Geometric Modeling and 3D Printing Using Recursively Generated Point Cloud

Tashi 1, AMM Sharif Ullah 2,* and Akihiko Kubo 2

1 Graduate School of Engineering, Kitami Institute of Technology, 165 Koen-cho, Kitami 090-8507, Japan; tashi.cst@rub.edu.bt
2 Faculty of Engineering, Kitami Institute of Technology, 165 Koen-cho, Kitami 090-8507, Japan; kuboak@mail.kitami-it.ac.jp
* Correspondence: ullah@mail.kitami-it.ac.jp; Tel./Fax: +81-157-26-9207

Received: 15 June 2019; Accepted: 16 September 2019; Published: 17 September 2019

Abstract: Previous studies have reported that a recursive process called the point cloud creation algorithm (PCA) that generates a point cloud is useful for reverse engineering a planner shape. This study elucidates the characteristics of the parameters used in the recursive process as well as its ability in geometric modeling and 3D printing of 3D shapes. In the recursive process, three constants (center point, initial distance, and initial angle) and two variables (instantaneous distance and instantaneous rotational angle) are employed. The shape-modeling characteristics of the constants and variables are elucidated using some commonly used shapes (straight-line, circle, ellipses, spiral, astroid, S-shape, and leaf-shape). In addition, the shape-modeling capability of the recursive process as a whole is quantified using two parameters called the radius of curvature and aesthetic value. Moreover, an illustrative example that shows the efficacy of the recursive process in virtual and physical prototyping of a relatively complex 3D object is presented. The results show that reverse engineering performed by the recursive-process-created point cloud is free from computational complexity compared to reverse engineering performed by the 3D-scanner-created point cloud. As such, the outcomes of this study enrich the field of reverse engineering.

Keywords: geometric modeling; additive manufacturing; reverse engineering; point cloud; design for additive manufacturing

1. Introduction

Additive manufacturing (AM) (in common terms, 3D printing) has emerged as a useful process for producing complex shapes directly from the solid 3D CAD models of the shapes [1–7]. Numerous industrial sectors have been benefitting from AM, including the automotive industry [8], the aerospace industry [1], the biomedical engineering field (e.g., dental braces, artificial tissues, hearing aids, dummy cadavers, and prosthetics) [9–15], the education field [16–18], the fashion industry (e.g., cloth, shoes, and jewelry) [19–23], the food industry (e.g., pizza, chocolate, crackers, and pasta) [24–28], and the construction industry (e.g., 3D printed houses) [29–32]. AM has also been applied to preserving and restoring cultural heritage [33]. Thus, the advent of AM has created a new field called Design for Additive Manufacturing (DfAM). One of the objectives of DfAM is to manage the complexity in the geometry and topology of a given object for the ease of AM [34].

The typical steps underlying an AM process are Steps 4 to 7, as shown in Figure 1. In some cases, particularly when a given object is reverse engineered [35–37], Steps 1 to 3 in Figure 1 must be added to make the process meaningful. Therefore, some of the challenges of DfAM are rooted in Steps 2 and 3, as described below.
Thus, the abovementioned data acquisition processes require either image processing or point cloud processing, a given object is scanned by a 3D scanner and then represented by a relatively large set of points known as a point cloud. Afterward, the data are processed by an appropriate computational method (Step 3) for making the information meaningful to a CAD package for solid modeling. In the case of sketch processing, a given object is sketched and then the sketch data are processed by an appropriate computational method for making the information meaningful to a CAD package for solid modeling. In the case of scanned point-cloud processing, a given object is scanned by a 3D scanner and then represented by a relatively large set of points known as a point cloud. Afterward, the point cloud is processed by an appropriate computational method for making the information meaningful to a CAD package for solid modeling. Thus, the abovementioned data acquisition processes require either image processing or point cloud processing. The computational complexity and the respective solutions have been studied by numerous authors, as reported in [38]. For the sake of better understanding, the following articles can be noted that show the computational details regarding raster to vector graphics conversion [39–41], sketch processing [42–44], data registration [9,45–47], noise removal [48,49], data reduction [50,51], curve-based surface reconstruction [42,52–58], polygon-based surface reconstruction [59–63], and direct slicing [64–68].

Compared to the image processing, sketch processing, and scanned point-cloud processing, analytical-point-cloud-processing-based reverse engineering is still in infancy. However, one of the advantages of analytical-point-cloud-based reverse engineering is that it can restrict the size of the point cloud to a small one. This way, some of the computational complexity as mentioned above can be avoided, as shown by the authors in [69]. On the other hand, there is an inherent problem, that is, what is the best approach to create an analytical point cloud? In this regard, there are multiple approaches. An analytical point cloud can be created using a deterministic approach [38,70], a semi-random approach or affine mapping-based approach [34,71], and a fully random approach [72]. In each case, the 3D solid model construction process can be systematized using the information underlying the analytically created point cloud. This is perhaps not the case when the other approaches are used to create the point cloud.

Figure 1. Additive manufacturing from the viewpoint of reverse engineering.
In this article, the focus is on the deterministic point cloud that is created by a recursive process as shown in [38,69,70]. This process uses two variables and three constants, which are user-defined. The role and the characteristics of these variables and constants must be known beforehand to facilitate the modeling process. Whether or not a given shape has been modeled as per the requirement has not been elucidated yet. The deterministic point cloud creation process has been applied to modeling the planner shapes only. Therefore, the following questions are yet to be answered.

