Self-Organizing Approach To Moving Surface Reconstruction

MIKHAIL BESSMELTSEV
Novosibirsk State University
Mechanics and Mathematics Department
Pirogova st.,2, Novosibirsk, 630090
RUSSIA
bmpix@mail.ru

Abstract: In this paper we introduce a novel self-organizing method of moving surface reconstruction from the data obtained from measuring real-world objects.

The core of the approach is using Kohonen’s Self Organizing Maps model. As that model is not traditionally applied to mesh deformation or surface reconstruction, in the paper we propose its modification enforcing input data approximation and time-space smoothing.

The main idea is to generate the moving surface via deformation. Since the mesh is to be adapted only around areas of surface changes, we deform the existing mesh where it is necessary in order to fit the new point cloud corresponding to the surface. In place of processing all the sample points of unknown surface, we choose the points randomly and therefore it is possible to avoid the issues of oversampling and control the mesh quality in the case of undersampling. We can vary the mesh nodes density by picking the sampled points in the specified areas more often.

Due to stochastic nature of the proposed method, in many cases it is not necessary to employ additional denoising or data filtering. Moreover, while deforming a mesh, no user assistance is needed. Underlying self-organizing principles make the technique human-free, efficient and easy to parallelize.

Key–Words: animation, surface reconstruction,self-organizing maps

1 Introduction

Modeling of realistic three dimensional objects is one of the fundamental problems in geometry processing. Its applications include 3D computer graphics and movie making, industrial computer-aided design, numerical modeling of physical processes on real objects, acquiring CAD model of a part with lost CAD data, etc. [8]

Many approaches of 3D shape acquisition by measuring were developed, the most common of those are laser scanning [1], structured light techniques [2], and passive multi-view stereo [7]. The result of scanning is usually a point cloud or a height map. In this paper, we focus on one of the problems of reverse engineering of CAD models: moving surface reconstruction from point clouds.

Currently, numbers of methods exist that allow to reconstruct surfaces of static objects from unorganized set of points, particularly, methods of surface reconstruction of unknown topology, e.g. [8], methods of surface reconstruction via deformation, e.g. [3], methods of topology estimation [5], etc.

Recently, several approaches have been proposed to extend measuring techniques to capture animated scenes in real-time. That evidently opens up a large variety of interesting new applications, such as creating special effects for movies or creating content for interactive applications and games. Besides, new approaches for moving surface reconstruction appear, such as [17], [16], [11]. However, despite high research interest in the subject, currently available technologies of automatic moving surface reconstruction impose some significant restrictions. First of all, point clouds often contain remarkable level of noise, the input data suffer from oversampling or undersampling. Second, it is not evident how to evaluate the topology of an object being scanned and how it is changing in time. Third, in attempt to regenerate a moving surface using a static surface reconstruction method we may obtain not a moving mesh, but a sequence of meshes of varying connectivity. And finally, the methods of surface reconstruction are often difficult to parallelize whereas the complexity of those methods is a function of number of input points, which can seriously affect the reconstruction time in the case of oversampling.

Those are the problems this paper addresses. The main contribution of the paper is a new method ED-SOM (Expanding Dynamic SOM) which allows us to automatically reconstruct a moving surface with non-
uniform mesh nodes density, fixed topology and fixed number of nodes from unorganized set of points. Due to underlying self-organizing principles and stochastic nature, the new method is resistant to noise, can efficiently process oversampled or undersampled data and is easy to parallelize.

The core of the proposed method is Kohonen’s Self-Organizing Maps (SOM), a well-known neural network model [9]. In order to properly adjust the model to be applicable to the problem of moving surface reconstruction, the parameters of the model were fine-tuned. Also, some essential modifications are proposed which allow us to refine the surface approximation and make the quality of the final animation better. Particularly, we propose a stochastic technique which refines the mesh and decreases the gaps between the surface and the point cloud, moving the nodes in surface normal direction towards the point cloud and then additionally refining it with geodesic SOM [10]. To smooth trajectories, we use a stochastic K-means-like technique proposed in [4], which allows us to deform the mesh only in the areas of real surface deformations in time.

2 Related Work

We can divide the related work into two parts: moving surface reconstruction and surface reconstruction using self-organizing maps.

Moving surface reconstruction. Though many approaches were proposed to reconstruct surfaces from unorganized set of clouds, starting with [8], only a few techniques were developed to extend surface reconstruction to animated point clouds. Nevertheless, a full survey on the area is beyond the scope of the paper, and we will mention only some noticeable techniques.

