3D Printing of Generative Art using the Assembly and Deformation of Direction-specified Parts

Yasusi Kanada
Dasyn.com; Tokyo; Japan
yasusi@kanadas.com

Structured Abstract:

Purpose
A methodology for designing and printing 3D objects with specified printing-direction using fused deposition modelling (FDM), which was proposed by a previous paper, enables the expression of natural directions, such as hairs, fabric, or other directed textures, in modelled objects. This paper aims to enhance this methodology for creating various shapes of generative visual objects with several specialized attributes.

Design/methodology/approach
The proposed enhancement consists of two new methods and a new technique. The first is a method for "deformation." It enables deforming simple 3D models to create varieties of shapes much more easily in generative design processes. The second is the spiral/helical printing method. The print direction (filament direction) of each part of a printed object is made consistent by this method, and it also enables seamless printing results and enables low-angle overhang. The third, i.e., the light-reflection control technique, controls the properties of filament while printing with transparent PLA. It enables the printed objects to reflect light brilliantly.

Findings
The proposed methods and technique were implemented in a Python library and evaluated by printing various shapes, and it is confirmed that they work well and objects with attractive attributes, such as the brilliance, can be created.

Research limitations/implications
The methods and technique proposed in this paper are not well-suited to industrial prototyping or manufacturing that require strength or intensity.

Practical implications
The techniques proposed in this paper are suited for generatively producing various a small number of products with artistic or visual properties.

Originality/value
This paper proposes a completely different methodology for 3D printing than the conventional CAD-based methodology and enables products that cannot be created by conventional methods.

Keywords:
3D, Computer aided manufacturing, Fused deposition modelling, Manufacturing, Rapid manufacturing, Deformation

Article Classification:
Technical paper

Biographical Details:

Yasusi Kanada received a B.E. degree in mathematical engineering from the University of Tokyo in 1979 and an M.E. degree in information engineering from the University of Tokyo in 1981. He has been working for Hitachi, Ltd. since 1981. He is a part-time lecturer at Kogakuin University. He worked at Carnegie Mellon University from 1988 to 1990, and at Tsukuba Laboratory of the Real World Computing Partnership (RWCP) from 1992 to 1995. He received a Ph.D. from the University of Tokyo in 1992. He studied programming, programming languages, and emergent computation in the University of Tokyo and in Hitachi, studies and teaches computer networks in Hitachi and Kogakuin University; he also studies new 3D-printing methods for Dasyn.com.

1. INTRODUCTION

Conventional 3D design and printing methods cannot express and embody models with internal attributes and some surface attributes. One important attribute is *direction* (Kanada, 2014; Song *et al.*, 2013). 3D printing methods such as fused deposition modeling (FDM) can express a direction at each location in the model, including each internal point, based on the printing direction. The print heads of 3D printers usually move in horizontal directions, but the print results can express non-horizontal directions by moving and stacking strings (i.e., filament) in non-horizontal directions.

There are several advantages to non-horizontal printing. First, the so-called staircase effect (Song *et al.*, 2013) can be avoided by non-horizontal head motion. This effect weakens the mechanical strength of the object and also makes the object ugly. Second, the expressing direction is important, especially when the purpose is production printing rather than prototyping because objects with natural or artificial directions, such as the direction of hair or grass, and the details of objects can be expressed in a better way (Kanada, 2014).

A methodology for printing 3D objects with a specified printing direction using the FDM method was originally proposed by a poster for the Solid Free-form Fabrication (SFF) Symposium in 2013. This methodology, which is called the direction-specified 3D design and printing methodology, was described in detail, and design and printing methods based on this methodology were proposed in the previous paper (Kanada, 2014). Users can select predefined directed parts, give parameters, assemble them, and print the assembled model using these methods.

The major purpose of this methodology is to design and to print 3D artistic models rather than industrial models generatively (Kanada, 2014) or algorithmically. This methodology is also intended to be applied to the production of final products but not of prototypes.

This paper proposes two methods and a technique for generatively designing and printing direction-specified 3D models based on the above methodology. First, a method for the *deformation* of direction-specified 3D models is proposed; that is, this paper extends the direction-specified methodology by adding another step called deformation to enable varieties of shapes to be generated much more easily by a generative design process and proposes a method for this step. Unlike computer graphics such as generative art generated using Processing (Pearson, 2011), the preservation of 3D printability is important for deformation in direction-specified 3D printing. Second, the spiral/helical printing method is proposed as a method for direction-specified 3D printing methodology. The print direction (filament direction) of each part of a printed object is made consistent by this method, and it also enables seamless or less-seam printing results and low-angle overhang. Third, a technique for controlling light reflection while using the spiral/helical printing method is introduced. By using this technique, the printed objects can reflect light brilliantly.

The rest of this paper is organized as follows. Section 2 outlines the proposed designing and printing methodology. Section 3 describes the method of deformation. This section includes the history of deformation and a description of the 3D printability concept and a method for preserving it in the deformation process. Section 4 describes the method for spiral/helical printing. Section 5 explains the technique for light-reflection control. Section 6 explains an implementation and evaluations using simple combinations and deformations of parts such as a helix (an empty cylinder) and a (filled) cylinder, which generate shapes of dishes, vases, wine grasses, and so on. Section 7 concludes this paper.

2. METHODOLOGY

This section outlines the proposed methodology for printing a 3D object with a specified printing direction, which is a refined version of the methodology proposed in the previous paper (Kanada, 2014).

