An Meta-heuristic Approach for Non Uniform Rational B-Spline Surface Reconstruction

Raazi Rizvi
Department of Electrical and
Computer Engineering
University of Ontario
Institute of Technology
Oshawa, Ontario L1H-7K4
Email: syed.rizvi@uoit.ca

***Abstract*—This paper outlines a novel technique used to convert un-organized point cloud data into a 3D surface. The technique relies heavily on upon Particle swarm optimization to fit a non-uniform rational B-spline (NURBS) surface to a set of data. The proposed approach is very robust and generalized, and as such does not require any prior knowledge other than the input point cloud data. The method outlined in this paper is robust, fast, and simple to implement.**

I. Introduction

3D reconstruction is computer vision/graphics problem in which an object is scanned by an image/video acquisition system. The resulting output is a set of un-organized data points with locations (x, y, z). This data then can be used to reconstruct a target object in the digital domain. Surface reconstruction comprises of generating/approximating a 3D surface from a set of data points (point cloud data). This approach is used in reverse engineering and CAD applications where a digital model for point cloud data is desired. Surface reconstruction is a growing field in computer graphics/vision. Research in surface reconstruction techniques will only increase with the increasing demand for digitization of real world objects. The primary application of surface reconstruction is in CAD and virtual reality model generation. Where a real world object is digitized into a 3D digital model.

II. Problem Overview

Generating a 3D model from point cloud data $(\mathbf{Q})$ is a challenging task. The data is un-organized and scattered. Before any surface generation technique can be applied, the data must first be structured or organized. Organizing the data refers to arranging the data according to its location such that a data point is surrounded by its nearest Cartesian neighbours in either a grid or mesh structure. After the data has been organized it must be parametrized and fit to a 3D generated surface $(\mathbf{S}(u, v))$. Parametrization allows the 3D data to be represented by two parameters (u, v) instead of conventional Cartesian coordinates (x, y, z). After the parametrization process the surface must be fit to the point cloud data with minimal error. The fitting process generates and fits a surface to the point cloud data. The surface is generated either from pre-existing models, or from the point cloud data itself. The actual fitting process is an optimization procedure selects the best model based on a reconstruction error metric (difference between the point cloud data and the generated surface model).

III. System Framework & Overview

Previous approaches have aimed to improve the reconstruction process by either improving parametrization, or by fitting more robust surface models to the data. Optimizing both procedures is key in order to achieve a good reconstruction. Graph based methods are often used to parametrize and organize the point cloud data and surface fitting is usually done by non-uniform b-spline (NURBS) surfaces. However optimal methods for both techniques can be quite computationally taxing. Andre Iglesias and Akemi Galvez proposed that instead of relying on prior knowledge, particle swarm optimisation (PSO), could be used to preform both parametrization and surface fitting [1]. This approach relies heavily on the black box nature of meta-heuristic algorithms to find a solution to the 3D reconstruction problem.

A. Particle Swarm Optimization

Particle swarm optimization (PSO) is a population based meta-heuristic optimization technique. Population based meta-heuristics use a randomly initialized population of solutions to search the solution space (space containing all possible solutions) in order to find either a global maximum or minimum. PSO in particular uses the concept of swarm intelligence (SI). The population moves through the solution space in a manner similar to that of a swarm. Each individual follows the changing gradient of its local neighbourhood, while maintaining a threshold distance to other population elements. Analogously fish in a fish school avoid predators by following the fish surrounding them. If a predator approaches the school, the fish are excited and move away from the predator, since the excitation moves throughout the swarm much like a ripple-effect, fish who don't see the predator still end up avoiding it by following their local neighbourhood. Similarly PSO randomly initializes a population and uses SI to move around the solution space. Each member of the population is evaluated against a fitness function, which scores the individual against a set of solution criterion (ie. distance from a predator). After all individuals in a population have been scored, the population best (pbest) is selected, and all members are assigned a specific velocity and have their position updated according to Equations (1 & 2). (for a more detailed explanation see [2])

