JPMTR-2106
DOI 10.14622/JPMTR-2106
UDC 655.1-023.5:681.6|62-11

Original scientific paper | 150
Received: 2021-05-27
Accepted: 2021-09-10

Improving 3D printing methods to create versatile sleeves for the printing industry

Xueying Wei, Peter Urban and Sven Ritzmann

University of Wuppertal,
School of Electrical, Information and Media Engineering,
Rainer-Gruenter-Straße 21, D.42119 Wuppertal

xueying.wei@ovgu.de
purban@uni-wuppertal.de
ritzmann@uni-wuppertal.de

Abstract

In this work we show a method to produce sleeves for the printing industry by utilizing a modified 3D printer. The modification enables the printing process to take place on a rotating cylindrical surface. Due to this modification the corresponding G-code is generated by a MATLAB program. The advantage of this G-code based workflow, as we refer, is the inclusion of the sleeve dimensioning process, and the independence from generalized slicer algorithms providing the freedom to design every layer in respect to mechanical requirements. The motivation behind combining the ability to print on a cylindrical printing surface with the G-code based workflow is to increase the stability of 3D-printed sleeves in relation to their usage in the printing industry. The sleeves themselves consist of required elements, such as top layer and bottom layer, infill layer and air ducts. Our method allows load-appropriate extrusion orientation and therefore gives an additional choice to the common 3D printing workflow adopting Cartesian coordinates.

Keywords: additive manufacturing, G-code based workflow, rotating cylindrical surface, printing industry sleeves, cylindrical coordinate 3D printing

1. Introduction

In this paper we show a new method to produce sleeves for the printing industry. Sleeves in general are pipes of different wall thickness and often consist of layers of different materials in the radial direction. By this construct, they can fulfill requirements for rigidity, price, weight, precision, cut resistance, stickiness, viscoelasticity and so on. Sleeves are used very often in flexographic printing machines as plate cylinders or carriers for the plate cylinder. They are mounted on so-called air mandrels, lubricated by an air film blown out through small orifices on the cylinder's face. First, the sleeve is expanded in diameter by the air pressure and therefore can glide on the air mandrel. After switching off the air, the sleeve shrinks onto the air mandrel. In general, under the conditions of printing the sleeve then stays in its place. Sleeves have a number of advantages which have made them a common part of flexographic printing machines through the past 30 years. They are also used as so-called adapter sleeves to cover a larger diameter span with one air mandrel diameter. In such cases they function as an intermediate part between air mandrel and printing sleeve and often carry additional air guiding pipes internally to allow the mounting of the external printing sleeves.

Most commonly, the sleeves consist of different materials distributed radially, such as fiber-enforced epoxy resin, polyurethane materials with different densities, viscoelastic materials and so on. We want to present a different method for sleeve construction based on 3D printing with thermoplastic materials. The idea behind this is that different densities of any sleeve material can be reached easily by the control of the extruded path. A large number of 3D printers are available today; some of them are very well documented/open source and need only small changes for our purpose. These printers in general create their output in Cartesian coordinates layer by layer or slice by slice in the *z*-direction. There is a general workflow from the 3D model (given as STL- or OBJ-file) to the extruded path description (in G-code), which allows some influence but, for example, guiding the extruded path in the direction of expected mechanical stress is not possible within this limited workflow.

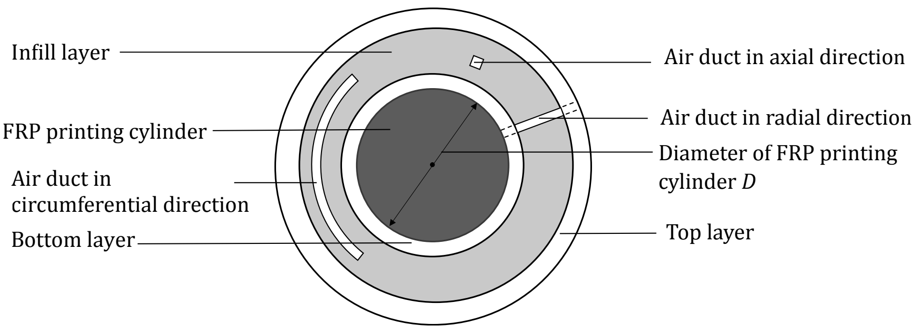

Infill layer
Air duct in axial direction
Air duct in radial direction
FRP printing cylinder
Diameter of FRP printing cylinder *D*
Air duct in circumferential direction
Top layer
Bottom layer

Figure 1: The structure and the basic layers of the 3D printed sleeves

Our approach is based on a cylindrical coordinate system, a cylindrical printing surface and a self-written software. The software generates G-code which is specialized to 3D-printed sleeves on a structurally changed commonly available type of Fused Filament Modeling (FFM) printer. The internal software of the printer, the so-called firmware, is not changed. To generate the necessary G-code we use the computer algebra system MATLAB (MathWorks, 2021), which allows for writing and storing of programs as M-files. The G-code is generated based on the sleeve dimensions defined by the user. Figure 1 shows the structure and the basic layers of each sleeve, namely the bottom layer, the infill layer, and the top layer. Each of these layers is built up from several printing layers. The printing plate itself is not a part of our 3D printing process as we concentrate on the production of adapter sleeves.

