SCIENCE CHINA
Information Sciences

• HIGHLIGHT •

July 2017, Vol. 60 078101:1–078101:3
doi: 10.1007/s11432-016-0311-8

B-spline surface fitting to mesh vertices

Yang LU[1,2], Junhai YONG[1,3,4], Kanle SHI[1,3], Hejin GU[1,5*], Ping ZHENG[6] & Jean-Claude PAUL[1,7]

[1]*School of Software, Tsinghua University, Beijing* 100084*, China;*
[2]*Department of Computer Science and Technology, Tsinghua University, Beijing* 100084*, China;*
[3]*Key Laboratory for Information System Security, Ministry of Education of China, Beijing* 100084*, China;*
[4]*Tsinghua National Laboratory for Information Science and Technology, Tsinghua University, Beijing* 100084*, China;*
[5]*Department of Research & Development, Jiangxi Academy of Science, Nanchang* 330096*, China;*
[6]*Department of Mathematics and Computer Science, Nanchang Normal University, Nanchang* 330032*, China;*
[7]*National Institute for Research in Computer Science and Automatic Control, Villers-les-Nancy* F-54600*, France*

Received May 12, 2016; accepted July 8, 2016; published online November 9, 2016

Citation Lu Y, Yong J H, Shi K L, et al. B-spline surface fitting to mesh vertices. Sci China Inf Sci, 2017, 60(7): 078101, doi: 10.1007/s11432-016-0311-8

Here we propose a novel method of fitting B-spline surfaces to input mesh vertices. For a mesh model that is difficult to fit with one B-spline surface, we divide it into several patches that form a tree structure (which is called hierarchical splines [1]), and for each patch, we use one B-spline surface for fitting. Normally, only a few patches are required, and G^2 continuity can be achieved between the adjacent surfaces.

Introduction. B-spline surface fitting is one of the common fundamental problems in computer aided design (CAD). It has been studied for decades. In this paper, we propose a novel method to construct B-spline surfaces from an input mesh and attempt to make the distance between the resulting surfaces and the mesh vertices as close as possible. The mesh is required to be topologically equivalent to a disc.

Since B-spline surfaces are a type of tensor product surface, it is difficult for them to efficiently express a surface shape that is much different from a plane. The existing fitting methods which use one surface patch for fitting (e.g., [2, 3]) often fail to fit the complex models well. Meanwhile, the other kind of methods which use multiple B-spline surfaces for fitting (e.g., [4, 5]) will divide the mesh into quadrilateral patches and generate a large number of surfaces (frequently more than one hundred). Reaching high geometric continuity for the adjacent surfaces is also difficult.

We present a novel method to deal with this problem. We also use multiple surfaces for fitting. However, the way we split the mesh is different from the traditional methods. The surface patches are not adjacent to each other, but rather constitute a tree structure (hierarchical splines). More precisely, we first construct a root surface to fit the major part of the model. Then, we construct one or more children surfaces, which are extracted and refined from the parent surface, to fit the remaining part of the model that is not handled yet. In this case, only a few surface patches are needed to fit the entire model, and handling the geometric continuity problem between the surfaces is convenient.

Our fitting method consists of three major steps, which are respectively described as follows.

Mesh division. First, we evaluate a parameter-

* Corresponding author (email: guhejin@vip.163.com)
The authors declare that they have no conflict of interest.

ization of the input mesh to help divide the mesh. We use circle-pattern conformal mapping [6] to compute the parameterization of the mesh, which supports flexible boundary conditions. We also compute the smallest-area enclosing rectangle [7] for the vertex parameters and normalize the parameter field into $[0, X] \times [0, Y]$ through rotating and scaling ($X = 1$ or $Y = 1$).

After parameterization, we divide the mesh progressively such that each mesh is suitable for fitting with one surface patch. The detail of mesh division is described as follows.

First, we digitize the parameter field into a 2D image of resolution $M \times N$. Then, each parameter (u, v) will be located at a pixel. For each pixel, we evaluate its vertex density by counting the number of vertices located in the square of $K \times K$ centered at its position.

Then, we detect whether the maximum density in the current image is higher than a given threshold T_1. If so, then we create a rectangle region that is initially located at the pixel of maximum density. The region is then expanded iteratively in its four directions until the mean vertex density on the four boundaries is lower than another threshold T_2. Then we construct a new mesh from the mesh elements in this rectangle region and remove it from the origin mesh. The above operation is repeated until the maximum density of the current mesh is no longer higher than T_1. Assuming the mean vertex density over the entire parameter field is T, we assign $T_1 = 60T$ and $T_2 = 10T$ in our experiment.

After new meshes are generated, we iteratively use the division method to detect whether these meshes meet our requirements. Finally, a set of meshes that constitute a tree structure can be obtained.

Constructing the outer surface. The fitting process proceeds stepwise from the root mesh to the leaf meshes, and we construct one B-spline surfaces for each mesh model. First, we construct an outer surface to fit the root mesh, and the fitting method for other meshes is introduced in the next section.

Two steps are required for constructing an outer B-spline surface to fit the root mesh. First, we estimate two suitable knot vectors for the u/v direction. The basic idea of our method is simple. On the basis of the definition of B-spline surfaces, in each internal knot span $[u_i, u_{i+1}]$, the same number of valid control points affects the surface shapes. So we estimate the knot vectors such that almost the same number of data points is presented in each internal knot span.

Second, we evaluate the control points of the surface by minimizing several energy functions. The first energy term E_1 measures the distance between the mesh vertices and surface points, which can be expressed as

$$E_1 = \sum_{\boldsymbol{v} \in \boldsymbol{V}} ||\boldsymbol{S}(u_{\boldsymbol{v}}, v_{\boldsymbol{v}}) - \boldsymbol{v}||^2,$$

where $\boldsymbol{S}(u, v)$ is the resulting surface, $\boldsymbol{V}$ denotes the vertex set, and $(u_{\boldsymbol{v}}, v_{\boldsymbol{v}})$ represents the parametric value for each vertex $\boldsymbol{v}$. The second term E_2 measures the smoothness of the surface shape, which is expressed as

$$E_2 = \iint (\alpha f_1 + \beta f_2) \mathrm{d}u\mathrm{d}v,$$
$$f_1 = ||\boldsymbol{S}_{uu}||^2 + 2||\boldsymbol{S}_{uv}||^2 + ||\boldsymbol{S}_{vv}||^2,$$
$$f_2 = ||\boldsymbol{S}_{uuu}||^2 + 3||\boldsymbol{S}_{uuv}||^2 + 3||\boldsymbol{S}_{uvv}||^2 + ||\boldsymbol{S}_{vvv}||^2.$$

The constants α and β are usually assigned as 10^{-8}–10^{-6}.

The traditional methods use the two energy terms above ($E_1 + E_2$) to evaluate the control points. However, we find that this process is often very unstable and generates unexpected shapes in practice. To stabilize this process, we construct a rectangle grid over the parameter field and attempt to constraint the surface position at each grid points, which results in the third energy term E_3 as follows:

$$E_3 = \sum_{i=0} \sum_{j=0} ||\boldsymbol{S}(a_i, b_j) - \boldsymbol{F}(a_i, b_j)||^2,$$

where $\{a_i\}$ and $\{b_j\}$ are the sampling values in the u/v direction, respectively, and the function $\boldsymbol{F}(*)$ returns the expected space positions for each parameter.

Finally, the total energy function is expressed as

$$E_{\text{total}} = E_1 + E_2 + E_3.$$

It is a quadratic form related to the control points and can be minimized through solving a linear system of equations. Then the control points are obtained.

Constructing the inner surfaces. After constructing an outer surface for the root mesh, we continue to construct inner surfaces for each non-root mesh. The geometric continuity problem has to be handled when constructing the inner surfaces. We construct the surfaces from the parent node to the children node, and for each inner surface, we need to consider the continuity issue between the surface and its parent.

Without loss of generality, consider a non-root mesh $\boldsymbol{T}_k$ and its corresponding surface $\boldsymbol{S}_k$. Assume its parent mesh is $\boldsymbol{T}_h$ and the corresponding surface $\boldsymbol{S}_h$. We recall that the mesh $\boldsymbol{T}_k$ is extracted

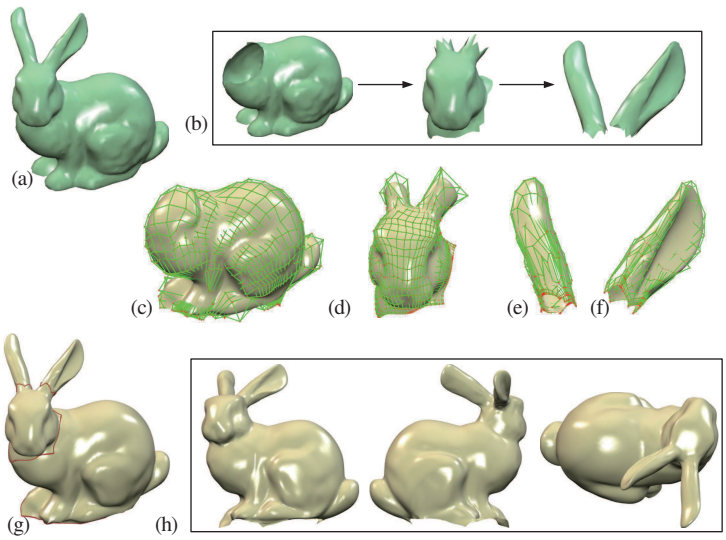

Figure 1 (Color online) An example of the bunny model. The original mesh is shown in (a), and the result of mesh division is shown in (b). (c)–(f) show the B-spline surface and its control points for each patch. The surfaces are combined in (g) and (h).

from a rectangle region in the parameter field of $\boldsymbol{T}_h$. Thus, we initialize the current surface $\boldsymbol{S}_k$ as the subsurface of $\boldsymbol{S}_h$ within that rectangle region. We can further modify the surface $\boldsymbol{S}_k$ to fit the mesh $\boldsymbol{T}_k$, while preserving its boundary conditions to ensure G^n continuity with the parent surface $\boldsymbol{S}_h$.

We first use knot refinement for the surface $\boldsymbol{S}_k$ to gain more control points, and then fit it to the mesh $\boldsymbol{T}_k$. We preserve the boundary conditions of $\boldsymbol{S}_k$ to ensure its continuity with the parent surface. Assuming G^g continuity is required (generally $g = 2$), then we preserve the first and last $(g + 1)$ rows and columns of the control points, and use the remaining control points for optimization. The energy function is similar with that of the outer surface.

Examples. Figure 1 shows the result of constructing the bunny model. G^2 continuity is required for the example, and the numbers of control points are assigned manually. The original mesh (a) has 2334 vertices. We divide the mesh into four patches (b), and use the B-spline surfaces to fit each patch. The number of control points for the B-spline surfaces (c)–(f) is 30×30, 33×33, 22×22, 20×20, respectively. The final shape of the combined surfaces is shown in (g) and (h).

Conclusion. We propose a method to construct B-spline surfaces from input mesh vertices. We divide the origin mesh progressively such that each mesh is suitable for fitting with one surface patch. Then, we construct an outer surface for the root mesh and construct inner surfaces for the other patches. G^2 (or higher) continuity can be achieved between the adjacent surfaces, and several examples show that our method can construct expected mesh shapes efficiently.

Acknowledgements The research was supported by International Science & Technology Cooperation Program of China (Grant No. 2013DFE13120). The second author was supported by National Natural Science Foundation of China (Grant No. 61272235). The third author was supported by National Natural Science Foundation of China (Grant No. 91515103). The fifth author was supported by National Natural Science Foundation of China (Grant No. 61562063).

References

1 Forsey D R, Bartels R H. Hierarchical B-spline refinement. ACM SIGGRAPH Comput Graph 1988, 22: 205–212
2 Lai Y K, Hu S M, Pottmann H. Surface fitting based on a feature sensitive parametrization. Comput-Aided Design, 2006, 38: 800–807
3 Bo P, Ling R, Wang W. A revisit to fitting parametric surfaces to point clouds. Comput Graph, 2012, 36: 534–540
4 Eck M, Hoppe H. Automatic reconstruction of B-spline surfaces of arbitrary topological type. In: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques. New York: ACM, 1996. 325–334
5 Yoshihara H, Yoshii T, Shibutani T, et al. Topologically robust B-spline surface reconstruction from point clouds using level set methods and iterative geometric fitting algorithms. Comput Aided Geom Design, 2012, 29: 422–434
6 Kharevych L, Springborn B, Schröder P. Discrete conformal mappings via circle patterns. ACM Trans Graph, 2006, 25: 412–438
7 Toussaint G T. Solving geometric problems with the rotating calipers. Proc IEEE Melecon, 1983, 83: A10