Robust NURBS Surface Fitting from Unorganized 3D Point Clouds for Infrastructure As-Built Modeling

Andrey Dimitrov[1] and Mani Golparvar-Fard[2]

[1] Graduate Student, Depts of Civil Eng and Engineering Mechanics, and Computer Science, Columbia University, New York, NY 10027; PH (713) 854-7744; e-mail: ad2895@columbia.edu
[2] Assistant Professor and NCSA Faculty Fellow, Departments of Civil and Env. Engineering, and Computer Science, Univ. of Illinois, Urbana-Champaign, Urbana, IL 61801, PH (217) 300-5226; FAX (217) 265-8039; e-mail: mgolpar@illinois.edu

ABSTRACT

The mapping of real-world objects to 3D geometry is of particular importance for engineering applications such as as-built modeling for progress monitoring and energy performance simulation. The state-of-the-art methods for fitting NURBS to point clouds still fail to account for all the topological variations or struggle with the mapping of physical space to parameter space given unordered, incomplete, and noisy point clouds. To address these limitations, we present a new method which starts by successively fitting uniform B-spline curves in 2D as planar cross sectional cuts on the surface. An intermediate B-Spline surface is then computed by globally optimizing the cross sections and lofting over the cross sections. This surface is used to parameterize the points and perform final refinement to a NURBS surface. Assuming an input of points that can be described by a single, non-self-intersecting NURBS surface, the algorithm outputs a NURBS surface. Experimental results on several real-world point clouds show the applicability of the proposed method for as-built modeling of civil infrastructure systems.

INTRODUCTION

The mapping of real-world objects to 3D geometry is of particular importance for many engineering applications such as as-built modeling for progress monitoring and energy performance simulation. This need has led to the development of a variety of capturing devices (e.g. structured light, structure from motion) that output 3D information in form of point cloud models. To become beneficial to Computer Aided Design (CAD), computational techniques can be used to translate point clouds to manageable geometric descriptions such as meshes, primitives, and parametric surfaces. Among these descriptions, Non-Uniform B-Spline (NURBS) curves and surfaces are of particular interest to engineering as they offer precise modeling control, descriptive flexibility (ability to construct conics), and a modeling logic that is in tune with fabrication considerations. Thus, NURBS have been widely adopted in CAD software (e.g., CATIA, Rhino). However, existing automated techniques to fit

NURBS to point clouds are still limited. This paper identifies and addresses these limitations by presenting a method for automatically fitting of NURBS surfaces to unorganized point clouds in the presence of noise and incomplete data.

PREVIOUS WORK

The task of fitting a B-Spline curve or surface to a set of points can be expressed as a linear optimization and solved for the control point positions if the 3D points can be associated with corresponding *u–v* parameter values (Peigl and Tiller 1997). Additional smoothness terms (Dierckx 1995) can be added to the optimization to produce aesthetically pleasing results in regions where no data is available or when the roughness of the point cloud needs to be attenuated. The main challenge thus lies in the proper parameterization of the input points.

Over the past decade, several research groups have proposed methods that can overcome the challenges associated with parameterization. For example, a simple surface can be used as an initial guess (Bo et al. 2012, Richtsfeld et al. 2012, Liu et al. 2006). The points can be projected onto it by Newton–Raphson steps in order to find their corresponding parameters. However, except for case where the final surface is somewhat planar –i.e., does not fold on it itself– the initial surface needs to be pre-defined by the user. Another alternative for somewhat planar surfaces is to automatically refine parameterization by progressive addition of control points (Weiss et al. 2002), or by additional feature sensitive optimization (Lai et al. 2006) given an initial state. Parameterization can also be obtained if it is assumed that the data is complete and the boundaries of the point cloud coincide with those of the fitted surface (Berhak 2001, Azariadis 2004). For topologies that form a loop in one direction, finding cross sections (Xie et al. 2012) can lead to proper parameterization, assuming the point cloud is complete. However, completeness is not a given in the general case as point clouds generated in the field (e.g., laser scan of an interior building scene, a part from a mechanical system/assembly) are often subject to occlusions and limited view points of the scanning device. To tackle the problem of incomplete data, more constrained approaches have been devised. For example, B-Splines constrained to developable surfaces (Peternell 2004) or sweep surfaces (Wu et al. 2012) can be fitted efficiently if assumptions about a ground plane can be made.