- What are the characteristics and roles of the variables and constants used in the recursive process that deterministically creates a point cloud?
- How do we quantify the shape information modeled by a set of a point cloud?
- Does the recursive process that deterministically creates a point cloud correctly model a given shape?
- Does the deterministically created point cloud help create 3D solid models of some given objects?

This article addresses answers to the aforementioned questions. The rest of the article is organized as follows. Section 2 describes the characteristics of the point-cloud-creating algorithm. Section 3 describes the quantification of point cloud by the radius of curvature and aesthetic value. Section 4 describes the 3D solid modeling and physical prototyping of a given shape guided by some points. Section 5 describes the discussion on a point-cloud-based surface texturing and processing of point cloud obtained from a 3D scanner. Section 5 provides the concluding remarks of this study.

2. Characteristics of the Point-Cloud-Creating Algorithm

Recall the following question. What are the characteristics and roles of the variables and constants used in the recursive process that deterministically creates a point cloud? The recursive process, called the point cloud creation algorithm (PCA), that creates a point cloud on a plane is shown in Appendix A. As shown in Appendix A, PCA needs three constants as inputs—namely, the center point, \( P_c \); initial distance, \( d \); and initial angle, \( \theta \)—that remain the same for all iterations. In addition, PCA needs two variables as inputs—namely, instantaneous rotational angle, \( \theta_i \), and instantaneous distance, \( r_i \), \( i = 0, 1, ..., n \)—that may change in every iteration. The PCA creates the output, denoted as \( PC = \{ P_{ei} \mid i = 0, 1, ..., n \} \), which is a small-size point cloud. As such, the output of PCA, \( PC \), depends on \( d, \theta, \theta_i, \) and \( r_i \). This input–output relationship is characterized by using some well-known shapes, namely, straight-line, circle, ellipse, spiral, astroid, S-shape, and leaf-shape. The description is as follows.

First, consider the point clouds that evolve due to the linear settings of the instantaneous rotational angle and distance. These settings create some point clouds presenting straight lines, circles, and spirals, as shown in Figures 2–5. For example, consider a point cloud that models a straight line. In this case, \( \theta_i \) must be kept constant and \( r_i \) must linearly increase or decrease. When \( \theta_i = 0^\circ \) and \( r_i \) linearly increases for all iterations, the PCA creates a point cloud, \( PC \), presenting a horizontal line, as shown in Figure 2. When \( \theta_i = 90^\circ \) and \( r_i \) linearly increases for all iterations, the PCA creates, \( PC \), presenting a vertical line, as shown in Figure 3. The slope of the straight line can be modeled using \( \theta_i \). When an opposite strategy is adopted, that is, \( \theta_i \) is increased or decreased linearly and \( r_i \) is kept constant, the point cloud, \( PC \), presents a shape called a circle, as shown in Figure 4. In this case, the radius of the circle can be modeled using \( r_i \). On the other hand, when both \( \theta_i \) and \( r_i \) are increased or decreased simultaneously, then the point cloud, \( PC \), presents a shape called a spiral, as shown in Figure 5.

In the next step, consider the relatively complex settings of the instantaneous angle and distance while keeping the instantaneous rational angle increase linear. For example, when \( \theta_i \) is increased linearly and \( r_i \) follows a tent function, then the point cloud, \( PC \), presents a shape called leaf-shape, as shown in Figure 6. When \( \theta_i \) increases linearly and \( r_i \) follows a sine or cosine function with two cycles, then the point cloud, \( PC \), presents a shape called ellipse, as shown in Figure 7. Similarly, when \( \theta_i \) increases linearly and \( r_i \) follows a u-shape function with four cycles, then the point cloud, \( PC \), presents a shape called astroid, as shown in Figure 8. When \( \theta_i \) increases linearly and \( r_i \) follows a bathtub function with two cycles, then the point cloud, \( PC \), presents a shape called S-shape, as shown in
As shown in Figures 2–9, keeping \( \theta_i \) very simple (constant or straight-line) and varying \( r_i \) using some simple functions (straight-line, tent function, sine/cosine, \( \text{u-shape function}, \) and bathtub functions) can model a variety of shapes. The findings are summarized in Table 1. These characteristics of the algorithm can be used while modeling an arbitrary planner shape or a segment of it.

As shown in Table 1, besides the instantaneous rotational angle and distance described above, the other input parameters of the PCA, the center point, \( P_c \) and the initial angle, \( \varphi \) also play an important role. For instance, the leaf-shape shown in Figure 9 can be rotated by resetting the values of \( \varphi \). For example, when the initial angle is set in the following manner

\[
\varphi = 0^\circ, 45^\circ, \text{ and } 90^\circ,
\]

respectively, as shown in Figure 10. Similarly, \( P_c \) can perform the function of translation in both \( x \)-axis and \( y \)-axis directions. The initial distance, \( d \), on the other hand, does not play many roles, i.e., it does not affect the shape to be modeled; it is only necessary for placing the starting point.