2. Previous work

At a past drupa, the company Leapfrog demonstrated the Xcel Leapfrog (released in 2016) for the production of sleeves by 3D printing (Leapfrog, 2020; 2021). The sleeves exhibited at the drupa consisted of a cylindrical outer shell and a cylindrical inner shell both connected by struts in an axial direction. Their printed sleeves seemed to meet requirements such as light weight and easy customization in diameters and lengths. However, these sleeves are produced by using a general 3D printing workflow, that is to say, the sleeves are printed in the axial direction layer by layer using a Cartesian coordinate system. It could be argued that printing layers in an axial direction of the sleeve could lead to a lack of stability when bending the sleeve.

Elsey (2011) describes a method to print cylindrical objects on a rotating cylindrical surface. A printer was proposed that uses a cylinder as a printing bed and an applicator to place the object forming material layer by layer in a circumferential direction. Smid (2008) gives an introduction to G-code for the control of NC-machines. Zheng, et al. (2019) shows a method to create G-code from programming.

3. Methods

The 3D printer taken for this work is a Geeetech Prusa I3A Pro (Shenzhen Getech Technology, 2016). Originally built from a kit, it was left from another project. The Prusa construction is a kind of portal robot, linked with the person of Josef Prusa, who brought it into the world of 3D-enthusiasts and thus became the identifier for this type of printer. The Shenzhen Getech Technology (2021) is a Chinese company that offers this type of printer worldwide. The printer and its components are controlled by a microcontroller. Its firmware interprets and executes the Marlin subset of G-code commands (Lahteine, et al., 2020). We exclusively use polylactid acid (PLA) filament for our prints and have not done any further experiments related to the material itself. This is due to the fact that the focus of our work is the printing device and the implementation of the G-code based workflow. Still the usage or combination of different available materials keeps the possibility open to enhance specific properties of a sleeve.

The modification of the 3D printer is described in section 4.2. As mentioned before, we used the software MATLAB to write our G-code generator. To visualize and validate the generated G-code we used Repetier-Host (Hot-World, 2021). Repetier-Host is a slicer software and is usually used to generate and analyze G-code for 3D printing systems with a Cartesian coordinate system.

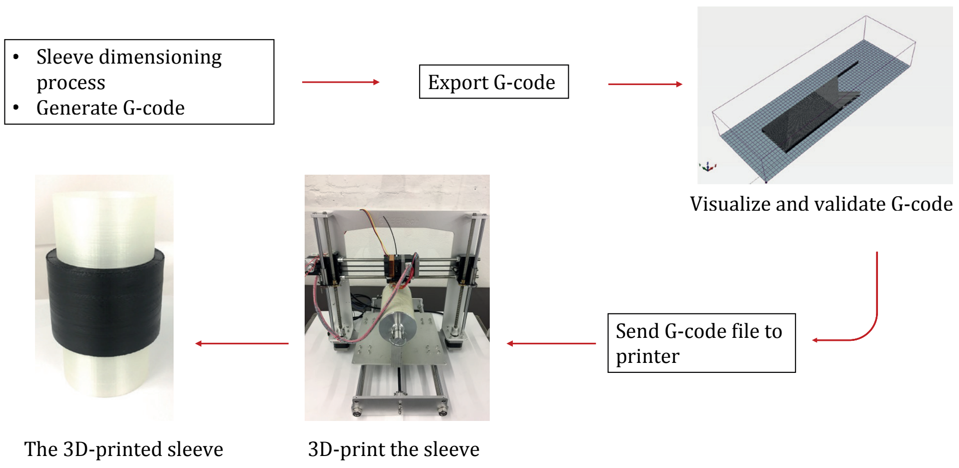
4. Developing of cylindrical printer and sleeves

The G-code-based workflow from our approach simplifies the computer aided sleeve dimensioning process to a limited number of user-defined parameters which are directly processed into well-optimized G-code (detailed in 4.1). As a first step, we modified a Geeetech Prusa I3A Pro printer to meet the needs of cylindrical coordinates (4.2). Then we wrote the mentioned software by defining crucial parameters to describe a single sleeve and by using selected G-code commands (4.3). Furthermore, we applied a commonly used model for calculating the filament length to be fed into the extruder for printing an extruded path of a given length (4.4). We used the same model to ensure an even and closed surface by overlapping every extruded path with its adjacent paths (4.4). After that we finally gained all tools to write our G-code generator. Its core processes are the implementation of sleeves in a cylindrical coordinate system (4.5) as well as the implementation of the different basic layers of sleeves and the needed air ducts (4.6).

4.1 The G-code based workflow and the advantages of cylindrical coordinates

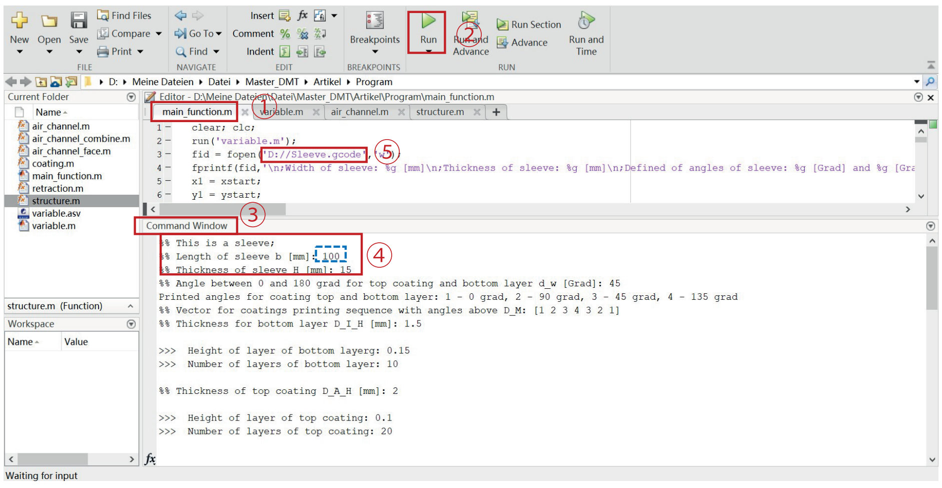
The core element of our approach is the G-code based workflow as shown in Figure 2. The conventional workflow strictly separates the creation of a 3D object from the generalized G-code generation. The G-code based workflow reduces the dimensioning process to the key parameters of the object, in our case the sleeve. As a result, it concentrates on an application-optimized and therefore specialized G-code generation for an also application-optimized 3D printer.