Overall, previous NURBS fitting methods form unorganized point clouds have the following limitations: (1) they cannot account, automatically, for all possible surface topologies; (2) often rely on user defined initial guesses; and (3) they struggle with imperfections in the input data, in particular lack of completeness.

NURBS SURFACE FITTING

Goal of the proposed method is to produce a reference B-Spline surface from an input point cloud. This surface can be used to parameterize the points and to perform the refinement to a NURBS surface through iterative methods. As in (Peigl and Tiller 1997), given control points $\{P_{i,j}\}$ and p-th degree basis functions $\{N_{k,p}(\cdot)\}$, and weights $\{w_{i,j}\}$ are the weights, B-spline and NURBS surfaces take the form:

$$S^{B-spline}(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u)N_{j,p}(v)P_{i,j}$$

$$S^{NURBS}(u,v) = \frac{\sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u)N_{j,p}(v)w_{i,j}P_{i,j}}{\sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u)N_{j,p}(v)w_{i,j}}$$

The addition of weights allows a NURBS to describe conics (e.g. circles, ellipses, spheres) but makes fitting non-linear. An initial parameterization by a B-spline facilitates this process. Assuming an input of points that can be described by a single, non-self-intersecting NURBS surface, the algorithm will output a NURBS surface (Figure 1). In this paper, our contribution is a new method with three novel algorithms to account for the incompleteness of the input data. These include: (1) finding cross sections through surface normal alignment; (2) curve extension and blending to keep partial cross sectional curves in a consistent state; and (3) cross sectional refinement through global optimization.

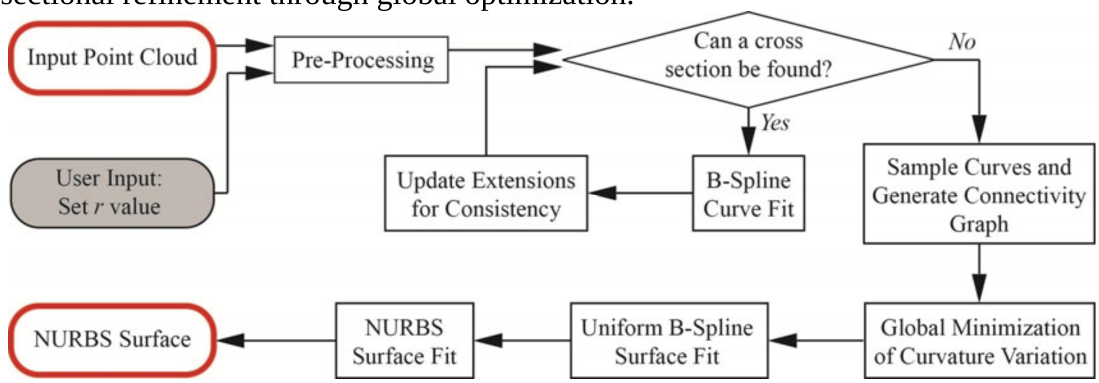

Figure 1. Process and Data Overview.

Pre-processing. We first perform several pre-processing steps. Point clouds from capture devices exhibit a certain degree of *roughness*. In general, roughness cannot simply be associated with noise from inaccurate sensing (which can be estimated) because real-world surfaces can be physically textured or have features that may or may not be significant depending on scale. Thus, the user defines r – a radius – which represents the level of detail for modeling, any roughness smaller that this value will tend to be smoothed out. Then, surface normal vectors (if not available as input) are extracted and oriented consistently using (Hoppe et al. 1992) a neighborhood of radius r around each point. Finally, the point cloud is then structured simultaneously as: (1) a k-d tree T by Euclidian distance; (2) a graph G where two points are connected by an edge if the distance between them is smaller than $c \times r$. In this paper, we use $c = 0.5$ for all experiments. If the point cloud originates from a segmentation procedure such as (Dimitrov and Golparvar-Fard 2014), these data structures are given as input.