A close strategy to ours is to generate moving meshes by fitting template meshes to the data [11], [16], [7]. The limitation of those methods is that those methods are able to work only with low-noise or noise-free data and sufficient sampling density. Those limitations are overcome by Wand et al. [17] at the cost of significant performance issues, though producing impressive results. Moreover, most of those methods cannot be efficiently parallelized.

Surface reconstruction using Self-Organizing Maps. One of the pioneer papers in the area is [18]. In this paper, static surface reconstruction using Kohonen’s SOM is considered. The algorithm proposed in that paper yields reasonable though non-optimized meshes: the number of triangles in the mesh tends to the number of points, what is unacceptable in most applications. Moreover, even with proposed technique called edge swap, the method possibly can not be applied to large model with small features. In this paper we solve this problem by employing a geodesic metric.

The idea of surface deformation by SOM is not new itself [6], [15]. Though the key difference of our approach that the initial mesh for deformation is irregular, unstructured and has non-uniform nodes density when most of the papers concerning SOM deal with regular uniform grids.

Finally, in [4], moving meshes are generated using Self-Organizing Maps. As stated in the paper, the technique is limited to surface without holes in it, the input data is a height map while in our paper we deal with unstructured sets of points. Though, our paper may be considered as a significant extension of that method.

3 The EDSOM reconstruction pipeline

3.1 Scheme

The proposed method of moving surface reconstruction belongs to the class of methods of moving mesh generation via deformation. The main goal of the deformation at each time step is to fit the template mesh to current point cloud, preserving structure and topology of the template mesh.

The initial mesh may be generated by any suitable method of static surface reconstruction, e.g. mentioned above. It has to be noted that the only condition we require is that the initial mesh has the same topology as the real object, because we use the initial mesh as a template one, generating moving mesh by its deformation.

Our moving surface reconstruction pipeline at each time step consists of three major components: estimation of changes in the point cloud in time (Section 4), deformation of the current mesh in the areas of point cloud changes with Self-Organizing Maps algorithm (Section 5), and the so-called Expanding - the final refinement of the mesh (Section 6).

4 Motion tracking

In order to track motion in the animated point cloud and deform the mesh only in the areas where the input data changes, we use a stochastic technique proposed in [4]. The main idea of the technique is to investigate changes of sample points near each node by its Voronoi cell centroid variation in time. Following this technique, we calculate the distance between each
node and the centroid of its Voronoi cell computed using the point cloud at the next time step. This distance actually shows how much the object under consideration has changed in the node’s neighborhood. The last step is to normalize this distance and perform Gaussian smoothing on it. The obtained factor \( v(t, x) \) is called the function of deformation area:

\[
v_c(t, q_i) = \begin{cases} \frac{w(t, q_i)}{\max_{i \in [1:N]} w(t, q_i)} , & \text{if } n_i(t) > 0 \\ 1 , & \text{if } n_i(t) = 0. \end{cases}
\]  

where \( w(t, q_i) \) is Euclidian distance between the centroids of Voronoi cells of \( i \)-th node at two adjacent time steps and \( n_i \) is the number of points in corresponding Voronoi cell. The obtained factor will be used as a mask for the deformation.

5 Deformation via SOM

The main goal of SOM deformation stage is to roughly fit the current mesh to a new point cloud, making the global structure of the mesh correspond the point cloud.

Let \( U(t) \) be the surface of an animated object in a 3D Euclidean space with spatial coordinates \( x \in \mathbb{R}^3_t \), where \( t \) is a discrete time step. A moving triangle mesh \( M(t) = \{m_1(t), ..., m_N(t)\} \) is to be generated on \( U(t) \), where \( m_i(t), i = 1, ..., N \) are moving mesh nodes. Let \( Q = M(0) \) be a fixed template mesh. The moving mesh will be acquired via deformation of that template mesh in time.

For simplicity, here we assume that the global topology of \( U(t) \) is not changing in time (i.e. no large holes appear or disappear in time in the object under consideration). This is actually a weak requirement since most of contemporary applications imply 3D modeling of a single moving object, e.g. actor in movie making.

