3D surface generation using digital geometric approach

Soumi Dhar and Shyamosree Pal

National Institute of Technology Silchar

Abstract. This paper presents a simple number theoretic method for creating digital concentric and circular objects like potteries based on some effective digital geometric approaches. This paper introduces a novel algorithm to create digital 3D surface using certain simple but efficient digital geometric techniques. The aim of the work is to generate a 3D surface with the given a generatrix and an axis of revolution. The created digital surface is digitally connected and irreducible when the generatrix is an irreducible digital curve segment, which ensures no missing voxels in the generated surface. Many kinds of three dimensional surface can be generated successfully and efficiently using different generatrix. Our aim is to design an algorithm to generate points of a 3D surface, give a realistic image which include missing voxels to have final product ultimately as a smooth and real life like surface of an object.

Keywords: 3D Surface, Digital Circle, 3D objects

1 Introduction

The technique of representing any surface of a real 3D object mathematically using some specialized software is known as 3D surface generation. The outcome of the process is better known as a 3D model. 3D models generally depict a real life structure by the means of connected voxels, splines, Non-Uniform Rational B-Splines etc. Another way of creating such models is 3D printing [1]. 3D surface generation [2] gives us the opportunity to visually represent the exterior parts and contoures of an object. This objects can be components of a car engine, teeth, skin, bone structures etc. The prerequisite for developing a 3D surface is to define the shape and and exterior curve of the desired object. The most mind intriguing fact is that it provides the designer the ability to change the model accordingly, but the solid modeling is not capable of giving such relaxation. The wide range of interdisciplinary applications in real life has make us compelling to contribute in this area. Some of them are as follows:

- An algorithm for generation of a complete sphere that is both efficient and effective is expected to find numerous applications in 3D imaging, graphics and rapid prototyping.
- Improved quality of geometric primitives give rise to quality 3D models and improved quality of 3D printouts, even in situations where bitmap representation of voxel space is not feasible or is expensive. Presently, manufacturing

R. Murugan et al. (eds.), *Proceedings of the International Conference on Signal Processing and Computer Vision (SIPCOV 2023)*, Advances in Engineering Research 239,
https://doi.org/10.2991/978-94-6463-529-4_7

hollow spheres with conventional machining and rapid prototyping is quite difficult.
– However, an improved sphere discretization algorithm would make this process convenient and this would greatly impact manufacturing. For instance, magnetic ball valves are used in aeroplanes and automobiles, if these ball bearings can be made hollow, this would enable the valve to react quickly, which would translate to the vehicle becoming more fuel-efficient

2 Preliminaries

As mentioned earlier the geometric primitive used in this paper is the digital circle. There are several definitions of a digital circle available in the literature based on the fact that whether the center and radius of the circle have real or integer values. Here we are considering Bresenham's digital circle having integer values for center and radius and its construction is as proposed in [1]. If Z denote the set of non negative integers and Z^+ denote the set of positive integers such that radius $r \in Z^+$ and the center $S = O(0,0)$, then the first octant of the corresponding digital circle is given by eqn. 1

$$M_1^z(O,r) = \left\{ (p,q) \in Z^2 | 0 \leq i \leq j \leq r \wedge |j - \sqrt{r^2 - l^2}| < \frac{1}{2} \right\} \tag{1}$$

Hence, octant 1 of an arbitrary digital circle $M^z(P,r)$ having center at $P(i_P, j_P) \in Z^2$ and radius $r \in Z^+$ can be obtained by the following equation (2):

$$M^Z(P,r) = \{(p + p_P, q + q_P) | (p,q) \in M^Z(O,r)\}. \tag{2}$$

The digital circle possesses 8-octant symmetry [1], thus generating the set of points of one octant of the circle is sufficient to generate the complete circle by reflection about the respective axes of symmetry.

3 Related Works

The idea of mathematically representing a surface of any object whether inanimate or living, as well as replicating real life objects as 3D objects in the digital space has always proven to be a challenging task. The most important task of the process is to create as realistically accurate model as possible. By recreating the object it is meant that the object is replaced by a discrete set of its points using digital geometry [2]. Our problem statement of concern is as follows: finding a closed digital surface defined by a set of points in digital space such that they optimally approximate a real sphere with integer radius. Several works exist in the literature for sphere voxelation, however, they suffer from either the problem of theoretical discrepancies of underlying algorithms or are associated with high computational complexities in correcting the discrepancies in different applications. Even though voxelation techniques based on integer operations are comparatively faster and more efficient than the others, some of them do not

ensure completeness of the voxel set, giving rise to absentee voxels - this happens because they have been designed by extending the principle of Bresenham's circle drawing algorithm. Even though not much work has been done in the generation of Andres arithmetical circles [**3**] in the recent years, the circle's definition can be extended to to generate hole-free sphere covering all the points of the discrete space which is not possible with a digital circle [**1**] (the arithmetical sphere - which is free of 6-connected holes, is an immediate 3D extension of an arithmetical circle, however a sphere made using conventional digital circle has absentee voxels). But unlike the digital circle generation algorithm there are no such run length based efficient algorithms for arithmetic circle generation. In [**4**], the discrete sphere generation algorithm resorts to performing complex operations in real space. An algorithm to fill up the absentee voxels that arise on rotating a generatrix about its diameter has been proposed in [**5**] but a closed-form solution still remains unavailable and also, the discrepancy rectification here results in an increase in computational complexity. In [**6**], a wall-distance search algorithm based on voxelized marching spheres has recently been proposed. However, a sphere produced by this technique is not always mathematically accurate as it contains voxels with isothetic distance greater than $\frac{1}{2}$ from the real sphere Also, the algorithm does not make full use of the 48-symmetry property of a sphere. In [**7**], two number-theoretic approaches for sphere generation have been proposed, but voxels are generated in radii bands and the sphere obtained is not a thin sphere.

3.1 Motivation of the work

From the existing literature it can be concluded that, almost all existing methods for generating 3D surfaces gives user a straight control over the geometry of the 3D objects to be constructed, which is not a good practice as it becomes very difficult to represent the surface mathematically. Few major existing methods of representing 3D surfaces [8] are: Polyhedra representation, finite element method, cylindrical element method and circular sector element method. On the other hand, using our method one can represent the complete 3D surface in terms of a finite number of control points and the axis of rotation. In this paper, authors have used digital geometric approach which does not involve complex calculation like trigonometric functions which are costly in terms of time complexity.

4 Proposed Algorithm

In this section, the authors have proposed the algorithm that we have used to generate a 3D surface. The whole algorithm is divided into 2 sections:

4.1 Generation of a generatrix in 2D plane.

We are generating a cubic B-spline curve. The user provides the control points in an ordered sequence [9]. A cubic segment of the curve is generated taking four

consecutive points at a time. Suppose $L = L_1, L_2, L_3.....L_n$ is the set of control points. Four consecutive points, $S_i = L_i, L_i + 1, L_i + 2, L_i + 3$, where ($1 \leq$ i $\leq$ n-3) and a parameter u is passed to the algorithm where ($0 \leq$ u ≤ 1). Now, the points will be calculated according to the following equation 3:

$$f(x,y) = 1/6[(1 - 3u - 3u^2 - u^3)L_i + (3u^3 - 6u^2 + 4)L_{i+1} \\ (1 + 3u + 3u^2 - 3u^3)L_{i+2} + (u_3 L_{i+3})] \tag{3}$$

The value of u is incremented by 0.01 and we get a point in 2D plane for each value of u. The points are approximated to fit the discrete nature of digital space.

4.2 Generation of 3D Surface without any missing voxels

In this step the authors have used the above generated points of a curve to generate all the points which will lie on the 3D surface after the curve is revolved around the axis of revolution [10]. To generate the points we can pick each point from the above set and revolve it around the axis of revolution such that the axis is perpendicular to the plane of the circle formed by revolving the point around the axis of revolution. In order to generate points of this circle we have used modified digital circle algorithm(DCS), which generates all the points of this circle using number theoretic properties and digital symmetry of circle. Repeating the above method for each point in the set, we get all the points lying on the 3D surface. But this digital circle construction algorithm [1] suffers some problem. As we construct digital circles of consecutive radii, starting with radius r = 1, we face the problem of missing pixel generation. To cover these points the authors have given an algorithm in their paper that covers these points once the circle has been constructed, but in our case it would lead to complex calculations to handle and ultimately the execution time will also be increased. So we have to fill those missing pixels while constructing the digital circle itself so that an additional step for detection of missing voxels and then covering them with extra voxels could be avoided. Moreover it leads to lower the execution time. The occurence of those missing pixels follows a particular pattern, some certain observations are listed below:

- A missing pixel having coordinates (x, y) is generated between digital circles of two consecutive radius in octant 1, if and only if for the given ordinate value $y + 1$ for the digital circle with radius $r + 1$ the abscissa value x is maximum (i.e., the pixel $(x, y+1)$ is the last pixel in the run of pixel having the same ordinate value, $y+1$, in octant 1) and the pixel $(x-1, y)$ is the last pixel in the run of pixel having the same ordinate value, y, for the digital circle of radius r in octant 1.
- Also, it is evident that the 4-neighborhood of an missing pixel are either points from the digital circle of radius $r+1$, namely, $(x, y+1)$ and $(x+1, y)$ or from the digital circle of radius r for the same generating points, namely, $(x - 1, y)$ and $(x, y - 1)$.

Now, from the digital circle construction algorithm in [1]. As mentioned in Lemma 2, it is mathematically proven that $S^Z(P, r)$ with ordinate value $r - m$ always lies between $[U_m, V_m] = U_m + V_m - 1]$. DCS was designed using the concept of square numbers of $V_m(m > 0)$. It is clear that the points in the digital circle of a given radius r are included for a particular ordinate value only if the square of the abscissa falls in the range given by Lemma 1. Hence we perform a lookahead in each step of run generation for ordinate value j (corresponding to the m-th run) in the construction of the digital circle of radius r using the DCS algorithm for the first octant. We check by looking ahead whether the maximum abscissa value i for the ordinate value $(j+1)$ [corresponding to the $(m-1)$ th run] for the circle of radius $r+1$, about the same point as center, is one greater than the maximum abscissa value for the ordinate value j (corresponding to the m-th run) for the circle of radius r which we are presently constructing. Our approach guarentees that there would not be a single missing voxel in the generated 3D surface.

The algorithm $DCS\_without\_missing\_pixels()$ is almost the same as the DCS algorithm mentioned in [1]. The only difference being that we declare in the above mentioned procedure three additional variables $psum$, $csum$ and k with initial value 0, are required to keep track of the lookahead aspect for missing pixel finding. For each of the ordinate value y in the first octant we check whether the square number stored in s when it comes out of the inner **while** loop is $\leqslant w$. If it is so then it signifies that (x, y) is an missing pixel. This pixel and its 8-symmetric pixels are filled. At the end of the procedure in steps and $psum$, $csum$ and k are updated and made ready for the next value of y.

LookAhead table is simple to integrate with the existing digital circle algorithm. The algorithms shown above work together to generate 3D surface - a surface which is continuous and irreducible and does not have any missing voxel. After the generation of points all the points have been saved in a file.The points are further plotted using matplotlib library of python.
It is clear from the algorithm that only 2 LookAhead tables are required at any point of time. While generating points for circle C^Z_{r+1}, only the LookAhead table for circle C^Z_r is required. The LookAhead table for the circle C^Z_{r+1} is generated during the generation of the points for the circle C^Z_{r+1} itself. Searching for a point (x, y) in the LookAhead table is processed in O(1) time, so it does not add any major time quantum for execution of the algorithm.

5 Experiments and Results

The proposed algorithms are used to generate different 3D objects. A smooth surface was achieved by using a smooth curve as generatrix. In order to achieve a smooth surface, B-Spline curve is used to generate a smooth 2D curve. Few control points are taken as input by the user. With the help of these control

Y	X
7	0
7	1
7	2
6	3
6	4
5	5

Table 1. Coordinates of a digital circle(Octant 1) having radius 7

Y	X
6	0
6	1
6	2
5	3
4	4

Table 2. Coordinates of a digital circle(Octant 1) having radius 6

points, the points a 2D curve using B-Spline curve interpolation has been generated. The points of the 2D curve is stored to be used as generatrix. Now, at each point of the 2D curve we draw a digital circle using digital circle algorithm. The radius of this digital circle is the euclidean distance between the point on the 2D curve and the axis of rotation. This gives us our final 3D object. However, it is found out that, this method produces some of the unwanted missing voxels in our 3D surface In order to remove these missing voxels, we introduced the concept of LookAhead table.

Analysis on the the pattern of positions has been done, where missing voxels tend to appear and came up with an algorithm to overcome this problem of missing voxels.

X	Y
3	68
13	60
20	45
11	71
15	65
12	54
3	25
1	28
4	3
1	1
1	1

Table 3. Control Points for B-Spline curve generation

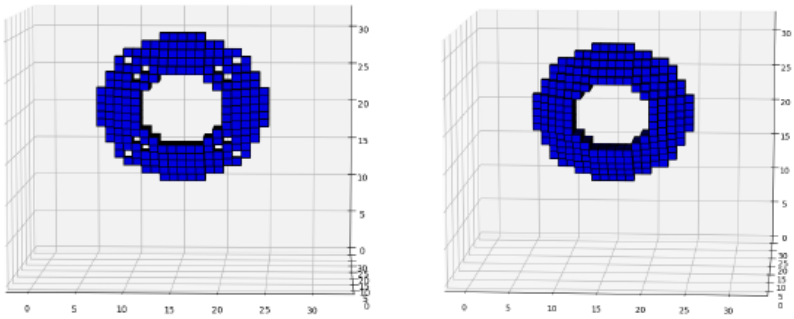

Fig. 1. Concetric digital circles (a) With missing Voxels and (b) Without missing voxels

5.1 Generated 3D surface

Motion of a generatrix in 3D space gives us a surface [11]. This is the key idea behind our implementation. If we revolve a curve in 2D plane along an axis of revolution which is passing through the plane of curve and is parallel to the plane then we get a connected 3D surface [12]. This idea has been used to generate a 3D surface from a generatrix.

Below the results of the various experiments we performed to generate different 3D shapes has been attached:

No. of control points	No. of points in B-Spline	Time taken(ms)
3	24	18
4	26	22
5	33	27
7	37	42
11	47	111
15	117	131

Table 4. Time for the generation of hole free 3D surface

6 Conclusions

This paper surveyed many algorithms [13] to generate and plot circles and 2D curves, for the generation of 3D surfaces, from different literatures. It has been observed that there are 2 different curves which can be used as generatrix namely Bezier curve and B-Spline curve. It is concluded that Bezier curves can be used to generate points of 3D surface but more smoothening can be achieved using B-spline curves [14]. Moreover B-splines provide more flexibility. Thus, we have used B-spline curves to generate a smooth 2D curve as generatrix which is further

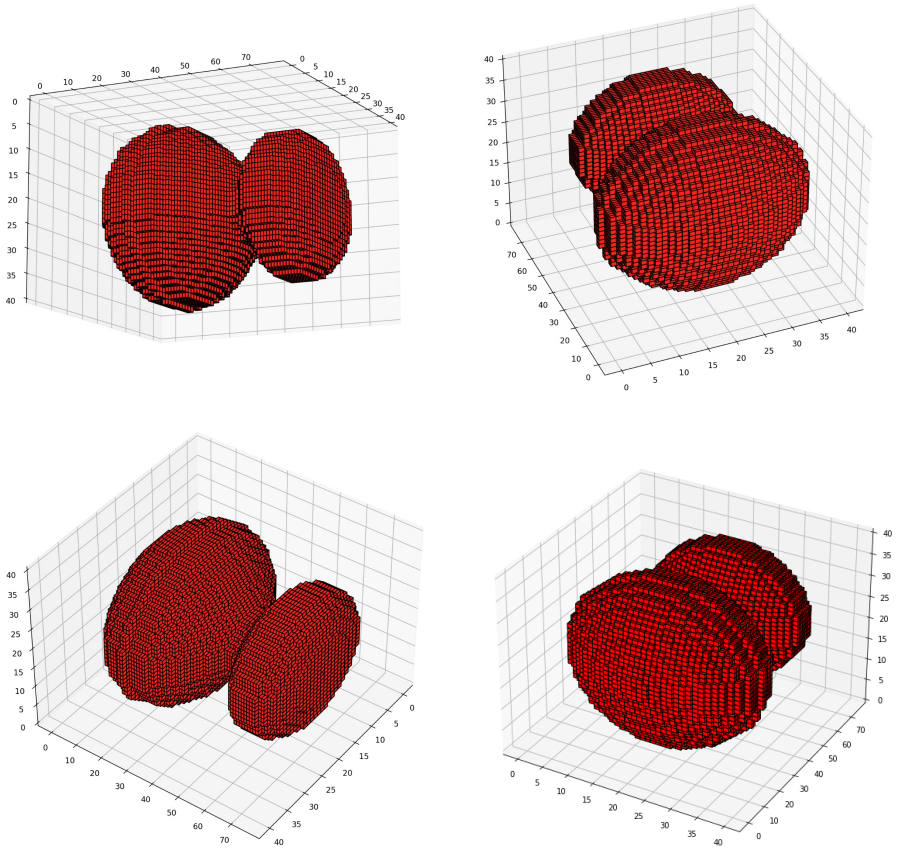

Fig. 2. Various views of a sample output object using our proposed methods

used to generate a smooth 3D surface. Different control points are taken as input and 2D curves are generated with these control points using B-Spline curve interpolation [15]. It has been analysed that the proposed algorithm integrated with an additional LookAhead table succesfully generates different 3D surfaces without any missing voxel. Some of the common shapes like frustum and bowling pin has been generated using this algorithm. Other than that generation of cubical, cuboidal and similar surfaces can be equally challenging topic.

References

1. P. Bhowmick, B. B. Bhattacharya, Number-theoretic interpretation and construction of a digital circle, Discrete Applied Mathematics 156 (12) (2008) 2381 – 2399. doi:https://doi.org/10.1016/j.dam.2007.10.022.

URL http://www.sciencedirect.com/science/article/pii/S0166218X07004817
2. J. Mukhopadhyay, P. P. Das, S. Chattopadhyay, P. Bhowmick, B. N. Chatterji, Digital Geometry in Image Processing, 1st Edition, Chapman & Hall/CRC, 2013.
3. E. Andres, L. Richaume, G. Largeteau-Skapin, Digital surface of revolution with hand-drawn generatrix, Journal of Mathematical Imaging and Vision (02 2017). doi:10.1007/s10851-017-0708-6.
4. E. Andres, Discrete linear objects in dimension n: The standard model, Graph. Models 65 (1-3) (2003) 92–111. doi:10.1016/S1524-0703(03)00004-3.
URL http://dx.doi.org/10.1016/S1524-0703(03)00004-3
5. P. K. Bhunre, P. Bhowmick, J. Mukherjee, On efficient computation of inter-simplex chebyshev distance for voxelization of 2-manifold surface, Inf. Sci. 499 (2019) 102–123. doi:10.1016/j.ins.2018.03.006.
URL https://doi.org/10.1016/j.ins.2018.03.006
6. O. R. Bingol, A. Krishnamurthy, Nurbs-python: An open-source object-oriented nurbs modeling framework in python, SoftwareX 9 (2019) 85–94. doi:https://doi.org/10.1016/j.softx.2018.12.005.
URL https://www.sciencedirect.com/science/article/pii/S2352711018301778
7. P. K. Bhunre, P. Bhowmick, Carve in, carve out: A bimodal carving through voxelization and functional partitioning, Vis. Comput. 34 (6-8) (2018) 1009–1019. doi:10.1007/s00371-018-1527-5.
URL https://doi.org/10.1007/s00371-018-1527-5
8. A. Rosenfeld, A. C. Kak, Digital Picture Processing: Volume 1 and 2, 2nd Edition, Computer Science and Applied Mathematics, Academic Press, Orlando, FL, 1982.
9. D. Salomon, Curves and surfaces for computer graphics, 2005.
10. P. Bhowmick, B. Bhattacharya, Real polygonal covers of digital discs - some theories and experiments., Fundam. Inform. 91 (2009) 487–505. doi:10.3233/FI-2009-0053.
11. E. Andres, G. Largeteau-Skapin, Conference slides "digital surfaces of revolution made simple"' (04 2016).
12. D.-J. Yoo, Three-dimensional surface reconstruction of human bone using a b-spline based interpolation approach, Computer-Aided Design 43 (8) (2011) 934–947. doi:https://doi.org/10.1016/j.cad.2011.03.002.
URL https://www.sciencedirect.com/science/article/pii/S0010448511000637
13. J. Foley, A. van Dam, S. Feiner, J. Hughes, Computer graphics (2nd ed. in c): principles and practice (01 1996).
14. I. K. Park, I. D. Yun, S. U. Lee, Constructing nurbs surface model from scattered and unorganized range data, in: Second International Conference on 3-D Digital Imaging and Modeling (Cat. No.PR00062), 1999, pp. 312–320. doi:10.1109/IM.1999.805361.
15. Y. Kineri, M. Wang, H. Lin, T. Maekawa, B-spline surface fitting by iterative geometric interpolation/approximation algorithms, Computer-Aided Design 44 (7) (2012) 697–708. doi:https://doi.org/10.1016/j.cad.2012.02.011.
URL https://www.sciencedirect.com/science/article/pii/S0010448512000528