Virtual prototyping for the textile industry: A framework supporting the production of crocheted objects

ARTICLE TEMPLATE

Virtual Prototyping for the Textile Industry: A Framework Supporting the Production of Crocheted Objects

Alberto Cannavò[a], Massimo Gismondi[a] and Fabrizio Lamberti[a]

[a]Department of Control and Computer Engineering, Politecnico di Torino, Corso Duca degli Abruzzi, 24, Torino, Italy

ARTICLE HISTORY
Compiled August 24, 2023

ABSTRACT
Over the last years, many progresses have been made in the field of virtual prototyping, pushed by the interest of industries and artisans. Especially in the context of the textile industry, the digitizing of the prototyping stage offers the possibility to validate the product design choices before committing to the market.

This paper presents a framework for the virtual prototyping of crocheted objects. The core of the framework is an algorithm that is capable of generating the crocheting patterns for a given object and the corresponding instructions. The instructions are leveraged by the framework to visualize the 3D geometry of the object, and can be also used to craft it. Compared to previous works, the proposed algorithm combines a number of features (primarily, the use of parametric surfaces and the support for short rows) that can reduce the distortions in crafted object shape while also lowering computational cost; the algorithm is also able to consider material- and style-related information.

The results of a comparison between the proposed algorithm and state-of-the-art approaches showed improved performance in terms of similarity of the generated shape with the target one, computation time, and appearance of the crafted object.

KEYWORDS
Virtual prototyping; Crocheting instruction generation algorithm; Short rows; Digital crafting; Parametric design

1. Introduction

When dealing with product development, one of the most common applications of computer graphics (CG) is virtual prototyping, i.e., the digitalization of the prototyping phase, aiming at validating a design before committing to making a physical object (Lee 2001). Virtual prototyping is generally characterized by an iterative process (depicted in Fig. 1), which encompasses the modeling, simulation, and visualization of a product to assess it under representative operating conditions, and refine its design to possibly accommodate new needs. The process typically includes the production of computer-generated geometrical shapes, which are tested for, e.g., appearance, mechanical properties or functioning (Wu, Shao, and Liu 2019; Choi and Cheung 2008). The output of the above process can be additionally exploited to automatically produce also the instructions needed for crafting the object, either as a physical prototype

CONTACT A. F. Lamberti. Email: fabrizio.lamberti@polito.it

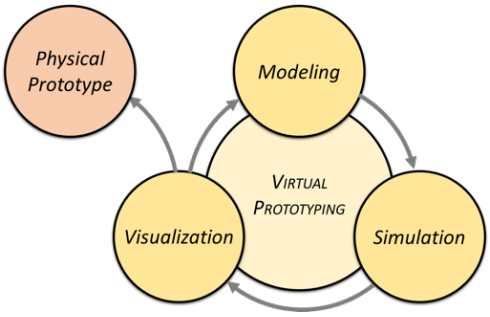


Figure 1. A schematic representation of the virtual prototyping cycle.

or final product (McCann et al. 2016).

In the textile industry and, more specifically, for crafted objects, virtual prototyping allows to save time (Jimeno-Morenilla et al. 2021), as the geometry of the final object can be visualized, e.g., to discuss materials, colors, and possible customer-specific personalizations to be applied to it (Kaiser, Vogt, and Tilebein 2017) or to simulate complex phenomena like yarn relaxation (the settlement of the sewed structure according to the existing forces between the single meshes) without the need to make a physical prototype or before actually producing it (Rubio, Sanz, and Sebastián 2005). Like other sectors (Feldmann and Christoph 2003; Carvalho Filho, Nunhes, and Oliveira 2019; Leite et al. 2019; Dong et al. 2019; Baurmann and Taimina 2013; Zhao et al. 2023; Zheng et al. 2021), the textile industry is facing an increasing interest towards the development of novel methods able to improve the manufacturing process (He et al. 2021).

As a matter of example, the work in (Jevšnik, Stjepanovič, and Rudolf 2017) investigated the use of virtual prototyping to design garments for people with special needs. In particular, the work presented tools and techniques to check the fit of garments and provide indications regarding the mechanical properties of fabrics without the need to have people physically present. The application proposed in (Vitali, D'Amico, and Rizzi 2016), in turn, mainly focused on the collection of customers' measures. It proposed to use Virtual Reality simulation and consumer devices (i.e., Oculus Rift DK2, Leap Motion Controller, and Microsoft Kinect v2) to let tailors perform the measurements as they would do in the real world. The work in (Han, Wohn, and Ahn 2021) presented a web application to support the students enrolled in fashion university programs in the design of e-textiles, i.e., clothes embedded with electronic devices. The results of a user study showed improvements in terms of perceived usability and learning performance compared to traditional methods. As shown in works like (de Oliveira Neto et al. 2020, 2021; Viganò et al. 2004), virtual prototyping could also play a crucial role in supporting more efficient decision-making strategies, or reducing the environmental impact. More examples, as well as trends and shortcomings of virtual prototyping applied to the textile and fashion industry can be found in (Sayem 2020; Papahristou and Bilalis 2017; Cugini, Bordegoni, and Mana 2008)

Despite these advantages, many of the commercial tools available today to support the textile industry provide only a limited set of functionalities for the virtual prototyping of crafted objects like, e.g., techniques for defining parametrically or modeling its shape (Kapllani et al. 2022), for simulating its physics (Sperl et al. 2022), etc.; in most of the cases, they require fine manipulations to be performed by human operators with a deep understanding of crafting operations (Popescu et al. 2018).

Based on the above considerations, the goal of the present paper is to introduce

a framework supporting virtual prototyping in the textile industry focusing, in particular, on objects produced through *crocheting*. Crocheting is a crafting technique based on the use of a single hook to realize the *stitches*, i.e., loops of yarn resulting from one pass of the hook. The presented methodology, however, could be extended also to the other major crafting technique, i.e., *knitting*, where fabrication proceeds by transferring stitches between two needles (Guo et al. 2020).

The core of the framework is represented by an algorithm in charge of generating the crocheting patterns and related instructions for the given shape. The output of the algorithm can be exploited either to visualize a computer-generated 3D geometry of the crocheted object or to produce a physical version of it. Like in (Çapunaman, Bingöl, and Gürsoy 2017), it was chosen to represent the instructions in a human-readable format. This conventional representation is crucial for sharing the crocheting procedure with others who are not familiar with CAD tools (Çapunaman, Bingöl, and Gürsoy 2017). The instructions can be used by a crafter to manually make a prototype, or converted into a specific machinery language format for automated production.

The strength of the algorithm proposed in the present paper lies in the ability to combine the following features.

First, like in (Popescu et al. 2018) and (Çapunaman, Bingöl, and Gürsoy 2017), the algorithm operates with parametric representations for computing the stitches. In particular, the algorithm receives in input a so-called target shape and converts it to a parametric surface from which it is possible to extract key information that facilitates the processing steps. More specifically, the algorithm adopts a *helix progression* for the crocheting pattern, which is the natural way of manually crafting a 3D object (Wu, Swan, and Yuksel 2019). This aspect is partially disregarded (or not considered at all) in other works like (Igarashi, Igarashi, and Suzuki 2008a; Narayanan et al. 2018; Osinga and Krauskopf 2004; Popescu et al. 2018; McCann et al. 2016), which are based on a *separate row* progression, a technique that makes use of patterns in which the yarn pieces are handled separately for each row. The use of the helix progression makes the devised algorithm not applicable to surfaces for which is not possible to compute a UV parametric representation; nevertheless, it avoids the need to rely on the non-realistic assumption of the separate row progression, which can have a negative impact on the quality of the crafted object (Wu, Swan, and Yuksel 2019).

Second, the algorithm implements strategies aimed to fit as much as possible the target shape based on the use of *short rows*, i.e., local increases in the number of rows within the stitches (Popescu et al. 2018). Short rows complement the basic crocheting patterns that contain only increase (widening) and decrease (narrowing) stitches. With short rows, the single crochet approach, which is able to create only cylindrical shapes since each knot of a level is connected with the corresponding knot of the upper level, is enhanced to support the generation of curved and doubly curved shapes.

Third, the algorithm can receive in input the size of the thread and the tension to be applied while crocheting, as well as the stylistic characteristics of the crafting, which can influence the actual size and shape of the final object.

In order to validate the effectiveness of the devised algorithm, it was compared with relevant state-of-the-art approaches by means of objective and subjective metrics. The obtained results showed improved performance with respect to the references, as the algorithm was capable to generally guarantee higher similarity to the target shape and reduced stitch distortion in the generated geometries, lower computation time, and better visual appearance of the crafted objects.

2. Related works

Nowadays, industries are more and more recognizing the value of developing novel methods supported by computer technology to improve their processes. As a matter of example, it is possible to mention the works in (Zhao et al. 2023; Zhao, Jia, and Shao 2023), which proposed solutions based on machine learning for machinery fault diagnosis. Other relevant examples of automatic methods applied to both industrial and civilian applications are reported in (Zheng et al. 2021, 2022; Jin and Vai 2014).

Narrowing the field of interest to the textile industry, the literature offers several solutions for the automatic generation of crocheting and knitting instructions and the representation of the corresponding crafting patterns. Although the focus of this paper is on crocheted objects, in the remaining of this section, both the crafting techniques are discussed, since it is possible to fairly adapt crocheting instructions to knitting and vice-versa, as suggested in (Guo et al. 2020). The review of the literature was carried out by leveraging the Scopus and Web of Science databases. Works were selected based on the relevance with the topic and the rigor of the adopted methodology. The following keywords were used in refining the selection: crocheting, knitting, instruction generation, short rows, parametric design, fabrication, digital craft.

A first example dates back to more than 25 years ago, when an approach leveraging mathematical formulas to describe the surface of the object to be crafted was proposed (Hong et al. 1994). This approach, however, could only generate primitives shapes, like cylinders, spheres, and boxes. A solution for free-form surfaces was presented in (Thomsen and Hicks 2008), where 3D shapes are translated into 2D knitting patterns by unrolling the developable surfaces, i.e., smooth surfaces with zero Gaussian curvature (Nelson et al. 2019). The drawback of this solution was that it could not be adapted to non-developable surfaces.