Initialization. The initialization step selects a random seed point and finds a planar cross section (CS) passing through it, and fits a B-Spline curve in the 2D subspace defined by it. First, an energy function is defined to find the most appropriate plane. This function favors planes that cuts through points whose surface normal lies most closely on the plane. Given a set of points, the energy value of a candidate cross sectional plane ${n_{cs}}^T x - d_{cs} = 0$ where $x \in \mathbb{R}^3$ is given by

$$E(CS) = \frac{\sum_i (1 - |{n_{cs}}^T . n_i|) e^{-8r{d_i}^2}}{\sum_i e^{-8r{d_i}^2}}$$

where n_i is the unit normal vector of point p_i and d_i is its distance to the candidate CS plane. To reduce computation time, we sum only over points with distance to plane smaller than $2r$. From the seed point, a set of planes rotating around its estimated surface normal are considered and their energy values are computed. The plane with the highest energy is deemed the best (see Figure 2). If the energy value and the number of point used in its computation are higher than set thresholds, the CS is accepted, otherwise no CS can be found and the algorithm terminates.

Assuming a satisfactory cross section plane is found, all points within r distance from it are projected in the 2D subspace defined by it. This subset of points might form multiple connected components and might represent one or more open or closed curves. For the initial CS, we consider only the largest connected component.

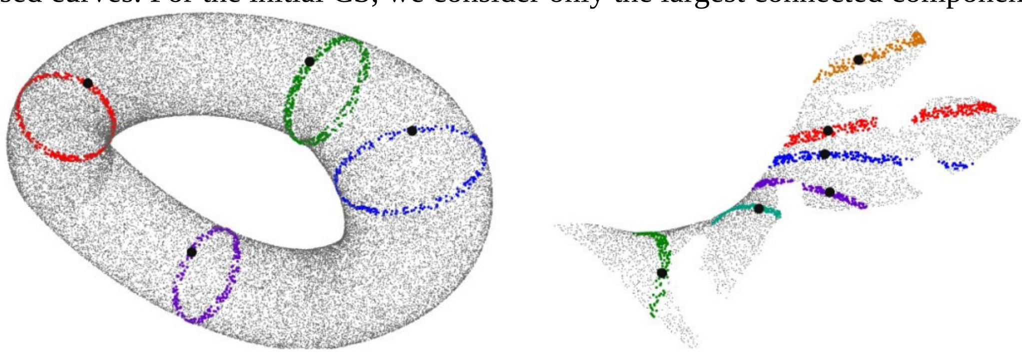

Figure 2. Seed points and the best cross sectional planes through them.

B-spline curve fit. A cross sectional curve is found by fitting a B-spline to 2D points in a given cross sectional plane subspace. In order to use the linear fit proposed in (Peigl and Tiller 1997), an initial parameterization is found by tracing a path between two edge points (Figure 3a-c). Each point is scored based on the largest open angle between its neighbors. The path finding procedure finds a minimum span path starting at the point with highest score and ending with the point with the second highest score. The second point is chosen from points not including immediate neighbors of the first point, to assure a minimum distance between the two. An open path is thus found and points associated with it are removed. If no points remain, the points did not form a loop and procedure terminates. Otherwise, another open path is found with the same end points and the two paths are joined into a loop (Figure 3d). The final path describes a polygon of straight lines (Figure 3a). To avoid unexpected spikes in the curve, the polygon is sampled uniformly (l_T/r; $l_T = total\ length$) along its length and interpolated (Figure 3b). The resulting B-spline is used to parameterize the points and fit the cross sectional curve (Figure 3c). If not closed, this curve is assumed to be partial and some degree of extension is added. Two possibilities are considered: (1) the curve might be part of a closed loop; in that case, we complete it by adding a second curve with $G2$ continuity; otherwise, (2) the ends are simply extended in a straight line. In our experiments, the extension length was set to the diagonal of the bounding box of the points. Finally, uniform samples from the curve are labeled as *confirmed* if they lie in the point cloud, and *potential* if they lie on the extended part of the curve.

