
6. To generate the zig-zag, build horizontal lines from the lowest point of *zigzagPolygon*, varying the separation between them according to the side of the turn of the zig-zag, until the highest point is reached and look for the intersections with *zigzagPolygon*. Join the intersections by the right and left side alternatively following the contour of *zigzagPolygon*. Figure 10(a) shows the evolution of the zig-zag and Figure 10(b) shows the final zig-zag pattern. At this point, we have implemented an automatic adjustment of the chosen line separations (which can be enabled or disabled) to ensure that the polygon is completely filled at the top.
7. Extend the zig-zag, following if possible the slope of the corresponding edge of *zigzagPolygon*, to *contourPolygon*. If the extension of the zig-zag intersects the left-chain of *contourPolygon* (see Figure 11(a)), a line parallel, at a distance *D*, to the first segment of the left-chain is defined and its point of intersection with *contourPolygon* will be the point of union between the zig-zag and the contour (see Figure 11(b)). This step turns to be especially relevant when the variation of the slope of adjacent edges is very slight (curved polygons, for example) and when the chosen line separations is very big, where the number of vertices of *contourPolygon* and *zigzagPolygon* is not the same. Once the union is made, go through *contourPolygon* in clockwise direction until the lowest part of its left-chain is reached.

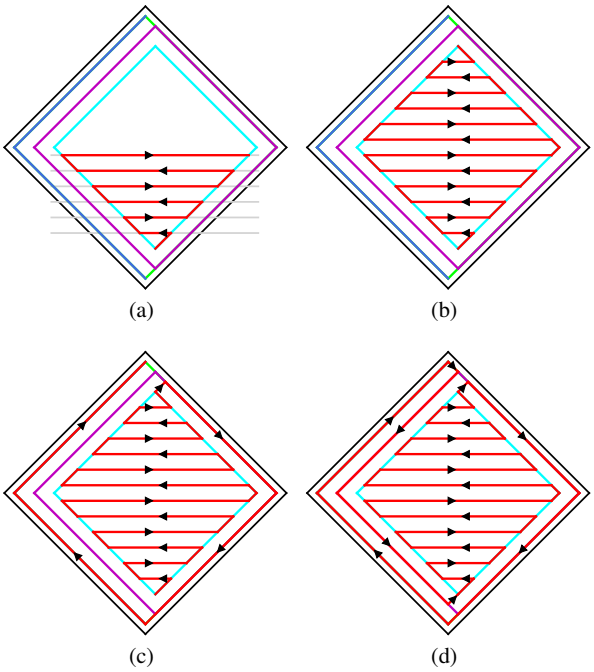


Fig. 10 Hybrid zig-zag and contour infill pattern generation for convex polygons **a)** Generation of the zig-zag **b)** Zig-zag pattern **c)** Extension to contour and traverse of *unionPolygon* **d)** Finish the contour, using *contourPolygon* and union with the zig-zag

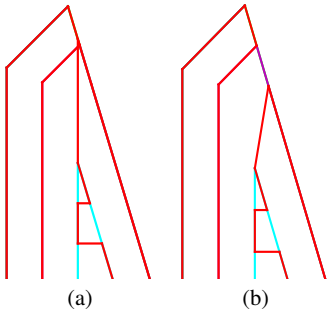


Fig. 11 Extension of the zig-zag to the contour **a)** self-intersecting path because the slope of the corresponding edge of *zigzagPolygon* makes the extension intersect the left chain of *contourPolygon* **b)** alternative extension to get a non-self-intersecting path

8. Traverse *unionPolygon* in a clockwise direction until the highest part of the left-chain of *contourPolygon* is reached, as shown in Figure 10(c).
9. Go through *contourPolygon* in counter-clockwise direction, as shown in Figure 10(d), until the point where the extension of the first line of the zig-zag and *contourPolygon* would intersect, and join it with the beginning of the zig-zag. Figure 10(d) shows the final path of a convex polygon with a horizontal continuous hybrid zig-zag and contour pattern.
10. Transform the original polygon and the whole pattern to the original coordinate system. Figure 12 shows the infill of the same polygon with the same separation between lines and different orientations.