To overcome the limitations regarding the representation of the surface to be crafted, two different approaches have been devised. The first approach relies on the use of polygon meshes to describe the geometry of polyhedral objects. When using this approach, the input is represented by a collections of vertices, edges and faces. For instance, the system described in (Igarashi, Igarashi, and Suzuki 2008a) is able to automatically generate instructional diagrams to create a 3D knitted object starting from a triangulated geometry. The system first samples the geometry surface with parallel winding strips of constant width. Then, the samples are converted into a knitting pattern that can be used for crafting the object. Another example is the system named "Knittty" described in (Igarashi, Igarashi, and Suzuki 2008b). The system relies on sketching interfaces that allow the users to define a 3D surface model. The generated model is then leveraged by the system to automatically produce the knitting patterns and present a 3D visualization of the corresponding shape that is reconstructed by applying a physical simulation.

The second approach for representing the target shape is based on parametric surfaces. The advantage of this approach is that it facilitates the algorithmic steps reducing the need for human supervision. For instance, a parametric representation maintains information regarding the development direction of the surface, which could be used as the crafting direction. With polygonal meshes, this information is missing, and the algorithms leveraging such representation require a manual or semi-automatic labeling of the faces (e.g., in (Wu, Swan, and Yuksel 2019)) or the assignment of weights to the vertices (e.g., in (Narayanan et al. 2018)) for guiding the process. Depending on the target shape, these operations may not be always possible. An example of use of parametric surfaces is given in (Osinga and Krauskopf 2004). This work reports the

mathematics behind the crocheting of the Lorenz manifold. The mathematical representation is used to created a 3D mesh of the surface, which is then automatically converted into the crocheting patterns for manual crafting.

When dealing with complex target shapes, the level of control on the crocheting pattern becomes of fundamental importance. In this regard, short rows represent a key contribution to the creation of bend or bulge shapes, as they can be used to push apart adjacent courses (McCann et al. 2016). Support for short rows has been considered in works like (Popescu et al. 2018), (McCann et al. 2016), and (Narayanan et al. 2018). More specifically, in (Popescu et al. 2018), a system is proposed to generate 2D knitting patterns for a given 3D geometry expressed as a parametric surface. Differently than in previous studies, the courses generated by sampling the surface with the specified loop height may include short rows. Like in (Yuksel et al. 2012) and (Igarashi, Igarashi, and Suzuki 2008a), however, the system asks the user to specify the knitting direction and the loop parameters to be used for crafting the object with a given machine. The work in (McCann et al. 2016) presents a knitting design representation that offers the possibility to easily manipulate the structure and the high-level structure of a knitted object by leveraging generalized tubes and sheets that are connected each other at their boundaries with glueing instructions. Short rows can be added to make the assembled parts bend or bulge. A knitting assembly language is used to convert the above high-level representation into low-level operations that can be used by industrial machines to produce the objects. In (Narayanan et al. 2018), a system is illustrated that converts 3D meshes generated by common modeling software into instructions for a computer-controlled knitting machine. Once the user has specified a start and end point on the input geometry, the system incrementally generates a new mesh with a uniform edge length. To this aim, it makes use of short rows. A tracing procedure is then run on the generated mesh to create a knitting path and the corresponding instructions, which are compatible with possible constraints set by the crafting machine.

All the algorithms seen so far rely on a separate row progression that, as already mentioned, has some important drawbacks. The literature proposes an alternative approach based on a helix progression. Although the considered crafting techniques make it possible, in principle, to produce objects with either approach, the helix progression presents a number of benefits. Separate row progression requires adding different-looking stitches used as bridges between the rows or exposing the inner side of the loops to the outside, thus causing aesthetic issues in the crafted object. The helix progression, in turn, ensures a higher uniformity of the surface appearance, since the stitching direction is maintained, there are no jumps between different loops, and each stitch always has its outer side facing the outside of the object. Moreover, especially in the case in which objects are manually crafted, the instructions generated with a helix progression are usually easier to follow as they represent the natural way of manually crafting a 3D object (Wu, Swan, and Yuksel 2019). A possible limitation of helix progression is that, compared to separate row progression, can make the handling of branched shapes more sophisticated.

In the literature, only a few attempts to use the helix progression have been made, precisely in (Wu, Swan, and Yuksel 2019; Çapunaman, Bingöl, and Gürsoy 2017). In (Wu, Swan, and Yuksel 2019), however, the helix structure is generated only at the end of the modeling process, and the algorithm relies on polygon meshes rather than on parametric surfaces. The authors of (Çapunaman, Bingöl, and Gürsoy 2017), in turn, proposed an algorithm for generating a crocheting geometry leveraging the helix progression since from the beginning of the computation process (like the work in (Wu, Swan, and Yuksel 2019), it supports branches, although implementation details are

not described). Moreover, differently than in previous works, the algorithm receives in input not only the target shape, but also additional crafting parameters. These parameters are introduced to consider both *determinate* variables like, e.g., material properties (yarn weight) and tool size (crochet hook), as well as *indeterminate* variables accounting for the crafting process (like, e.g., the tension of the thread, grip of the hand on the yarn, the stylistic characteristics of the crafter, etc.). In this way, the flexibility of the approach is enhanced, supporting both industrial production and manual crafting.

Considering the advantages associated with the approach adopted in (Çapunaman, Bingöl, and Gürsoy 2017), this work was considered as a reference for the development of the framework proposed in the present paper. Differently than in (Çapunaman, Bingöl, and Gürsoy 2017), however, the devised algorithm also integrates the support for short rows (as proposed in (Wu, Swan, and Yuksel 2019)) that, as said, are of paramount importance to control the shape of crocheted objects. The assumption behind the above choice is that their use can improve the similarity between the target shape and the crafted object.

3. Proposed framework

As illustrated in the previous sections, the core of the devised framework is represented by an algorithm that takes in input a target shape and several crafting parameters and generates the corresponding crocheting instructions.

The algorithm was implemented by using the Rust programming language and integrated as a script into the Godot Engine, a MIT-licensed graphics engine, to achieve fast 3D visualization of the resulting geometry.

The algorithm operates with a parametric representation of the target shape and is in charge of computing the actual placement and connection of the stitches. The generated instructions are expressed in a human-readable format; they are leveraged by the framework to produce the 3D geometry and preview it, but can also be used to physically make the object.

Fig. 2 depicts the main steps of the algorithm, which are described in detail in Sections 3.1–3.5. The visualization aspects are discussed in Section 3.6. The major contribution with respect to the reference algorithm concerns the handling of vertices to cope with short rows, though all the other steps were affected by the introduction of this feature.

3.1. *Initialization*

During this step, the target shape provided in input – e.g., the cylinder considered as an example in Fig. 2(a) – is represented by means of a parametric surface, in the following referred to as $S(u, v)$. More specifically, the target shape is divided into loops (i.e., profiles of the control curve computed along the u direction) so that the minimum geodesic distance between each row corresponds to the desired target stitch height. The loops computed for the target shapes can be circles, ellipses, and curves composed of concatenated non-planar segments. Thus, if the target shape has a linear control curve like in the considered example, this operation converts it into a sequence of equidistant circles (Fig. 2(b)). At present, the control curve is described by making use of Bèzier curves; in the future, however, the support for different parametric representations, e.g., based on Non-Uniform Rational B-Splines (NURBS) could be added to increase

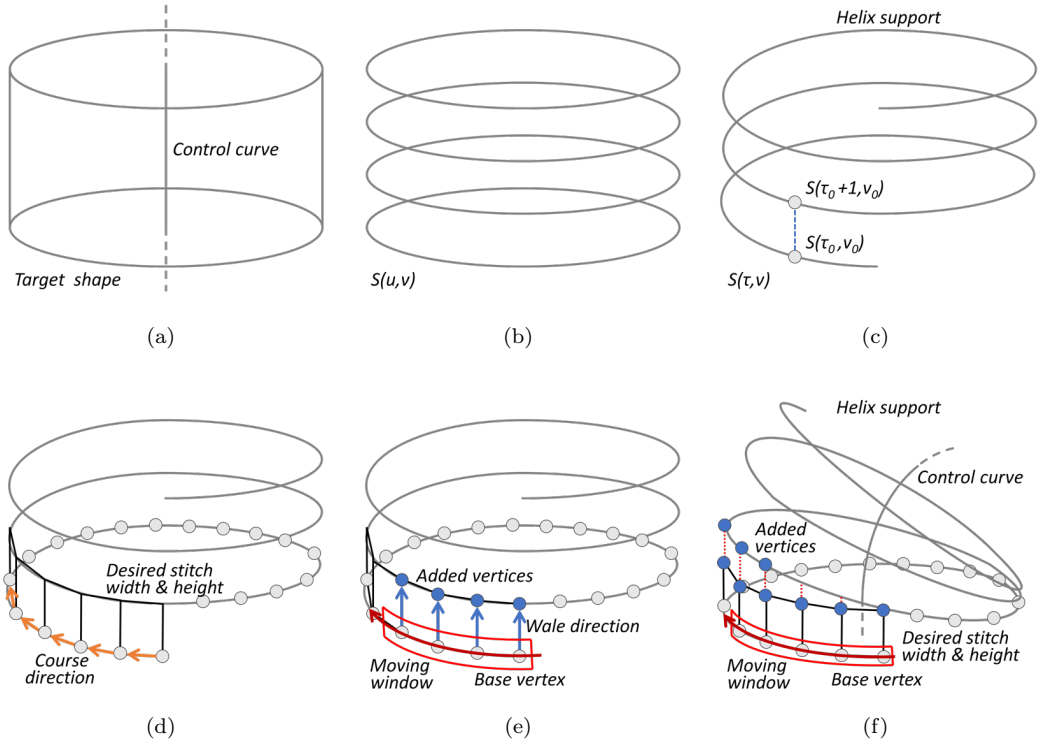


Figure 2. Steps of the devised algorithm and intermediate outputs: (a) cylinder target shape considered as an example for presenting the algorithm, (b) initialization and parametric representation of the target shape, (c) reparametrization of the surface and definition of the helix progression, (d) graph construction and generation of the stitches in the first row, (e) computation of vertex positions and generation of the stitches in the next rows, and (f) conditions requiring the addition of more than one vertex over the base vertices.

flexibility. The target shape can be passed to the framework by providing directly the parametric surface described by using the Rust programming language. Alternatively, if the target shape is expressed as a polygonal mesh, a script running in an external 3D modeling tool (Blender) is provided by the framework to convert it into the required format. More specifically, after modeling or importing a polygonal mesh in Blender, the script automatically computes the parametric surface $S(u, v)$ of the mesh and generates a text file that can be parsed to initialize the data structure used in the Rust program. In the generation of the loops, the algorithm first samples the control curve at desired steps (i.e., the target stitch height) to obtain a sequence of points. Then, cross-sections are computed by estimating the intersection between the input polygonal mesh and the planes that are orthogonal to the direction of the control curve and contain one of the above mentioned points.