Therefore, the first step of the G-code based workflow is the sleeve dimensioning process. It starts with opening the main_function.m script in MATLAB (① in Figure 3) and its execution (② in Figure 3). In the command window (③ in Figure 3) the user will see prompts (④ in Figure 3) that guide one through the whole dimensioning process. The prompts actually ask the user to define each parameter for a sleeve, such as length of the sleeve, thickness of top layer, angles of the extruded bands for each printing layer and so on. A full list of the user-definable parameters can be seen in Table 1. By varying these parameters, the user can customize the sleeve to his or her needs. The user defined parameters will be sent back to the program and a G-code file will be created after the last prompt has been answered. The save path of the G-code file can be changed directly in our MATLAB script (⑤ in Figure 3). In this example



• Sleeve dimensioning process
• Generate G-code
Export G-code
Visualize and validate G-code
Send G-code file to printer
3D-print the sleeve
The 3D-printed sleeve

Figure 2: The G-code based workflow



```
clear; clc;
run('variable.m');
fid = fopen('D://Sleeve.gcode','w');
fprintf(fid,'\n;Width of sleeve: %g [mm]\n;Thickness of sleeve: %g [mm]\n;Defined of angles of sleeve: %g [Grad] and %g [Gra
x1 = xstart;
y1 = ystart;
```

```
%% This is a sleeve;
%% Length of sleeve b [mm]: 100
%% Thickness of sleeve H [mm]: 15
%% Angle between 0 and 180 grad for top coating and bottom layer d_w [Grad]: 45
Printed angles for coating top and bottom layer: 1 - 0 grad, 2 - 90 grad, 3 - 45 grad, 4 - 135 grad
%% Vector for coatings printing sequence with angles above D_M: [1 2 3 4 3 2 1]
%% Thickness for bottom layer D_I_H [mm]: 1.5

>>>  Height of layer of bottom layerg: 0.15
>>>  Number of layers of bottom layer: 10

%% Thickness of top coating D_A_H [mm]: 2

>>>  Height of layer of top coating: 0.1
>>>  Number of layers of top coating: 20
```

Figure 3: The GUI of MATLAB and an example of a user input (dotted rectangle) as reaction on a prompt from the program on the command window

the G-code file is named “Sleeve.gcode” and is saved to the hard disk drive D. The software is very simple in its structure and is mainly designed to work in laboratory application.

Table 1: Sleeve parameters and their meanings

Parameter	Definition
eb	Width of infill layer [mm]
α	Angle of infill layer [°]
L	Length of sleeve [mm]
H	Thickness of sleeve [mm]
D	Diameter of printing cylinder [mm]
O	Overlap distance [mm]
β	Angle of printed top and bottom layers [°]
D_A_H	Thickness of top layer [mm]
D_I_H	Thickness of bottom layer [mm]

As a next step, we use Repetier-Host to visualize and validate the G-code file. It appears as a flat printed model because the software environment to process our G-code is left untouched. The only exception is the flat bed size in the *x*-direction, which must have the ability to take up several revolutions of the cylindrical printing surface. On a flatbed printer without our modification the printout would appear as shown in Figure 2. Herein it simply saved us from writing our own visualization software. The code can then be sent either directly from the PC to the 3D printer managed by Repetier-Host or by inserting an SD card with the G-code in the 3D printer. Finally, the 3D printer interprets the G-Code and prints the sleeve. As Cartesian coordinate 3D printers print any object from bottom to top, layer by layer they are not well suited to print a cylindrical object with its center axis parallel to the *x*-*y*-plane. Support material necessary for overhanging parts has to be printed and then later removed again. The G-code-based workflow could create these kinds of objects in combination with a necessary mechanical change of the printer. In addition to this, it gives us full control over the extrusion path in terms of orientation, temperature, speed, and so on, which cannot be achieved by a standard slicing workflow.

A sleeve manufactured on a Cartesian coordinate system printer will have different mechanical properties depending on slicing parameters and special arrangement of the sleeve. As a result, it may become weak in certain load situations or even change mechanical properties like stiffness with the angle of rotation. If the layers of a sleeve could be printed on a cylindrical surface, each layer would correspond to the circumference for a specific radius. Furthermore, if each layer could be printed following any direction between circumferential to the axial direction, this would allow alternating spiral helical applied layers. As a result, the only major weakness may be the connection from one layer to the next. This we consider to be a minor problem as the stress in the printing machine would be mainly compressive rather than tensile.

4.2 Modifying a Cartesian coordinate system 3D printer to operate with a cylindrical coordinate system

In the following, we describe how to implement sleeves in a cylindrical coordinate system and how we modified a Cartesian coordinate system 3D printer to operate with a cylindrical coordinate system.