![Figure 2](image2.png)

**Figure 2.** Settings for a horizontal straight line using the point cloud creation algorithm (PCA). (a) Input1: \( \theta_i \); (b) Input2: \( r_i \); (c) Output: Point cloud.

![Figure 3](image3.png)

**Figure 3.** Settings for a vertical straight line using the PCA. (a) Input1: \( \theta_i \); (b) Input2: \( r_i \); (c) Output: Point cloud.

![Figure 4](image4.png)

**Figure 4.** Settings for a circle using the PCA. (a) Input1: \( \theta_i \); (b) Input2: \( r_i \); (c) Output: Point cloud.
Figure 5. Settings for a spiral using the PCA. (a) Input1: \( \theta_i \); (b) Input2: \( r_i \); (c) Output: Point cloud.

Figure 6. Settings for a leaf-shape using the PCA. (a) Input1: \( \theta_i \); (b) Input2: \( r_i \); (c) Output: Point cloud.

Figure 7. Settings for an ellipse using the PCA. (a) Input1: \( \theta_i \); (b) Input2: \( r_i \); (c) Output: Point cloud.

Figure 8. Settings for an astroid using the PCA. (a) Input1: \( \theta_i \); (b) Input2: \( r_i \); (c) Output: Point cloud.
Figure 9. Settings for an S-shape using the PCA. (a) Input1: $\theta_i$; (b) Input2: $r_i$; (c) Output: Point cloud of half of an S-shape.

Figure 10. Role of an Initial Angle. (a) When $\varphi = 0^\circ$; (b) when $\varphi = 45^\circ$; (c) when $\varphi = 90^\circ$.

<table>
<thead>
<tr>
<th>Instantaneous Rotational Angle, $\theta_i$</th>
<th>Instantaneous Distance, $r_i$</th>
<th>Point Cloud, PC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constant</td>
<td>Increase or decrease linearly</td>
<td>Straight-line</td>
</tr>
<tr>
<td>Increase or decrease linearly</td>
<td>Constant</td>
<td>Circle</td>
</tr>
<tr>
<td></td>
<td>Increases linearly</td>
<td>Spiral</td>
</tr>
<tr>
<td></td>
<td>Increase and decrease linearly like a tent function</td>
<td>Leaf-shape</td>
</tr>
<tr>
<td></td>
<td>Decrease and increase gradually like a sine/cosine for two cycles</td>
<td>Ellipse</td>
</tr>
<tr>
<td></td>
<td>Decrease and increase gradually like a U-shape function for four cycles</td>
<td>Astroid</td>
</tr>
<tr>
<td></td>
<td>Decrease and increase gradually like a bathtub function for two cycles</td>
<td>S-shape</td>
</tr>
</tbody>
</table>

| Center point, $P_c$                     | Translate a shape             |
| Initial angle, $\varphi$               | Rotate shape                  |
| Initial distance, $d$                   | No effect on a shape          |

3. Quantification of Point Clouds

Recall the following questions. How do we quantify the shape information modeled by a set of a point cloud? Does the recursive process that deterministically creates a point cloud correctly model a given shape?

In order to answer the abovementioned questions, two well-known parameters called a radius of curvature and aesthetic value, defined in Appendix B, are considered.

Let $A$ be a planner shape. Let $U$ be a method that creates a point cloud $V = \{V_i \mid i = 0, 1, \ldots \}$ to model $A$. In addition, $PC = \{P_{ci} \mid i = 0, 1, \ldots \}$ is created by PCA to model $A$. As such, both point clouds, $V = \{V_i \mid i = 0, 1, \ldots \}$ and $PC = \{P_{ci} \mid i = 0, 1, \ldots \}$ must look alike. At the same time, their quantitative
characteristics must be the same. Based on this consideration, this section quantifies two point clouds of the same shape using the radius of curvature and aesthetic value, respectively.

3.1. Quantification Based on Radius of Curvature

Consider the parameter called a radius of curvature [73], as defined in Appendix B. A point in a point cloud has a radius of curvature, \( \rho_i \), \( i = 0, 1, \ldots \). To calculate \( \rho_i \), the point itself and two more preceding consecutive points are needed. The variability in \( \rho_i \) is calculated for circle, ellipse, and spiral, as shown in Figures 11–13.

Figure 11 shows two point clouds of the same shape, circle. Figure 11a shows the point cloud of the circle modeled by Equation (A1), where the radius, \( a = 100 \) and center, \( P_c = (150, 150) \). In this case, 49 points are created to model the circle. Figure 11b shows the values of \( \rho_i \), which is equal to 100.21. Theoretically, \( \rho_i \) must be equal to \( a = 100 \). This small error (100.21 – 100 = 0.21) is due to calculation precision error. On the other hand, Figure 11c shows the point cloud of the same circle modeled by a PCA-generated point cloud. For this point cloud, \( r_i \) plays the role of the radius of curvature of a circle.