The purpose of this methodology is to design and print generative art objects; thus, generative design is first explained. There are two types of 3D CAD tools, i.e., manual-design tools and generative (or algorithmic) design tools, and both types of tools can be used for both free-form design and design by parts assembly. Manual-design tools are used for specifying parts and models Many commercial and free manual-design tools, such as AutoCAD (http://www.autodesk.co.jp/products/all-autocad), SolidWorks (http://www.solidworks.com/), or Blender (http://www.blender.org/) are available. Models can be created as free-form models, i.e., freely designed by drawing lines or other shapes by using manual-design tools by pointing devices, or created by assembling predefined parts and specifying parameters manually. In contrast, generative-design tools generate models or specify parts algorithmically using a declarative or procedural language, such as Processing (Pearson, 2011). Therefore, the users of the tools, i.e., the programmers, write programs to generate models. Several generative-design tools, such as OpenSCAD (WikiBooks, 2015), are available.

The design and printing process of the proposed method consists of five steps: field-oriented modeling, partitioning, deformation, field-based tool-path generation, and non-horizontal 3D printing (**Fig. 1**) (Kanada, 2014). The first three steps are the design steps. These steps are briefly reviewed below using **Fig. 2**.

In the field-oriented modeling step, an object model with a vector field is created. This model is represented by a modeling language, which can express a vector field. It is an extension of a conventional solid model. A direction is defined at each (3D) point in the object as a vector. A field-oriented model can be created by assembling parts in a palette (Fig. 2(a)) (Kanada, 2014). Although the palette in this figure contains various parts, some of them are composite parts and still abstract (that is, the tool-paths are not yet specified in the descriptions of the parts).

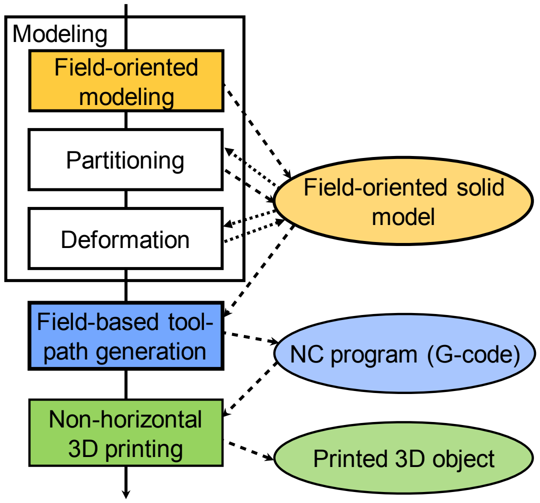
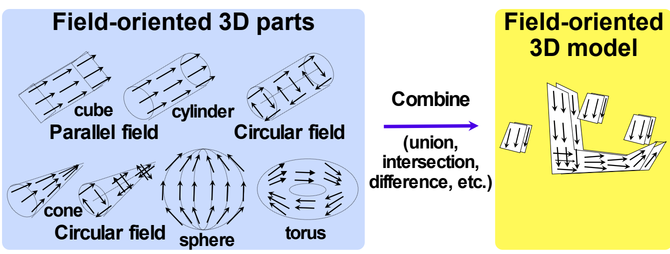
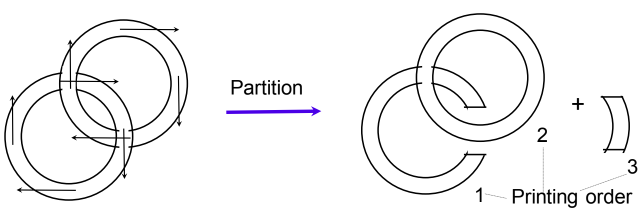


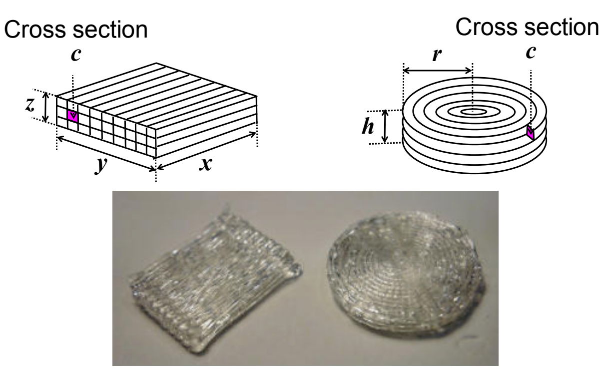
Fig. 1 Design and printing process for direction-specified 3D printing



(a) Field-oriented modeling by parts combination



(b) Printability enhancement sub-step of the partitioning step



(c) Peeling (hashing) sub-step of the partitioning step

Fig. 2 Partitioning a model for printability enhancement

In the partitioning step, the designed model is partitioned to a list of strings (fragments) that can be printed. This step consists of two sub-steps, *printability enhancement* and *peeling* (or *hashing*). Printability enhancement, which is explained in the previous paper (Kanada, 2014) and thus not explained here, is applied only when the model becomes printable by dividing it into two or more sub-models (see Fig. 2(b) for example). In the peeling sub-step, the model is "peeled" ("hashed") along the field vectors (Fig. 2(c)) and a set of *strings* is generated as a result. These strings can be regarded as filaments. Each string S*i* can be represented by

$$S_i = (P_{i_{start}}, P_{i_{end}}, c_i, v_i)$$

where $P_{i_{start}}$ means the start point of the string, $P_{i_{end}}$ means the end point of the string, c_i means the cross section of the string (which may be replaced by a filament-density parameter), and v_i means the printing speed. v_i is conceptually unnecessary, but it is useful in implementing (printing) the string. Although the strings in a model can be concretely specified by coordinates as described above, alternatively the strings, i.e., tool-path fragments, can still be abstract in this methodology; that is, concrete tool-path and printing attributes have not yet been determined.

The "peeled"-model generation process is completely different from processes used in conventional 3D printing, which is based on the Standard Triangulated Language (STL) and slicing. If the field vectors are not in parallel, peeling follows them; that is, the directions of strings are not necessarily horizontal. Advanced methods for hashing are briefly described in the previous paper (Kanada, 2014).

In the deformation step, the designed model can be deformed (modulated) to various shapes. The detail is described in the next section.

The execution order of the design steps, i.e., the first three steps, is conceptually shown in Fig. 1; however, the order is not strict. One possible method is to assemble non-partitioned parts into a field-oriented solid model first, to partition it then, and to deform it. Another possible method is to select and to deform pre-partitioned parts and to assemble them into a field-oriented solid model. In this case, the first step is partitioning, which is included in the part design process, and the second step is deformation.

In the field-based tool-path generation step, a tool-path is generated as a computerized numerical-control (CNC) program such as G-Code. Not only the path of the print head, but also the amount of extrusion, is calculated for each string. A tool-path is generated by ordering strings. A method for ordering strings is described in the previous paper (Kanada, 2014).

Finally, in the non-horizontal 3D printing step, the CNC program is executed by a 3D printer and a printed 3D object is outputted.

3. DEFORMATION

In this section, first, the history of deformation in computer graphics and generative art is briefly reviewed. Second, a problem concerning deformation in 3D printing, which does not exist in graphics, and a solution to the problem are described. Third, the printability preservation concept is described.

3.1 Deformation in computer science, technology, and art

The deformation concept plays important roles in computer graphics and in generative art.

In computer graphics, "free-form deformation" has been considered important for the solid modeling or surface modeling of objects with free-form surfaces (Sederberg and Parry, 1986; Coquillart, 1990), and it was used because it eases the control and rendering of 3D geometric shapes (Barr, 1984). In free-form deformation, a user specifies the shape of the object using a GUI.

In generative art, deformation is also an important technique and is used everywhere (Pearson 2011; Unemi *et al.*, 2008). However, in contrast to many other computer-graphics applications, in generative art, artistic objects are generated algorithmically, and deformations used for this purpose are also generative.

Deformation has not yet been focused on in 3D printing. Although deformation is essential in drawing graphics including generative art, and free-form deformation can be used in CAD for 3D printing, deformation is not useful in the slicing and printing steps of 3D printing. This is due to the fact that conventional 3D printing methodology does not take the attributes generated by the deformation into account because they are not (cannot be) used in slicing and printing steps. However, such attributes are important in direction-specified 3D printing.

3.2 Deformation method for directed 3D printing

This paper extends the direction-specified methodology by adding another step called "deformation" to enable varieties of shapes to be generated much more easily in generative design processes. Although deformation has not been discussed in a 3D-printing context, this paper focuses on generative 3D design and printing, which is based on a deformation-based method.

To preserve the printability of the model, deformation for 3D printing requires controlling two attributes of strings that comprise the peeled model. They are cross sections and printing velocity. This requirement is 3D-printing specific. The 3D-printability concept is explained and the preservation issues are explained in the next subsection.

Two types of deformations are defined. One is Cartesian-coordinate-based deformation and the other is cylinder-coordinate-based deformation. Both translate coordinates, cross sections, and printing speed.

Describing a deformation using Cartesian coordinates is sometimes useful, and thus the following function is defined (and implemented in the software library for deformation, which is described in Section 6).

$$\text{deform\_xyz}(fd(x, y, z), fc(c, x, y, z), fv(v, x, y, z))$$

Function $fd(x, y, z)$ (i.e., the first argument) maps a location (x, y, z) to a new location (x_1, y_1, z_1), so it returns three values. Function $fc(c, x, y, z)$ (i.e., the second argument) maps a cross-section at (x, y, z) to a new cross-section at (x_1, y_1, z_1). Function $fv(v, x, y, z)$ (i.e., the third argument) maps a printing speed at (x, y, z) to a new printing speed at (x_1, y_1, z_1). Function fc must be monotonically increasing about c and function fv must be monotonically increasing about v. Theoretically, any mapping can be specified for fd, fv, and k. However, they must at least be continuous and smooth to get a correct printing result. There may have to be other conditions on these functions. Such conditions will be described in future work.

It would be better if the cross section could automatically be optimized; however, it is currently difficult. Therefore, the cross section must be manually specified in the method proposed in this paper.

A cylinder coordinate is more useful for describing a deformation, especially when axisymmetric models are deformed, so the following function is defined.

$$\text{deform\_cylinder}(fd(r, \theta, z), fc(c, r, \theta, z), fv(v, r, \theta, z))$$

Function $fd(r, \theta, z)$ (i.e., the first argument) maps a location (r, θ, z), which is expressed in cylinder coordinates, to a new location (r_1, θ_1, z_1). Function $fc(c, r, \theta, z)$ (i.e., the second argument) maps a cross-section at (r, θ, z) to a new cross-section at (r_1, θ_1, z_1). Function $fv(v, r, \theta, z)$ (i.e., the third argument) maps a printing speed at location (r, θ, z) to a new speed at (r_1, θ_1, z_1). The same monotonicity conditions as deform_xyz exist for fv and fc for deform_cylinder. Functions fd, fv, and fc must be continuous and smooth.