In case the control curve includes branches, a methodology to handle the construction of the parametric surface is needed. As said, the work in (Çapunaman, Bingöl, and Gürsoy 2017) did not provide implementation details about this aspect; therefore, the present work proposes an hoc strategy. In particular, the algorithm identifies the branch with the smaller curvature, by comparing the derivatives of the last segment of the Bezier curve with the first segments of all the branches. The selected branch is regarded as the continuation of the main branch in the construction of $S(u, v)$, whereas the remaining branches are considered as independent sub-shapes. At the end of the algorithm, once the stitches have been computed for all the sub-shapes, a merging

process is activated to join them. Details regarding the merging will be provided in Section 3.4.

Besides the target shape, the algorithm receives in input additional parameters. These parameters are included to consider the determinate variables (the physical properties of the thread and the tool size) and indeterminate variables (the effects of the crafter or the crafting process) defined in (Çapunaman, Bingöl, and Gürsoy 2017). Differently than determinate variables, indeterminate variables cannot be quantified or standardized a priori, since they can, e.g., vary from person to person, from time to time, etc. To determine suitable values for these variables, it is possible to consider that they influence the width and height of the stitches. Hence, it is reasonable to compute the dimensions of the stitch to indirectly obtain the above information. To this aim, the approach proposed in (Çapunaman, Bingöl, and Gürsoy 2017) based on a 10-by-10-stitch swatch is adopted. This approach requires to make a physical object which contains 10 (horizontal) $\times$ 10 (vertical) stitches. The width and height of the crafted object are measured to retrieve the size of the single stitch and the distance between the rows. The computed stitch dimensions (in the following referred to as W) are then passed in input to the algorithm.

3.2. *Reparametrization and graph construction*

Once the rows are computed, the system applies a reparametrization $S(u, v) \mapsto S(\tau, v)$ of the surface. Through this operation, the coordinate u is converted to τ, which indicates the number of the row a point on the surface belongs to. Given an arbitrary point $S(\tau_0, v_0)$, $S(\tau_0 + 1, v_0)$ represents the corresponding point in the next row. By adopting this representation, it is possible to easily reconstruct the helix progression of $S(u, v)$. As said, the helix progression was chosen to improve the quality of the results, as it represents the natural way of crafting objects (thus, it eases the job for a manual crafter), and can increase the uniformity of their surface appearance (Wu, Swan, and Yuksel 2019). The helix progression $H(\tau)$ is computed by means of the following equation:

$$H(\tau) = S(\tau, frac(\tau)) \tag{1}$$

where $frac(\tau)$ is the function which returns the fractional part of τ. The result of this step is illustrated in Fig. 2(c).

Once the helix progression is available, the algorithm determines the number of required stitches by dividing the overall length of the helix support by the width of the desired stitch. This operation is performed only for $0 \leq \tau \leq 1$ (i.e., the first row of the helix progression), as shown in Fig. 2(d).

The positions of the bottom vertices of each added stitch (the gray circles in Fig. 2(d)) are recorded into a directed graph $G = (N, A)$, where N is the set of nodes (representing the vertices of the stitches) and A is the set of arcs that describe how nodes (representing the edges of the stitches) are connected. Fig. 3 shows the relationship between vertices/edges of the stitches and nodes/arcs of the graph. In Fig. 2(d), different colors have been used for the arcs, in order to distinguish the *wale* direction (blue), i.e., arcs connecting nodes in the vertical direction, from the *course* direction (orange), i.e., arcs that creates the horizontal connections. The vertices in the first row are connected to each other with course edges.

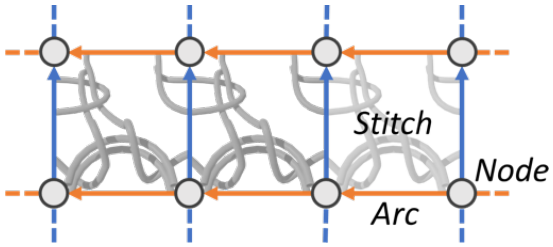


Figure 3. Graph representation used by the algorithm: nodes (gray circles), arcs (blue and orange arrows) and stitches (defined by vertices and edges). Each vertex and edge that composes the stitches of the object being created is represented by a node and arc in the graph, respectively.

3.3. *Computation of vertex positions*

After the initialization of the first row, the algorithm is executed on the remaining of the surface with the goal of calculating the positions of the vertices in the next rows. In order to connect the new vertices, wale type edges and short rows are introduced.

The algorithm is designed to use a moving window approach. The moving window can be regarded as a block that receives in input a sequence of vertices and outputs the vertices of the next row. The vertices currently processed by the moving window on top of which the new vertices are added are called *base vertices* (Fig. 2(e)).

The size of the window is customizable: in the experiments, a window size equal to six (vertices) was used, as it was found to produce the best results.

After the generation of every new vertex, the window is shifted forward by one position, discarding the oldest vertex and adding the next vertex found in the course direction. The moving window is implemented as an array of linked lists, which forms a growable bidimensional matrix. More specifically, each element in the window is identified by two indices. The first index selects the base vertex to which one or more vertices can be connected using the wale type edges. The second index addresses the nodes in the wale direction on top of each base vertex. At each step, more than one vertex can be generated on top of each base vertex, depending on the surface.

The number of needed vertices in the wale direction is computed by dividing the geodesic distance between the vertex itself and the target helix position on top of it. The result is rounded down to an integer odd number in order to maintain a valid structure. In the target shape of the considered example, the height of the stitch already fits the desired height (the cylinder does not contain any radius variation or curvature in the direction of the control curve). For this reason, a single vertex is added on top of each base vertex. For more complex shapes, e.g., containing large variations of the control curve, it may be necessary to add more vertices to reach the next row, as shown in Fig. 2(f).

Once the number of vertices on top of each base vertex has been determined, the algorithm defines the way in which vertices have to be connected to the adjacent vertices by choosing the type of stitches to use. To this aim, the geodesic distance from a given added vertex to the previous one in the course direction is calculated, as shown by the red segments in Fig. 4(a), Fig. 4(c), and Fig. 4(e). Depending on such distance, the course edge connecting the vertices is either maintained (Fig. 4(b)), joined (Fig. 4(d)), or split (Fig. 4(f)), leading respectively to a *single crochet* stitch, an *increase* stitch (needed to widen the fabric), or a *decrease* stitch (needed to narrow it). The vertices supporting the insertion of increase or decrease stitches are added to the graph and connected to the existing structure through course and wale edges.

It may happen that the number of vertices added on top of two consecutive base

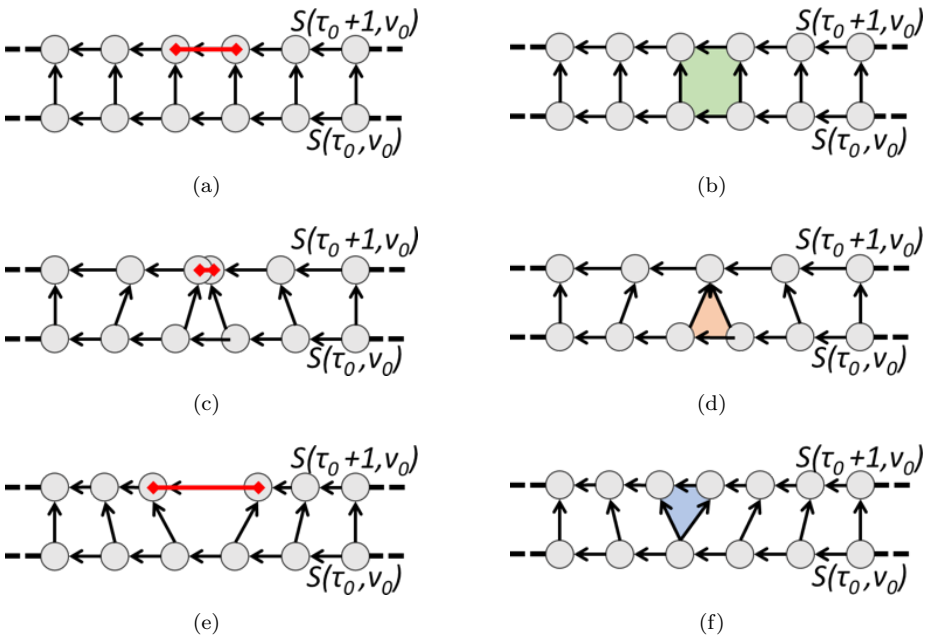


Figure 4. Computing the type of stitch for connecting the added vertices: (a) precondition for a single crochet stitch, (b) single crochet stitch, (c) precondition for an increase stitch, (d) increase stitch, (e) precondition for a decrease stitch, and (f) decrease stitch.

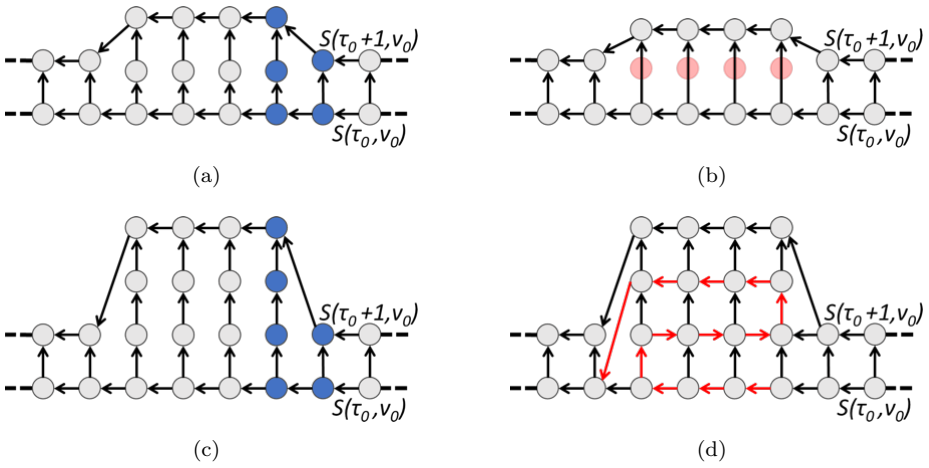
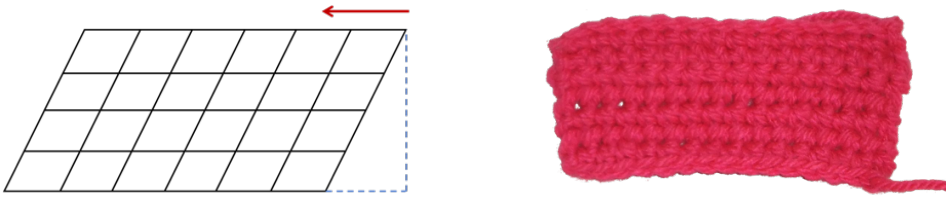


Figure 5. Short rows: (a), (c) scenarios that could require the introduction of a short row to level the different number of vertices on top of two consecutive base vertices, and (c), (d) results of the proposed algorithm.

vertices is different, as exemplified by the blue vertices in Fig. 5(a) and Fig. 5(c). When the difference is an odd number, like in Fig. 5(a), the algorithm forces the introduction/removal of vertices with the aim to obtain differences equal to zero or to an even number: for instance, in Fig. 5(b), the red vertices are removed to obtain a difference equal to zero. In the case of differences equal to an even number like in Fig. 5(c), the algorithm introduces short rows, as shown in Fig. 5(d) (red edges).