$$V_i(k+1) = wV_i(k) \quad +c_1R_1(P_g^b(k) - P_i(k)) \quad (1)$$
$$+c_2R_2(P_i^b(k) - P_i(k))$$

$$P_i(k+1) = P_i(k) + V_i(k) \quad (2)$$

where V_i and P_i are the velocity and position of the population element, and c_1 & c_2 are learning factors, R_1 & R_2 are uniformly distributed random numbers in the range [0,1]. P_b^g & P_b^i are the pbest and gbest (gbest is simply the best of the population after k iterations whereas pbest is the population best in each iteration). In Equation (1) the velocity for each population element is calculated and in Equation (2) its position updated. Pseudo-code for PSO is shown in Algorithm (1).

Algorithm 1 Algorithm for Particle Swarm Optimization

```
Create swarm population
Start
$k \leftarrow 0$
initialize initial population randomly
evaluate the fitness of the initial population
while not termination condition do
    Calculate the best fitness particle $P_g^b$
    for each particle i in Pop(k) do
        Calculate particle position $P_g^i$ with best fitness
        Calculate velocity $V_i$ for particle i according to (1)
        while not feasible $P_i + V_i$ do
            Apply scale factor to $V_i$
        end while
        Update position $P_i$ according to (2)
    end for
    k = k + 1
end while
Finish
```

B. *Non-Uniform Rational B-Spline Surfaces*

Non-Uniform Rational B-Spline Surfaces (NURBS) are surfaces which are created from a series of B-Spline basis functions, and a bi-directional net of control points ($\mathbf{P_{i,j}}$). A NURBS curve is a curve which is formed a linear combination of series of control points ($\mathbf{P_i}$) and a set of B-spline basis functions over the range of some parametric value, u, as shown in Equations (5–7) (see [3]). Similarly a NURBS surface can be expressed as a rationalized function of two parameters u and v, as shown in Equation (8). A NURBS surface is dependant on three things: (1) control points ($\mathbf{P_{i,j}}$), (2) a knot vector (3), and a set of basis functions ($N_{i,k}\&N_{j,l}$). Control points, knot vectors, and basis functions are explained below:

1) Control Points: Control points are a series of points which lie near or on the NURBS curve/surface. These points form a skeleton from which a surface can be generated. In order to generate NURBS surfaces the coefficients of the B-spline basis function must multiplied by the coefficients of the control points (x, y, z) in order to produce generated surface points at the parameter values of u and v (ie. $\mathbf{S}_{x,y,z}(u, v) \rightarrow S(x, y, z)$). Thus the 3D control points are parametrized by two variables, namely u and v.

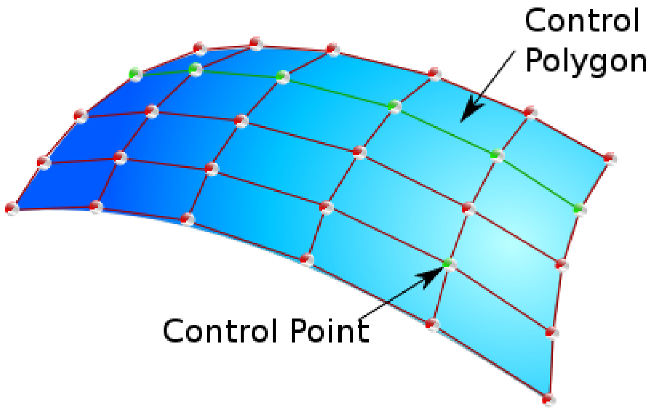

Fig. 1: An typical NURBS surface

2) Knot Vector: A knot vector comprises of r monotonically increasing values (knots) in the range [0,1] (Equation (3). These values work to create control polygons (commonly known as bi-linear patches) which restricting newly generated B-Spline basis functions to the control polygon. Practically the knot vector can be thought of as sequence which activates and deactivates specific control points in order to generate a NURBS surface a given control polygon (see Figure. **??**). In the term NURBS, non-uniform, refers to non-uniform knot vectors, which contains k (curve order) outer knots on either side with a values of 0 and 1 respectively, the inner knots , which are not linearly spaced Equation (4).