Examples of deformation are visualized in **Fig. 3**. (A 3D-printing tool called Repetier Host was used for visualization.) Figure 3(a) shows a cup, which consists of an empty cylinder and a thin cylinder (i.e., the bottom); that is, the cup is an assembly of two direction-specified components. This cup can be transformed to a dish shown in Fig. 3(b) by applying the following deformation to the empty cylinder part.

$$\text{deform\_cylinder}(fdd(r, \theta, z), fcd(c, r, \theta, z), fvd(v, r, \theta, z))$$
$$\text{where } fdd(r, \theta, z) = (r + 1.05\, z, \theta, 0.3\, z). \qquad \ldots\ldots (1)$$

The forms of functions fvd and fcd will be described in the implementation section because they depend on the techniques described in Sections 3.3 and 4. The thin cylinder, i.e., the bottom, must be resized to fit the deformed empty cylinder. The above deformation can be applied before or after the assembly.

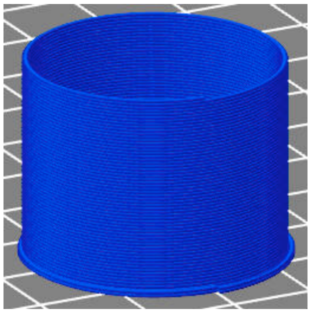
The cup can also be transformed to a vase shown in Fig. 3(c) by applying the following deformation.

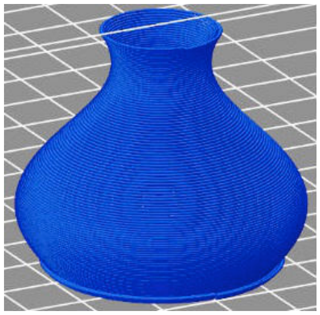
$$\text{deform\_cylinder}(fdp(r, \theta, z), fcp(c, r, \theta, z), fvp(v, r, \theta, z))$$
$$\text{where } fdp(r, \theta, z) = (r\,(0.8 + 0.4 \sin(z / 8 + 6.5)), \theta, \text{z}). \qquad \ldots\ldots (2)$$

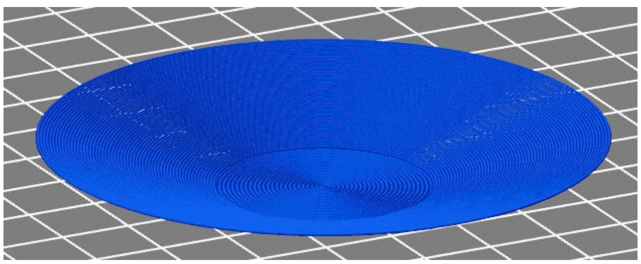
Fig. 3(d) shows an empty cylinder (without a bottom). This cylinder can be transformed to a sphere shown in Fig. 3(e), which preserves the filament pitch, by applying the following deformation.

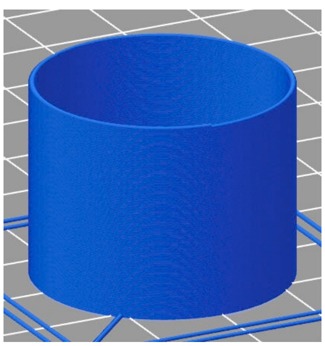
$$\text{deform\_cylinder}(\,fds(r,\theta,z), fcs(c,r,\theta,z), fvs(v,r,\theta,z))$$
$$\text{where } fds(r,\theta,z) = (Radius * \sin(\pi z \,/\, cylinderHeight), \theta, \; r - Radius * \cos(\pi z \,/\, cylinderHeight)) \quad \ldots\ldots (3)$$

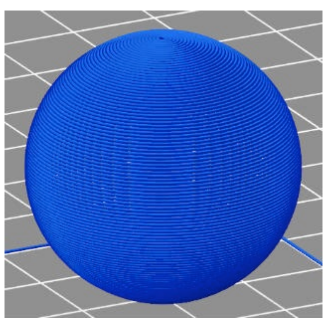
Parameter *cylinderHeight* is the height of the empty cylinder before the deformation, and it is half of the equator length of the sphere generated by the deformation.


(a) Cup – original shape


(c) Vase – deformation result


(b) Dish – deformation result


(d) Empty cylinder – original shape


(e) Sphere – deformation result

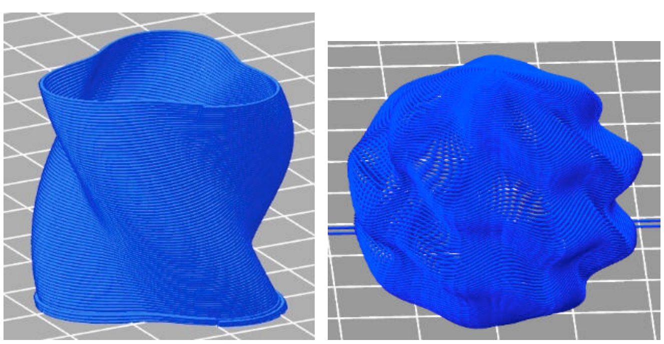

(f) Non-axisymmetric shapes – deformation results

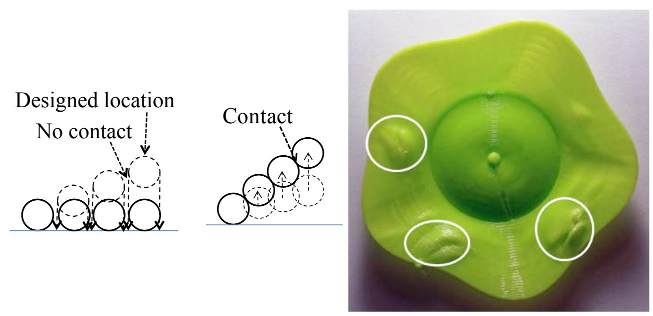
Fig. 3 Examples of deformation

Moreover, non-axisymmetric shapes such as those shown in Fig. 3(f), which can be generated by deforming axisymmetric shapes shown in Fig. 3(a) and (d), can be easily generated using cylinder-coordinate based deformations. Although various shapes exist in Fig. 3, all these shapes are generated only using these trigonometric functions. However, other types of functions can of course be used.

3.3 Printability preservation

The concept of 3D printability plays an important role in avoiding printing failures in the proposed method. By giving the 3D printability measure to each string of a peeled model and by managing the measure while deforming the model, the printability of the model can be preserved. Thus if the deformation reduces the string's density, the cross section must be increased to preserve the printability. If the amount of extrusion is constant, a new string may fail to stack and drop (**Fig. 4**(a)). On the contrary, if the deformation increases the string's density, the cross section must be decreased. Otherwise, excess filament may cause bumps because of the elasticity of printed filament (Fig. 4(b)).

If extruded filament becomes solid immediately, there is probably no need to update the printing speed. The printing speed is dependent on the thermal and mechanical features of the filament and the 3D printer. However, because it takes some time for filament to become solid, if the deformation makes the round-trip time of the print head shorter, the printing speed must be reduced to keep the round-trip time longer (maybe constant). The *round-trip time* described above means the time required to touch old and new extruded filament. Therefore, deformation updates the printing speed of the model by *fv*.



(a) Dropped sparse filament (cross section)

(b) Overstacked dense filament (cross section and top-face)

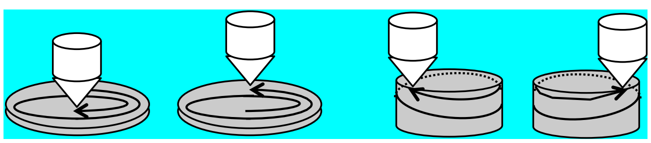
Fig. 4 Printing failures

A peeled model is defined to be *3D printable* (or to have 3D printability) if it can be translated into a CNC program that prints a 3D object of a designed form. If the translated program usually or often fails to generate a designed object, the model is not 3D printable. For example, the coordinate of printed filament may have large error compared with the specified coordinate of the string in the model. This means the model is not 3D printable. This 3D printability concept still has many ambiguities, which must be resolved to make the concept unambiguous.

Model parts used for this method must be 3D printable, and the modeling and deformation steps for this method must preserve 3D printability. To preserve 3D printability, the cross section and the printing speed specified in the strings in the model must be updated correctly. If a deformation increases or decreases the density of strings, the cross section must also be increased or decreased. If the density of strings is initially $n0$ (/ mm^2) and it becomes $n1$ and the cross section is initially $c0$, the cross section should become $c0 * n0 / n1$ (approximately).

4. SPIRAL/HELICAL PRINTING METHOD

The *spiral/helical printing method* enables the printing of axisymmetric direction-specified 3D models such as cylinders or spheres with consistent print-direction vector field, i.e., filament directions, and without seams or with less seams. The spiral/helical printing method means a method for printing an object spirally or helically with adjusting cross section and printing speed to preserve the 3D printability (**Fig. 5**). If a model to be printed (or a part in assembled parts) is axisymmetric, it is natural to include a circumferential-direction vector in each location in the model. Such a model is peeled to be a collection of circumferential strings, and a natural method for approximately printing such a model is to print it spirally or helically. This method enables non-printing head motions to be reduced; that is, it enables more seamless printing. Seamlessness matters when printing final products or artistic objects. To print a widening or narrowing helix (i.e., a helix with increasing or decreasing radius) using this method, cross section and printing speed must be properly controlled, as described in the previous section. In addition, this method can also generate a thin and strong structure.



(a) Spiral printing (b) Helical printing

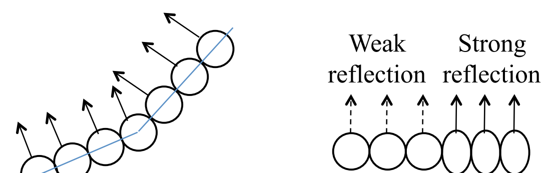
Fig. 5 Spiral/helical printing method

By using the spiral/helical printing method, a quickly widening helix such as a dish-like shape or a narrowing helix such as an umbrella-like or dome-like shape can be printed without support material. As such, this print method allows high-angle overhangs. Conventional 3D-printing methods do not allow low-angle overhang, e.g., more than 75 degrees, without support material. Because support material causes several problems such as extra material consumption and difficulty in removing it, it is beneficial if no support is required. Although shapes that can be printed by this method are restricted, it is still beneficial.

However, when printing a widening or narrowing helix at a higher speed, the author found that the cross section should be decreased or increased to stabilize printing (to preserve printability). Otherwise, printing failures such as those shown in Fig. 4 may occur. To enable high-angle overhang when printing a widening helix, the cross section should be smaller than that of a normal vertical helix with the same pitch (i.e., with a constant helix), because the filament is pulled toward the center of the curvature. The force on the extruded filament enables high-angle overhang. On the contrary, when printing a narrowing helix, the cross section should be larger than that of a vertical helix because the filament must be pushed toward the outer direction of the curvature (see Fig. 5(b)). This force enables high-angle overhang.

5. LIGHT-REFLECTION CONTROL TECHNIQUE

Some material used for FDM printers reflects light on the surface and brilliantly shines, and the amount and the direction of reflection can be controlled by certain techniques. This is an attractive attribute for artistic or visual-design purposes. For example, transparent polylactic acid (PLA), especially pure PLA, is quite attractive in reflection (see Fig. 8); that is, reflection becomes strong in 3D-printed objects made of PLA because strings increase the surface area that reflects light. This attribute is caused not only by the surface, but it is also affected by the internal structure of an object. If the filament is not transparent, the brilliance disappears. The reflection can be controlled by the overhang angle, filament density, or some other attribute of the spiral/helical printing method (**Fig. 6**). Fig. 6(a) shows reflection controlled by the overhang angle. If the angle of the light is changed, different portions of the object more strongly reflect the light. Fig. 6(b) shows reflection controlled by the filament density. Even if the angle of the light is changed, the dark part never reflects light brilliantly.



(a) Reflection controlled by the overhang angle (b) Reflection controlled by the filament density

Fig. 6 Reflection control

6. IMPLEMENTATION AND EXAMPLES

6.1 Python library

A preliminary version of a modeler program library, **`draw3.py`**, for generating G-Code for 3D printers was developed and used for evaluation, and an updated version, **`draw3dp.py`**, is publicly available at http://www.kanadas.com/program-e/2014/10/-3d_printing_library_for_parts.html. By using this library, although no GUI-based modeler is available, the user can write a program to select parts, such as a line, a helix, a cylinder, to specify parameters of the parts, and to combine them. Deformations can be applied to the combined model and to parts or partially combined model as well.

Model data representations in a program can be described as follows. A peeled model is represented by a list of strings in this library program, as described in Section 2. A string is a named tuple (or structure) that contains coordinates x, y, and z, a cross section, and a printing speed.

A model part can be generated by a function defined in the library. For example, the following function generates a string list that represents a cylinder.

draw.cylinder(radius, height, vpitch, hpitch, x0, y0, z0)

The parameters “radius” and “height” determine the shape of the cylinder; the parameters vpitch (i.e., vertical string-pitch) and hpitch (i.e., horizontal string-pitch) determine the cross section; and x0, y0, and z0 determine the bottom center coordinates of the cylinder. Parts can be concatenated (i.e., assembled) to form an object.

```
import draw                        # Import the library
from math import sin, cos          # Use of standard library

## Constants ##
x0 = 0; y0 = 0;                    # Print bed center for Rostock MAX
FilamentDiameter = 1.75            # Filament diameter (mm)
HeadTemperature = 220              # For PLA (°C)
BedTemperature = 0 # For PLA
DefaultCrossSection = 0.196        # (mm^2)

## Cup generator ##
def cup(radius, height, bottomHeight, vpitch, hpitch,
        x0, y0, z0, speed):        # Generate a model of a cup
    draw.speed(speed)
    draw.cylinder(radius + 1.5 * hpitch, bottomHeight,
                  vpitch, hpitch, x0, y0, z0)
    draw.speed(speed)
    draw.helix(radius, height – bottomHeight + vpitch/2.0, vpitch,
               x0, y0, z0 + bottomHeight – vpitch/2.0)
    return

## Initialization ##
draw.init(FilamentDiameter, HeadTemperature,
          BedTemperature, DefaultCrossSection, x0, y0, 0.4)

skirt()    # Definition of skirt() is omitted here

## Define initial model ##
radius = 15; height = 24; bottomHeight = 0.81;
vpitch = 0.4; hpitch = 0.6; x0 = 0; y0 = 0; z0 = 0.4; # (mm)
cup(radius, height, bottomHeight, vpitch, hpitch,
    x0, y0, –bottomHeight, 20)

## Deformation ##
draw.deform_cylinder(
  lambda r, theta, z:
    ((r + 0.94*z) * (1 + sin(z/7)*cos(4*theta) / 12) if z > 0 else r,
     theta,
     (0.6*z if z > 0 else z + 3*(1 – (r/radius)**2)) + \
       bottomHeight + z0),
  lambda v, r, theta, z: v,
  lambda c, r, theta, z: 1.3 * c if z > 0 else 2.1 * c)

## Generate G-code ##
draw.draw()
```

Fig. 7 Usage example of `draw.py` (a reflection-controlled dish (Fig. 8(c))

Operations of the deformations can be described as follows. A Cartesisan-coordinate based deformation functions to operate directly on these values. A cylinder-coordinate based deformation translates the Cartesian coordinates, i.e., *x*, *y*, and *z*, to cylinder coordinates, functions on the translated values, and translates the results back to a Cartesian coordinates in the current version of the library. This method may be optimized to reduce errors in a future version.

A usage example, which generates a G-code program for a 3D printer, is shown in **Fig. 7**. This program assumes that a delta-type 3D printer called Rostock MAX is used (but, of course, the proposed methods and technique can also be used for other types of printers). Delta-type printers are good for quick vertical motions. Most of the parameters in this program are not yet theoretically determined, but experimentally determined. However, they meet the theory qualitatively. The radius of the nozzle hole is 0.25 mm, so the default cross-section (DefaultCrossSection) is set to 0.196 (= $0.25^2\pi$) mm^2. In this program, function draw.cylinder generates a peeled model of a filled cylinder, as described above, and function draw.helix generates a peeled model of an empty cylinder. They are concatenated to form a cup. The generated cup is deformed to a dish by function draw.deform_cylinder. The arguments for this function include the functions $fd(r, \theta, z)$, $fc(c, r, \theta, z)$, and $fv(v, r, \theta, z)$, which are described in Section 3.2.

