The seventh International Conference on Neural Networks and Artificial Intelligence
October 10-12, 2012, Minsk, Belarus

Computational Intelligence for Surface Modeling

Francesco Bellocchio, N. Alberto Borghese, Stefano Ferrari, and Vincenzo Piuri
Department of Computer Science, Università degli Studi di Milano, Milan, Italy
{francesco.bellocchio, alberto.borghese, stefano.ferrari, vincenzo.piuri}@unimi.it

***Abstract* - *The surface reconstruction problem, which consists in the search of the surface that best describes a given set of points, is of interest in many application fields (e.g., design, archeology, medicine, and entertainment). This can be viewed as a supervised learning problem, where the vector coordinates (or other features) of each point is an input instance, while a further coordinate is an output label. The approximation function provides a law to obtain labels from instances.*

Several effective computational intelligence paradigms have been developed for solving the surface reconstruction problem, e.g., Multi-layer Perceptron Networks, Radial Basis Function (RBF) Networks, Self-Organizing Maps (SOM), and Support Vector Machines (SVM). However, other paradigms such as Genetic Algorithms has been used to improve the performances of traditional approaches of surface reconstruction. In general, the performance of a single paradigm depends on the application context. Since the real objects has generally a complex structure, that can be described at different levels of detail, a hierarchical multi-scale representation allows for a more accurate tuning of the reconstruction, with a lower complexity of the final model.

In this paper, the basic concepts of surface reconstruction will be introduced and the approaches based on computational intelligence paradigms will be presented. In particular, the approaches based on some hierarchical techniques (namely, HRBF and HSVR) will be analyzed and discussed in detail.

***Keywords* - *Surface reconstruction, computational intelligence paradigms, multiscale models, hierarchical models.*

I. INTRODUCTION

The problem of surface modeling by reconstruction consists in the search of the surface that best describes a given set of points. In computational intelligence, specifically in neural-based approaches, this can be viewed as a supervised learning problem, where the vector coordinates (or other features) of the single point is an input instance, while a further coordinate is an output label. The approximation function identifies how to obtain labels from instances.

The tri-dimensional surface reconstruction case has particular interest in many applications encompassing, e.g., the fields of design, archeology, medicine, and entertainment. A digital 3D model can be obtained, for example, by means of physical object measurements performed by using a 3D scanner. In this approach, an important step of the 3D model building process consists of creating the object's surface representation from a cloud of noisy points sampled on the object itself. This process can be viewed as the estimation of a function from a finite subset of its points.

Several effective computational intelligence paradigms have been developed for solving the surface reconstruction problem. For the solution of the function approximation problem, neural techniques, generally, show a good trade-off between computational complexity, accuracy and robustness of the solution with respect to other methods. In this context, there are many different paradigms which are able to find the approximation function, e.g., Multi-layer Perceptron Networks, Radial Basis Function (RBF) Networks, Self-Organizing Maps (SOM), and Support Vector Machines (SVM). However, other paradigms such as Genetic Algorithms has been used to improve the performances of traditional approaches of surface reconstruction. In general, there is not a single paradigm better than the others, but each one performs differently depending on the application context. Since the real objects has generally a complex structure, that can be described at different levels of detail, a hierarchical multi-scale representation allows for a more accurate tuning of the reconstruction, with a lower complexity of the final model.

This paper is directed to introduce the basic concepts of surface reconstruction by using approaches based on computational intelligence. In particular, the approaches based on two hierarchical techniques (namely, HRBF and HSVR) will be analyzed and discussed in detail.

II. THE SURFACE RECONSTRUCTION PROBLEM

A 3D scanner is a device that provide a measurement of the geometry (and possibly also the visual features) of an object by means of a sampling of the points of the object's surface [1][2][3][4][5]. Depending on the particular setup of the 3D scanner, the set of points can enjoy some additional properties, such as a structural relationship on the points. In the most general case, however, the acquisition procedure provides a set of unstructured 3D points, also called a 3D points cloud. The surface reconstruction procedure provides a generalization of the points cloud, obtaining a continuous description of the surface from which the points belong to. When the object has a complex structure, its surface cannot be captured in a single acquisition session. Hence, the data belonging to several sessions have to be properly registered and fused together. This operation can be performed before or after the generalization phase. In the first case, the registration and the fusion can be operated on the partial surface, while in the second case the surface has to be recovered from the whole (registered) dataset.