In the Cartesian coordinate system (Figure 4a) the extruder moves on smooth rods horizontally and is driven by a toothed belt to realize the *x*-axis. The printing bed moves perpendicular to the *x*-axis horizontally along the *y*-axis. The two vertical screw rods control the *z*-axis. Contrary to Cartesian coordinates, cylindrical coordinates consist of a radial (*r*), an azimuthal (*θ*) and a linear (*l*) component.

To implement these components our modifications (Figure 4b) leaves the *y*-axis and the *z*-axis mechanically unchanged from the Cartesian coordinate system. The difference is that the former *z*-axis now controls the *r*-component of the cylindrical coordinate system, while the former *y*-axis controls the *l*-component. To implement the *θ*-component, the printing bed is now designed as a rotating fiber-reinforced plastic (FRP)

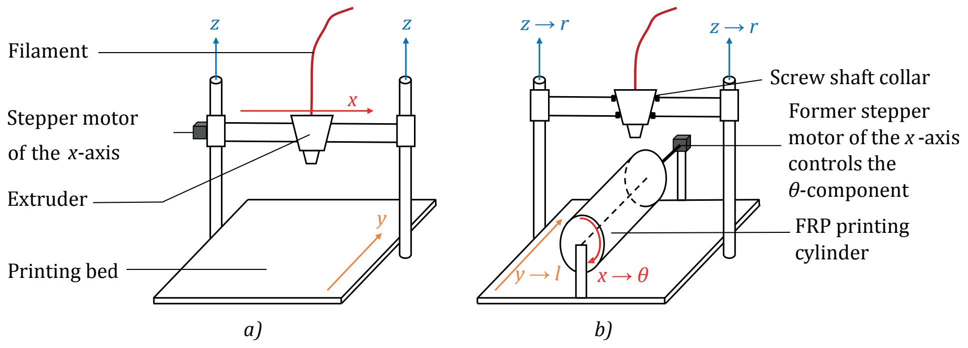


Figure 4: (a) Cartesian coordinate system before, and (b) cylindrical coordinate system after modification of the 3D prnter

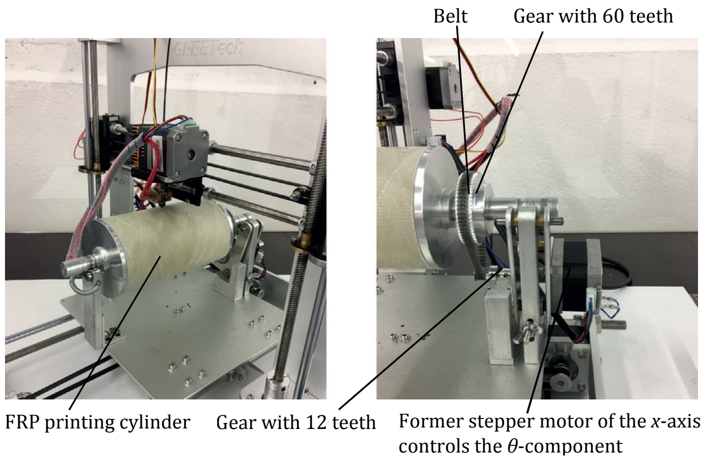


Figure 5: Modified Geeetech Prusa I3A Pro with mounted FRP printing cylinder and 1 : 5 gearbox

printing cylinder carried by the former printing platform. As the FRP printing cylinder is driven by the former x-axis motor, the former x-axis controls the θ-component. The extruder is kept fixed above the top center line of the FRP printing cylinder.

In contrast to the formerly limited build area, the cylindrical printing surface is virtually infinite in circumferential direction (formerly: x-direction). By that, seamless objects and also free choice of angle for extruded paths are possible. Practically it can also save time because for a repositioning of the extruder in circumferential direction the maximum distance is always less than, or maximally, one-half circumference.

In practice we modified a Geeetech Prusa I3A Pro as shown in Figure 5. The printing bed was replaced by an FRP printing cylinder mounting system. This system can be mounted with a FRP sleeve with a diameter of 79 mm, which complies with the standard basic FRP-sleeves used in the industry. In our case they serve as a changeable printing cylinder.

A stepper motor (200 steps/rev., 32 microsteps each) drives the FRP printing cylinder via a gearbox. The gear ratio of 5 ensures a circumferential resolution of 0.00776 mm/microstep (Equation [1]) which is approximately equivalent to the given axial resolution. Furthermore, the gearbox was built in to keep accuracy with increasing diameters and to avoid problems from the larger inertia of the FRP printing cylinder or additional mechanical system.

$$Res_{\mathrm{cf}} = \frac{d_{\mathrm{pc}} \times \pi}{st_{\mathrm{res}} \times mst \times gr} \qquad [1]$$

where Res_{cf} indicates a circumferential resolution [mm/microstep], d_{pc} is a diameter of an FRP printing cylinder [mm], st_{res} is a stepper resolution [steps/revolution], mst denotes a microstepping [microsteps/step], and gr indicates a gear ratio.

4.3 G-code and relevant commands

G-code is a CNC programming language used for many different types of NC machines. The command set depends on the type of machine and the implemented hardware. The unmodified Geeetech Prusa I3A Pro uses the so-called Marlin firmware. A description of the included commands is documented on the Marlin website (Lahteine, et al., 2020). A general G-code command consists of a letter which stands for a group of commands (e.g., G for standard commands like movement or T for tool related commands) most often followed by a number which specifies the command. Table 2 shows basic G-code commands most commonly used to control Cartesian coordinate system printers.