The use of short rows plays a crucial role to preserve the similarity between the target shape and the crafted object, as it allows to reduce the skew among stitches. The goal of an algorithm like the proposed one should be to limit the skew as much as possible, since stitches that present large skew values will be stretched in the physical



(a) (b)

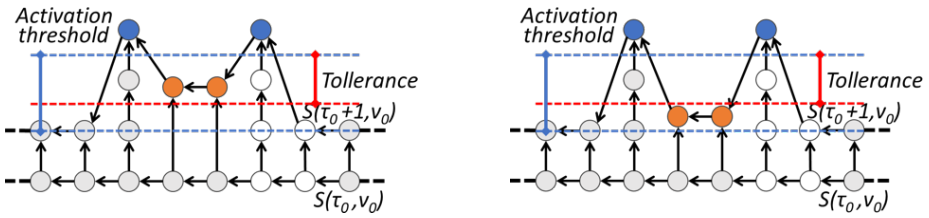
Figure 6. Comparing (a) calculated, and (b) crafted structure in case of large skew.

object (losing their target position) due to the tensile and elastic forces of the thread. As a matter of example, Fig. 6 shows the resulting effect in the crafted object when the stitches present large skew values (the arrow in the figure indicates the direction of the tensile and elastic forces acting on the thread). Comparing Fig. 6(a) and Fig 6(b) it is possible to observe that the intended shape is not preserved in the object due to large skew.

Once short rows have been computed, it is necessary to limit the presence of small and frequent short rows (i.e., numerous short row endings followed by short row beginnings). These conditions can introduce visual stretches that would worsen the visual appearance of the crafted object. In order to avoid small short rows, those containing a small number of elements are discarded, as proposed in (Narayanan et al. 2018). The number of minimum stitches that a short row should contain is a parameter of the algorithm. In the experiments, it was set to four, as this value proved to be capable of providing the best results. The other objective of limiting the presence of short row endings which are followed by short row beginnings is addressed through a mechanism based on a double threshold. An *activation* threshold is used to determine whether a short row should be introduced or not, whereas a *tolerance* threshold is used to control the sensibility of the algorithm. When the activation threshold identifies close vertices which are possible candidates for generating two separate short rows, the algorithm operates as follows: if the height of all the in-between vertices found between the two vertices that satisfied the activation threshold is under the tolerance value, a single short row is created by merging the two consecutive short rows; otherwise, the two short rows are maintained only if their lengths satisfy the first criterion, hence they can be removed if they are too short or maintained if they contain enough vertices. Fig. 7 provides two examples showing how the mechanism based on the double threshold works. The outer vertices in blue trigger the insertion of two very close short rows. In the first case (Fig. 7(a)), all the in-between vertices (orange vertices) respect the tolerance value: thus, the two short rows are merged. Conversely, in the second case (Fig. 7(b)), it is not possible to merge the two short rows since some of the in-between vertices do not respect the tolerance. The two short rows triggered by the blue vertices are removed since they only contain a limited number of vertices that do not satisfy the first criterion.

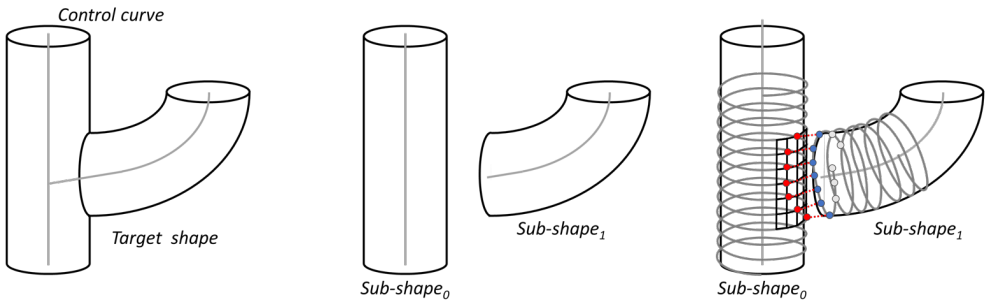
3.4. *Branches support*

As described in Section 3.1, when the control curve contains branches, the original target shape is subdivided into a number of sub-shapes. An example is depicted in Fig. 8(a) and Fig. 8(b). The algorithm considers each sub-shape independently, and the steps seen so far are executed in parallel for all the sub-shapes. Once the posi-



(a) (b)

Figure 7. Activation threshold and tolerance: examples of conditions for (a) generating or (b) discarding a short row.



(a) (b) (c)

Figure 8. Steps of the devised algorithm to handle branches: (a) original target shape, (b) sub-shapes, and (c) interlocking mapping to join the sub-shapes.

tions of the vertices have been computed for all the sub-shapes, they are merged to reassemble the original target shape. To merge the sub-shapes, the algorithm identifies the vertices in the main shape (the red ones in Fig. 8(c)) that are closest to the vertices belonging to the first row of the sub-shape to be merged (the blue vertices in Fig. 8(c)). The identified stitches are marked with a placeholder to indicate that they will have to be used as a bridge for connecting the two shapes. In the instructions generation process, the placeholder is replaced with the corresponding stitch type, and interlocking instructions are added.

3.5. *Instructions generation*

Once the graph has been obtained, it is navigated in order to generate the crocheting instructions. Starting from the first vertices in the first row and following the helix progression, the graph is explored with the aim of recognizing and reconstructing the sequence of stitch types. More specifically, by comparing the lengths of the edges connecting vertices in both the wale and course directions, the algorithm identifies the types of stitches needed (single crochet, increase, or decrease). The output of this step is a text-based file containing the crocheting instructions in the form of a conventional, human-readable description. The file presents the instructions starting from the first stitch in the first row and ending at the last stitch in the last row. The text-based instructions report the number of stitches in a row, as well as the position in which increases, decreases and short rows are needed. Stitches are identified by the conventional acronyms: *sc* (single crochet), *inc* (increase), *dec* (decrease). The short rows are represented by means of the symbols $\rightarrow$ and $\leftarrow$, which indicate the change of

the sewing direction. In order to simplify the readability of the instructions, consecutive stitches of the same type are aggregated. Moreover, instructions are reported per row, to help the artisans who may manually craft the object by providing them with possible reference points (i.e., bookmarks) that can be used after an error to recover the work. Like in (Çapunaman, Bingöl, and Gürsoy 2017), it was chosen to adopt a format that crafters are usually familiar with. Nevertheless, custom scripts could be easily developed to convert the output of the algorithm into machine-readable instructions enabling automated manufacturing.

3.6. *Visualization*

For the purpose of visualization, stitches are represented as tiles (as proposed in (Guo et al. 2020)). In each tile, it is possible to represent the yarn as a series of parametric curves each specified by a number of control points contained in the unit cube. To accommodate the concatenation of adjacent stitches, the control points of the curves are transformed by applying a trilinear interpolation aimed at positioning the tiles in their target positions and scaling them to fit the assigned area. The parametric curves used to define all the stitch types are multiple Bézier curves modeled by means of a graphics suite. The vertices of the unit cube that defines a tile are leveraged as control points for lattice-based deformations. The starting and ending control points of each yarn curve are automatically moved to maintain the derivative direction between contiguous tiles.

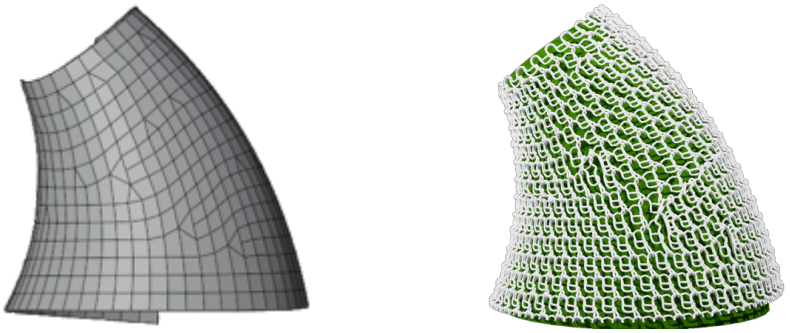
Once the 3D geometry is generated, the algorithm provides two alternative representations, named *wire-frame* and *realistic* mode. Fig. 9 shows the supported visualizations. The former aims at creating a visualization that highlights the structure and the composition of stitches. The latter represents an aesthetically faithful visualization of the final object.

The implementation of the above algorithm into Godot Engine eases possible integration efforts and supports future extensions. For instance, by leveraging the functionalities of Godot Engine, it is possible to export the 3D mesh of the generated objects in a standard `.stl` format for further processing in external graphics suites. A direct integration with software supporting photorealistic rendering could also be easily pursued.

4. Experimental evaluation

As mentioned, the work in (Çapunaman, Bingöl, and Gürsoy 2017) was considered as a reference for the design and development of the proposed algorithm. Both the algorithms receive in input the desired width and height for the stitches, and generate crocheting instructions by working on a parametric surface and leveraging the helix progression. The reference algorithm, however, does not support short rows. Hence, a comparison was first performed with the aim to assess the benefits associated with the introduction of this feature. As said, no implementation details were provided in the reference algorithm about the methodology adopted to handle branches; for this reason, in the evaluation, the strategy proposed in the present paper was used to generate the instructions with both the algorithms.