From the 3D scanning pipeline above briefly described, it is evident that the surface reconstruction problem can be characterized by several features that can be exploited for tailoring the solution to the particular application:

- the inherent structure of the data, for instance from contour based scanning, is an important source of information and is a sort of a-priori information that can be easily incorporated in some paradigms for surface approximation;
- the topology of the surface can be obtained from the data if they are enough dense; otherwise, it should be known a-priori and can be incorporated in the surface approximation paradigm;
- sometimes, the class of the object is a-priori known and its general shape can be described by a parametrized model; the actual surface can be recovered by estimating the value of the model parameters from the data.

The traditional approaches for surface reconstruction describe the surface as a triangular mesh or a polynomial surface (e.g., NURBS [6]). They can be categorized as deterministic or iterative, depending on the structure of the fitting algorithm [7], or, depending on the technique used to optimize the surface description to the data, in spatial subdivision and surface fitting techniques. However, since advanced methods can use a mix of the above cited techniques (e.g., spatial subdivision to obtain a coarse description and surface fitting for refining the description [8]), the categorization can be quite arbitrary. A more structured taxonomy can be found in [9].

III. Computational Intelligence Approaches to Surface Reconstruction

Computational intelligence methods are used in the context of the surface reconstruction for their learning-from-data, generalization, and robustness capabilities. Since the reconstruction is operated through several stages, each with its optimization subproblem, computational intelligence techniques can found application for several tasks.

The unsupervised learning neural paradigms for clustering are composed of neurons that are characterized for a position in the space of the training data. This allows a direct application in the surface reconstruction, when the training data is composed of a 3D point cloud.

The SOM are characterized by a topological structure defined by the neighborhood relationship of the neurons. When the neurons are 3D points and the topological structure is a 2D lattice (which can be implemented as a mesh), the SOM can be used to approximate a 3D surface. This approach has been followed in [10][11], where particular attention has been paid to address the problem of correctly distributing the SOM units on the boundary of the training set; in [7], the mesh is iteratively refined, by making the SOM denser at each refinement iteration. A SOM is used in [12][13] to obtain a surface from a set of points structured in lines. The 3D points lines are transformed in a 6-dimensional point that describe the line, and in that 6D space the SOM training is carried out. A backward transformation of the position of the SOM units allows to obtain the ordered set of lines that defines the surface. The smooth mapping properties of the SOM are exploited in [14], where a dense general 3D mesh model of a face is adapted to a particular face by computing the mapping between few feature points in the input face and the corresponding points in the general model. In [15], a topology preserving self-organizing clustering model is presented and used for filtering a dense, noise-affected 3D point cloud. A similar approach is proposed in [16][17][18], where an extension of the Neural-Gas model [19], optimized for low dimensionality spaces, is applied to the filtering of the noise in 3D point cloud. Clustering is also used as an intermediate processing in [20], where the normals of the surfaces points are clustered using the fuzzy k-means algorithm, obtaining a description of the surface as a collection of planar patches.

Since surface fitting belongs to the optimization problems, the search of the optimal value of parameters used to describe the surface has been carried out also using evolutionary techniques. In [21], two evolutionary algorithms are challenged for fitting a triangular mesh to a point cloud. In [22], the surface is described as a patch of planar or quadratic surfaces; a genetic algorithm is used in the segmentation process as a tool for robustly estimating the surface that fits a cluster of points. The B-splines form a widely used class of parametric models for describing a curve or a surface, where the parametrization is controlled by particular points of the parameter space, called knots. Since their position can heavily affect the shape of the surface, their estimation is critical in surface fitting problems. Once they are set, the other parameter of the B-spline model can be computed using traditional (e.g, least-square) optimization. In [23], an artificial immune system is used for positioning the knots of a B-spline. A similar approach is followed by [24], where genetic algorithms are used to determine the parametric value of the points and to choose the knots position.

Computational intelligence paradigms can be used also in processing stages that are usually included in the acquisition of the points cloud. This is the case of the shape from shading approach described in [25], where a multi-layer feed-forward neural network is used for computing the surface normals from the reflectance information coded by three images of the same object, but taken using three different illumination sources. In [26], instead a neural network is used to estimate the components of the reflectance of a surface, from the pixel values of an image of the surface. In [27], instead, a shape-from-shading algorithm targeted to face reconstruction is presented. The input image is segmented in face features (e.g., lips, nose, eyes) and for each face parts, a neural network provides the corresponding surface. The shading pattern of each face part is obtained training the corresponding networks with the images of the target face part of several individual, taken with different poses and illuminations. The whole face surface is then obtained merging the partial surfaces.

The adaptiveness of the computational intelligence paradigms to the data can be exploited also for post-processing operations, like surface filtering. For instance, in [28] a fuzzy filtering is proposed for smoothing the face normals of a triangular mesh, while preserving the features of the surface, such as sharp edges.

IV. Hierarchical Models

Hierarchical models describe the surface at different levels of detail (LOD) and allow to perform operation related to surface processing in a more effective and robust way [29]. In this Section, two models, namely the Hierarchical Radial Basis Function (HRBF) network and the Hierarchical Support Vector Regression (HSVR), are introduced. They are extension of respectively of the Radial Basis Function (RBF) networks and the Support Vector Machines (SVM) for regression (also called Support Vector Regression, SVR). HRBF and HSVR are both based on a pool of subnetworks (called layers), which act at different scales, organized in a hierarchical structure, with a training algorithm that is based on a coarse-to-fine surface fitting procedure.

Given a target surface, $y = f(x)$, the first layer, $a_1(x)$, provides a low resolution description of the surface, while the second layer, $a_2(x)$, refines this description by approximating the residual, $r_1(x) = f(x) - a_1(x)$. This schema can be repeated until some stop criterion is met, fitting the layer a_k to the function $r_{k-1}(x) = r_{k-2}(x) - a_{k-1}(x)$. Then the reconstruction can be operated summing up the contribution of all the layers: $\tilde{f}(x) = \sum_k a_k(x)$.

Every layer acts on a different scale and is characterized by a scale parameter, σ_k, with $\sigma_k < \sigma_{k-1}$. Hence, this schema allows to obtain also L reconstructions with different resolution:

$$\tilde{a}_l(x) = \sum_{h=1}^{l} a_h(x) \tag{1}$$

Besides, the coarse-to-fine schema is usually more robust, since errors in one layer can be taken care by the following layer [30][31]. Despite there are not theoretical constraints, the set $\{\sigma_1, ..., \sigma_L\}$ is computed, generally, such that each scale is halved with respect to the previous one. In this way a wide range of scales can be covered by using a small number of layers.

Since usually the target function $f(\cdot)$ is known only through a sampling of the target function:

$$\{(x_i, y_i)\,|\, x_i \in \mathbb{R}^2,\, y_i \in \mathbb{R},\, i = 1, \ldots, n\} \tag{2}$$

the training algorithms have to work on a finite set of samples, usually affected by noise (which is assumed to be Gaussian white noise).

The layers of both the HRBF and HSVR, are constituted of a linear combination of Gaussian units:

$$G(x; \mu, \sigma) = \frac{1}{\pi\sigma^2} e^{-\frac{||x-\mu||^2}{\sigma^2}} \tag{3}$$

where μ is the center of the Gaussian (and it is a point that belong to the input domain of the training set) and σ is the width of the Gaussian, which regulate the extension of the neighborhood of the center where the Gaussian mostly contributes to the reconstruction. This property is also called locality.

Although their overall operations are very similar, the training algorithms differs, since they are based on the features of the paradigms they extend (i.e., RBF and SVM).

They can be used to realize mappings $\mathbb{R}^N \to \mathbb{R}^M$, with arbitrary M and N, but in Sects. IV-A and IV-B they will be briefly described with reference to the $\mathbb{R}^2 \to \mathbb{R}$ case. More detailed descriptions can be found in [32][33][34][35][36][37].

A. Hierarchical Radial Basis Function Networks

The Hierarchical Radial Basis Function (HRBF) network is composed of a pool of layers of Gaussian units. The units of each layer are regularly distributed in the input domain space (which centers are positioned on the node of a 2D lattice) and have the same width, σ. These properties make each layer equivalent to a Gaussian digital filter, which, in turn, can be exploited to motivate the computation of the length of the units' lattice side, Δx, and the weights, w of the linear combination which compute the output of the layer. The parameters $\{\mu_{l,k}\}$, $\{\sigma_l\}$, and L define the structure of the HRBF network.

The HRBF configuration algorithm can be summarized as follows. Let us consider an input data set: $\{(x_i,\, y_i)|i = 1, \ldots, n\}$.

1) A set of L values of σ_l is chosen and the corresponding Δx_l are computed for each layer:

$$\Delta x_l = \frac{\sigma_l}{1.465} \tag{4}$$

2) The first residual is set as: $r_0 = \{r_0(x_k) = y_k\}$.
3) For each layer, $l = 1, \ldots, L$:
    4) The Gaussians center is defined, $\{\mu_{l,j}|\mu_{l,j} - \mu_{l,j-1} = \Delta x_l\}$, such that they cover the bounding box of the input data. This will define the number, M_l of the Gaussians of the l-th layer.
    5) For each Gaussian, j, only a subset, $A_{l,j}$, of the training data is considered for the

computation of the weight, w_j. These are the samples that fall in a neighborhood of the Gaussian center, called receptive field of the unit. This is defined as the circular region centered in $\mu_{l,j}$ with radius equal to $2\Delta x_l$.

6) A Gaussian unit is inserted only if the approximation obtained with the previous layers is not enough accurate, that is if the residual is larger than a given threshold, ε, and if enough samples in the receptive fields are available (for practical cases, 3 to 5 samples is enough). If $R_k = \frac{\sum_{x_r \in A_k} ||r_r||}{|A_k|} > \varepsilon$, the coefficient $w_{l,j}$ is computed as a weighted mean of the residuals in the receptive field:

$$w_{l,j} = \Delta x_l^2 \frac{\sum_{x_r \in A_{l,j}} r_{l-1}(x_r)\, e^{-\frac{||x_r - \mu_{l,j}||^2}{\sigma_l^2}}}{\sum_{x_r \in A_{l,j}} e^{-\frac{||x_r - \mu_{l,j}||^2}{\sigma_l^2}}} \tag{5}$$

The weights are set proportional to the estimate of the surface in the Gaussian center, $\mu_{l,j}$, which is obtained through a kernel regression approach [38].

7) The approximation function $a_l = \{a_l(x_i) = \sum_j w_{l,j} G(x_i; \mu_{l,j}, \sigma_l)\}$ is computed.

8) The residual is computed as: $r_l = \{r_{l-1}(x_i) - a_l(x_i)\}$.

9) If $l < L$, go back to step 3.

The output function is the sum of all the approximation functions computed:

$$\tilde{f}(x) = \sum_{l=1}^{L} \sum_{j=1}^{M_l} w_{l,j} G(x; \mu_{l,j}, \sigma_l) \tag{6}$$

The number of layers has not to be set a priori, but a termination criterion can be used: for example, the procedure can be iterated until the residual goes under a given threshold on the entire domain.

From (6), we see that the output of all the Gaussians in all the layers should be computed for each point in the input space, that would require a huge computational effort. However, due to its exponential decay, the Gaussian assumes values significantly larger than zero only in the region close to its center and some windowing to zero can be introduced. In fact the Gaussian value at the distance of 3σ from its center is $1.24 \cdot 10^{-4}$ times its value in the center. This allows to limit the computation of the contribution of each Gaussian to the output of the layer, only in to a suitable neighborhood of its center, called *influence region* of the Gaussian k, I_k. For surface reconstruction application, this can be defined as the circular region with radius equal to $3\sigma_l$.

The set of scale parameter, $\{\sigma_l\}$, can be set using some a priori knowledge on the frequency content of the function [32]. However, when this information is missing, σ_0 can be set equal to the extension of the input domain to which the data belong to (e.g., to the diagonal of the input data bounding box). If no information on the scale of the details is available, the width of the Gaussians can be halved at each layer. The position of the units of each layer can be easily determined starting from the center of the domain moving laterally by Δx (4) in the axes direction. The number of layers, L, can be determined a priori considering the computational resources available (e.g., the number of units), or can be determined runtime considering the error achieved by the last configured layer. Otherwise it can be determined by a trial and error strategy or by cross-validation.

The error threshold, ε, determines if the Gaussians of the new layer will be added to the network and then it determines the accuracy of the approximation of the whole HRBF network. It should be set from the accuracy of the measurement instruments used for obtaining the data set. In the case of the 3D scanning, it can be estimated by using several techniques (e.g., average distance of sampled points on a plane with respect to the optimal plane), or from the calibration procedure. If a priori information is not available, a trial and error strategy could be used to estimate the error threshold.

In order to compute the parameters, the configuration procedure of a HRBF network requires operations that local to the data [39][40]. This produces a very fast configuration algorithm that is also suitable to be parallelized. An on-line version of the HRBF training algorithm has been proposed in [41] and [34]. A different coarse-to-fine schema for the HRBF have been explored in [42].

B. Hierarchical Support Vector Regression

The training of the layers that composes a Hierarchical Support Vector Regression (HSVR) model is based on the Support Vector Machine optimization [43][44][45][46] and its extension to the regression (Support Vector Regression, SVR). The surface fitting problem is formulated as convex optimization problem whose computational complexity is independent from the data input dimensionality. The quality of the solution computed by the SVR paradigm depends on a three hyperparameters (namely σ, ε, and C), generally, selected using a trial-and-error strategy. The solution is searched optimizing a cost function composed of two additive terms; the first term measures the closeness of the approximation to the data, while the second term measures the complexity of the solution. The trade-off between the two terms is controlled through the hyperparameter C.

In order to guarantee the sparsity of the solution, the closeness of the approximation to the data is measured through a *ε-insensitive* loss function:

$$L_1^{\varepsilon}(\zeta) = \begin{cases} 0 & \text{if } |\zeta| \leq \varepsilon \\ |\zeta| - \varepsilon & \text{otherwise} \end{cases} \tag{7}$$

where ζ is the difference between the real and the approximated surface, $\zeta = y - \hat{f}(x)$, and ε is the hyperparameter that controls the accuracy of the reconstruction.

Although initially formulated for linear regression, the SVR can be extended to the non-linear case using a kernel function. One of the most used, and here considered, is the Gaussian kernel, which is controlled by the width parameter, σ.

The optimization procedure selects a subset of the training examples, called Support Vectors (SVs), and the output of the SVR model results:

$$\tilde{f}(x) = \sum_{i=1}^{M} w_i \, G(x;\, x_i, \sigma) + b \tag{8}$$

where M is the number of SVs, on which the Gaussian units are centered, w_i are the weights that weight the contribution of each Gaussian to the reconstruction and b is a bias. Due to the optimization procedure, the weights enjoy the following property:

$$w_i = \begin{cases} C, & y_i > f(x_i) + \varepsilon \\ [0,\, C], & y_i = f(x_i) + \varepsilon \\ 0, & |y_i - f(x_i)| < \varepsilon \\ [-C,\, 0], & y_i = f(x_i) - \varepsilon \\ -C, & y_i < f(x_i) - \varepsilon \end{cases} \tag{9}$$

Hence, the SVR optimization procedure partitions the points in three categories, depending by the region they fall in. The region close to the approximation surface for less than ε, called ε-tube, identifies the points that do not contribute to the reconstruction (their weight is zero). The points that lie outside the ε-tube have a constant weight, equal (in absolute value) to the hyperparameter C (these SVs are also called "bounded"). The weight of the points that lies on the ε-tube cannot exceed C, in absolute value.

The HSVR configuration algorithm can be summarized as follows. Let us consider an input data set: $\{(x_i,\, y_i) | i = 1, \ldots, n\}$.

1) A set of L values of σ_l is chosen.
2) The first residual is set as: $r_0 = \{r_0(x_k) = y_k\}$.
3) For each layer, $l = 1, \ldots, L$:
    4) The value of C_l is set as proportional to the standard deviation of the training data:

    $$C_l = J \, \mathrm{std}(r_{l-1}(x_i)) \tag{10}$$

    5) The SVR optimization procedure is run on the training set $S_l = \{(x_1, r_{l-1}(x_1)), \ldots, (x_n, r_{l-1}(x_n))\}$ using $\{\sigma_l,\, \varepsilon,\, C_l\}$ as hyperparameters, obtaining the parameters of the layer ($\{w_{l,j}\}$ and b_l).
    6) The approximation function $a_l = \{a_l(x_i) = \sum_j w_{l,j} G(x_i; \mu_{l,j}, \sigma_l)\} + b_l$ is computed.
    7) The residual is computed as: $r_l = \{r_{l-1}(x_i) - a_l(x_i)\}$.
    8) If $l < L$, go back to step 3.

The above scheme provide an effective approximation for the given dataset, but lack of efficiency, since the standard SVR optimization produces a large number of SVs also for the first layers. These describe very smooth approximations and should require only few SVs. The SVR solution, however, selects as SVs all the training examples outside the ε-tube (which in the first layers are the majority of the training examples). The problem can be faced by applying a reduction of the training set used to compute the layer parameters, selecting only the most meaningful for being passed to the SVR optimization procedure. Since the approximation error flows in the residual, which can be reconstructed by the following layers, a small increase in the approximation error achieved by the first layers are well accepted if it allows to obtain a more compact layer. Hence, the examples closer to the surface resulting from the step 5 can be selected and used to compute the approximation of a_l. This can be done defining the following additional steps:

7a) Selects the examples that lies on the tube and those that are close to a_l:

$$\begin{aligned} S'_l \;=\; & \{(x_i,\, r_{l-1}(x_i)) \mid \text{ s.t. } |r_l(x_i)| - \varepsilon = 0 \\ & \vee |r_l(x_i)| < \frac{\varepsilon}{2}\} \end{aligned} \tag{11}$$

7b) As the density of the samples in S'_l is significantly less than in S_l, the value of C_l is increased proportionally to the change in data density:

$$C'_l = C_l \frac{|S_l|}{|S'_l|} = J \, \mathrm{std}(r_{l-1}(x_i)) \frac{|S_l|}{|S'_l|} \tag{12}$$

7c) The SVR optimization procedure is run on the training set S'_l using $\{\sigma_l,\, \varepsilon,\, C'_l\}$ as hyperparameters, obtaining the layer parameters ($\{w'_{l,j}\},\, b'_l$).
7d) The approximation function $a'_l = \{a'_l(x_i) = \sum_j w'_{l,j} G(x_i; \mu_{l,j}, \sigma_l)\} + b'_l$ is computed.
7e) The residual is computed as: $r'_l = \{r_{l-1}(x_i) - a'_l(x_i)\}$.

V. Conclusions

In this paper, an overview of the computational intelligence approaches used in literature for solving the problem of surface reconstruction has been presented. In particular, two multiscale, hierarchical models, namely HRBF and HSVR, has been described.

References

[1] N. A. Borghese, G. Ferrigno, G. Baroni, A. Pedotti, S. Ferrari, and R. Savarè, "Autoscan: a flexible and portable 3D scanner," *Computer Graphics and Applications, IEEE*, vol. 18, no. 3, pp. 38–41, May–June 1998.

[2] S. Ferrari and N. A. Borghese, "A portable modular system for automatic acquisition of 3D objects," in *Proceedings of IMTC'99 (16th IEEE Instrumentation and Measurement Technology Conference)*, V. Piuri and M. Savino, Eds., vol. 3, May 1999, pp. 1823–1827.

[3] N. A. Borghese and S. Ferrari, "A portable modular system for automatic acquisition of 3D objects," *IEEE Trans. on Instrumentation and Measurement*, vol. 49, no. 5, pp. 1128–1136, Oct. 2000.

[4] F. Bellocchio and S. Ferrari, *Depth Map and 3D Imaging Applications: Algorithms and Technologies*. IGI Global, 2011, ch. 3D Scanner, State of the Art, pp. 451–470.

[5] C. Rigotti, N. A. Borghese, S. Ferrari, G. Baroni, and G. Ferrigno, "Portable and accurate 3D scanner for breast implants design and reconstructive plastic surgery," in *Proceedings of SPIE's International Symposium on Medical Imaging 1998*, vol. 3338, Feb. 1998, pp. 1558–1567.

[6] L. Piegel and W. Tiller, *The NURBS book*. Springer, 1997.

[7] A. de Medeiros Brito Jr., A. D. Dória Neto, J. Dantas de Melo, and L. M. Garcia Gonçalves, "An adaptive learning approach for 3-D surface reconstruction from point clouds," *IEEE Trans. on Neural Networks*, vol. 19, no. 6, pp. 1130–1140, Jun. 2008.

[8] C. L. Bajaj, F. Bernardini, and G. Xu, "Automatic reconstruction of surfaces and scalar fields from 3D scans," in *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '95, 1995, pp. 109–118.

[9] R. Mencl and H. Müller, "Interpolation and approximation of surfaces from three-dimensional scattered data points," in *Proceedings of the Eurographics 98 Conference*, 1998, pp. 51–67.