$$s = \{s_0, s_1, s_2, \ldots, s_{r-1}, s_r\} \quad (3)$$

$$s = \{s_0 = s_1 = s_k = 0, s_{k+1} = t_1, \ldots, \quad (4)$$
$$s_{r-k-1} = t_a, s_{r-k} = s_{r-1} = s_r = 1\}$$
$$t = 1, \ldots, a$$

3) B-Spline Basis Functions: B-Spline basis functions are the building blocks of NURBS curves since weighted linear combination of control points and B-spline basis functions forms the actual NURBS surface. The basis functions are polynomial functions of order k, or degree k-1 (depending on whether k starts from 0 or 1). A unique attribute of B-spline basis functions is that are dependant on previously constructed basis functions of the a lower order ($N_{i,k-1}\&N_{i+1,k-1}$). This property, often called local support, implies that a basis function of order k is constructed from a linear combination of its two nearest neighbours of order k-1. Equations (4 & 5) show the general form for B-spline basis functions.

B-Spline Basis Functions:

$$N_{i,1}(u) = \begin{cases} 1 & \text{if } s_i \leq u < s_{i+1} \\ 0 & \text{if } otherwise \end{cases} \quad (5)$$

$$N_{i,k}(u) = \frac{u - s_i}{s_{i+k-1} - s_i} N_{i,k-1}(u) + \frac{s_{i+k} - u}{s_{i+k} - s_{i+1}} N_{i+1,k-1}(u) \quad (6)$$

NURBS Curve:

$$C(u) = \sum_{i=0}^{m} w_i N_{i,k} \mathbf{P}_i \tag{7}$$

Rational B-Spline Basis Functions:

$$R_{i,j}(u,v) = \frac{w_{i,j} N_{i,k}(u) N_{i,l}(v)}{\sum_{i=1}^{m+1} \sum_{j=1}^{n+1} w_{i,j} N_{i,k}(u) N_{i,l}(v)}, \quad i = 0, \ldots, m; j = 0, \ldots, n \tag{8}$$

NURBS Surface:

$$\mathbf{S}(u,v) = \sum_{i=1}^{m+1} \sum_{j=1}^{n+1} \mathbf{P}_{i,j} R_{i,j}(u,v) \tag{9}$$

IV. PSO-NURBS Reconstruction

Although robust and efficient, the NURBS surface fitting technique has several serious limitations. Chief among these is the fact that the knot vectors and the control points, must be known for any object being reconstructed. Practically this is impossible because the points are un-organized and the knot vectors are unknown. However, instead of surface reconstruction, if the problem is changed to an approximation problem, then PSO can be applied to solve to the problem.

A. Initialization

The problem of surface approximation of point cloud data, as mentioned in previous sections, is a difficult one. The problem can be sub-divided into three smaller problems: (1) organization of control point data for parametrization, (2) generating surfaces according to the control points, (3) fitting the generated surfaces to the point cloud data. Each one of these problems is computationally intensive by conventional approaches. However, because of the meta-heuristic nature of PSO, it can solve all three sub problems at the same time. The reason for this is that by measuring the error of generated surface points $\mathbf{S}_{x,y,x}(u,v)$ against the point cloud data we can optimize for minimum error, thus should obtain the best reconstruction for out input point cloud data.