6.2 Creation of dishes and vases

Various dishes and vases can be created by deforming a cup, as shown in Fig. 3(a); that is, a combination of an empty cylinder and a thin cylinder. The results of printing a basic form of a dish, as shown in Fig. 3(b), and several variations by Rostock Max 3D-printer are shown in **Fig. 8(**a) to (c). The results of printing a basic form of a vase, as shown in Fig. 3(c), and several variations of deformed cups are shown in Fig. 8(d) to (f). The dish and vase were created by deformations using horizontal and/or vertical sine waves. Deformation techniques described in Section 3 were applied to both. Each of these objects takes 10 to 20 minutes to be printed.

A dish shown in Fig. 8(a) was printed by deformation (1) in Section 3.2. The translation functions for the printing speed and cross section used for the deformation described in Section 3.2 are as follows.

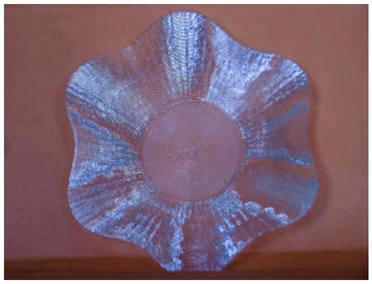
$$fcd(c, r, \theta, z) = 0.96\, c$$
$$fvd(v, r, \theta, z) = v$$

These functions qualitatively follow the techniques described in Sections 3.3 and 4; however, the functional forms and constant values were defined experimentally.

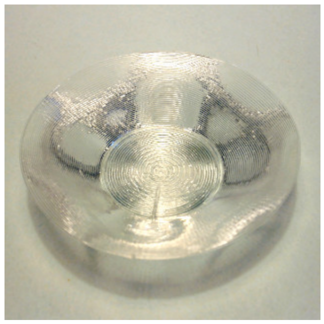
Figure 8(b) shows a variation of a dish (with 6-cycle modulation), and Fig. 8(c) shows a dish printed by the program shown in Fig. 7. The deformation function contains $\cos(4\theta)$, which generates the 4-cycle patterns shown in Fig. 8. The deformation function uses the light-reflection control technique. Moreover, the photos in this figure show the dish with two light-source directions. The brighter areas move while changing the light-source direction. A printing process of the dish shown in Fig. 8(c) can be seen at http://youtu.be/5P1vaahzW98 .

The plain vase shown in Fig. 8(d) was printed by deformation (2). The translation functions are as follows.

$fcp(c, r, \theta, z)) = 1.9\ c$
$fvp(v, r, \theta, z) = (1 + 0.4 \sin(z / 4) \cos(3\theta))\ v$

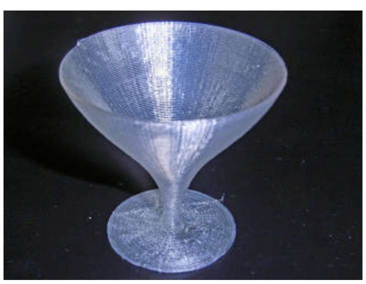



(a) Plain dish (b) 6-cycle dish

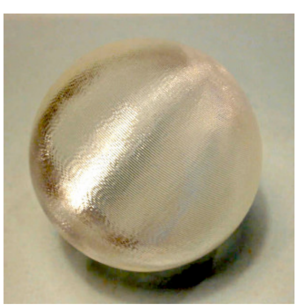
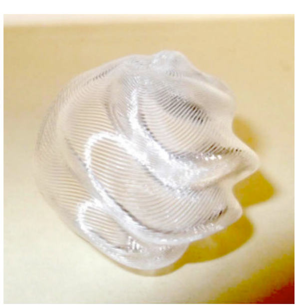



(c) Reflection-controlled dish with two light directions


(d) Plain vase (e) Skewed cup



(f) “Wine glass”

(g) Sphere (h) Waved sphere

Fig. 8 Dishes and pods generated by various deformations

Figure 8(e) shows a skewed cup, and Fig. 8(f) shows a “wine glass,” which is different from the usual wine glass in shape; that is, wine goes into the stem because there is no raised bottom. They are generated from a cup as well. The shapes of all the dishes and pods are approximated by 72 linear lines per round trip of the head. The turning points of the head and filament can be observed in these photos, especially in Fig. 8(a) to (c). The dishes will look better if they consist of finer lines, but it will take more time to print them.

6.4 Creation of spheres

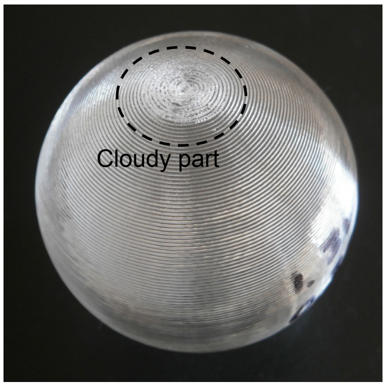
A sphere can be created by deforming a thin cylinder, as shown in Fig. 3(d). An example of a printed sphere is shown in Fig. 8(g), and a modulated sphere is shown in Fig. 8(h). The printing speed must be reduced and the cross sections of strings must be decreased in the lower and upper parts of the sphere (or modulated sphere). The amount of increase and decrease was still determined experimentally. An optimized set of translation functions for the sphere.