The sets of sample points \( X(t) \) are assumed to lie on the unknown surface \( U(t) \) near it. We do not make assumptions about the level of noise in the input data since SOM deformation algorithm is stable even on data with high noise level. Though it is obvious that very high level of noise will affect the mesh quality, but the process is stochastic, so the influence of noise may be lower than in conventional algorithms of surface reconstruction.

Taking into account that we deform the mesh only in the areas of object variation (Stage 1), the SOM deformation algorithm is as follows:

**Algorithm 1.** Procedure Deform \((S_0, S)\)

For each \( s = S_0, ..., S \):

1. **Point selection.** Take a random point \( y \in X(t) \) from the current point cloud.
2. **Winner determination.** Calculate all the Euclidean distances between \( y \) and all the nodes \( m_i^s(t) \) and choose the node \( m_i^s(t) \) which is the closest to \( y \), i.e.\n
\[
\|y - m_i^s(t)\|_U \leq \|y - m_j^s(t)\|_U, \quad (2)
\]

for all \( i = 1, .., N \). The node \( m_i^s(t) \) is called a winner.

3. **Node coordinates correction.** Adjust locations of the mesh nodes according to the following rule:

\[
m_i^{s+1}(t) = m_i^s(t) + \theta^S_{q_i}(s, q_i)(y - m_i^s(t)), \quad (3)
\]

for \( i = 1, .., N \), where \( \theta^S_{q_i}(s, q_i) \in [0, 1] \) is a learning rate.

The learning rate controls the overall quality of the mesh and affects the preservation of mesh structure throughout the iterative process. According to [12] our experiments, the learning rate has been thoroughly selected to provide acceptable mesh quality with reasonable computational speed, and looks like \( \theta^S_{q_i}(s, q_i) = \delta(s) \chi_{r_{q_i}^S}(s, q_i) \), where \( \delta(s) = s^{-0.2} \chi(s) \) is a learning step, \( \chi_{r_{q_i}^S}(s, q_i) = \zeta \frac{r_{q_i}^S - r_{q_i}}{r_{q_i}} \) (\( \zeta \) is close to zero, e.g. \( \zeta = 0.001 \)) is a neighborhood function, \( \chi(s) = 1 - e^{(s - S)/S} \), \( r(S) = r(1) > r(S) \) is a learning radius.

The maximum number of iterations \( S \) is fixed beforehand proportional to \( N \), e.g. \( S = 10N \); \( r(1) \) and \( r(S) \) are values of a learning radius that are selected depending on the distances between nodes of the template mesh \( Q \), \( r(1) > r(S) \). The \( r(s) \) function is shown in Fig. 1.

The learning step indicates the displacement magnitude the winner node receives, and the learning radius controls the radius of the neighborhood of the winner node that moves along with the winner. Obviously, there is always a compromise between low and high learning radius. When the radius is high, a large neighborhood of the winner node moves, therefore producing smooth, but poorly approximating the point cloud surface. On the other hand, low learning radius may yield fine approximation, though possibly making mesh non-smooth and even disordering its structure.

In our experiments we have chosen the starting value of radius greater than maximum Euclidian distance between nodes of the mesh: \( r(1) = 3 \cdot \max_{i,j \in [1:N]} \|m_i - m_j\|_U \), and final radius value \( r(S) \) equal to an average edge length in the initial triangle mesh \( M(0) = Q \).
Following Kohonen [9], the learning process of the SOM model can be divided into 2 parts: ordering stage and refinement stage. During the ordering stage, mesh nodes get roughly distributed on the surface of the input geometry, since nodes get large displacements.

During the refinement stage, mesh nodes’ displacements are relatively low, and that makes the mesh correspond to the probability distribution and approximate surface more precisely. This stage affects the final overall quality of the mesh (smoothness, topology correctness, etc.).

To avoid regeneration of the mesh from scratch at each time step, only deforming it to fit a new point cloud, we skip the ordering stage, performing only last iterations of the SOM deformation process. In our experiments, we chose $s = \frac{4}{5}$ as the beginning iteration of the refinement stage, therefore using only 75% of whole iterative process.

The mesh after Stage 2 of reconstruction - SOM deformation. Points were sampled on an ape model, this is its left hand.

Figure 2: Hand of an gorilla: Stage 1 of EDSOM.

6 Expanding

In this section, we introduce Expanding technique which is aimed to refine the approximation of the mesh, obtained as the result of Stage 1. The goal of the technique is to eliminate a distance between the mesh and the point cloud or at least decrease it.