In order to implement PSO, an initial population must be generated, in accordance with the point cloud data, the sets of data which must be randomly generate/initialized are: (1) The control points, (2) The knot vectors, (3) The fitting parameters (u,v). The search space (length of each population element) is given by by (10)

$$4(m+1)(n+1) + (m-k+n-l+2) + 2(p+1)(q+1) \tag{10}$$

There are $(m+1)(n+1)$ control points each with four parameters (x,y,z,w) which are randomly sampled from the point cloud data $(\mathbf{Q})$. The knot vectors used in this approach can be represented by the values of the inner knots (see Equation(4)) and thus have a size of $(m-k+n-l+2)$ where $(m-k+1)$ represents the number of inner knots in a non-uniform non-periodic knot vector. Both surface parameters, $u\&v$, each generate a vector of size $(p+1)(q+1)$. These

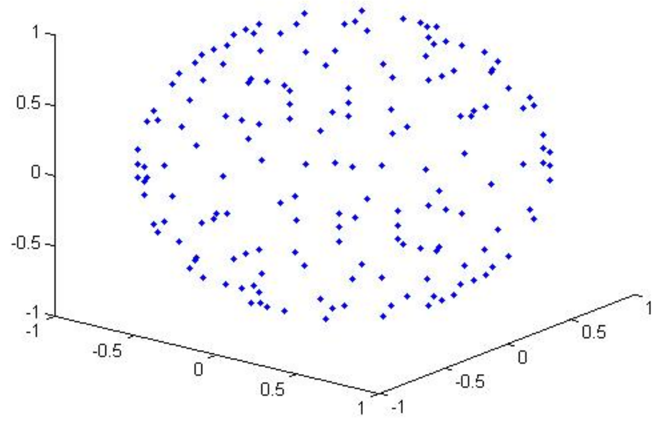

Fig. 2: Point Cloud Data to be fitted

parameters initialized and run through the PSO as shown below:

$$\begin{aligned}
\mathbf{P}_{x,y,x,w} &\rightarrow \mathbf{P}_{i,j} && (11)\\
t_{b,u} &= t_1 + \ldots + t_a, b = 0, \ldots, a && (12)\\
t_{d,v} &= t_1 + \ldots + t_b, d = 0, \ldots, c && (13)\\
u &= 0 : \frac{1}{p-1} : 1 && (14)\\
v &= 0 : \frac{1}{q-1} : 1 && (15)
\end{aligned}$$

where $P_{i,j}$ are a series of control points, t_b, t_d are knot vectors, u and v are bi-variate parameters. The internal knots for t_b and t_d are randomly generated from [0,1]. Similarly the weights for the control points are also randomly generated from the range of [0,2]. Even still the values for m, n, k, l, p, and q come from visual inspection of the point cloud data as shown in Figure (2). Where m+1 and n+1 is the size of the bi-directional net created from the control points, (p+1)(q+1) is the size of the point cloud data, and k and l are the curve orders.

B. Evaluation Criterion

The PSO randomly generates all relevant parameters to reconstruct a NURBS surface points, and compare them to the initial point cloud data. In this method each point on the generated surface is compared to each point cloud data point. The sum of all the errors gives the total error of reconstruction. As shown in Equation (16–18) the point cloud is first arranged into a matrix of size $(p+1)$x$(q+1)$ by stacking rows in the vector $(\mathbf{P})$ by splitting the point cloud data into $(q+1)$ groups of length $(p+1)$. Equation (18) simply re-parametrizes α and β into a single variable σ done by stacking the columns (in the uv matrix) to form a vector of length D $((p+1)(q+1))$. After parametrization a D points are sampled from the generated surface $(\mathbf{S})$ and compared to the point cloud data. The resulting expression gives the least square error between the point cloud data and the surface.

$$Elsq = \sum_{\alpha=0}^{p}\sum_{\beta=0}^{q}(\mathbf{Q}_{\alpha,\beta} - \mathbf{S}_(u_\alpha, v_\beta))^2 \tag{16}$$

$$Elsq = \sum_{\sigma=0}^{D}(\mathbf{Q}_\sigma - \mathbf{S}_(u_\sigma, v_\sigma))^2 \tag{17}$$

$$Elsq = \sum_{\sigma=0}^{D}\left(\mathbf{Q}_\sigma - \frac{\sum_{i=1}^{m+1}\sum_{j=1}^{n} w_\sigma \mathbf{P}_\sigma N_{i,k}(u_\sigma)N_{j,l}(v_\sigma)}{\sum_{i=1}^{m+1}\sum_{j=1}^{n} w_\sigma N_{i,k}(u_\sigma)N_{j,l}(v_\sigma)}\right)^2 \tag{18}$$