Table 2: Basic commands with definitions of commands with variables of G-code

Command	Definition
G1	Extruder goes straight
F	Printing speed F [mm/min]
X	Coordinate X in x-axis [mm] (Cartesian coordinates)
Y	Coordinate Y in y-axis [mm] (Cartesian coordinates)
Z	Coordinate Z in z-axis [mm] (Cartesian coordinates)
E	Extruded length E [mm]

On a flat printing bed, the G-code text in Table 3 appears as parallel lines (Figure 6) with layer height h = 0.2 mm. The extruder locates at coordinate (0, 50) first, then prints to coordinate (248, 50) with a speed of F = 800 mm/min while feeding E = 7.4 mm filament into the extruder. Next, the extruder travels quickly to a new position (0, 150) with a speed of F = 1500 mm/min. Finally, a parallel line from coordinate (0, 150) to coordinate (248, 150) is printed with identical parameters to those of the first line.

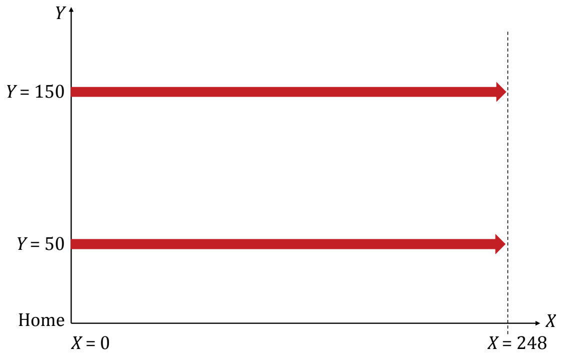


Figure 6: The printed paths from G-code presented in Table 3

Table 3: Example of G-code

G1 Z0.2	; Positioning
G1 X0 Y50 F800	; Positioning
G1 X248 Y50 E7.4 F800	; Printing
G1 X0 Y150 F1500	; Positioning
G1 X248 Y150 E14.8 F800	; Printing

As mentioned before, the firmware of our 3D printer is unchanged. As a consequence, we have to use these Cartesian coordinate system-based G-code commands to control our cylindrical coordinate system-based 3D printer. Therefore, we assume the printing surface of the FRP cylinder to be a plane with a finite height (l-component) and an infinite width (θ-component). So, we are still using Cartesian coordinates for calculating the G-code commands for each printing layer.

As the circumference raises with each printing layer (given by the r-component), we calculate each printing layer based on the current circumference. If one printing layer layout remains unchanged, we iteratively scale the results and the parameters up to the next printing layer (see section 4.5).

4.4 Calculating the extruded length and the overlap between two extruded paths

Figure 7 shows names and variables of the extruder and the extruded band. The extruded length E depends on the diameter of the filament d_f and the length l, width b and thickness h of the extruded path.

Assuming a constant material density before and after extrusion we calculate the E value by use of Equation [2] as part of the model described by Hodgson, Ranellucci and Moe (2019), which we explain as follows:

$$E = \frac{V_b}{\pi(d_f/2)^2} \qquad [2]$$

where E denotes the extruded length [mm], V_b is a volume of extruded band [mm^3], and d_f is a diameter of filament [mm].

The width b of a single extruded band can vary from values near the nozzle diameter up to wider values. The maximum width depends on the quality in terms of smooth and even surface of the band. Figure 8 shows different widths of an extruded band and a structure formed by several bands.

The cross section of the extruded path is not rectangular but shaped as shown in Figure 9a. It appears like a rectangle in the center with two semicircles on the sides. The volume V_P of the extruded path is calculated as shown in Equation [3]:

$$V_p = l[\pi(h/2)^2 + h(b-h)] \qquad [3]$$

with h, b and l as shown in Figures 7, 8, and 9.

If two extruded paths are printed side by side without overlapping, there will be a gap between them shown as area ① in Figure 9b. To solve this towards a more