$fcs(c, r, \theta, z) = 1.2\ c$
$fvs(v, r, \theta, z) = 1.2\ ((fr(r, \theta, z) / Radius)^2 + 0.1)\ v$

where *fr* is the first component of *fds* (i.e., the radius). The printability preservation techniques described in Section 4 are not applied to this set of functions. However, if the value of *fvs* is increased, that is, the printing speed is optimized, some techniques including the stabilization of printing (i.e., the printability preservation techniques) are required. A printing process of the sphere shown in Fig. 8(g) can be seen at http://youtu.be/xr6zg0Z07HA .

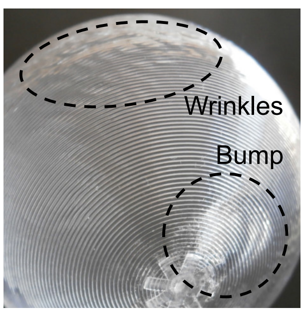
The printability preservation techniques were also qualitatively evaluated. When they were applied, the printer seldom failed to print a sphere even when the printing speed was faster. The cross section of string was decreased in the lower half of the sphere, and it was increased in the upper half. The amount of increase/decrease was still determined experimentally. Automation of these parameters will be included in future work. However, if the techniques were not applied, i.e., the cross section and the printing speed were constant, the printer easily would fail to print a sphere (see **Fig. 9**).

The evaluation results are summarized as follows. If the printing speed and the cross section of the filament are properly selected, spheres with a radius of 20 mm can be printed mostly fine with transparent PLA without changing the cross section. Figure 9(a) shows an example of such a result. The printing speed and the cross section in this case are defined as *v*0 and *c*0. Although this sphere is mostly well-formed, the top of the sphere has a small cloudy part (i.e., non-transparent white part), which can be avoided by increasing the printing speed. If the printing speed is decreased, clouds also appear at the bottom (Fig. 9(b)). (This figure shows a “star mark,” which is part of the support, at the center. This support is required because it is not possible to print a sphere supported only at a single point.) If the printing speed is increased, bumps are generated at the bottom even when the cross section is decreased to $0.79\,c0$ (Fig. 9(c)). If the cross section is decreased to reduce bumps, other defects, such as gaps or wrinkles, may appear, as shown in Fig. 9(c) and (d). If the printing speed is to be increased and defects are to be avoided, the cross section must be sloped. If the printing speed is $1.38\,v0$, the cross section at the top is $c0$, and that at the bottom is $0.79\,c0$, the printed sphere is fine and better than the sphere shown in Fig. 9(a).

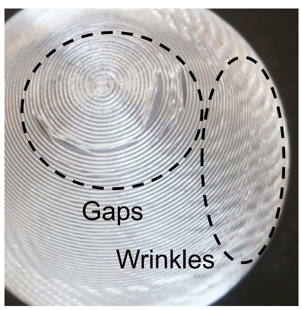


(a) Top of the sphere (when $v = v0$ and $c = c0$)

(b) Bottom of the sphere (when $v = 0.92\,v0$ and $c = c0$)



(c) Bottom of the sphere (when $v = 1.38\,v0$ and $c = 0.79\,c0$)



(d) Top of the sphere (when $v = 1.38\,v0$ and $c = 0.79\,c0$)

Fig. 9 Sphere printing failures

7. CONCLUSION

This paper aims to enhance a previously proposed direction-specified 3D-printing methodology for creating various shapes of generative artistic objects with several specialized attributes. This enhancement consists of two new methods and a new technique.

The first method, i.e., “deformation,” enables simple 3D models to be deformed to create varieties of shapes much more easily in generative design processes. The second method, i.e., the spiral/helical printing method, enables thin and strong structure and low-angle overhang to be generated. The third, i.e., the light-reflection control technique, controls light reflection and enables the printed objects to reflect light brilliantly. The proposed methods and technique were implemented in a Python library and evaluated by printing various shapes.

By this implementation and evaluation, two conclusions are confirmed. The first conclusion is that the proposed method works well for thin axisymmetric shapes, such as cylinders and spheres. The second conclusion is that objects with a variety of shapes, such as dishes, vases, and waved spheres, and attractive attributes, such as brilliant reflection, can be obtained by deforming thin axisymmetric shapes.

REFERENCES

Barr, A. H. (1984), “Global and local deformations of solid primitives”, *ACM SIGGRAPH Computer Graphics*, Vol. 18, Vol. 3, pp. 21–30.

Coquillart, S. (1990), “Extended free-form deformation: a sculpturing tool for 3D geometric modeling”, *ACM SIGGRAPH Computer Graphics*, Vol. 24, No. 4, pp. 187–196.

Kanada, Y. (2014), “Method of Designing, Partitioning, and Printing 3D Objects with Specified Printing Direction”, *2014 International Symposium on Flexible Automation* (*ISFA 2014*).

Pearson, M. (2011), “Generative Art: A Practical Guide Using Processing”, Manning Publishing Co.

Sederberg, T. W. and Parry, S. R. (1986), “Free-form deformation of solid geometric models”, *ACM SIGGRAPH Computer Graphics*, Vol. 20, No. 4, pp. 151–160.

Xuan Song, Yayue Pan, and Yong Chen (2013), “Development of a Low-Cost Parallel Kinematic Machine for Multi-Direction Additive Manufacturing”, *SFF Symposium 2013*.

Unemi, T., Matsui, Y., and Bisig, D. (2008), “Identity SA 1.6: An Artistic Software that Produces a Deformed Audiovisual Reflection Based on a Visually Interactive Swarm”, *2008 International Conference on Advances in Computer Entertainment Technology* (*ACE '08*), pp. 297-300.

WikiBooks (2015), “OpenSCAD User Manual”, available at http://en.wikibooks.org/wiki/OpenSCAD_User_Manual (accessed March 2015)