Figure 12 shows two point clouds of the same shape, ellipse. Figure 12a shows the point cloud of the ellipse modeled by Equation (A2), where the major radius, \( a = 54 \); minor radius, \( b = 46 \); and center, \( P_c = (80, 80) \). In this case, 49 points are created to model the ellipse. Figure 12b shows the variation of \( \rho_i \). On the other hand, Figure 12c shows the point cloud of the same ellipse modeled by a PCA-generated point cloud. For this point cloud, \( d = 15; \varphi = 0; r_i \) is decreased linearly from 54 to 46 and increased linearly from 46 to 56 for two cycles; \( \theta_i = 2\pi i/n; n = 48; i = 0, 1, \ldots, 48; \) and \( P_c = (80, 80) \). In this case as well, 49 points are created to model the ellipse. Figure 12d shows the variation of \( \rho_i \). As seen in Figure 12a,c, it is clear that both point clouds are identical and their radius of curvature (Figure 12b,d) also varies in the same trend. This means that for an ellipse, \( r_i \) plays the role of the radius of curvature of an ellipse.

Figure 13 shows two point clouds of the same shape, spiral. Figure 13a shows the point cloud of the spiral modeled by Equation (A3), where an arbitrary, positive, real constant, \( a = 28.647 \) that controls the distance between the successive turnings of the spiral. In this case, 49 points are created to model the spiral. Figure 13b shows the variation of \( \rho_i \). On the other hand, Figure 13c shows the point cloud of the same spiral modeled by a PCA-generated point cloud. For this point cloud, \( d = 15; \varphi = 0; r_i \) is decreased linearly from 54 to 46 and increased linearly from 46 to 56 for two cycles; \( \theta_i = 2\pi i/n; n = 48; i = 0, 1, \ldots, 48; \) and \( P_c = (600, 600) \). In this case as well, 49 points are created to model the spiral. Figure 13d shows the variation of \( \rho_i \). As seen in Figure 13a,c, it is clear that both point clouds are identical and their radius of curvature (Figure 13b,d) also varies in the same trend. This means that for spiral, \( r_i \) plays the role of a radius of curvature of a spiral.

It is clear from Figures 11–13, the shape modeled by the parametric equation based and the PCA are visually identical. This means that the instantaneous distance plays the role of the radius of curvature when the shape is modeled by the PCA. The variation of the instantaneous distance is comparatively lower than the radius of curvature, which shows that the geometry of the shape can be easily controlled using the instantaneous distance compared to the radius of curvature. It is worth mentioning that there are several key elements for shape designing, and mathematical formulation of the radius of curvature or instantaneous distance can be one effective approach [74].
Figure 11. Circle shapes and their radii of curvature. (a) Point cloud of a circle modeled by a parametric equation; (b) radius of curvature of (a); (c) point cloud a circle modeled by the PCA; (d) radius of curvature of (c).

Figure 12. Ellipse shapes and their radii of curvature. (a) Point cloud of an ellipse modeled by a parametric equation; (b) radius of curvature of (a); (c) point cloud of an ellipse modeled by the PCA; (d) radius of curvature of (c).
3.2. Quantification Based on Aesthetic Value

This subsection describes the quantification of shapes based on a parameter called aesthetic value, denoted as $\alpha$ [75–77]. This parameter is defined in Appendix B. In order to know the aesthetic value, a plot called a logarithm curvature histogram (LCH) is used. The procedure to construct LCH for a given curve (in this case from a given point cloud) is also shown in Appendix B. For quantification of the aesthetic value, the ellipse and spiral shape are considered.

Figure 14a shows the point cloud of the ellipse modeled by Equation (A2), which is repeated from the previous section (Figure 12a). The corresponding LCH of the ellipse is shown in Figure 14b. Figure 14c shows the point cloud of the ellipse modeled by the PCA (repeated from Figure 12c). The corresponding LCH of the ellipse is shown in Figure 14d. Both point clouds have the same aesthetic value, $\alpha = 1.25$. On the other hand, Figure 15a shows the point cloud of a spiral modeled by Equation (A3) (repeated from Figure 13a). The corresponding LCH of the spiral is shown in Figure 15b. Figure 15c shows the point cloud of the spiral modeled by the PCA (repeated from Figure 13c). The corresponding LCH of the spiral is shown in Figure 14d. Both point clouds of the spiral shape have the same aesthetic value, $\alpha = 1.22$. As seen in Figures 14 and 15, even though the shapes are created using the PCA instead of the respective parametric equations, their aesthetic natures remain unaffected. It is also observed that the length of the line segment in the LCH depends on the variation range (minimum value to maximum value) of the radius of curvature. If the variation in the radius of curvature is high, the length of LCH becomes larger and vice versa.
The point clouds are used in a sequential order to produce a solid model of the object within a CAD package. Finally, a physical prototype is built from digital data of the solid model.

### 4. 3D Solid Modeling and Physical Prototyping

Recall the last question, as follows. Does the deterministically created point cloud help create 3D solid models of some given objects?

In order to answer this question, this section presents an illustrative example where a relatively complex object is first modeled by some point clouds. The point clouds are generated by the PCA. The point clouds are used in a sequential order to produce a solid model of the object within a CAD package. Finally, a physical prototype is built from digital data of the solid model.

---

Figure 14. Logarithm curvature histogram (LCH) of ellipses. (a) Point cloud of an ellipse modeled by a parametric-based equation; (b) LCH of (a); (c) point cloud of an ellipse modeled by the PCA; (d) LCH of (c).