[10] J. Barhak and A. Fischer, "Parameterization and reconstruction from 3D scattered points based on neural network and PDE techniques," *IEEE Trans. on Visualization and Computer Graphics*, vol. 7, no. 1, pp. 1–16, Jan.-Mar. 2001.

[11] ——, "Adaptive reconstruction of freeform objects with 3d som neural network grids," *Computers & Graphics*, vol. 26, no. 5, pp. 745–751, 2002.

[12] M. Hoffmann and E. Kovács, "Developable surface modelling by neural network," *Mathematical and Computer Modelling*, vol. 38, no. 7–9, pp. 849–853, 2003.

[13] M. Hoffmann, "Numerical control of Kohonen neural network for scattered data approximation," *Numerical Algorithms*, vol. 39, pp. 175–186, 2005.

[14] L. Sajó, M. Hoffmann, and A. Fazekas, "A 3D head model from stereo images by a self-organizing neural network," *Journal for Geometry and Graphics*, vol. 13, no. 2, pp. 209–220, 2009.

[15] A. Baraldi and E. Alpaydin, "Constructive feedforward ART clustering networks. II," *IEEE Trans. on Neural Networks*, vol. 13, no. 3, pp. 662–677, May 2002.

[16] S. Ferrari, G. Ferrigno, V. Piuri, and N. A. Borghese, "Reducing and filtering point clouds with enhanced vector quantization," *IEEE Trans. on Neural Networks*, vol. 18, no. 1, pp. 161–177, Jan. 2007.

[17] S. Ferrari, I. Frosio, V. Piuri, and N. A. Borghese, "Enhanced vector quantization for data reduction and filtering," in *Proceedings of 3DPVT 2004 (2nd International Symposium on 3D Data Processing, Visualization and Transmission)*, Sep. 2004, pp. 470–477.

[18] N. A. Borghese and S. Ferrari, "Mesh construction with fast soft vector quantization," in *Proceedings of IJCNN 2000 (IEEE-INNS-ENNS International Joint Conference of Neural Networks)*, S.-I. Amari, C. L. Giles, M. Gori, and V. Piuri, Eds., vol. 5, Jul. 2000, pp. 473–478.

[19] T. Martinetz, S. Berkovich, and K. Schulten, "Neural-gas network for vector quantization and its application to time-series prediction," *IEEE Trans. on Neural Networks*, vol. 4, no. 4, pp. 558–568, 1993.

[20] A. Sampath and J. Shan, "Segmentation and reconstruction of polyhedral building roofs from aerial lidar point clouds," *IEEE Trans. on Geoscience and Remote Sensing*, vol. 48, no. 3, pp. 1554–1567, Mar. 2010.

[21] Y. Fujiwara and H. Sawai, "Evolutionary computation applied to mesh optimization of a 3-D facial image," *IEEE Trans. on Evolutionary Computation*, vol. 3, no. 2, pp. 113–123, Jul. 1999.

[22] P. F. U. Gotardo, O. R. P. Bellon, K. L. Boyer, and L. Silva, "Range image segmentation into planar and quadric surfaces using an improved robust estimator and genetic algorithm," *IEEE Trans. on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 34, no. 6, pp. 2303–2316, Dec. 2004.

[23] E. Ülker and A. Arslan, "Automatic knot adjustment using an artificial immune system for B-spline curve approximation," *Information Sciences*, vol. 179, no. 10, pp. 1483–1494, 2009.

[24] A. Gálvez, A. Iglesias, and J. Puig-Pey, "Iterative two-step genetic-algorithm-based method for efficient polynomial B-spline surface reconstruction," *Information Sciences*, vol. 182, no. 1, pp. 56–76, 2012.

[25] K. V. Rajaram, G. Parthasarathy, and M. A. Faruqi, "A neural network approach to photometric stereo inversion of real-world reflectance maps for extracting 3-D shapes of objects," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 25, no. 9, pp. 1289–1300, Sep. 1995.

[26] C.-T. Lin, W.-C. Cheng, and S.-F. Liang, "Neural-network-based adaptive hybrid-reflectance model for 3-d surface reconstruction," *IEEE Trans. on Neural Networks*, vol. 16, no. 6, pp. 1601–1615, Nov. 2005.