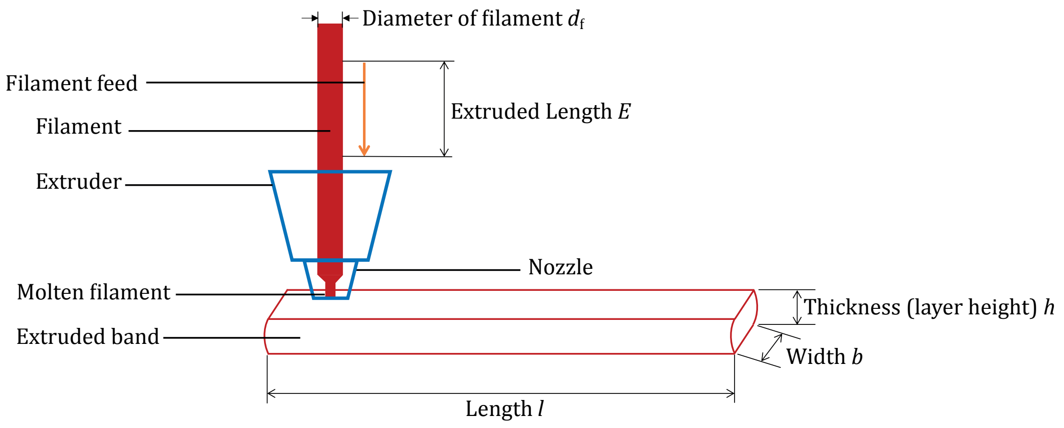


Figure 7: Extruder and extruded band

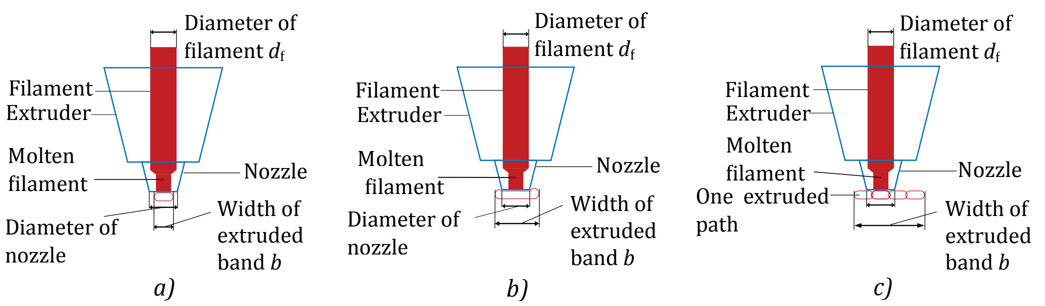


Figure 8: (a) A band with a width near to the extruder's nozzle diameter, (b) an over-squeezed band, (c) a combination of several paths printed in parallel to form a wider structure

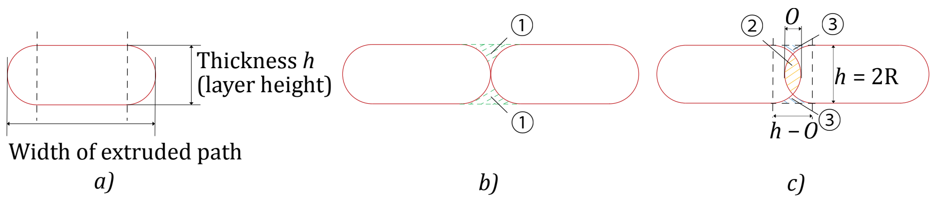


Figure 9: (a) Cross section of one extruded path, (b) two extruded paths without overlap, (c) two extruded paths with accurate overlap to get a more even and closed surface

even surface there should be some overlap (area ② in Figure 9c), so that the overlap area equals the gap area (area ③ in Figure 9c). The overlap distance O is calculated by Equation [4]. The value depends only on the layer of extruded filament thickness (height of printing layer) h [mm] (Hodgson, Ranellucci and Moe, 2019).

$$O = h - \frac{\pi(h/2)^2}{h} \qquad [4]$$

4.5 Dynamic parameters depending on layer

For each extruded band in the first printing layer the extruded length is calculated as described before (4.4). Because of the cylindrical coordinates some parameters depend on the current printing layer. By changing the r-component from one layer to the next the diameter and perimeter of the rotating surface also changes. We have to consider the extruded length and printing speed at identical angular positions in different printing layers.

Figure 10 illustrates the increase in length for each new printing layer. Therefore, we iteratively calculate the extruded length (extruded material) for the current printing layer based on the previous layer as shown in Equation [5].

$$E_{n+1} = E_n \frac{D + 2h(n+1)}{D + 2hn} \qquad [5]$$

where E_n denotes the extruded length of the nth printing layer [mm], and E_{n+1} is extruded length of the (n+1)th printing layer [mm], accordingly.

In 3D printing with FFM printers an increasing printing speed generally has a decreased print quality as a result. Therefore, the growing perimeter should be considered in a way that the extruder can travel at the same surface speed on each radius.

Equation [6] shows the calculation of an adapted printing speed.

$$F_{n+1} = F_n \frac{D + 2hn}{D + 2h(n+1)} \qquad [6]$$

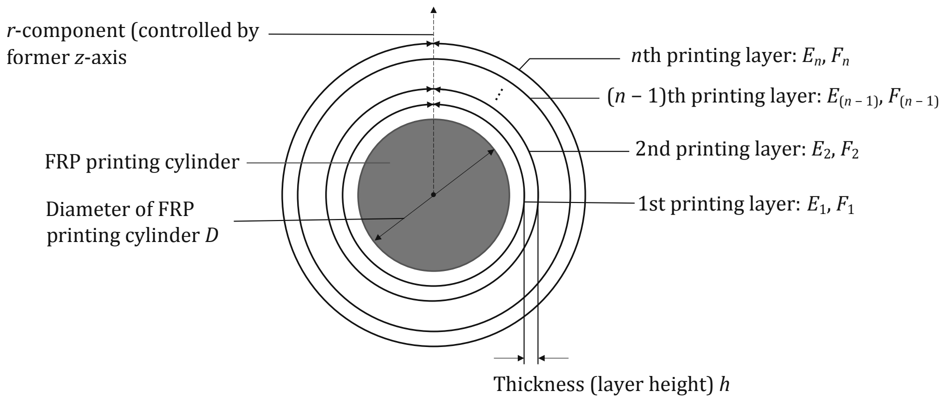
where F_n denotes printing speed of the nth printing layer [mm/min], and F_{n+1} is printing speed of the (n+1)th printing layer [mm/min], accordingly.

Figure 11 illustrates how the extruded length of an infill path E_i depends on the infill angle α. By Equation [7], we calculate the length of an infill path l_i (dashed red line in Figure 11).

According to Equations [2] and [3], the extruded length depends on the length of the extruded path. Therefore, the length of an infill path l_i also determines the extruded length of infill E_i. This extruded length E_i will be increased proportionally to the growing r-component and is calculated printing layer by printing layer similarly to Equation [5].

$$l_i = \frac{L}{\sin(\alpha)} \qquad [7]$$

where L denotes the length of sleeve [mm], α is the angle of infill layer [°], and l_i is the length of infill layer [mm].



r-component (controlled by former *z*-axis
*n*th printing layer: E_n, F_n
(*n* – 1)th printing layer: $E_{(n-1)}$, $F_{(n-1)}$
FRP printing cylinder
2nd printing layer: E_2, F_2
Diameter of FRP printing cylinder *D*
1st printing layer: E_1, F_1
Thickness (layer height) *h*

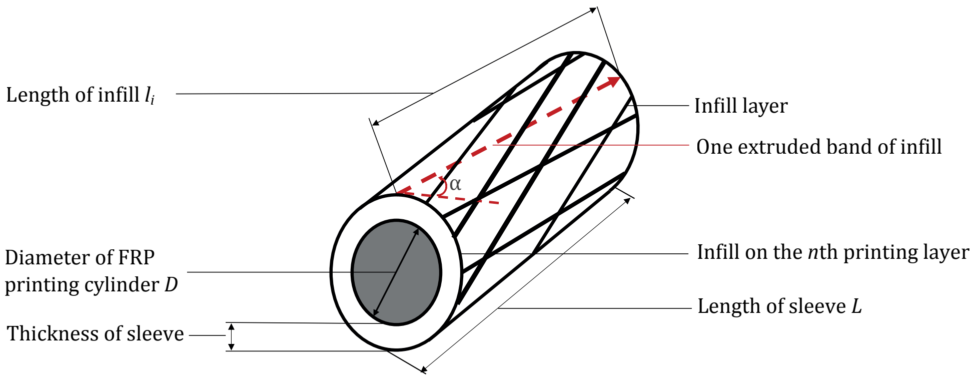
Figure 10: Dynamic extruded length for circumference and printing speed in cylindrical coordinate system

4.6 The structure of the basic layers and the air ducts

The sleeves consist of a top layer, a bottom layer and an infill layer, which includes air ducts in the case of adapter sleeves (see Figure 1). The following sections will explain each basic layer in more detail.

4.6.1 Top layer and bottom layer

The top layer and the bottom layer should be 100 % material. The top layer is the outside hull of a sleeve. The bottom layer is the inside hull of a sleeve (shown with white filling in Figure 1 and printed out in Figure 12). Together with the front face and the back



Length of infill l_i
Infill layer
One extruded band of infill
α
Diameter of FRP printing cylinder *D*
Infill on the *n*th printing layer
Length of sleeve *L*
Thickness of sleeve

Figure 11: Angle, length and extruded length of single infill

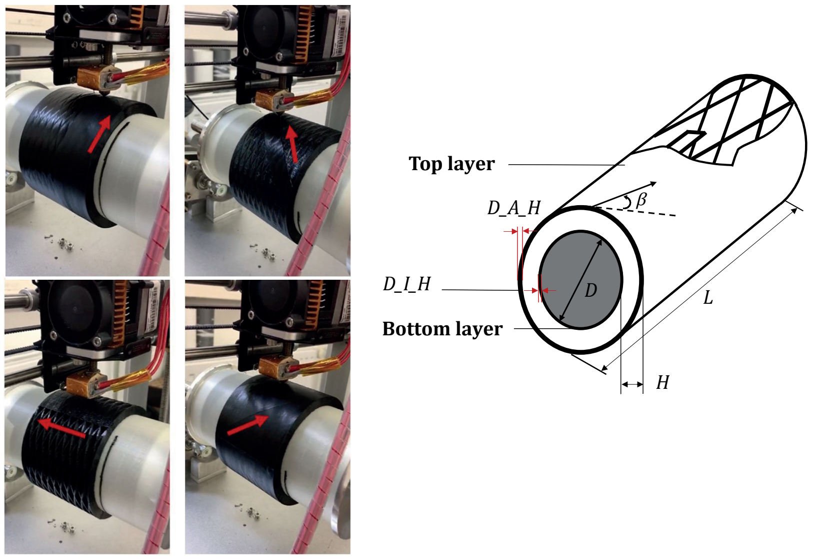


Figure 12: Four different angles of the extruded bands for the top layer during the printing process (left images) and the schematic (right image) with related parameters, for parameter descriptions see Table 1

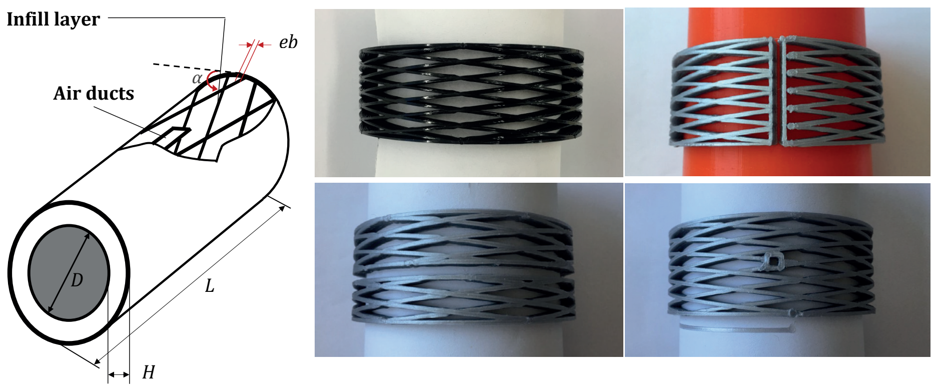


Figure 13: Schematic with infill related parameters (left image) and the printed infill with included air ducts (right images), for parameter descriptions see Table 1

face of the sleeve they build a totally enclosed body. To adjust the stability, we could use several layers to build the bottom layer and top layer. Furthermore, the angle of the extruded bands α can vary between each layer. We define $\alpha = 0^\circ$ to be in the circumferential direction and $\alpha = 90^\circ$ to be in the axial direction. The angle can be freely selected in MATLAB and the angles can be defined as a sequence for successive printing layers. The top layer and bottom layer will then be printed following the defined sequence.

4.6.2 Infill

The infill, shown with gray filling in Figure 1 and printed out in Figure 13, connects the top layer with the bottom layer. The infill is enclosed by the bottom layer, the top layer and the front face and the back face of the sleeve. Its main function is to save weight and material while delivering enough support for the top layer. The infill can be defined with any angle or sequence of angles and any necessary built-up area.

4.6.3 Air ducts

The printing sleeves in flexographic printing machines need an air cushion to be mounted as mentioned in the introduction to this paper. If there is an adapter sleeve between the air mandrel and the printing sleeve, some systems deliver the necessary air for the air cushion by air ducts through the adapter sleeve. In our MATLAB program, the air ducts can be defined in axial, radial and circumferential directions as shown in Figures 1 and 13.

5. Results

Our main result is a simple way to implement a 3D printer based on cylindrical coordinates with the corresponding G-code generator specialized for sleeves. One of the key elements of this implementation is the software for dimensioning the desired sleeves within the intended limitations. Beyond the outer dimensions of the sleeve the extruded bands angle of each printing layer is user-definable to fit mechanical needs. Based on the user-defined parameters the software automatically generates the corresponding G-code. The other key element is the modified 3D printer that interprets the G-code to produce the sleeves. As we described the results as parts of section 4 we just want to refer to the corresponding subsections that are solely our work. The whole workflow in general is described in section 4.1. The way we modified a cartesian coordinate system 3D printer to work with a cylindrical coordinate system we discussed in section 4.2. The structure of the 3D printed sleeve and the needed calculations are presented in sections 4.5 and 4.6.

6. Conclusion and future work

Recent 3D printers permit the creation of a large variety of objects. For some objects with special demands like sleeves, a less generalized and more specialized workflow in combination with a specialized 3D printer would be preferable.

In this work, we proved that this concept is fairly simple to implement. We modified a Geeetech Prusa I3A Pro FFM printer with a stepper driven FRP printing cylinder as a replacement for the *x*-axis. We provided a method based on the modified 3D printer to have additional choices for printing sleeves. We ignore the conventional workflow and develop the sleeves on a rotating surface directly via a G-code-based workflow. The matched MATLAB program allows to print a sleeve without programming within the foreseen limits. The method is suitable for the given task and will serve as a tool to optimize 3D printed sleeves.

With our work we aim not only for printing sleeves but also for adapter sleeves with imprinted air ducts. The option for air ducts is currently only implemented in principle and not adapted to a specific system.

Our future work will follow different ideas. One is to build a more professional sleeve printing device with increased 3D printing speed to produce sleeves for narrow web printing machines. In terms of sleeve variation, we have a large number of material options, post processing options and structural design options. Also, so far, we have only focused on the proof of concept – the next step will be to use a sleeve inside a flexographic machine.

References

Elsey, J., Zydex Pty Ltd., 2011. 3*D printing on a rotating cylindrical surface.* Australia. Pat. 2010,278,663.

Hodgson, G., Ranellucci, A. and Moe, J., 2016. *Slic3R manual: flow math.* [online] Available at: <https://manual.slic3r.org/advanced/flow-math> [Accessed 28 January 2019].

Hot-World, 2021. *Repetier-Host* (Windows 2.2.2). [Computer program] Hot-World GmbH & Co. KG. Available at: <https://www.repetier.com/download-software/> [Accessed 25 May 2021].

Lahteine, S., Neufeld, R., Bennett, K., Oliveira, V., Pepper, C., Smith, J., Lac, L., Kuhn, B. and van der Zalm, E. eds., 2020. *G-code Index.* [online] Available at: <https://marlinfw.org/meta/gcode/> [Accessed 16 December 2020].

Leapfrog, 2020. *3D Flexographic printing sleeve*s. [online] Available at: <https://www.lpfrg.com/testimonials/tech-sleeves-engineering-testimonial/> [Accessed 27 November 2020].

Leapfrog, 2021. *Large scale industrial 3D printer*. [online] Available at: <https://www.lpfrg.com/products/leapfrog-xcel/> [Accessed 15 February 2021].

MathWorks, 2021. *MATLAB* (R2021a). [Computer program] MathWorks. Available at: <https://www.mathworks.com/products/matlab.html> [Accessed 25 May 2021].

Shenzhen Getech Technology, 2016. *Prusa I3 A pro 3D printer*. [online] Available at: <https://www.geeetech.com/wiki/index.php/Prusa_I3_A_pro_3D_Printer> [Accessed 26 February 2021].

Shenzhen Getech Technology, 2021. *Geeetech*. [online] Available at: <https://www.geeetech.com/about-us-ezp-14.html> [Accessed 05 September 2021].

Smid, P., 2008. *CNC programming handbook: a comprehensive guide to practical CNC programming*. 3rd ed. New York, U.S.: Industrial Press.

Zheng, H., Darweesh, B., Lee, H., and Yang, L., 2019. Caterpillar – a G-code translator in grasshopper. In: M. Haeusler, M.A. Schnabel, and T. Fukuda, eds. *Intelligent & Informed: Proceedings of the 24th International Conference on Computer-Aided Architectural Design Research in Asia*. Wellington, New Zealand, 15–18 April 2019. Hong Kong: Association for Computer-Aided Architectural Design Research in Asia (CAADRIA), pp. 253–262.