Figure 15. Logarithm curvature histogram of spirals. (a) Point of a spiral modeled by a parametric-based equation; (b) LCH of (a); (c) point cloud of a spiral modeled by the PCA; (d) LCH of (c).
Figure 16a shows an ewer from Seto ware (Seto is a city located in Aichi Prefecture, Japan) produced during the Nanboku-chō period in the 13th century [78]. The ewer consists of three parts—the body, spout, and handle—as shown in Figure 16b. Few line segments and a concave as a reference, plane 3 is created and then the ellipse is constructed on plane 3, as shown in Figure 19c. On the mid-section of the outer and inner surface of the body, shown in Figure 19b. Taking PC as shown in Figure 18d. Afterward, a hollow is created inside the solid CAD model by off as reference points, plane 2 is created and then another ellipse is constructed on plane 2, as shown in Figure 18c. Similarly, taking PC, presenting an elliptical shape that can be modeled by customizing the inputs, θ, and r, as described in Section 2 (Figure 7). Regarding the point cloud PC, presenting the concave curve that can be modeled by setting the input parameters of the PCA as follows: PC = (90,90); d = 15; \( \varphi = 0; \theta_i = \pi/2n; n = 30; i = 0,1,\ldots,30; \) and \( r_i \) is linearly decreased from 73.55 to 0.

![Figure 16. An Ewer from Seto ware. (a) A photograph of the Ewer [78]; (b) an illustration of the ewer; (c) individual point clouds.](image)

The steps for solid CAD modeling are described as follows. In the first step, the individual point clouds PC, PC, PC, PC, and PC are sequentially synthesized by a C^0 continuity approach which represents the basic point cloud of the ewer body. The resulting point cloud is shown in Figure 17a. These point clouds are then transferred to a commercially available 3D CAD package for solid modeling. By revolving the point cloud 360°, a solid CAD model of the ewer body is constructed. The result is shown in Figure 17b.

In the second step, the point clouds PC, PC, and PC are sequentially synthesized by a C^0 continuity approach which represents the basics point cloud of the spout. The resulting point cloud is shown in Figure 18a. The point clouds are then injected into the solid CAD model of the body as shown in Figure 18b. Here, P_1 and P_2 represent the start point and endpoint of PC and are connected to the endpoint of PC and start point of PC, respectively. For the other points, P_3 denotes the endpoint of PC and P_4 denotes the starting point of PC. The point clouds must be iterated so that P_3 and P_4 must lie on a tangent to the surface of the body. Taking P_1 and P_2 as reference points, plane 1 is created and then an ellipse is constructed on plane 1, as shown in Figure 18c. Similarly, taking P_3 and P_4 as reference points, plane 2 is created and then another ellipse is constructed on plane 2, as shown in Figure 18c. Now, these two ellipses are joined by applying the lofted function to form the spout, as shown in Figure 18d. Afterward, a hollow is created inside the solid CAD model by offsetting surface and hiding the solid body and applying the thicken function on the offset surface. The resulting solid CAD model is shown in Figure 18e.

In the third step, the point cloud PC, shown in Figure 19a, is injected on the solid CAD model, as shown in Figure 19b. In this case, PC must iterate so the start point P_5, and endpoint P_6, must lie on the mid-section of the outer and inner surface of the body, shown in Figure 19b. Taking P_5 and PC as a reference, plane 3 is created and then the ellipse is constructed on plane 3, as shown in Figure 19c. Then, applying the sweep function on the ellipse and taking PC as a sweep path, a handle is created. The resulting solid CAD model is shown in Figure 19d.
Finally, the rendered solid CAD model of the ewer is shown in Figure 20a. Once the solid CAD modeling is done, triangulation modeling is performed simply by saving the solid CAD model into an STL data file using a CAD package. Figure 20b shows the triangulation model of the ewer. The STL data generated from the triangulation modeling can be used to build a physical model using commercially available 3D printing devices (e.g., 3D printer). Figure 20b shows one of the ewer physical models built by a 3D printer available at the Kitami Institute of Technology.

![Figure 17](image1.png)

**Figure 17.** Solid CAD modeling of the ewer body. (a) Basic point cloud of the body; (b) solid CAD model constructed using the basic point cloud.

![Figure 18](image2.png)

**Figure 18.** Solid CAD modeling of the ewer spout. (a) Basic point cloud of the spout; (b) point cloud of the spout injected into the solid CAD model; (c) reference planes and ellipses to construct spout; (d) result of spout construction; (e) a solid CAD model with hollow inside.

![Figure 19](image3.png)

**Figure 19.** Solid CAD modeling of the ewer handle. (a) Point cloud of the handle; (b) point cloud of handle injected into the solid CAD model; (c) reference plane and ellipse to construct handle; (d) result of handle construction.
This section provides some insights into this type of surface texturing.

Figure 20. Ewer model. (a) Rendered solid CAD model; (b) triangulation model; (c) 3D printed model.

5. Discussions

The above three sections answer the questions coined in the introduction. Apart from this, the presented methodology can be used in many different ways. One of the ways is surface texturing. This section provides some insights into this type of surface texturing.

The PCA-based point cloud can be used for surface texturing. For example, consider the solid model of the ewer shown in Figure 20. A texture can be added to it. In this case, the PCA-based point cloud can be used. For example, consider the point clouds that represent a pattern taken from Ainu cultural heritage. The settings for creating the point clouds are available in [69].