Furthermore, in the proposed algorithm, short rows were implemented following the approach pursued in (Wu, Swan, and Yuksel 2019). This work focuses on knitting, did not use a parametric representation for the target surface, and did not exploit the helix



(a) (b)

Figure 9. Supported visualization modes: (a) wire-frame, and (b) realistic.

progression. To analyze the impact of these features, the evaluation further considered also this work.

4.1. *Target shapes*

Fig. 10 shows the target shapes selected for the evaluation. The experiments considered two parametric surfaces already used in (Çapunaman, Bingöl, and Gürsoy 2017), in order to make the results comparable with those obtained in the reference work. In the following, they will be referred to as Shape A (Fig. 10(a)) and Shape B (Fig. 10(b)). Shape A is nearly cylindrical, with small asymmetric radius variations, whereas Shape B shows large and asymmetrically increases in the surface radius. Both the surfaces keep the control curve direction constant. These shapes have been selected to confirm the ability of the proposed algorithm to model asymmetric variations of the surface, which was also the goal of the reference work. A custom Blender script was used to convert the surfaces from the Rhinoceros format adopted in the reference algorithm to the format used by the proposed algorithm (i.e, the parametric surface $S(u, v)$ composed of stacked loops along the u direction).

In addition, two more surfaces, i.e., Shape C and Shape D, were considered to highlight the benefits associated with the use of the proposed algorithm. Both the shapes present a variation in the direction of the control curve, which is small in the former (Fig. 10(c)) and more accentuated in the latter (Fig. 10(d)).

Finally, the surface referred to Shape E was taken into account to study the performance of the considered algorithms on branches. This shape was previously used in (Wu, Swan, and Yuksel 2019), and is freely available at https://github.com/textiles-lab/autoknit-tests/tree/master/models.

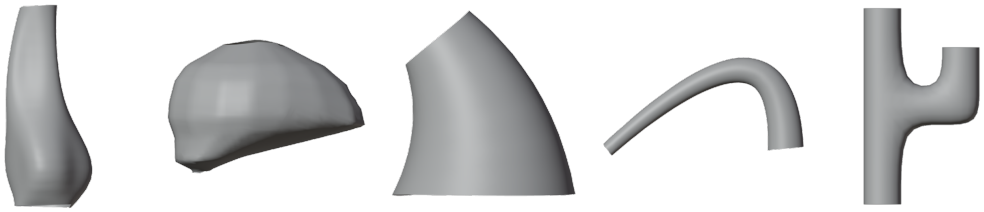
4.2. *Metrics*

The algorithms have been compared in both objective and subjective terms.

The objective evaluation leveraged the metrics reported below.

i) Length error (Narayanan et al. 2018). This metric is computed for the i-th edge (with $i \in [1, E]$ and E equal to the number of edges) as:

$$length\ error_i = \frac{\|PA_i - PB_i\| - l}{l} \tag{2}$$



(a) (b) (c) (d) (e)

Figure 10. Target shapes considered in the experiments.

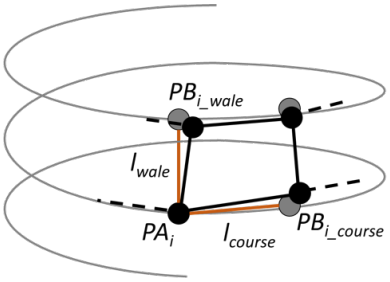


Figure 11. Computing length error for wale and course edges.

where PA_i and PB_i are the two vertices that delimit the edge under analysis, whereas l is the target length (i.e., the desired length of the edge). The metric is used both on the wale and course edges. Fig. 11 shows the position of PA_i and PB_i in the two cases. The metric assumes positive/negative values when the generated edges are longer/shorter than expected.

ii) Skew (EngMorph 2019): The skew for the j-th stitch (with $j \in [1, S]$ and S equal to the number of stitches) is computed as:

$$skew_j = \left|cos(\alpha)\right| \tag{3}$$

where α is the angle between the middle segments shows in Fig. 12. The metric defined in (EngMorph 2019) was adopted since it supports both quadrilateral and triangular faces, i.e, the tiles in the 3D geometry to which a stitches are mapped to.

The metric provides an indication of the skew that could be observed in the object when it is physically produced. Besides being important for the quality assessment of 3D meshes (it is used, e.g., in existing computer graphics software such as (CUBIT 2019)), the metric plays an important role in the analysis of virtual prototypes. In fact, as said, the stitches that present large skew will be stretched due to the tensile and elastic forces of the thread.

iii) Computation time (Narayanan et al. 2018): This metric takes into account the time required by the algorithm to compute the crocheting instructions. Computation time represents a critical performance indicator since, in the considered context, virtual prototyping can be regarded as an iterative process that encompasses changing the parameters and observing the result several times before actually reaching the intended design.

As anticipated, given a target shape, it is possible to adjust the input parameters that define the width and height of the stitches, thus creating different versions of the same shape. Each version differs in the number of vertices, as the smaller the size of

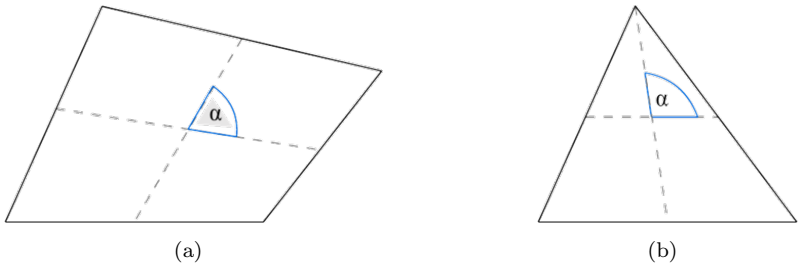


Figure 12. Computing the skew for (a) quadrilateral, and (b) triangular faces in the 3D geometry stitches are mapped to.

the stitches, the more the vertices generated. For this reason, the objective metrics presented above were evaluated considering three different values of W, namely $W_0 = 0.25$cm, $W_1 = 0.20$cm, and $W_2 = 0.15$cm.

The subjective evaluation, in turn, was based on a visual comparison of the physical objects crafted by following the crocheting instructions generated by the algorithm in (Çapunaman, Bingöl, and Gürsoy 2017) and the proposed one; visual comparison with (Wu, Swan, and Yuksel 2019) was not considered, since it adopts a different technique to craft the objects.

4.3. *Experimental setup*

Experiments were conducted by using an AMD Ryzen 9 3900X 12-Core Processor (2.2GHz), 32GB RAM, and a GNU/Linux operating system (Ubuntu 20.04 LTS). The machine was characterized by an x86_64 architecture and presented the following levels of cache: 384KiB (L1), 6MiB (L2), and 64MiB (L3).

As for the reference work, it was chosen to re-implement the algorithm by following the detailed description in (Çapunaman, Bingöl, and Gürsoy 2017). For what it concerns the work in (Wu, Swan, and Yuksel 2019), it was possible to leverage the implementation available at https://github.com/textiles-lab/autoknit.

5. Results and discussion

Fig. 13 shows the results obtained for the different shapes by running the three algorithms, i.e., (Çapunaman, Bingöl, and Gürsoy 2017) (left), proposed (center), and (Wu, Swan, and Yuksel 2019) (right).

5.1. *Length error*

Regarding the length error, data are reported as percentages in Table 1. It can be noticed that the proposed algorithm outperformed both (Çapunaman, Bingöl, and Gürsoy 2017) and (Wu, Swan, and Yuksel 2019), as the average values were lower than the two algorithms for all the considered shapes. The standard deviations suggest that, except for Shape A, the proposed algorithm was also able to generate more frequently stitches with the desired size than the others. Large differences were observed for Shape B, C, D, and E. These results indicate that the three algorithms behave in a similar manner when nearly cylindrical shapes with small asymmetric radius variations are considered (like for Shape A). Conversely, the advantages of the proposed algorithm in