[27] D. Nandy and J. Ben-Arie, "Shape from recognition: a novel approach for 3-D face shape recovery," *IEEE Trans. on Image Processing*, vol. 10, no. 2, pp. 206–217, Feb. 2001.

[28] Y. Shen and K. E. Barner, "Fuzzy vector median-based surface smoothing," *IEEE Trans. on Visualization and Computer Graphics*, vol. 10, no. 3, pp. 252–265, May–June 2004.

[29] N. Borghese, S. Ferrari, and V. Piuri, "A methodology for surface reconstruction based on hierarchical models," in *Proceedings of HAVE 2003 (IEEE International Workshop on Haptic Virtual Environments and Their Applications)*, Sep. 2003, pp. 119–124.

[30] S. Ferrari, N. A. Borghese, and V. Piuri, "Multiscale models for data processing: an experimental sensitivity analysis," *IEEE Trans. on Instr. and Meas.*, vol. 50, no. 4, pp. 995–1002, Aug. 2001.

[31] ——, "Multi-resolution models for data processing: an experimental sensitivity analysis," in *Proceedings of IMTC 2000 (17th IEEE Instrumentation and Measurement Technology Conference)*, vol. 2, May 2000, pp. 1056–1060.

[32] N. A. Borghese and S. Ferrari, "Hierarchical RBF networks and local parameter estimate," *Neurocomputing*, vol. 19, no. 1–3, pp. 259–283, 1998.

[33] S. Ferrari, M. Maggioni, and N. A. Borghese, "Multiscale approximation with hierarchical radial basis functions networks," *IEEE Trans. on Neural Networks*, vol. 15, no. 1, pp. 178–188, Jan. 2004.

[34] S. Ferrari, F. Bellocchio, V. Piuri, and N. A. Borghese, "A hierarchical RBF online learning algorithm for real-time 3-D scanner," *IEEE Trans. on Neural Networks*, vol. 21, no. 2, pp. 275–285, Feb. 2010.

[35] ——, "Multi-scale support vector regression," in *Proceedings of IJCNN 2010 (IEEE International Joint Conference on Neural Networks)*, Jul. 2010, pp. 1–7, runner-up Best Paper.

[36] F. Bellocchio, S. Ferrari, V. Piuri, and N. A. Borghese, "Hierarchical approach for multiscale Support Vector Regression," *IEEE Trans. on Neural Networks and Learning Systems*, vol. 23, no. 9, pp. 1448–1460, Sep. 2012.

[37] F. Bellocchio, N. A. Borghese, S. Ferrari, and V. Piuri, *3D Surface Reconstruction: Multi-Scale Hierarchical Approaches*. Springer-Verlag New York, LLC, 2012.

[38] ——, "Kernel regression in HRBF networks for surface reconstruction," in *Proceedings of HAVE 2008 (IEEE International Workshop on Haptic Audio and Visual Environments and Games)*, Oct. 2008, pp. 160–165.

[39] S. Ferrari, I. Frosio, V. Piuri, and N. A. Borghese, "Automatic multiscale meshing through HRBF networks," *IEEE Trans. on Instrumentation and Measurement*, vol. 54, no. 4, pp. 1463–1470, Aug. 2005.

[40] ——, "The accuracy of the HRBF networks," in *Proceedings of IMTC 2004 (21th IEEE Instrumentation and Measurement Technology Conference)*, May 2004, pp. 482–486.

[41] F. Bellocchio, S. Ferrari, V. Piuri, and N. A. Borghese, "Online training of hierarchical RBF," in *Proceedings of IJCNN 2007 (IEEE International Joint Conference on Neural Networks)*, Aug. 2007, pp. 2159–2164.

[42] S. Ferrari, F. Bellocchio, N. A. Borghese, and V. Piuri, "Refining hierarchical radial basis function networks," in *Proceedings of HAVE 2007 (IEEE International Workshop on Haptic Audio and Visual Environments and Games)*, Oct. 2007, pp. 166–170.

[43] V. Vapnik and A. Lerner, "Pattern recognition using generalized portrait method," *Automation and Remote Control*, vol. 24, pp. 774–780, 1963.

[44] V. Vapnik and A. Chervonenkis, "Theory of pattern recognition," *Nauka, Moscow*, 1974.

[45] V. Vapnik, *Statistical learning theory*. Wiley-Interscience, 1989.

[46] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and Computing*, vol. 14, pp. 199–222, 2004.