Figure 21a shows the point cloud of the texture. This point cloud is injected into the solid CAD model of the ewer, as shown in Figure 21b. Then, the texture on the front side is constructed by extruding the point cloud from the surface of the ewer. The texture on the backside is constructed by mirroring the front side texture. The result is shown in Figure 21c. Finally, the physical model is realized using a 3D printer, as shown in Figure 21d.

Figure 21. Surface texturing. (a) Point cloud of surface texture [69]; (b) point cloud of surface texture injected into the solid CAD model of ewer; (c) solid CAD model of the ewer with surface texture; (d) 3D printed physical model of the ewer with surface texture.

In addition to texturing, the 3D printed physical model of ewer shown in Figure 21 is scanned using a 3D scanner available at the Kitami Institute of Technology. Three different orientations are implemented to scan the ewer, as shown in Figure 22. Point clouds are then preprocessed (removing noise points, removing redundant points, and combining all point clouds into a single coordinate) and a triangulation model is created directly from the point cloud. The triangulation model is shown in Figure 22. Afterwards, a solid CAD model is constructed from the triangulation model. As seen
in Figure 23a, the texture is a blur and there is also surface distortion near the handle. To make the texture sharp, clear, and have a smooth surface, it requires the user efforts while performing solid CAD modeling. It is worth mentioning that the triangulation model from the 3D scanned point cloud (Figure 23) consists of 271,803 facets and 815,409 vertices, whereas the triangulation model from the PCA-based point cloud has 25,336 facets and 76,008 vertices for the model without texture (Figure 20b); and 51,244 facets and 153,732 vertices for the model with texture (Figure 21d). It is clear that the 3D scanned point cloud has 10 times higher facets than the PCA-based point cloud. As a result, the 3D scan-based approach becomes user-dependent and requires heavy computation. Table 2 shows a summary of the number of points, facets, vertices, and printing time required to manufacture the physical model of the given ewer.

<table>
<thead>
<tr>
<th>Scanning orientations</th>
<th>Point clouds</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rotating platform</td>
<td></td>
</tr>
<tr>
<td>Laser</td>
<td></td>
</tr>
<tr>
<td>Camera</td>
<td></td>
</tr>
</tbody>
</table>

Figure 22. Point cloud registration from different orientations.

Figure 23. Triangulation model obtained from the 3D scan point cloud data set. (a) Front view; (b) side view; (c) top view; (d) bottom view.

The article has demonstrated geometric modeling of a given shape which is completely controlled by a few analytically generated points and free from computational complexity. This way, the user
has the freedom to design any given shape without being dependent on the data acquisition system. Moreover, the point clouds are used in a sequential order to produce the solid CAD model of the given object.

Table 2. Summary of results for different models.

<table>
<thead>
<tr>
<th>Models</th>
<th>Model without Texture</th>
<th>Model with Texture</th>
<th>3D Scanned Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of points</td>
<td>109</td>
<td>789</td>
<td>52,1404</td>
</tr>
<tr>
<td>Number of facets</td>
<td>25,336</td>
<td>51,244</td>
<td>271,803</td>
</tr>
<tr>
<td>Number of vertices</td>
<td>76,008</td>
<td>153,732</td>
<td>815,409</td>
</tr>
<tr>
<td>Printing time without support</td>
<td>7 h, 50 min, 38 s</td>
<td>8 h, 6 min, 58 s</td>
<td>-</td>
</tr>
<tr>
<td>Printing time with support</td>
<td>12 h, 20 min, 28 s</td>
<td>13 h, 14 min, 56 s</td>
<td>-</td>
</tr>
</tbody>
</table>

6. Concluding Remarks

In this article, the characteristics and roles of the two variables (instantaneous distance and instantaneous rotational angle) and three constants (center point, initial angle, and initial distance) used in the recursive process that deterministically creates a point cloud are studied. It is found that varying the pair of the instantaneous distance and instantaneous rotational angle in different combinations can create many different shapes. For examples, in this study, a straight-line, circle, ellipses, spiral, astroid, S-shape, and leaf-shape are created using five different combinations of instantaneous distance and instantaneous rotational angle. The first constant (center point) can perform the function of translation in both x- and y-axis directions. The second constant (initial angle) can perform the function of rotation of shape around the center point. However, the third constant does not affect the shape to be modeled; it is only necessary for placing the starting point.

A planner shape can be modeled by a recursive-process-created point cloud and by a parametric-equation-created point cloud. If both point clouds are the same in terms of some quantitative measures, then it can be said that the recursive-process-based shape modeling approach is as good as the parametric-equation-based shape modeling approach. This is verified using three well-known shapes, namely, circle, ellipse, and spiral. In the verification process, the parameters called the radius of curvature and aesthetic value are used. It is observed that both point clouds of a given shape (one generated by the recursive process and the other generated by the parametric equation) have the same radius of curvature and aesthetic value. It is also found that the instantaneous distance plays the role of a radius of curvature. This means that it is not difficult to predict the values of the instantaneous distance compared to those of the radius of curvature while modeling an arbitrary shape.