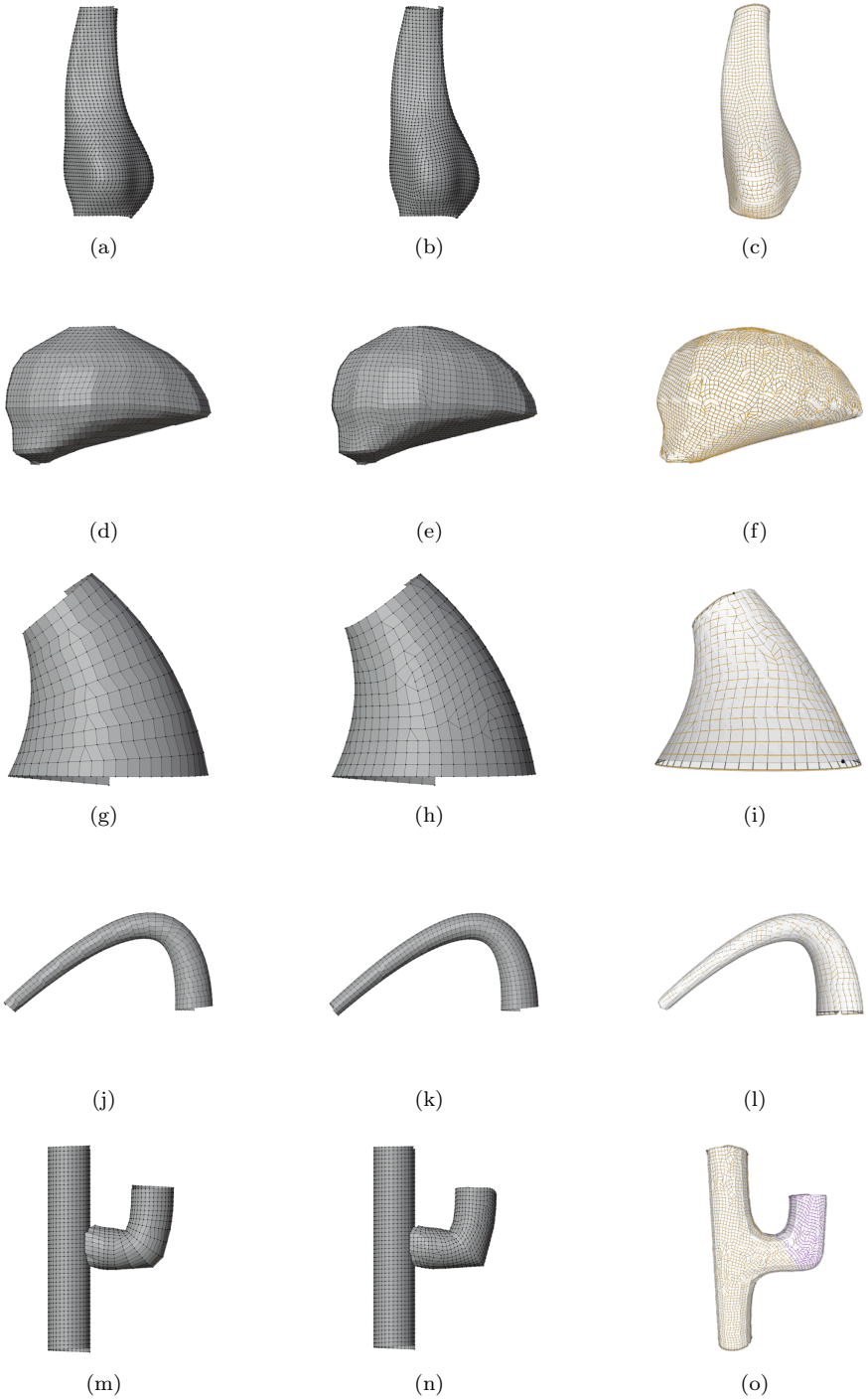


Figure 13. Objects generated by three algorithms: (a), (d), (g), (j), (m) algorithm in (Çapunaman, Bingöl, and Gürsoy 2017) (left), (b), (e), (h), (k), (n) proposed algorithm (center), and (c), (f), (i), (l), (o) algorithm in (Wu, Swan, and Yuksel 2019) (right). The different colors are due to the fact that, for the first two algorithms, a common base implementation in Rust was leveraged; for the last algorithm, the implementation provided by the authors was used.

terms of length error become more evident when the shapes are characterized by large and asymmetrically increases in the surface radius (Shape B), as well as variations in the control curve (Shapes C and D). Differences became again less evident in Shape

Table 1. Objective results for the length error metric (percentages) obtained by the three algorithms for the considered shapes; the lower the better.

Shape	(Çapunaman, Bingöl, and Gürsoy 2017)	Proposed	(Wu, Swan, and Yuksel 2019)
Shape A	M = 4.2%	**M = -1.9%**	M = -4.72%
	SD = 0.069	SD = 0.074	SD = 0.098
Shape B	M = 14.9%	**M = -3.3%**	M = -29.94%
	SD = 0.419	SD = 0.176	SD = 0.263
Shape C	M = 17.6%	**M = -2.9%**	M = -7.93%
	SD = 0.249	SD = 0.101	SD = 0.142
Shape D	M = 8.8%	**M = -1.5%**	M = -37.63%
	SD = 0.193	SD = 0.083	SD = 0.318
Shape E	M = 8.9%	**M = -4.9%**	M = -9.89%
	SD = 0.384	SD = 0.119	SD = 0.240

E, which presents a combination of nearly cylindrical shapes and large variations in the control curve.

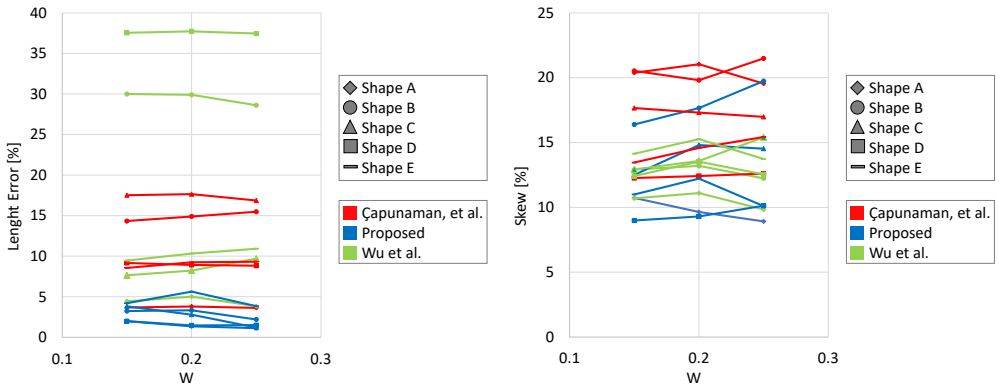
In order to analyze the behaviour of the three algorithms depending on the stitch size, the length error was computed for the three different values of W (i.e., W_{0-2}). For the sake of readability, in Fig. 14(a) absolute values are shown. It worth observing that values remain almost constant over the values of W for all the algorithms. This result suggests that the metric is not influenced by the number of vertices generated by the algorithm.

5.2. *Skew*

Similarly to what was observed for the length error, the proposed algorithm was found to be characterized by lower values of skew than both (Çapunaman, Bingöl, and Gürsoy 2017) and (Wu, Swan, and Yuksel 2019) for Shapes A, D, and E, as reported in Table 2 (like for the length error, percentages are tabulated). For the other two shapes, it is possible to notice that, even though the proposed algorithm outperformed (Çapunaman, Bingöl, and Gürsoy 2017), (Wu, Swan, and Yuksel 2019) was the best algorithm among the three.

Considering values collected for the proposed algorithm and (Çapunaman, Bingöl, and Gürsoy 2017), it can be noticed that marked differences were observed only for Shape A. Comparing these results with the previous ones, a complementary behavior can be observed for these two algorithms. In fact, a smaller difference in terms of error length (observed for Shape A) corresponds to a large difference for what it concerns the skew, and vice-versa. This aspect seems to indicate that a higher fidelity to the desired stitch size (obtained for small values of the length error) is paid with objects that present a larger value of skew. As said, the algorithm in (Wu, Swan, and Yuksel 2019) obtained the best results for Shapes B and C. This finding suggests that, for these kinds of shapes, the use of short rows and separate row progression can reduce the skew.

Regarding the behaviour of the skew metric as a function of W (Fig. 14(b)), it appears that values remain again almost constant for all the algorithms. This outcome indicates that the metric is not influenced by the number of vertices generated by the algorithm.



(a) (b)

Figure 14. Representation of the (a) length error and (b) skew as a function of W.

Table 2. Objective results for the skew metric (percentages) obtained by the three algorithms for the considered shapes.

Shape	(Çapunaman, Bingöl, and Gürsoy 2017)	Proposed	(Wu, Swan, and Yuksel 2019)
Shape A	M = 22.4% SD = 0.160	**M = 9.7%** SD = 0.084	M = 10.90 SD = 0.121
Shape B	M = 19.8% SD = 0.138	M = 17.7% SD = 0.149	**M = 13.06%** SD = 0.142
Shape C	M = 17.3% SD = 0.116	M = 14.9% SD = 0.146	**M = 13.26%** SD = 0.137
Shape D	M = 12.6% SD = 0.115	**M = 10.1%** SD = 0.089	M = 12.97% SD = 0.140
Shape E	M = 14.0% SD = 0.144	**M = 11.6%** SD = 0.171	M = 14.70% SD = 0.151

5.3. *Computation time*

Table 3 shows the behavior of the computation time as a function of W for the considered shapes. The performance of (Çapunaman, Bingöl, and Gürsoy 2017) and the proposed algorithms was almost comparable for a small number of generated vertices ($W = W_2$). However, as the number of generated vertices increases (approaching $W = W_0$), the reference algorithm becomes slower than the proposed one. This is due to the linking mechanism between vertices in adjacent rows, as the reference algorithm checks, for each vertex, which of the vertices in the next row is closest to it. The optimization of this part of the algorithm is mentioned among future works by the authors of (Çapunaman, Bingöl, and Gürsoy 2017). Conversely, the proposed algorithm adopts a moving window with a fixed, finite size, which reduces the number of operations to be executed at each step. As a result, a linear trend can be observed for this metric, which highlights the scalability of the devised approach.

For what it concerns the third algorithm included in the comparison, it can be noticed that it represents the slowest solution among the three. The observed results are in line with data reported in (Wu, Swan, and Yuksel 2019). More specifically, as stated by the authors, the slowest stages of their pipeline regard hand knitting, simulating yarn-level and relaxation, and mesh-based relaxation. For the experiments, the time associated to the first stages was not considered since they are executed only for visualization purposes. Therefore, the time accounted in the evaluation was mainly related to mesh-based relaxation that, as reported in (Wu, Shao, and Liu 2019), could

Table 3. Objective results for the computation time obtained by the three algorithms for the considered shapes.

Shape	Size [cm]	(Çapunaman, Bingöl, and Gürsoy 2017) [ms]	Proposed [ms]	(Wu, Swan, and Yuksel 2019) [s]
Shape A	0.25	117	67	3.13
	0.20	231	106	5.77
	0.15	559	183	15.03
Shape B	0.25	50	20	4.25
	0.20	96	30	9.08
	0.15	241	55	23.83
Shape C	0.25	17	12	4.36
	0.20	39	19	10.43
	0.15	82	38	27.06
Shape D	0.25	33	39	3.39
	0.20	63	64	6.66
	0.15	153	119	18.19
Shape E	0.25	91	58	5.51
	0.20	180	93	12.21
	0.15	437	163	30.05

take several seconds up to about a minute depending on the complexity of the mesh.

5.4. *Visual comparison*

Fig. 15 shows the manually crafted objects obtained by following the crocheting instructions generated by the proposed algorithms and (Çapunaman, Bingöl, and Gürsoy 2017) for the considered shapes. As said, the algorithm in (Wu, Swan, and Yuksel 2019) was not considered for the comparison, as it relies on a different crafting technique and the visual comparison of the stitches would be biased.

All the objects were aligned for the shooting. The inner parts of the objects were filled in with soft material, in order to preserve the volume and highlight the shapes of the crafted objects. Moreover, for Shapes B, D, and E, the objects were hung up using a transparent thread, in order for them not to collapse during the shooting (though making sure not the alter the shape resulting from the crafting).

Considering Shape A (Fig. 15(a)), the visual comparison highlights the presence of some irregularities in the bottom part of the object obtained with the algorithm in (Çapunaman, Bingöl, and Gürsoy 2017) that are missing in the other object. These issues are due to the connection between a part of the object characterized by a small skew with another that presents larger values. Another aspect revealed by the comparison of the two objects is the higher fidelity of the overall curvature in the object obtained with the proposed algorithm. It can easily observed that, as for all the shapes, the starting point for crocheting the two objects is the same (the right bottom corner, in this case). With the reference algorithm, however, the larger skew distorted the crafted object making it assume a different shape.