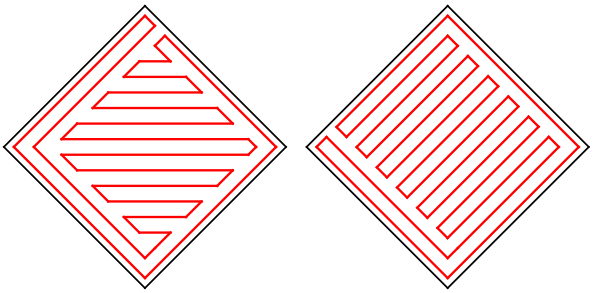


Fig. 12 Possible infills for a convex polygon following a hybrid zig-zag and contour pattern infill **a)** Horizontal pattern **b)** pattern with 45°

3.4 Joining of subpolygon paths into a single path

To carry out the union of the paths of the different subpolygons, which will be made around the edges shared by two of them, we make a graph of adjacency, where each subpolygon is a node and its degree is the number of shared edges it has. From the adjacency graph we construct a spanning tree as follows:

1. We identify the root node as the node with lowest degree and, in case of a tie, we choose the first-generated node among them.
2. We mark that root node as visited.
3. Starting from the root node, we proceed with a recursive traversal of the graph: we mark the edges that connect the current node of the traversal with a still-unmarked node, mark those still-unmarked nodes, and then recurse the traversal to them in order.

The spanning graph, which is shown in Figure 13(b), is formed by all the marked edges and has the identified root. This way it is assured that the paths of all the subpolygons will be joined at least with another path, and globally all of them will be joined by the paths of other subpolygons.

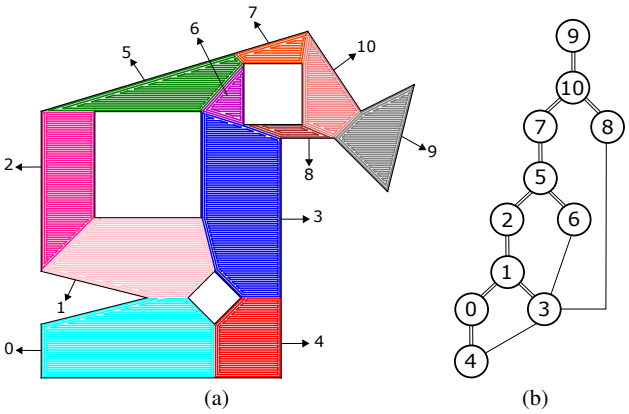


Fig. 13 Proposed method to define which paths should be joined to get a single continuous path **a)** paths of convex subpolygons that have to be joined **b)** adjacency graph used to identify which subpolygons have to be joined. The double lines indicate which is the spanning tree

The steps to make the union of the paths of two subpolygons are stated below:

1. Identify the two points of the same path that delimit the line parallel and closer to the edge that is shared by the subpolygons to be joined. These points are plotted in grey in Figure 14.
2. Decide around which two points the union is going to be made, and identify them as p_1 and p_2. To try to avoid very sharp turns, the internal angle that would form each subpolygon in the environment of each point is calculated and if any of these angles is lower than 25 degrees, it is proposed to make the union around the other end. If this is not the case, the angles of the two subpolygons around the same end are summed up and the union is made at the end at which there is an angle closer to 0, 90, 180 or 270 degrees, prioritizing the angles closer to 0 and 180 over 90 and 270 and the convex ones over the concave ones.

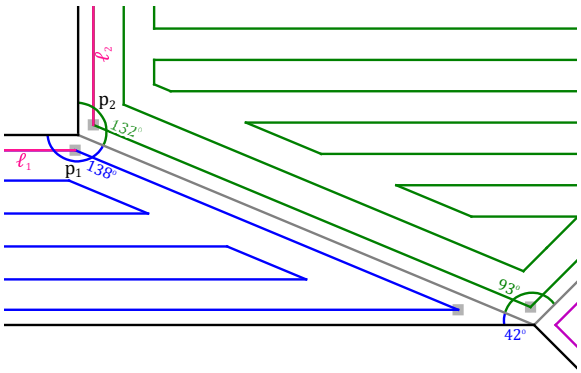


Fig. 14 Selection of the area where the union is going to be made

3. Extend ℓ_1 and ℓ_2 and look for their point of intersection, which is called $p_{1,2}$ (see Figure 15).
4. Generate two lines parallel to ℓ_1 and ℓ_2 at a distance D (see ℓ'_1 and ℓ'_2 in Figure 15). The intersection points of these lines with their respective zig-zags are labelled as p'_1 and p'_2 and the intersection of both as $p'_{1,2}$. There are 2 cases that can vary the way to obtain these points.