In addition, an illustrative example that shows the efficacy of the recursive process in virtual and physical prototyping of a relatively complex 3D object is presented. The results show that, compared to a 3D-scanner-created point cloud, the recursive-process-created point cloud helps avoid computational complexity in reverse engineering. It also makes reverse engineering more systematic of complex shape. Moreover, the recursive-process-created point cloud can be used to perform surface texturing.

In the future, this study can be extended to develop a system that automatically generates a shape based on the user-defined inputs (instantaneous distance and rotational angle, and alike). This way, human cognition-based reverse engineering can be achieved—which is one of the frontiers of digital manufacturing.

Author Contributions: Tashi, A.M.M.S.U., and A.K. conceived the idea. Tashi and A.M.M.S.U. worked on the methodology and manuscript preparation. Tashi and A.K. conducted the scanning experiment in Section 5. Tashi and AM.M.S.U. produced the point cloud and solid CAD models. A.M.M.S.U. administered the work.

Acknowledgments: The authors acknowledge Kitami Institute of Technology for providing materials and financial support during the course of this study.

Conflicts of Interest: The authors declare no conflicts of interest.
Appendix A  Point Cloud Creation Algorithm

The PCA needs three constants, namely, the center point ($P_c$), initial distance ($d$), and the initial angle ($\phi$), that remain the same during the iterations. In addition, the PCA needs two variables—namely, the instantaneous rotational angle ($\theta_i$) and instantaneous distance ($r_i$), $i = 0, 1, \ldots, n$—that change in every iteration. The PCA consists of four steps. The first is input, the second is calculation, the third is iteration, and the last is output. In the input step, $P_c = (P_{cx}, P_{cy}) \in \mathbb{R}^2$; $d > 0$; $\phi \in \mathbb{R}$; $r_i \in \mathbb{R}$ for $i = 0, 1, \ldots, n$; and $\theta_i \in \mathbb{R}$ for $i = 0, 1, \ldots, n$ are defined. In the calculation step, an initial point $P_0 = (P_{0x}, P_{0y})$ is calculated first at distance $d$ from $P_c$ on line $P_cP_0$, making angle $\phi$ in the counterclockwise direction with the x-axis. In the iteration step, $P_0$ is rotated counterclockwise from the x-axis using $\theta_i$ to create points $P_i = (P_{ix}, P_{iy})$, $i = 0, 1, \ldots, n$. Afterward, points $P_i = (P_{ix}, P_{iy})$ are placed at a distance $r_i$ from $P_c$ resulting in points $P_{ei} = (P_{ex}, P_{ey})$, $i = 0, 1, \ldots, n$. In the output step, the points $P_{ei} = (P_{ex}, P_{ey})$, $i = 0, 1, \ldots, n$, are collected for creating the desired point cloud denoted as $PC = \{P_{ei} | i = 0, 1, \ldots, n\}$.

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Define: Center Point $P_c = (P_{cx}, P_{cy}) \in \mathbb{R}^2$, Initial Length $d &gt; 0$, Initial Angle $\phi \in \mathbb{R}$, Instantaneous Distance ($r_i \in \mathbb{R}$ for $i = 0, 1, \ldots, n$), Instantaneous Rotational Angle ($\theta_i \in \mathbb{R}$ for $i = 0, 1, \ldots, n$)</td>
</tr>
<tr>
<td>2</td>
<td>Calculate: $P_0 = (P_{0x}, P_{0y})$ so that $P_{0x} = P_{cx} + d \cos \phi$ and $P_{0y} = P_{cy} + d \sin \phi$</td>
</tr>
<tr>
<td>3</td>
<td>Iterate: For $i = 0, 1, \ldots, n$, rotate $P_0$ by an angle $\theta_i$ around $P_c$ in the counterclockwise direction to create $P_i = (P_{ix}, P_{iy})$ so that $P_{ix} = P_{cx} + (P_{0x} - P_{cx}) \cos \theta_i - (P_{0y} - P_{cy}) \sin \theta_i$, $P_{iy} = P_{cy} + (P_{0x} - P_{cx}) \sin \theta_i + (P_{0y} - P_{cy}) \cos \theta_i$. Extend $P_i$ to $P_{ei}$ (a point on the line $P_cP_i$ at a distance $r_i$ from $P_c$) so that $P_{ex} = P_{cx} + (P_{ix} - P_{cx}) \frac{r_i}{d}$, $P_{ey} = P_{cy} + (P_{iy} - P_{cy}) \frac{r_i}{d}$</td>
</tr>
<tr>
<td>4</td>
<td>Output: $PC = {P_{ei}</td>
</tr>
</tbody>
</table>

Appendix B  Definition of Relevant Shapes and Parameters

B.1. Relevant Shapes

Let the point clouds of a circle, ellipse, and spiral be modeled by three parametric equations. Each shape is given by a point cloud consisting of the points $P_i = \{(P_{xi}, P_{yi}) | i = 0, 1, \ldots, n\}$. The mathematical formulations of the shapes are as follows.