For what it concerns Shape B (Fig. 15(b)), it is possible to note that the smaller values of the length error metric observed with the proposed algorithm translate in the possibility to better follow the marked asymmetrical changes of the surface. This aspect is particularly evident in the bottom part of the shape, where the proposed algorithm was able to preserve both the steep (on the left side) and smooth (on the right side) increase of the surface radius.

The ability of the proposed algorithm to make the final object maintain the intended curvature and the key role played, in this respect, by the inclusion of short rows, is confirmed by Shape C (Fig. 15(c)). Due to the forces acting in fabric relaxation, the

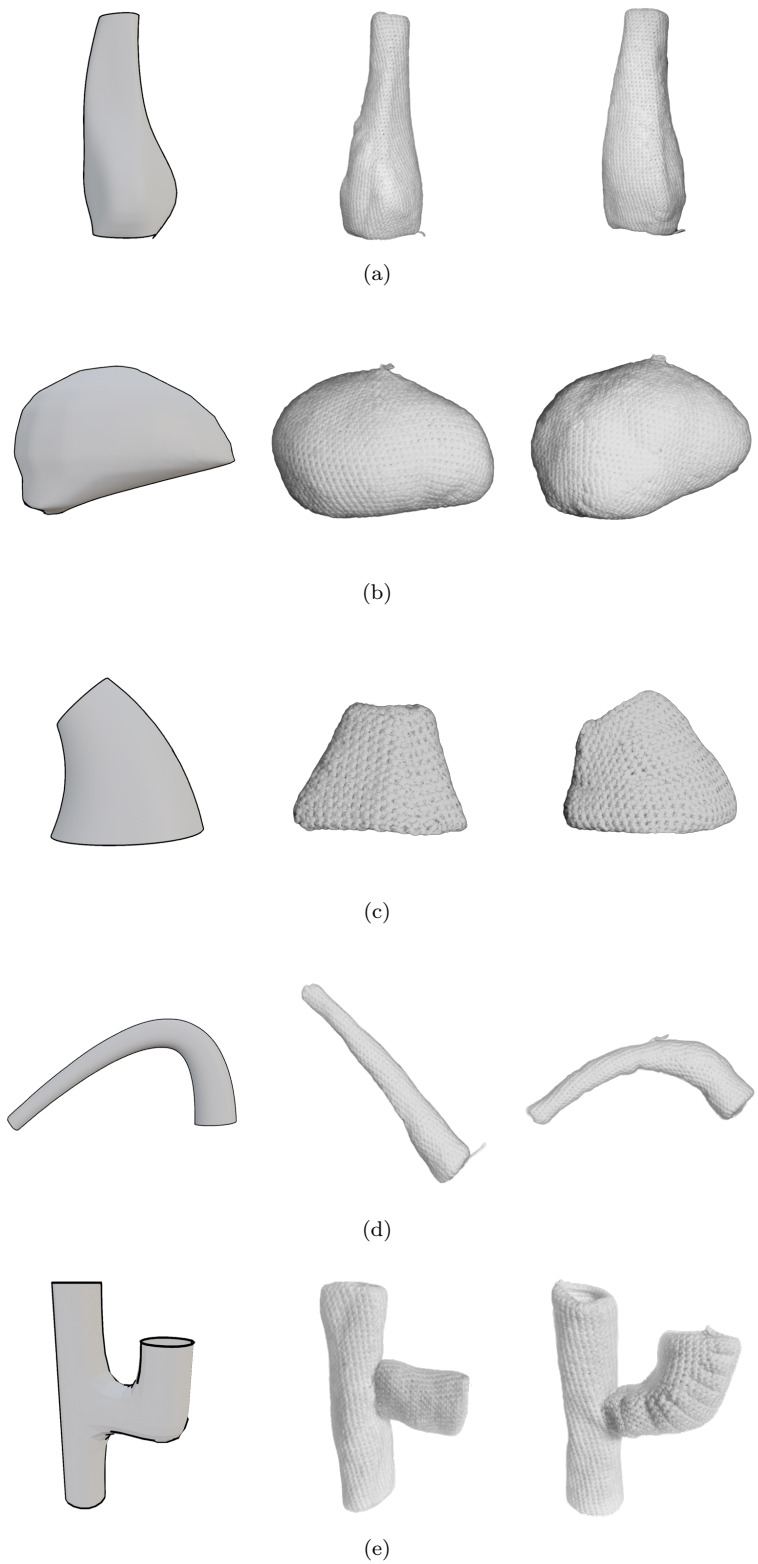


Figure 15. Outlines of the target shapes (left), and crafted objects produced by using the crocheting instructions generated by the reference (center) and proposed algorithm (right).

object obtained with the algorithm in (Çapunaman, Bingöl, and Gürsoy 2017) approximates a truncated cone, whereas the object created with the proposed algorithm follows the target shape more closely thanks to short rows, which allow the shape to bend naturally.

The results obtained on Shape D (Fig. 15(d)) highlight the even higher importance of short rows for shapes with marked curvatures. It can be easily seen that the possibility to use short rows where appropriate allowed the proposed algorithm to follow the original control curve better than the one in (Çapunaman, Bingöl, and Gürsoy 2017), producing an object much more similar to the target shape. With the algorithm in (Çapunaman, Bingöl, and Gürsoy 2017), fabric relaxation forced the structure of the shape to stiffen, as for Shape C.

Moving to Shape E (Fig. 15(e)) and focusing first on the straight part of the object, the visual comparison does not show significant differences between the two algorithms, as the shape is nearly cylindrical and does not present relevant irregularities on its surface. The branch on the right, in turn, shows a marked curvature; thus, outcomes similar to those mentioned for Shapes C and D are obtained. In particular, it is possible to notice the difficulties of the algorithm in (Çapunaman, Bingöl, and Gürsoy 2017) to follow the curvature due to the lack of short rows and the forces acting in fabric relaxation, which do not allow the branch to follow the original shape.

6. Conclusions and future work

The present paper proposes a framework for the virtual prototyping of crocheted objects. The core of the reported work is the design and development of an algorithm that is able to convert a target shape into the corresponding crocheting instructions. The output of the algorithm can be used to physically make the object or to visualize a 3D computer-generated geometrical reconstruction for it. Like other state-of-the-art works, the proposed algorithm is based on the helix progression and considers additional parameters regarding the thread size and the stylistic characteristics of the crafting process. Additionally, the algorithm supports the use of short rows, which are crucial for improving the quality of the final object. Experiments aimed at comparing the proposed algorithm with state-of-the-art techniques in (Çapunaman, Bingöl, and Gürsoy 2017) and (Wu, Swan, and Yuksel 2019)) generally showed its improved performance in terms of error length, skew, computation time, and visual appearance.

These results represent significant advancements in the process of validating the design of crocheted objects before moving to production, and make the proposed framework an effective tool that could support industrial processes from different perspectives. For instance, the improvements in terms of error length, skew, and visual appearance would enable a better analysis in the decision-making process, since the generated virtual prototypes are more similar to the final products. Moreover, the reduced computation time would allow to shorten the above process or to generate a larger number of possible alternatives to be evaluated in the same amount of time. These outcomes would translate into economic advantages, as already observed in different industries (Gomes et al. 2020; de Oliveira Neto and de Sousa 2014; Oliveira Neto et al. 2014). Besides its applicability in the industry for improving the virtual prototyping stages, the framework and the generated instructions could also be leveraged by artisans, as well as by hobbyists, to craft objects manually (e.g., the Amigurumi, small, stuffed, and stylized creatures traditional of Japanese art that can be realized using both the crocheting and knitting techniques.

Despite the promising results, the framework currently presents also some limitations. For instance, a graphics user interface could be integrated in the proposed framework to let the users interact in a more natural way with the generated surfaces. The virtual prototyping functionalities of the framework could be enhanced, e.g., by providing authoring tools to generate the object surfaces, or to apply changes whose effects can be visualized in real time. Another limitation regards the lack of physics simulation, which could prevent the users from visualizing the effects that forces (e.g., gravity) or different design choices (e.g., alternative materials) would have on the structural rigidity of the object. In addition, industrial crocheting machines could introduce some constraints regarding, e.g., the types of stitches supported; at present, it is not possible to instruct the algorithm on these constraints.

Incorporating the above features in the proposed framework could represent interesting directions for future research. Additional experiments could be also performed, e.g., to assess performance based on subjective evaluations, to analyze the usability of the framework, etc. Finally, tools for exporting the generated instructions to formats supported by industrial crocheting machines like, e.g., "Knitout" `.k` (McCann 2017) and "k-code" `.kc`, or to exchange formats such as XML (Zaharieva-Stoyanova and Bozov 2017) could be also integrated in the framework.

Acknowledgement(s)

The authors would like to thank Özgüç Çapunaman, co-author of the reference of work considered in the experimental evaluation, for making available the surfaces used in their research, thus enabling the comparison of achieved results.

Disclosure statement

The authors have no relevant financial or non-financial interests to disclose.

Funding

Research was supported by PON "Ricerca e Innovazione" 2014–2020 – DM 1062/2021 funds.

Notes on contributors

All the authors contributed to the study conception and design. The software was developed by Massimo Gismondi, under the supervision of Fabrizio Lamberti and Alberto Cannavò. The experimental analysis was performed by Massimo Gismondi with the support of Alberto Cannavò, and revised by Fabrizio Lamberti. The first draft of the manuscript was written by Alberto Cannavò, and all the authors worked on its revisions. All the authors read and approved the final manuscript.

References

Baurmann, Gisela, and Daina Taimina. 2013. "Crocheting algorithms." *Cornell J. Archit* 9: 105–112.

Çapunaman, Özgüç Bertuğ, Cemal Koray Bingöl, and Benay Gürsoy. 2017. "Computing stitches and crocheting geometry." In *International Conference on Computer-Aided Architectural Design Futures*, 289–305.

Carvalho Filho, José, Thais Vieira Nunhes, and Otavio Jose Oliveira. 2019. "Guidelines for cleaner production implementation and management in the plastic footwear industry." *Journal of Cleaner Production* 232: 822–838.

Choi, SH, and HH Cheung. 2008. "A versatile virtual prototyping system for rapid product development." *Computers in Industry* 59 (5): 477–488.

CUBIT. 2019. "Metrics for quadrilateral elements." https://bit.ly/38MOs0j. [Online; accessed 11-November-2022].