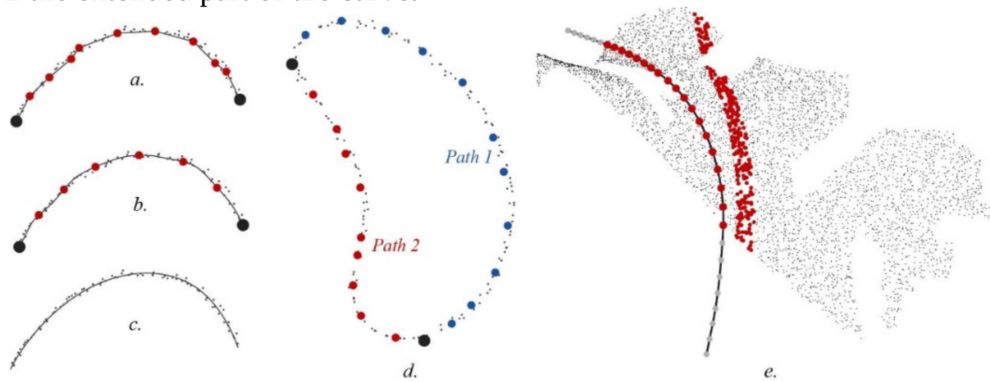

Figure 3. a-d) Curve fitting in 2D. e) Sampled curve with *potential* points shown in grey) and the next cross sectional plane.

Next plane. In the presence of one or more CS's, the energy function described above cannot be used freely, as it might find planes that intersect existing CS's. Thus, the user specifies an offset distance (for our experiments: $2r$) that guides the approximate spacing between CS's. The starting hypothesis is that the next CS plane will have the same normal direction as the previous one (the reference CS). To find the best normal direction for the new plane, the reference normal is rotated in respect to the starting position and the energy function is evaluated. The search is further constrained by bounding the rotation angles to planes that do not intersect any of the curve samples on the reference CS (see Figure 3e).

Update extension consistency. At each CS computation, an unknown collection of curve segments can be found. Segments are defined as connected components in the graph G. Without prior information, there is no way to know the best way to connect them, in part, because all segments might not belong to the same CS curve. In the case of a torus-like shape for example, a cut can produce two circular sections that do not belong together. To address this problem, we use a reference CS and project samples from its curve to the current CS. When a new CS curve is found, the sample points labeled as *confirmed* are matched to the reference samples. If they can be associated (closest point in reference samples) to a sample labeled as *potential*, this means that new information was introduced and needs to trickle down to the other CS's. Because the labeling in a CS has changed, we need to verify if additional segments need to be included in the curve. If no new segments are to be included, we do nothing, and only update labels. Otherwise, the curve needs to be refit with the additional segments. Figure 4 shows the progression and consistency at each step.

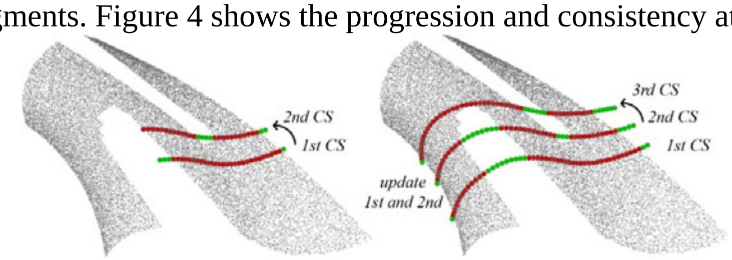

Figure 4. Progressive finding of cross sectional curves and extension updates.

Cross sectional optimization and NURBS fit. To coordinate between extended cross sectional curves, an optimization step is proposed. The goal is to minimize the variation of curvature across the sampled points and displace, in particular, parts of the curves that where labeled as *potential*. We sample the curves and generate a graph that links each sample point on each curve to its two adjacent samples on the same curve and the two corresponding samples on curves immediately adjacent to it using the following equation:

$$E_{Total} = \sum_i \sum_j |c_i - c_j| + \sum_i w_i |x_i - p_i|$$

where c_i is the curvature at point i (scalar), c_j is the curvature at point j that is a neighbor to point i. The second term minimizes the displacement of the points and gives more weight to *confirmed* points than *potential* points. We further constrain the displacement to the respective cross sectional plane. Figure 5a shows the initial curves with the *confirmed* sampled points shown in red. Figure 5b shows all sampled points after optimization. Additional cross sections can be interpolated (Figure 5c) for increased density. All sampled points can then be associated with a *u*–*v* parameter extracted from the *u* parameter and *v* order index of their respective B-spline curve.