Consider a circle with radius $a$, where $a > 0$. Let $P_c = (P_{cx}, P_{cy})$ be the center of the circle on the x-y plane. An arbitrary point on the circumference of the circle is denoted as $P_i = \{(P_{xi}, P_{yi}) | i = 0, 1, \ldots, n\}$ and is given by

$$
P_{xi} = P_{cx} + a \cos \theta_i, \quad P_{yi} = P_{cy} + a \sin \theta_i. \quad (A1)
$$

In Equation (A1), $\theta_i = 2\pi i/n$. While creating a point cloud for a circle using Equation (A1), the parameters are set as follows: $a = 100$; $n = 48$; and $i = 0, 1, \ldots, 48$. The point cloud shown in Figure 11a presenting the circle is created using the abovementioned formulation.
Consider an ellipse with major radius $a$ and minor radius $b$, where $a, b > 0$ and $a > b$. Let $P_c = (P_{cx}, P_{cy})$ be the center of the ellipse in the $x$-$y$ plane. An arbitrary point on the circumference of the ellipse is denoted as $P_i = \{(x_i, y_i) | i = 0, 1, \ldots, n\}$ and is given by

$$P_{xi} = P_{cx} + a \cos \theta_i, \quad P_{yi} = P_{cy} + b \sin \theta_i. \quad (A2)$$

In Equation (A2), $\theta_i = \frac{2\pi i}{n}$. While creating a point cloud for an ellipse using Equation (A2), the parameters are set as follows: $a = 54; b = 46; n = 48; i = 0, 1, \ldots, 48$. The point cloud shown in Figure 12a presenting the ellipse is created using the abovementioned formulation.

Consider a spiral. Let it be an Archimedean spiral. Let $P_c = (P_{cx}, P_{cy})$ be the center of the spiral in the $x$-$y$ plane. Consider $a$ to be the arbitrary positive real constant that controls the distance between the successive turnings of the spiral. An arbitrary point on the circumference of the spiral is denoted as $P_i = \{(x_i, y_i) | i = 0, 1, \ldots, n\}$ and is given by

$$P_{xi} = P_{cx} + a \theta_i \cos \theta_i, \quad P_{yi} = P_{cy} + a \theta_i \sin \theta_i. \quad (A3)$$

In Equation (A3), $\theta_i = \frac{6\pi i}{n}$. While creating a point cloud for a spiral using Equation (A3), the parameters are set as follows: $a = 28.647; n = 48; i = 0, 1, \ldots, 48$. The point cloud shown in Figure 13a presenting the spiral is created using the abovementioned formulation.

B.2. Radius of Curvature

Let $\rho_i$ be the instantaneous radius of curvature for point $P_i = \{(x_i, y_i) | i = 0, 1, \ldots, n\}$. Thus, the following relationship holds [73]:

$$\rho_i = \left(\frac{(\dot{x}_i)^2 + (\dot{y}_i)^2}{(\ddot{x}_i)(\ddot{y}_i) - (\dot{y}_i)(\ddot{x}_i)}\right)^{3/2}. \quad (A4)$$

In Equation (A4), $\dot{x}_i = P_{x,i+1} - P_{xi}; \dot{y}_i = P_{y,i+1} - P_{yi}; \ddot{x}_i = \dddot{x}_i - \dot{x}_i; \dddot{y}_i = \dddot{y}_i - \dot{y}_i$. The radii of curvature for the shapes described above are computed using Equation (A4).

B.3. Aesthetic Value

A parameter called aesthetic value, denoted as $\alpha$, quantifies the aesthetic nature of a curve [75–77], which is driven from the radius of curvature (Equation A4) and the arc length of a given curve, as shown in Figure A1. As seen in Figure A1a, $\rho, \Delta s$, and $\Delta \rho$ are the radius of curvature, arc length, and change in the radius of curvature, respectively. The logarithmic plot of $\Delta s/(\Delta \rho/\rho)$ versus $\rho$, which is called Logarithmic Curvature Histogram (LCH), becomes a straight line, and its slope is used to calculate the value of $\alpha$, as seen in Figure A1b. Therefore, the following relationship holds [76]:

$$\log\left(\frac{\Delta s}{\Delta \rho/\rho}\right) = \alpha \log(\rho) + c. \quad (A5)$$

In Equation (A5), $c$ is a constant. By knowing the value of $\alpha$, the degree of aesthetics of a given curve (shape) can be quantified.
Figure A1. Determining aesthetic value. (a) Parameters for determining the aesthetic value; (b) calculating aesthetic value using LCH.

References
39. Furfari, R.; Governi, L.; Volpe, Y.; Puggelli, L.; Vanni, N.; Carfagni, M. From 2D to 2.5D i.e. from painting to tactile model. Graph. Models 2014, 76, 706–723. [CrossRef]
45. Yang, J.; Cao, Z.; Zhang, Q. A fast and robust local descriptor for 3D point cloud registration. *Inf. Sci.* 2016, 346–347, 163–179. [CrossRef]
46. Li, W.; Song, P. A modified ICP algorithm based on dynamic adjustment factor for registration of point cloud and CAD model. *Pattern Recognit. Lett.* 2015, 65, 88–94. [CrossRef]
51. Budak, I.; Vukelić, D.; Bračun, D.; Hodolić, J.; Soković, M. Pre-Processing of Point-Data from Contact and Optical 3D Digitization Sensors. *Sensors* 2012, 12, 1100–1126. [CrossRef]


© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).