Cugini, Umberto, Monica Bordegoni, and Rossella Mana. 2008. "The role of virtual prototyping and simulation in the fashion sector." *International Journal on Interactive Design and Manufacturing (IJIDeM)* 2: 33–38.

de Oliveira Neto, Geraldo Cardoso, Paulo Cesar da Silva, Henrricco Nieves Pujol Tucci, and Marlene Amorim. 2021. "Reuse of water and materials as a cleaner production practice in the textile industry contributing to blue economy." *Journal of Cleaner Production* 305: 127075.

de Oliveira Neto, Geraldo Cardoso, and Washington Carvalho de Sousa. 2014. "Economic and environmental advantage evaluation of the reverse logistic implementation in the supermarket retail." In *Advances in Production Management Systems. Innovative and Knowledge-Based Production Management in a Global-Local World: IFIP WG 5.7 International Conference, APMS 2014, Ajaccio, France, September 20-24, 2014, Proceedings, Part II*, 197–204. Springer.

de Oliveira Neto, Geraldo Cardoso, Henrricco Nieves Pujol Tucci, José Manuel Ferreira Correia, Paulo Cesar da Silva, Victor Hugo Carlquist da Silva, and Gilberto Miller Devós Ganga. 2020. "Assessing the implementation of Cleaner Production and company sizes: Survey in textile companies." *Journal of Engineered Fibers and Fabrics* 15: 1558925020915585.

Dong, Longjun, Xiaojie Tong, Xibing Li, Jian Zhou, Shaofeng Wang, and Bing Liu. 2019. "Some developments and new insights of environmental problems and deep mining strategy for cleaner production in mines." *Journal of Cleaner Production* 210: 1562–1578.

EngMorph. 2019. "Skewness Calculation for 2D Elements." https://www.engmorph.com/skewness-finite-elemnt. [Online; accessed 11-November-2022].

Feldmann, K, and F Christoph. 2003. "Virtual prototyping of placement machines in electronics production." *International Journal of Computer Integrated Manufacturing* 16 (7-8): 479–484.

Gomes, Marcos Geraldo, Victor Hugo Carlquist da Silva, Luiz Fernando Rodrigues Pinto, Plinio Centoamore, Salvatore Digiesi, Francesco Facchini, and Geraldo Cardoso de Oliveira Neto. 2020. "Economic, environmental and social gains of the implementation of artificial intelligence at dam operations toward Industry 4.0 principles." *Sustainability* 12 (9): 3604.

Guo, Runbo, Jenny Lin, Vidya Narayanan, and James McCann. 2020. "Representing Crochet with Stitch Meshes." In *Symposium on Computational Fabrication*, SCF '20, 1–8.

Han, Ahyoung, Kwangyun Wohn, and Jaehong Ahn. 2021. "Towards new fashion design education: learning virtual prototyping using E-textiles." *International Journal of Technology and Design Education* 31: 379–400.

He, Zhenglei, Jie Xu, Kim Phuc Tran, Sébastien Thomassey, Xianyi Zeng, and Changhai Yi. 2021. "Modeling of textile manufacturing processes using intelligent techniques: a review." *The International Journal of Advanced Manufacturing Technology* 116 (1-2): 39–67.

Hong, H, R Fangueiro, MD de Araujo, et al. 1994. "The development of 3D shaped knitted fabrics for technical purposes on a flat knitting machine." *Indian Journal of Fibre & Textile Research* 19 (3): 189–194.

Igarashi, Yuki, Takeo Igarashi, and Hiromasa Suzuki. 2008a. "Knitting a 3D model." *Computer Graphics Forum* 27 (7): 1737–1743.

Igarashi, Yuki, Takeo Igarashi, and Hiromasa Suzuki. 2008b. "Knitty: 3D Modeling of Knitted Animals with a Production Assistant Interface." In *Eurographics (Short Papers)*, 17–20.

Jevšnik, Simona, Zoran Stjepanovič, and Andreja Rudolf. 2017. "3D virtual prototyping of garments: Approaches, developments and challenges." *Journal of Fiber Bioengineering and Informatics* 10 (1): 51–63.

Jimeno-Morenilla, Antonio, Philip Azariadis, Rafael Molina-Carmona, Sofia Kyratzi, and Vassilis Moulianitis. 2021. "Technology enablers for the implementation of Industry 4.0 to traditional manufacturing sectors: A review." *Computers in Industry* 125: 103390.

Jin, Bo, and Mang I Vai. 2014. "An adaptive ultrasonic backscattered signal processing technique for instantaneous characteristic frequency detection." *Bio-medical materials and engineering* 24 (6): 2761–2770.

Kaiser, Christian, Simeon Vogt, and Meike Tilebein. 2017. "Virtual development and production framework for textile orthotics." *International Journal of Computer Integrated Manufacturing* 30 (7): 680–689.

Kapllani, Levi, Chelsea Amanatides, Genevieve Dion, and David E Breen. 2022. "Loop Order Analysis of Weft-Knitted Textiles." *Textiles* 2 (2): 275–295.

Lee, JY. 2001. "Shape representation and interoperability for virtual prototyping in a distributed design environment." *The International Journal of Advanced Manufacturing Technology* 17: 425–434.

Leite, Roberto, Marlene Amorim, Mário Rodrigues, and Geraldo Oliveira Neto. 2019. "Overcoming barriers for adopting cleaner production: a case study in Brazilian small metal-mechanic companies." *Sustainability* 11 (17): 4808.

McCann, James. 2017. "The "Knitout" (.k) File Format v0.5.4." https://textiles-lab.github.io/knitout/knitout.html. [Online; accessed 11-November-2022].

McCann, James, Lea Albaugh, Vidya Narayanan, April Grow, Wojciech Matusik, Jennifer Mankoff, and Jessica Hodgins. 2016. "A compiler for 3D machine knitting." *ACM Transactions on Graphics* 35 (4): 1–11.

Narayanan, Vidya, Lea Albaugh, Jessica Hodgins, Stelian Coros, and James Mccann. 2018. "Automatic machine knitting of 3D meshes." *ACM Transactions on Graphics* 37 (3): 1–15.

Nelson, Todd G, Trent K Zimmerman, Spencer P Magleby, Robert J Lang, and Larry L Howell. 2019. "Developable mechanisms on developable surfaces." *Science Robotics* 4 (27).

Oliveira Neto, Geraldo Cardoso de, Maria Tereza Saraiva de Souza, Dirceu da Silva, and Leonardo Aureliano Silva. 2014. "An assessment of the environmental and economic benefits of implementing reverse logistics in the textured glass sector." *Ambiente & Sociedade* 17: 199–220.

Osinga, Hinke M, and Bernd Krauskopf. 2004. "Crocheting the Lorenz manifold." *Mathematical Intelligencer* 26 (4): 25–37.

Papahristou, E, and N Bilalis. 2017. "3D virtual prototyping traces new avenues for fashion design and product development: A qualitative study." *Journal of Textile Science & Engineering* 7 (2): 1–6.

Popescu, Mariana, Matthias Rippmann, Tom Van Mele, and Philippe Block. 2018. "Automated generation of knit patterns for non-developable surfaces." In *Humanizing Digital Reality*, 271–284. Springer.

Rubio, Eva María, A Sanz, and Miguel Angel Sebastián. 2005. "Virtual reality applications for the next-generation manufacturing." *International Journal of Computer Integrated Manufacturing* 18 (7): 601–609.

Sayem, ASM. 2020. "Virtual prototyping for fashion 4.0." In *Industry 4.0–Shaping The Future of The Digital World*, 193–196. CRC Press.

Sperl, Georg, Rosa M Sánchez-Banderas, Manwen Li, Chris Wojtan, and Miguel A Otaduy. 2022. "Estimation of yarn-level simulation models for production fabrics." *ACM Transactions on Graphics* 41 (4): 1–15.

Thomsen, Mette Ramsgard, and Toni Hicks. 2008. "To Knit a Wall, knit as matrix for compos-

ite materials for architecture." In *Ambience 08 International Scientific Conference – Smart Textiles – Technology and Design*, 107–114.

Viganò, Giampaolo, Stefano Mottura, Luca Greci, Marco Sacco, and Claudio Roberto Boër. 2004. "Virtual reality as a support tool in the shoe life cycle." *International Journal of Computer Integrated Manufacturing* 17 (7): 653–660.

Vitali, Andrea, Lorenzo D'Amico, and Caterina Rizzi. 2016. "Virtual Tailor for Garment Design." In *Virtual, Augmented and Mixed Reality: 8th International Conference, VAMR 2016, Held as Part of HCI International 2016, Toronto, Canada, July 17-22, 2016. Proceedings 8*, 653–661. Springer.

Wu, Kui, Hannah Swan, and Cem Yuksel. 2019. "Knittable stitch meshes." *ACM Transactions on Graphics* 38 (1): 1–13.

Wu, Weiwei, Xiaodong Shao, and Huanling Liu. 2019. "Automatic visibility evaluation method for application in virtual prototyping environment." *International Journal of Computer Integrated Manufacturing* 32 (10): 960–978.

Yuksel, Cem, Jonathan M Kaldor, Doug L James, and Steve Marschner. 2012. "Stitch meshes for modeling knitted clothing with yarn-level detail." *ACM Transactions on Graphics* 31 (4): 1–12.

Zaharieva-Stoyanova, Elena, and Stefan Bozov. 2017. "Application of XML-based language for digital representation of crochet symbols." In *Digital Presentation and Preservation of Cultural and Scientific Heritage*, Vol. 7, 181–190.

Zhao, Ke, Junchen Hu, Haidong Shao, and Jiabei Hu. 2023. "Federated multi-source domain adversarial adaptation framework for machinery fault diagnosis with data privacy." *Reliability Engineering & System Safety* 236: 109246.

Zhao, Ke, Feng Jia, and Haidong Shao. 2023. "A novel conditional weighting transfer Wasserstein auto-encoder for rolling bearing fault diagnosis with multi-source domains." *Knowledge-Based Systems* 262: 110203.

Zheng, Qinghe, Penghui Zhao, Yang Li, Hongjun Wang, and Yang Yang. 2021. "Spectrum interference-based two-level data augmentation method in deep learning for automatic modulation classification." *Neural Computing and Applications* 33 (13): 7723–7745.

Zheng, Qinghe, Penghui Zhao, Hongjun Wang, Abdussalam Elhanashi, and Sergio Saponara. 2022. "Fine-grained modulation classification using multi-scale radio transformer with dual-channel representation." *IEEE Communications Letters* 26 (6): 1298–1302.