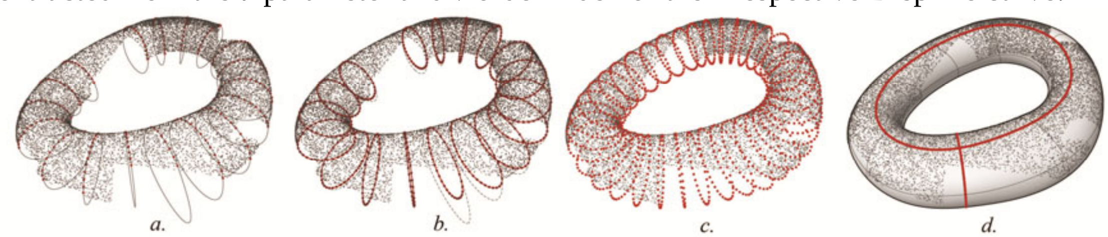

Figure 5. Cross sectional optimization and surface fitting.

A B-spline surface is fit to parameterized sampled points and becomes the base surface for parameterization of the entire point cloud. A final NURBS surface then can be fit by iterative methods, taking into account points in the original point cloud as well and the sample CS points to maintain stability in regions with no data.

EXPERIMENTAL RESULTS

The method was implemented in MATLAB and all experiments were performed on a PC with an Intel Core i7-2860QM CPU @ 2.50 GHz with 8GB of RAM. Figure 6 shows the results of NURBS fitting on a segmented point cloud from a building project for 3D meshing to be used for site verification purposes.

Figure 6. Segmentation and NURBS fitting of a scene from multiple laser scans.

Figure 7a (65k points, 17*sec* runtime) shows the fitting of a large NURBS surface on a parking garage floor registered from three laser scans. The results account for the variety of irregularities and slopes present in such an infrastructure project. Furthermore, the method can be used to generate additional data in regions where proper sensing was unavailable. Figure 7b (40k points, 15*sec* runtime) shows a twisted pipe segment scanned from one point of view. The B-spline surface used for parameterization (Figure 7c) completes the segment data. This surface can be used to generate a simplified model made up of cylindrical and tori sections for BIM.

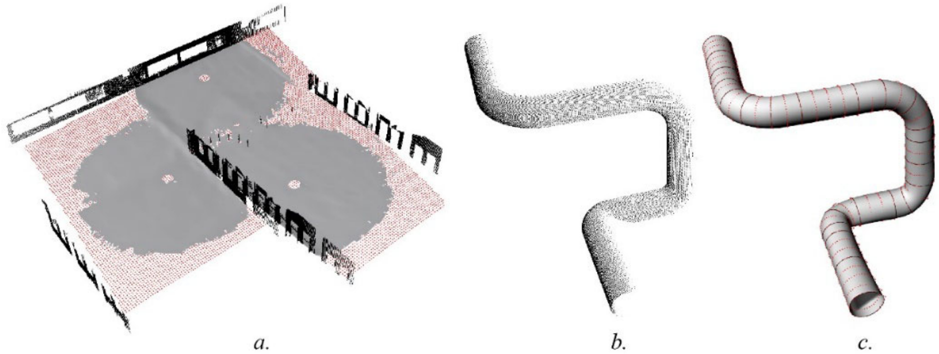

Figure 7. a) Parking garage floor NURBS fit and generated data. b-c) Scan and B-spline fit of a pipe segment.

The benefit of NURBS over primitives is their ability to precisely describe imperfect surfaces, which are often present in construction settings, with one object. In Figure 8a we show a NURBS fitting of a corridor floor segment. The segment appears flat and could be fitted with a plane, however a 10× exaggeration of the z-axis in Figure 8b reveals the subtle undulations and slopes that would have been missed by a single plane, or would have required multiple plane objects to describe.

CONCLUSIONS

In this paper we presented a new method for robust fitting on NURBS surfaces from unordered point clouds, accounting for noisy and incomplete input data. We further demonstrated the applicability of this approach for automated as-built modeling of civil infrastructure projects. Our future work focuses on further validating the method and parameters used on a range of Scan2BIM projects involving various types of civil infrastructure systems. Further results and data sets can be found at *http://raamac.cee.illinois.edu/NURBSfitting*.

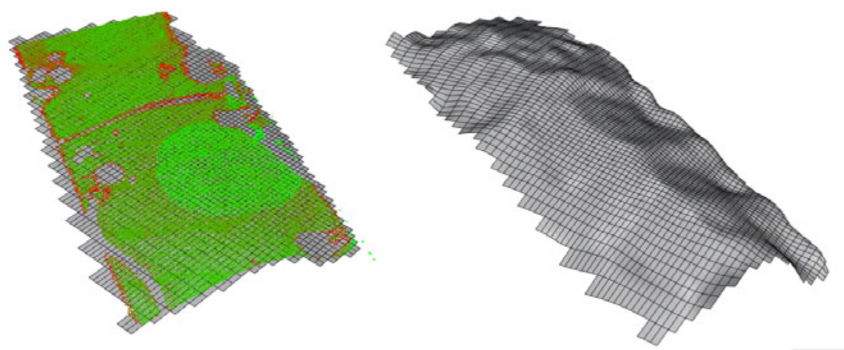

Figure 8. a) Corridor floor NURBS fit. B) Visualization of surface with 10x magnification of the z-axis.

REFERENCES

Azariadis, P. N. (2004). “Parametrization of clouds of unorganized points using dynamic base surfaces.” Computer-Aided Design, 36 (7), 607-623.

Berhak, J. (2001). “Parametrization and reconstruction from 3D scattered points based on neural network and PDE techniques.” IEEE Visualization and Computer Graphics 7 (1).

Bo, P., Ling, R., and Wanga, W.. (2012). “A revisit to fitting parametric surfaces to point clouds.” Computers &Graphics, 36, 534–540.

Dierckx, P. (1995). “Curve and Surface Fitting with Splines.” Clarendon Press, Oxford.

Dimitrov, A., and Golparvar-Fard, M. (2014). “Robust content-free segmentation for unordered 3D point cloud models.” Proc., Construction Research Congress (in press).

Hoppe H., DeRose, T., Duchamp, T., Mcdonald, J. and Stuetzle, W. (1992). “Surface reconstruction from unorganized points.” SIGGRAPH, 71–78.

Lai Y., Hu, S.-H. and Pottmann. H. (2006). “Surface fitting based on a feature sensitive parametrization.” Computer-Aided Design, 38(7):800-807.

Liu, Y., Pottmann, H. and Wang, W.. (2006). “Constrained 3D shape reconstruction using a combination of surface fitting and registration.” Computer-Aided Design, 38(6):572-583.

Peigl L., Tiller W. (1997). *The NURBS book*, 2nd Edition.

Peternell M. (2004). “Developable surface fitting to point clouds.” Computer Aided Geometric Design 21, 785–803.

Richtsfeld A., M¨orwald T., Prankl J. (2012). “Towards Scene Understanding – Object Segmentation Using RGBD-Images.” 17th Computer Vision Winter Workshop.

Wang, W., Pottmann, H., and Liu, Y. (2006). “Fitting B-Spline Curves to Point Clouds by Curvature-Based Squared Distance Minimization.” SIGGRAPH, 2 (2), 214–238.

Weiss V. Andor L., Renner G., Varady T. (2002). “Advanced surface fitting techniques.” Computer Aided Geometric Design, 19.1, 19–42.

Wu C., Agarwal S., Curless B., Seitz S. M. (2012). “Schematic Surface Reconstruction.” CVPR 2012, 1498–1505.

Xie, W. Zou, X. and Yang, J. and Yang, J. (2012). “Iteration and optimization scheme for the reconstruction of 3D surfaces based on non-uniform rational B-splines.” Computer-Aided Design, 44(11):1127-1140.