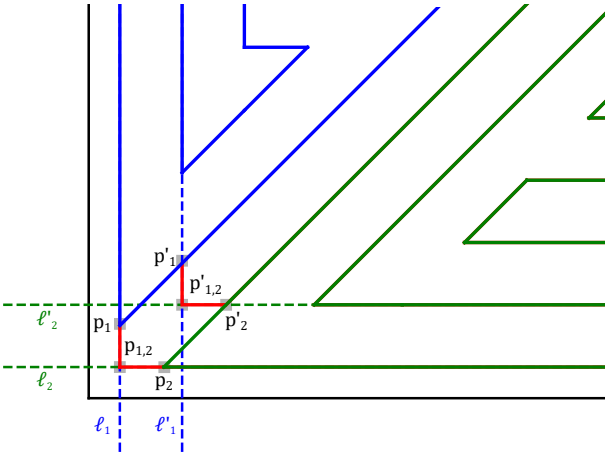


Fig. 15 Auxiliary points used to make the union of two paths

(a) Variation 1: $p'_{1,2}$ is inside one of the zig-zags. Find the intersection of the line of the zig-zag in which $p'_{1,2}$ is not contained with the zig-zag in which $p'_{1,2}$ is contained. This point is shown as a dot with the name $p'_{1,2_{aux}}$ in Figure 16.

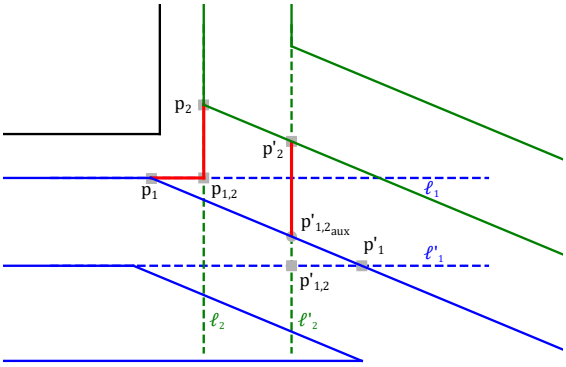


Fig. 16 Variation 1 to obtain the points needed to make the union

(b) Variation 2: ℓ'_1 or ℓ'_2 is left out of the zig-zag because the polygon is too small. In that case, the procedure used to obtain $p_{1,2}$ is repeated at the other end, such that p_1 and p_2 become p'_1 and p'_2 and their intersection $p'_{1,2}$. Figure 17 shows this type of union, where the green pattern, due to its small size, is the cause of this variation.

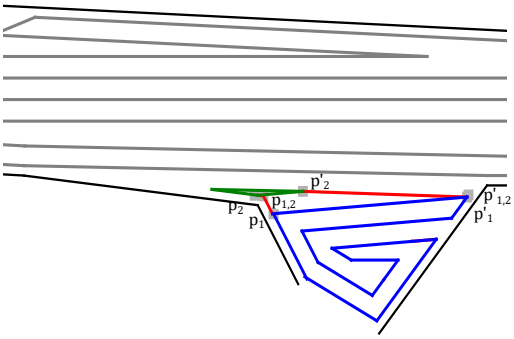


Fig. 17 Variation 2 to obtain the points needed to make the union

5. Modify the original paths to join them by means of the found points. p_1 and p_2 are joined with $p_{1,2}$, and the segments that joined p_1 with p'_1 and p_2 with p'_2 are removed. p'_1 and p'_2 will be joined between them, either directly or by crossing $p'_{1,2}$. To decide whether to pass through $p'_{1,2}$, calculate the internal angles that would result from going through it or not. From each of the options, the minimum angle is found and the greater of the two is chosen in order to avoid very abrupt turns, as it can be seen in Figure 18. In case of a tie, going through $p'_{1,2}$ is preferred over the direct connection, as it respects more the fidelity of the contour of the polygon

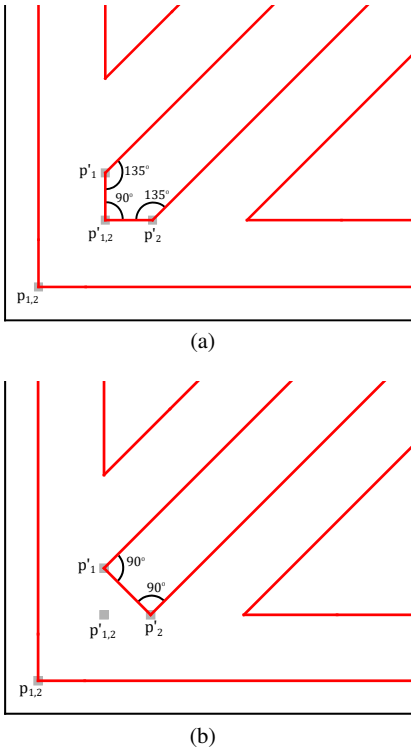


Fig. 18 Decide if p'_1 and p'_2 union should go through $p'_{1,2}$ or not **a)** Going through $p'_{1,2}$ the minimum angle is 90° **b)** Not going through $p'_{1,2}$ the minimum angle is 90°. As the angle is the same, the chosen option is **a** as it respects more the fidelity of the contour of the polygon

Once all the zig-zags have been joined and a unique continuous path has been achieved all the collinear segments that may have been generated are removed.

3.5 Evaluation

In order to validate the results obtained in both the non-convex polygon decomposition and the path generation, and given the lack of a 2D polygon dataset with which to check them, it has been decided to generate one. The repository[1] includes the coordinates of the polygons, together with a justification of why they are relevant, and the results obtained with our algorithm. In terms of the polygon decomposition the dataset allows the following situations to be evaluated:

- The extensions of the edges that form the reference notch intersect two different boundaries.
- There is a complete hole inside the area covered by the extensions of the edges that form the reference notch.
- Intersection of lines with the same slope (overlap of lines).

When it comes to path generation, the dataset covers the following cases of relevance:

- The edge of union of two subpolygons are not exactly the same, but one is longer than the other.
- The edge of one of the subpolygons is used as a double union area with two different subpolygons.
- Need to generate a graph that indicates which subpolygons are to be joined together.
- Ideal position of the points used to generate the union of zig-zags.
- The position of the points used to generate the union of zig-zags makes it necessary to use variation 1.
- The position of the points used to generate the union of zig-zags makes it necessary to use variation 2.

4 Results

Our algorithm has been successfully implemented for polygons with any number of holes. The algorithm aims to be able to fill polygons following any separation between lines (as long as they are smaller than the polygon itself, or any of the decomposed subpolygons) and zig-zag orientation, but a variant that allows to choose the orientation for each of the convex subpolygons has been developed in order to make a more realistic comparison with the results obtained by Ding et al. [5]. Both their results and those obtained in this work are shown in Table 1. As we do not have the exact coordinates of the polygons or the parameters of the fill patterns used by Ding et al. [5], we can only make a visual comparison of the results, with no conclusions about the number of path elements. Both the coordinates of the polygons and the parameters we have used to obtain our results are listed in the repository to ease quantitative comparisons with other works. The results obtained satisfy the needs that were sought to be covered: obtain a strictly continuous path, following any orientation and with any zig-zag separations, as long as they are not greater than that of the polygon itself (or any of the decomposed subpolygons). The number of toolpath elements is something that has no relevance in this work since it is conditioned by the separations that are imposed on the zig-zag. Figure 19 shows a result obtained for a complex polygon. However, the proposed method is not entirely appropriate for the generation of paths in polygons that have holes with curves. As circles are formed by vertices at infinitesimal distances, there would be many notches to eliminate and the resulting convex areas would be too narrow to be filled with the zig-zag pattern without overlapping the material. Besides, when the size of the subpolygon and the configured separation between lines does not allow to generate any infill (not even the simplified version using *unionPolygon*), it will not be possible to generate a continuous trajectory for the entire non-convex polygon if that subpolygon is the only connecting path between other subpolygons.

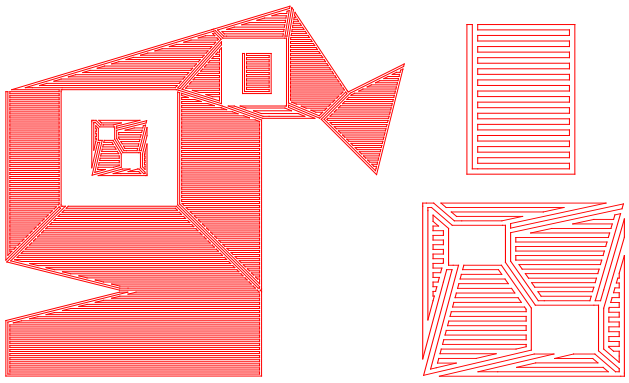


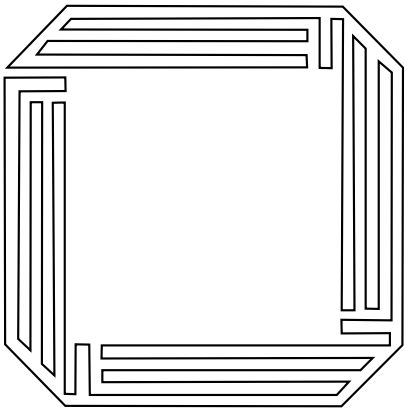
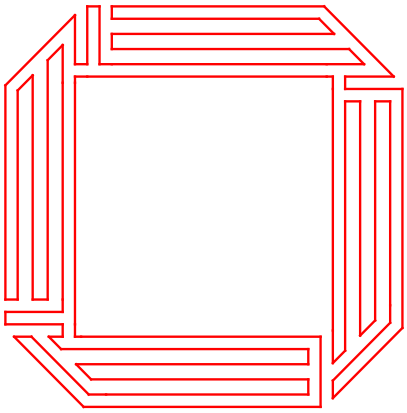
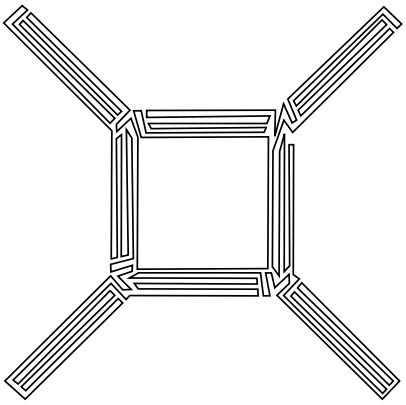
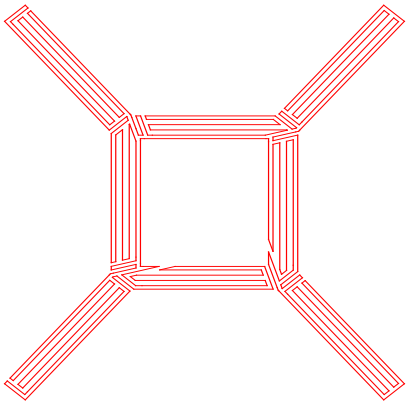
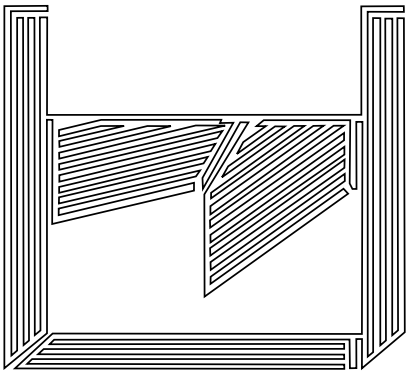
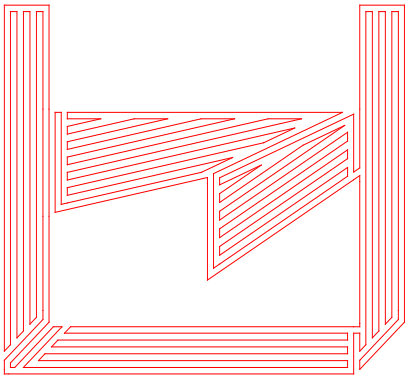
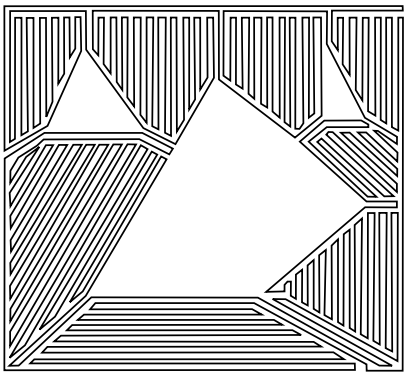
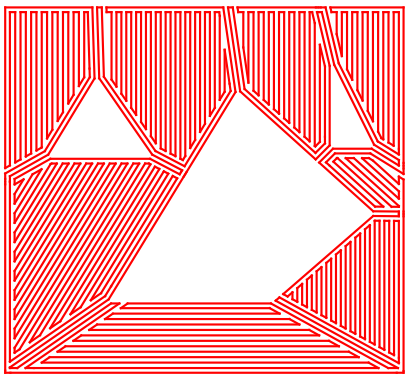
Fig. 19 Final result for a complex polygon **a)** complex polygon infill path **b)** zoom of the paths of the polygons found within the holes

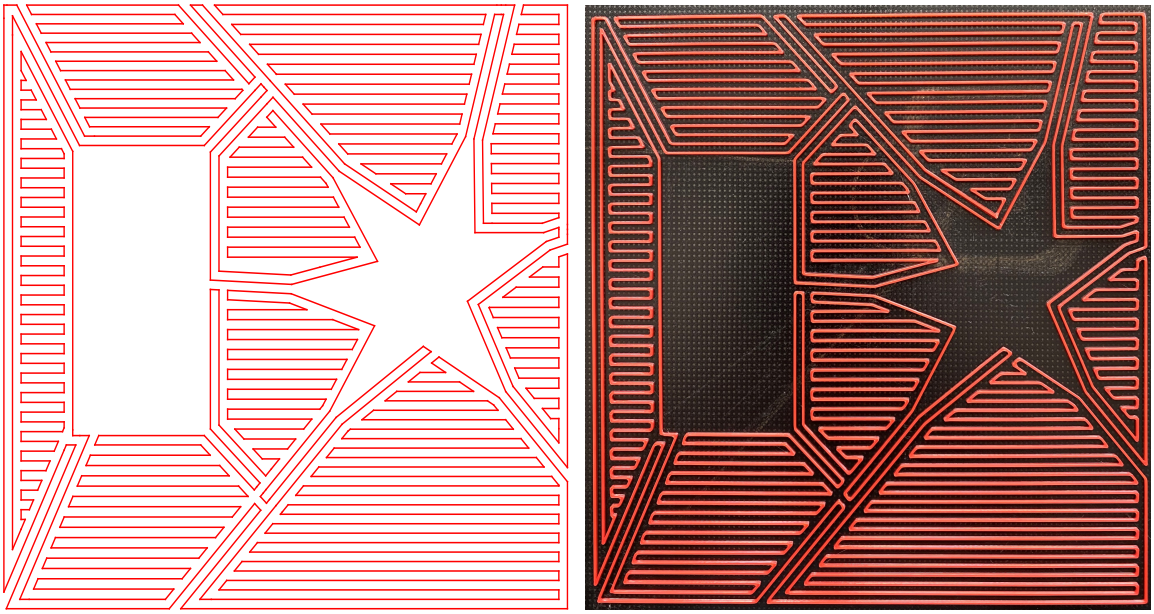
Experimental tests have been conducted using a Creality Ender-5 Plus printer. Polylactic acid (PLA) was extruded with no retraction during printing. The feed rate was set at 1.2m/min and the extruder speed was 45mm/s, obtaining a trace of 0.51mm of diameter. The object of Figure 20(b) was filled with 2mm of separation between lines, while the object of Figure 21(b) with 0.5mm.

The dataset of the repository introduced in section 3.5 contains 35 *.json* files with the coordinates of 15 convex polygons, with which we aim to check that the generation of the hybrid zig-zag and contour infill pattern is correctly

[1] https://github.com/Vicomtech/Dataset-of-2D-polygons-for-Additive-Manufacturing

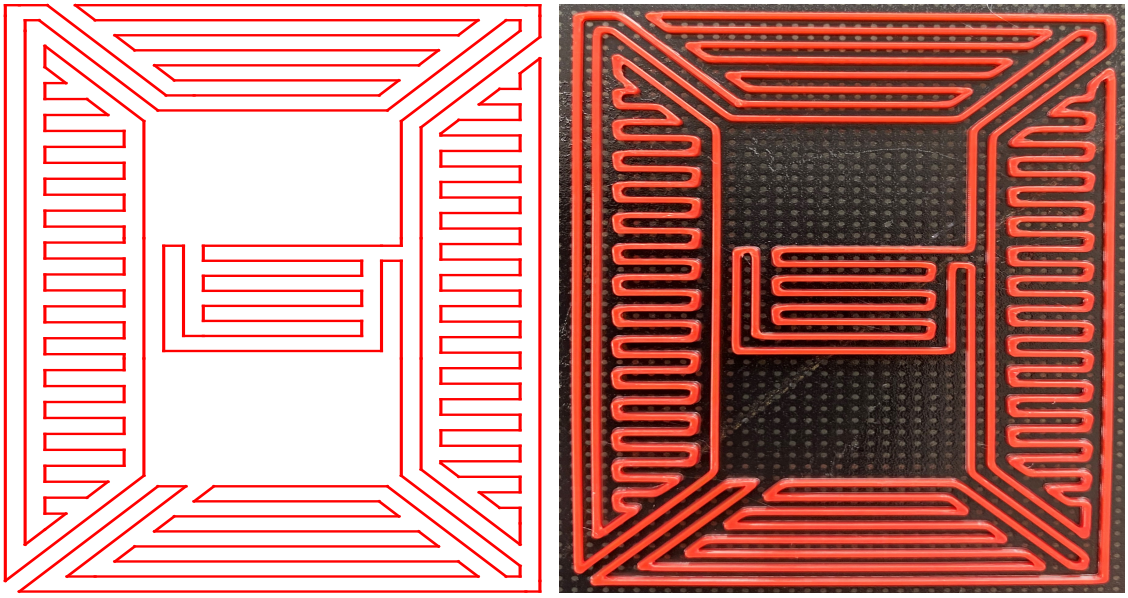
Table 1 Comparison of a reproduction of the results obtained with the method proposed by Ding et al. [5] and the method proposed in this work

	Replication of Ding et al. [5] method	Proposed method
1		
2		
3		
4		



(a) (b)

Fig. 20 Comparison of the expected (**a**) and the obtained (**b**) results using the experimental setup described in Section 4



(a) (b)

Fig. 21 Comparison of the expected (**a**) and the obtained (**b**) results using the experimental setup described in Section 4

performed, and 20 non-convex polygons, which aim to validate the algorithms of decomposition and generation of continuous paths. The repository also contains a *.txt* file corresponding to each of the polygons, where the coordinates of the infill paths obtained with 48 combinations of separation between lines and zig-zag orientation are stored, except for the *NCPolygon20.json*, which only has 1 single configuration for the separation between lines, which means a total of 1,633 infill paths. Finally, the repository also contains Python3 code that can be used to load and visualise the polygons and their infills.

5 Conclusions and future work

This work has presented the following contributions:

– Improvement of the current state of art of polygon decomposition into convex areas and although we do not

guarantee that we have a global optimal solution, we can get less subpolygons than in other methods of decomposition [1, 2, 5].
- A method to create a closed hybrid zig-zag and contour infill pattern for convex polygons.
- A method to join the hybrid zig-zag and contour patterns of two adjacent subpolygons, obtaining a single continuous path for the whole polygon, which makes it possible to be used in a wide variety of modes of additive manufacturing.
- A publicly available dataset of 2D polygons to check and compare the results.

This method, however, has certain limitations. On the one hand, although a valid result can be achieved, our method is not the most suitable for filling polygons with curved holes, as it would generate many subpolygons with very sharp areas where material could overlap. On the other hand, this work does not cover the way in which the different polygons that make up a layer should be joined, nor the union between two consecutive layers.

Declarations

Funding

Partial funding obtained from the Basque Government through the Elkartek KK-2019/00006 grant for project Bioprinting4Healing.

Conflicts of interest

The authors declare that they have no conflict of interest.

Ethical approval

No ethical approval was required for this research.

Consent to participate

Not applicable.

Availability of data and material

Data is available in the following repository: `https://github.com/Vicomtech/Dataset-of-2D-polygons-for-Additive-Manufacturing`

Code availability

Code is not available.

Author's contributions

References

1. Asano T, Asano T, Imai H (1986) Partitioning a Polygonal Region into Trapezoids. Journal of the ACM (JACM) 33(2):290–312, DOI 10.1145/5383.5387
2. Berg Md, Cheong O, Kreveld Mv, Overmars M (2008) Computational Geometry: Algorithms and Applications, 3rd edn. Springer-Verlag TELOS, Santa Clara, CA, USA
3. Bertoldi M, Yardimci Ma, Pistor CM, Giiveri SI (1998) Domain Decomposition and Space Filling Curves in Toolpath Planning and Generation. Proceeeding of the 1998 Solid Fabrication Symposium pp 267–274
4. Dávila JL, Freitas MS, Neto PI, Silveira ZC, Silva JV, D'Ávila MA (2016) Software to generate 3-D continuous printing paths for the fabrication of tissue engineering scaffolds. International Journal of Advanced Manufacturing Technology 84(5-8):1671–1677, DOI 10.1007/s00170-015-7866-8
5. Ding D, Pan Z, Cuiuri D, Li H (2014) A tool-path generation strategy for wire and arc additive manufacturing. International Journal of Advanced Manufacturing Technology 73(1-4):173–183, DOI 10.1007/s00170-014-5808-5
6. Domínguez-Rodríguez G, Ku-Herrera JJ, Hernández-Pérez A (2018) An assessment of the effect of printing orientation, density, and filler pattern on the compressive performance of 3D printed ABS structures by fuse deposition. International Journal of Advanced Manufacturing Technology 95(5-8):1685–1695, DOI 10.1007/s00170-017-1314-x
7. Dwivedi R, Kovacevic R (2004) Automated torch path planning using polygon subdivision for solid freeform fabrication based on welding. Journal of Manufacturing Systems 23(4):278–291, DOI 10.1016/S0278-6125(04)80040-2
8. Hedayati SK, Behravesh AH, Hasannia S, Bagheri Saed A, Akhoundi B (2020) 3D printed PCL scaffold reinforced with continuous biodegradable fiber yarn: A study on mechanical and cell viability properties. Polymer Testing 83(January):106347, DOI 10.1016/j.polymertesting.2020.106347, URL `https://doi.org/10.1016/j.polymertesting.2020.106347`
9. Hergel J, Hinz K, Lefebvre S, Thomaszewski B (2019) Extrusion-Based Ceramics Printing with Strictly-Continuous Deposition. ACM Transactions on Graphics DOI 10.1145/3355089.3356509, URL `https://hal.inria.fr/hal-02363424`
10. Ma Z, Wang J, Loskill P, Huebsch N, Koo S, Svedlund FL, Marks NC, Hua EW, Grigoropoulos CP,

Conklin BR, Healy KE (2015) Self-organizing human cardiac microchambers mediated by geometric confinement. Nature Communications 6(May):1–10, DOI 10.1038/ncomms8413, URL http://dx.doi.org/10.1038/ncomms8413

11. Oliveira JP, Santos TG, Miranda RM (2020) Revisiting fundamental welding concepts to improve additive manufacturing: From theory to practice. Progress in Materials Science 107(June 2019):100590, DOI 10.1016/j.pmatsci.2019.100590, URL https://doi.org/10.1016/j.pmatsci.2019.100590
12. Sugiyama K, Matsuzaki R, Malakhov AV, Polilov AN, Ueda M, Todoroki A, Hirano Y (2020) 3D printing of optimized composites with variable fiber volume fraction and stiffness using continuous fiber. Composites Science and Technology 186(November 2019):107905, DOI 10.1016/j.compscitech.2019.107905, URL https://doi.org/10.1016/j.compscitech.2019.107905
13. Urhal P, Weightman A, Diver C, Bartolo P (2019) Robot assisted additive manufacturing: A review. Robotics and Computer-Integrated Manufacturing 59(May):335–345, DOI 10.1016/j.rcim.2019.05.005
14. Yao Y, Li M, Lackner M, Herfried L (2020) A continuous fiber-reinforced additive manufacturing processing based on PET fiber and PLA. Materials 13(14), DOI 10.3390/ma13143044
15. Zhao D, Guo W (2020) Shape and Performance Controlled Advanced Design for Additive Manufacturing: A Review of Slicing and Path Planning. Journal of Manufacturing Science and Engineering, Transactions of the ASME 142(1):1–23, DOI 10.1115/1.4045055
16. Zhao H, Gu F, Huang QX, Garcia J, Chen Y, Tu C, Benes B, Zhang H, Cohen-Or D, Chen B (2016) Connected fermat spirals for layered fabrication. ACM Transactions on Graphics 35(4):1–10, DOI 10.1145/2897824.2925958