V. Experimental Results

The results from the surface reconstruction are shown in Figures. (3–7). Figure (3) shows a curve generated using the NURBS technique and Figure (5) shows the blending polynomial basis functions used to construct the curve. Note Due to some unresolvable errors in generation of the basis functions the curve does not follow the intended path set out by the control polygon however and returns to the origin. Thus making a closed curve, something which a non-uniform knot vector should prevent. This an erroneous result is due an error in the calculation of first basis function ($N_{1,k}/N_{1,l}$). As a result of this error the generated surfaces could not fit properly to the data for two primary reasons.

The first is the result of the basis functions giving an inaccurate output. Secondly the PSO was unable to organize the points properly because of the error plateauing at some value as a result of the errors in the basis functions. However there are some promising results for surface generation as shown in Figure (5). Although there is a very large error the surface does appear to be fitting in between the control points. For the generated sphere shown in Figure. (7), the reconstructed surface (shown in red) converges to the the middle of the sphere attempting to outline a circle. Again these errors can be attributed to incorrect calculation of the basis functions and / or in-correct organization of the point cloud data.

The errors for both the generated surfaces were staggeringly high as per the reasons mentioned above. However we can still see the effect the PSO as the error drastically falls from the first couple of generations as seen in Figures (6 & 8). Thus highlighting a characteristic property of the PSO as it will find the neighbourhood of the global solution very quickly.

VI. Conclusion

This paper attempted to utilize NURBS surfaces and PSO in order to create a surface reconstruction algorithm which could fit a NURBS surface to un-organized point cloud data. Although the results showed that the this particular surface reconstruction algorithm failed, it does provide insight into two key factors required for surface reconstruction. The first is organization of point cloud data into a mesh/grid, although this

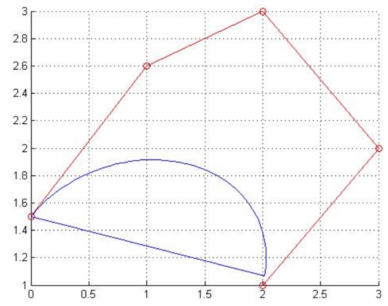

Fig. 3: A NURBS curve generated using the coded NURBS functions, curve is in blue, control polygon is in red (m=5,k=3)

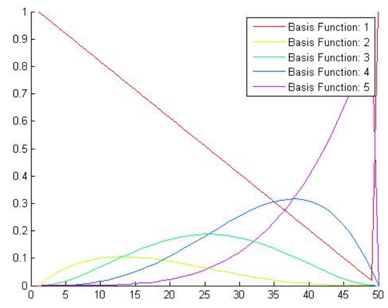

Fig. 4: Blending polynomials used to construct the NURBS curve in Figure. 3

can be handled by the PSO but requires very large population size. If the point cloud data is small it is more effective to use conventional techniques such as oct [4] or kd-tree [5] to organize the point cloud data. The second issue is the proper reconstruction of the NURBS surfaces. As the surface must be constructed in patches (control polygons) it is essential these patches are correctly computed otherwise the error will render the PSO to find the correct solution.

References

[1] Glvez, Akemi, and Andrs Iglesias. ”Particle swarm optimization for non-uniform rational B-spline surface reconstruction from clouds of 3D data points.” Information Sciences 192 (2012): 174–192.

[2] Kennedy, James, and Russell Eberhart. ”Particle swarm optimization.” Proceedings of IEEE international conference on neural networks. Vol. 4. No. 2. 1995.

[3] Rogers, David F. An introduction to NURBS: with historical perspective. Morgan Kaufmann, 2001.

[4] Lu, Mingyue, and Yongjian He. ”Organization and indexing method for 3D points cloud data.” Geo-Information Science 2 (2008): 013.

[5] Xu, Hui, Nathan Gossett, and Baoquan Chen. ”Knowledge and heuristic-based modeling of laser-scanned trees.” ACM Transactions on Graphics (TOG) 26.4 (2007): 19.

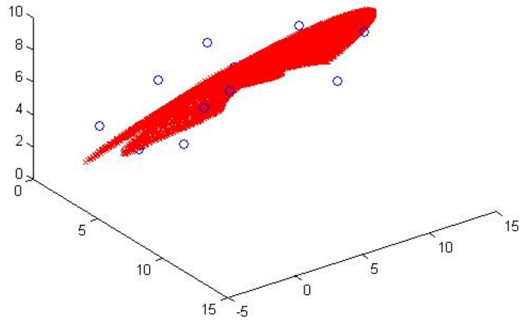

Fig. 5: A surface generated using the PSO NURBS reconstruction technique (red) and the point cloud data (blue) with parameters D = 16, k=l=3, and m=n=4. Note that the reconstructed surface has been sampled with 100000 points (red)

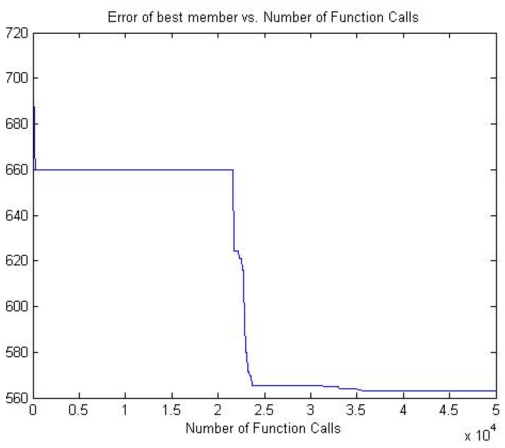

Fig. 6: The error of the PSO while generating the surface shown in Figure. **??**

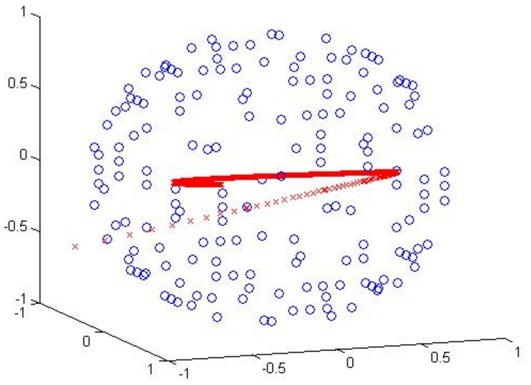

Fig. 7: A sphere generated using the PSO NURBS reconstruction technique (red) and the point cloud data (blue) with parameters D = 169, k=l=3, and m=n=4. Note that the reconstructed surface has been sampled with 100000 points (red)

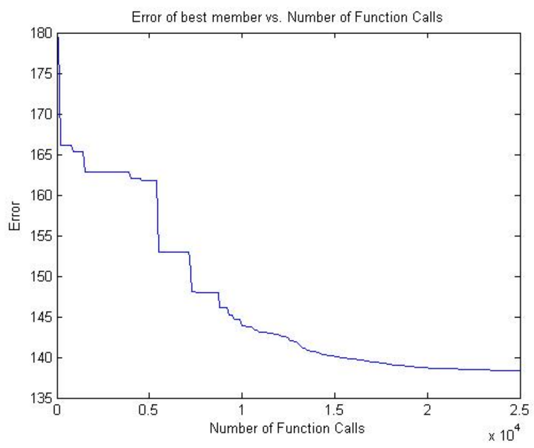

Fig. 8: The error of the PSO while generating the sphere shown in Figure. 7