Formally, the problem we have left is expanding the mesh to fit the point cloud which is an optimization problem. Even though after stage 1 we have a good guess of the reconstructed mesh, in the paper we prefer to avoid explicit minimization of non-linear functions since it would lead to notorious efficiency and parallelization problems, especially in the case of oversampling.

It has to be noted that the mesh after Stage 1 of reconstruction approximates the geometry of the surface and the distance between the mesh and the point cloud is greater in the areas where the surface is complex, i.e. has larger surface curvature. Taking that into account, the proposed additional refinement technique goes as follows:

**Step 1.** Move all the nodes of the mesh in the normal direction. The displacement value is proportional to surface curvature in node’s position:

$$m'_i(t) = m_i(t) + n_i(t) \cdot \xi(i, t),$$  \hspace{1cm} (4)

where $n_i(t)$ is normal of the surface in the $i$-th node. Since normal is a differential entity, it is not defined at the vertices of a mesh, instead of it we use pseudo-normal [19] which is angle-weighted sum of all the triangles adjacent to the node. $\xi(i, t)$ is a factor which is proportional to surface curvature. In our experiments we have chosen $\xi(i, t)$ proportional to maximum angle between neighborhood triangles’ normals.

**Step 2.** Perform last iterations of SOM deformation process with a special learning radius: instead of calculating Euclidian distance, we can obtain better refinement using geodesic distance over the triangle mesh between the nodes. It is vital to use the geodesic distance only in the end of iterative process since then the learning radius is low and we can easily optimize the mesh deformation without need to employ complex algorithms of real geodesic distance calculations. It is also important to underline that Step 1 is essential: last iterations of SOM with geodesic distance would be unable to refine the mesh. Also, due to computation reasons, it is inefficient to use whole SOM with geodesic distance.

In our experiments, instead of exact geodesic distance, we used a graph-based Dijkstra algorithm for shortest paths.
7 Experiments and Results

The proposed method was implemented in C++ and tested on synthetic data - random points generated on the surface of a moving gorilla model created in a commercial 3d modelling package. The initial mesh is the triangulation of the gorilla at the first frame. As shown in (Fig. 4), the mesh successfully adapts to the animated point cloud, the mesh quality is satisfactory and trajectories of the mesh nodes are smooth. Moreover, the surface has holes in it (mouth, eyes) and the method successfully tracks their motion. The input data is shown in (Fig. 5). The parameters of the proposed method were: $S = 20 \cdot N$, $S_0 = \frac{S}{2}$ for SOM deformation, $S = 10 \cdot N$, $r(1) = \max_{i,j \in [1:N]} ||m_i - m_j||/8$, $r(S) = 3.0$, $S_0 = \frac{S}{2}$ for geodesic SOM. For Stage 3 we used $\xi(i, t) = \sqrt{\rho(i, t)} \cdot 1.5$, where $\rho(i, t)$ is the maximum angle between the $i$-th node’s normal and normals of its neighbors.

Computation time on a single core Pentium-4 3.4GHz with 2GB of main memory for mesh with 23152 triangles and animated point cloud consisting of 20 frames, 155000 points per frame is 420 seconds per frame. The proposed algorithm has been also parallelized using MPI library and the shown efficiency of parallelization is above 95 percent.

8 Conclusions and future work

We have presented a novel self-organizing method of moving surface reconstruction from an animated point cloud based on self-organization. The method is human free, resistant to noise or oversampling, topologically correct and efficiently parallelizable. It is able to reconstruct complex geometry with holes from noisy animated point cloud. Instead of regenerating the surface at each time step, the method only deforms the existing mesh in order to fit a new point cloud. The proposed method was successfully parallelized.

The distribution we use for point selection step in Algorithm 1 influences the final nodes density. If the goal is to preserve the initial mesh nodes density, we can choose uniformly among all the sampled points. In the other case, to make the animated mesh track the model features appearing in time, we generate a random point from the point cloud the more often the more complex the point cloud in that area is. So one of the possible future directions is to carry out experiments with non-uniform density.

One of the limitations of method is that sometimes the gap between the reconstructed surface and the point cloud is still noticeable, this is the problem we are aiming at in future.

Also we are going to improve the quality of obtained meshes since sometimes the obtained mesh is not smooth enough.
The sample unorganized points at two time steps: $t = 2$ (grey), $t = 13$ (blue).

Figure 5: The input data, noise level = 1%

References:


