Journal of Fiber Bioengineering and Informatics 8:2 (2015) 301–308
doi:10.3993/jfbim00128

A Parallel Interpolation Subdivision Scheme for Curve and Surface Design ⋆

Aihua Mao [a], Jie Luo [b,*], Jun Chen [a]

[a] *School of Computer Science & Engineering, South China University of Technology Guangzhou 510006, China*

[b] *School of Fine Art and Artistic Design, Guangzhou University, Guangzhou 510006, China*

Abstract

This paper proposes a new scheme of interpolation subdivision curve and surface using Bezier curve to achieve more realistic surface design. Different from the previous interpolation subdivision schemes, the normal vectors are used to generate a circle from a triangle which results in only three vertices and three edges. This improvement not only makes a curve smooth, but also produces sharp angles during subdivision. Examples of subdivision surface by this scheme are illustrated and compared to show the advantages of this scheme.

Keywords: Interpolation Subdivision; Normal Vectors; Bezier Curve; Parallel Computation

1 Introduction

In the field of 3D computer graphics, subdivision surface is used to represent a smooth surface via the specification of a coarser piecewise linear polygon mesh [1,2]. It can be broadly classified into two categories: approximating and interpolating [3,4]. Approximating schemes adjust the original position of vertices as needed. In general, approximating schemes have greater smoothness. The first interpolating scheme was presented by Dyn, Levin and Gregory (1990), which was also called butterfly scheme [5,6]. They extended the four-point interpolation subdivision scheme for curves to a subdivision scheme for surface. In general, interpolating subdivision schemes are sensitive to the presence of sharp features and may produce low quality surfaces for some input meshes [7]. In recently years, some new interpolating schemes have been proposed, such as the variational subdivision scheme [8] and normal-based subdivision scheme [9]. Based on the CatmullCClark subdivision scheme, Zheng and Cai proposed a two-phase subdivision scheme to interpolate arbitrary topology meshes [10]. A similar method was also applied to the DooCSabin subdivision

⋆This paper is financially supported by the Science and Technology Project of Guangdong Province (No. 2013B021600011), the Science and Technology Project of Guangzhou City (No. 2014J4100158) and the MOE (Ministry of Education in China) Project of Humanities and Social Sciences (No. 13YJC890027), and the Fundamental Research Funds for the Central Universities.

*Corresponding author.
Email address: jieluo@gzhu.edu.cn (Jie Luo).

scheme [11]. More recently, surface interpolation based on similarity [12] and an efficient constructive algorithm [13] are presented for interpolating meshes by Catmull-Clark surfaces. Chen and Luo [14] have used normal vector and a linear system equations to subdivide surface scheme.

Though NURBS can provide a smooth surface with some control points, it can't be used for an arbitrary mesh, such as a triangle mesh. We utilize the ideas from the NURBS that using the Bezier curve to generate point. Since the 4-point subdivision curve and deducing interpolating subdivision scheme in [15] are both difficult to generate a circle from a square or a triangle, we refresh the mind to propose a scheme to make a curve smooth and circinal, then extend this scheme to the subdivision surface scheme. Meanwhile, it is difficult for the proposed interpolation schemes in [5,10,15] to generate a sphere from a tetrahedron, which has only 4 vertices and 4 faces. In the existing schemes, many have much computational cost of solving a large system of linear equations [15], or change the topology of the mesh [4] or modify the data of the vertices [1–3]. They need extra space to store data or spend much time during computation.

In order to improve these disadvantages, this paper proposes a parallel normal-based subdivision scheme in the simple way for interpolation subdivision curve, and further expands it for surface design. Firstly, we propose an interpolation subdivision scheme, in which the input parameters are very simple with only the position and the normal vector of the two endpoints of the edge. Then, we use the normal vector to construct the Bezier controls, and subdivide the sub-edges recursively to get the limit subdivision curve. The examples of subdivision results of the well-know schemes and our scheme are presented to show the advantages of our method.

2 Circle Generation from Cubic Bezier Curve

Bezier curves are widely used in computer graphics to model smooth curves. As the curve is completely contained in the convex hull of its control vertices, the vertices can be graphically displayed and used to manipulate the curve intuitively. Any series of any 4 distinct vertices can be converted to a cubic Bezier curve that goes through all 4 vertices in order. Let $P_i = (x_i, y_i, z_i), (0 \le i \le 3)$ are the four control vertices, then the explicit matrix form is

$$P = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix}; \begin{cases} P_0 = (\sin(-\theta), \cos(-\theta)) \\ P_1 - P_0 = (\cos(\theta), \sin(\theta))L \\ P_3 = (\sin(\theta), \cos(\theta)) \\ P_3 - P_2 = (\cos(\theta), -\sin(\theta))L' \end{cases} \tag{1}$$

When $t = 0$, $P = P_0$, and $T = 3(P_1 - P_0)$, which is the beginning point P_0, and going toward P_1; When $t = 1$, $P = P_3$, and $T = 3(P_3 - P_2)$, which is the destination point P_3, and coming from the direction of P_2; When $t = 0.5$, $P = \frac{1}{8}(P_0 + 3P_1 + 3P_2 + P_3)$, and $T = \frac{3}{4}(P_2 + P_3 - P_0 - P_1)$, which is the middle point of this curve, and it is important to generate the new interpolate point and this point can split a edge into 2 sub-edges. Assume we are going to make the Bezier curve to be an arc of a unit circle and its parametric form is $P = (\sin(u), \cos(u)), u \in [-\theta, \theta], \theta \in (0, \frac{\pi}{2}]$, then the direction of its tangent is $T = \frac{dP}{du} = (\cos(u), -\sin(u))$. The endpoint P_i can be calculated with the form of θ.