HAVE 2007 - IEEE International Workshop on
Haptic Audio Visual Environments and their Applications
Ottawa, Canada, 18–19 October 2008

Kernel Regression in HRBF Networks for Surface Reconstruction

F. Bellocchio[1], N.A. Borghese[2], S. Ferrari[1], V. Piuri[1]

[1] Department of Information Technologies,
University of Milano
via Bramante, 65 – 26013 Crema, Italy
E–mail: {bellocchio, ferrari, piuri}@dti.unimi.it.

[2] Department of Computer Science
University of Milano,
via Comelico, 39-41 – 20135 Milano, Italy
E–mail: borghese@dsi.unimi.it.

Abstract – *The Hierarchical Radial Basis Function (HRBF) Network is a neural model that proved its suitability in the surface reconstruction problem. Its non-iterative configuration algorithm requires an estimate of the surface in the centers of the units of the network. In this paper, we analyze the effect of different estimators in training HRBF networks, in terms of accuracy, required units, and computational time.*

Keywords – *Radial Basis Function Networks, HRBF, kernel regression.*

I. INTRODUCTION

A 3D scanning device measures the geometric properties of a real object and produce a 3D model of the object itself. This digitizing procedure is performed in two step: measuring the position of some points on the object surface (sampling) and processing the sampled data to obtain a representation of the surface. In computer graphics, the most used paradigm for representing the geometry of the model is the triangles mesh. Subsequent processing may be carried out on the mesh to obtain a more sophisticated representation in order to speed up operations such as editing or visualization, or to diminish the size of the representation.

Multiresolution representation [1][2][3][4] is widely used for this scope as it represents the surface at different levels of detail; it usually allows operating at the resolution required by the specific application or local processing by computing only a small fraction of the parameters that represent the surface.

The Hierarchical Radial Basis Function (HRBF) model is a neural network paradigm that can be used for surface reconstruction from a cloud of points affected by noise [5][6][7]. It uses a linear combination of Gaussians to represent the surface as an explicit function defined in the $\mathbb{R}^2$ domain. As its configuration procedure is not iterative and uses only local operations, it is suitable to fast processing. The multi-scale adaptive scheme of the HRBF model adds Gaussians only in those regions of the input domain that contain the surface details, allowing a compact representation of the surface.

The configuration procedure requires an estimate of the surface height in some points of its domain. This can be apparently a non-sense as the aim of the network is the reconstruction of the entire surface. However, due to the robustness of the configuration algorithm [7], the exact value of the function is not required, and a good estimate is usually sufficient for achieving an accurated reconstruction [5]. Besides, once configured, the network is able to generalize and to reconstruct the surface on the whole domain, not only in the given surface points.

Although the choice of the estimator does not affect the asymptotic behavior of the HRBF approximation [5], its effect may be important for real applications, as it can impact on the accuracy, as well on the computational time and the compactness of the final network. In this paper, we investigate the effect of different estimators on the HRBF machinery. In section II, the original formulation of the HRBF model is described, while in section III the most used estimation methods are introduced. A comparison of the results obtained with the different estimation methods is carried in section IV. In section V the results of the comparison are discussed, while in section VI conclusions are drawn.

II. HRBF NETWORKS

Let us assume that the manifold to be approximated can be described as a $\mathbb{R}^D \rightarrow \mathbb{R}$ function. This allows to consider the input dataset as a height field: $\{(P_i, z_i) \,|\, z_i = S(P_i),\, P_i \in \mathbb{R}^D,\, 1 \leq i \leq N\}$, and the manifold, output by the network, will assume the explicit analytical shape: $z = S(P)$. The output of a HRBF network is obtained by adding the output of a pool of Radial Basis Functions (RBF) networks, organized as a stack of hierarchical layers, each of which is characterized

978-1-4244-2669-0/08/$25.00

by a decreasing scale:

$$S(P) = \sum_{l=1}^{L} a_l(P; \sigma_l) \tag{1}$$

where σ_l determines the scale of the l-th layer, with $\sigma_l > \sigma_{l+1}$. If we suppose that the units are equally spaced on a grid support and a normalized spherical Gaussian function, $G(\cdot;\, \sigma) = \frac{1}{\sqrt{\pi\sigma^2}^D}\, exp\left(-\frac{||\cdot||^2}{\sigma^2}\right)$, is taken as basis function, the output of each layer can be written as a linear low-pass filter:

$$a_l(P; \sigma_l) = \sum_{k=1}^{M_l} w_{l,k} G(||P - P_{l,k}||; \sigma_l) \tag{2}$$

where M_l is the number of Gaussian units of the l-th layer. The $G(\cdot)$ are equally spaced on a D-dimensional grid, which covers the input domain of the data points: that is the $\{P_{l,k}\}$s are positioned in the grid crossings of the l-th layer. The side of the grid is a function of the scale of that layer: the smaller the scale, the shorter is the side length, the denser are the Gaussians and the finer are the details which can be reconstructed.

The actual shape of the surface in (1) depends on a set of parameters: the *structural parameters*, which are the total number, $\{M_l\}$, the scale, $\{\sigma_l\}$, and the position, $\{P_{l,k}\}$, of the Gaussians of the l-th layer; and the weights associated to each Gaussian: $\{w_{l,k}\}$. Each RBF grid, l, realizes a reconstruction of the surface up to a certain scale, determined by σ_l (low-pass filtered reconstruction). Considerations grounded on signal processing theory allow, given a certain scale, σ_l, to set the grid side, ΔP_l, as $\sigma_l = 1.465\, \Delta P_l$ and to determine consequently $\{P_{l,k}\}$ and M_l (and, hence, the total number of Gaussians, $M = \sum_l M_l$) [8]. From these observations, the weights $\{w_{l,k}\}$ are set equal to the manifold height in the grid crossings: $w_{l,k} = S(P_{l,k}) \cdot \Delta P_l^D$. As the data set usually does not include the $\{S(P_{l,k})\}$, these values should be estimated. We explicitly observe that, even if the $S(P_{l,\,k})$ were included, they would be corrupted by noise and an estimate would be the right solution. The data points that lie in an appropriate neighborhood of $P_{l,\,k}$ can be used to estimate $S(P_{l,\,k})$ as a weighted average of such subset of data points, $\tilde{S}(P_{l,\,k})$. This neighborhood, called *receptive field*, $A(P_{l,\,k})$, can be chosen as a spherical region centered in $P_{l,\,k}$ with the radius proportional to the grid side, ΔP_l. A possible weighting function is:

$$\tilde{S}(P_{l,k}) = \frac{\displaystyle\sum_{P_m \in A(P_{l,k})} S(P_m)\, e^{-\frac{||P_{l,k} - P_m||^2}{\sigma_l^2}}}{\displaystyle\sum_{P_m \in A(P_{l,k})} e^{-\frac{||P_{l,k} - P_m||^2}{\sigma_l^2}}} \tag{3}$$

which is strongly related to the Nadaraya-Watson estimator and maximizes the conditional probability density when the noise is normally distributed, zero mean [9] [10].

Although a single layer with Gaussians of very small scale could reconstruct the finest details, this would produce an unnecessary dense packing of units in all those regions which feature large scale details. Moreover, there might even be not enough points inside some $A(P_{l,k})$ to get a reliable estimate of $\tilde{S}(P_{l,k})$ in (3). A better solution is to adaptively allocate the Gaussian units, with an adequate scale, in the different regions of the input data domain. This can be achieved by adding and configuring one layer at time, proceeding from the layer featuring the largest scale to the layer featuring the smallest one. For sake of simplicity in the configuration stage, each new layer will feature half the scale of the previous one. However, arbitrary scales could be used for the different layers.

All the layers after the first one will be trained to approximate the residual, that is the difference between the original data and the actual output of the network output by the already configured layers. Hence, the residual, r_l, is computed as:

$$r_l(P_m) = r_{l-1}(P_m) - a_l(P_m) \tag{4}$$

and it is used for estimating the parameters of the l-th layer. $r_0(P_m) = z_m$ is also assumed.

The Gaussians of a new layer are inserted only where a poor approximation is obtained from the previous layers. This is evaluated, for each Gaussian, $P_{l,k}$, through an integral measure of the residuals inside the receptive field of that Gaussian, $A(P_{l,k})$. This measure, which represents the *local residual error*, $R(P_{l,k})$, is computed as the l_1 norm of the local residual as:

$$R(P_{l,k}) = \frac{\displaystyle\sum_{P_m \in A(P_{l,k})} |r_{l-1}(P_m)|}{|A(P_{l,k})|}. \tag{5}$$

When $R(P_{l,k})$ is over a given threshold, ϵ, the Gaussian is inserted in the corresponding grid crossing of the current layer under construction. As a result, Gaussians at a smaller scales are inserted only in those regions where there are still some missing details, forming a sparse approximation of the data. The introduction of new layers ends when the residual error is under threshold over the entire domain (uniform approximation).

This approach has been compared with classical multiresolution analysis through wavelet basis, and it has proved superior when approximation of noisy data is required [5].

III. KERNEL REGRESSION

In statistics, the estimate of the values of a function, $g(\cdot)$, given a set of observations affected by noise, $\{(X_i,\, Y_i)\,|\, i = 1,\, \ldots,\, n,\, Y_i = g(X_i) + \epsilon_i\}$, can be formulated as a regression problem, i.e., $g(x) = E(Y \,|\, X = x)$. Kernel regression is a technique used for non-parametric regression, which estimates the value of $g(x)$ as a weighted sum of those observations that belong to a suitable neighborhood of x. Different techniques can be found in literature [11], which are characterized by the

way they use the observations for computing the weights. The selected observations are weighted using a suitable function, $K(\cdot)$, called *kernel*, usually monotonically decreasing, which allows to contribute more to the observations closer to the estimation point. The size of the neighborhood is usually regulated by a parameter, h, called *bandwidth*. Hence, h allows to select the observations that will be used in the estimate. If h is too small, few observations may be selected causing an unreliable estimate. On the other hand, if h is too high, observations very distant from the estimation point will be considered, and the estimate will results in an average of points that belong to very different regions, loosing the local details. The choice of a suitable value of h for a given set of observations is known as the observations selection problem. For facing this problem, variable bandwidth selection techniques have been investigated [12].

Among all, we considered Nadaraya-Watson (NW), Local Polynomial (LP), and Gasser-Müller (GM) techniques.

The NW estimator has the following general form:

$$\hat{g}(x) = \frac{\sum_i K(\|X_i - x\|/h)\, Y_i}{\sum_i K(\|X_i - x\|/h)} \tag{6}$$

The LP estimator is based on fitting a n-th degree polynomial function, $\hat{p}(t;\, x)$, in the neighborhood of the estimation point, x. For an univariate estimation, the estimator is $\hat{p}(t;\, x) = \sum_{j=0}^{n} \hat{\beta}_j(x)\, (\|t - x\|)^j$, while for multivariate estimation it can be expressed by a more complex formula. As we deal with a surface reconstruction problem, we used a bivariate polynomial regressor. In particular, we used a linear ($n = 1$) and quadratic ($n = 2$) polynomial regressors, which we refer to as LP1 and LP2, respectively. In the estimation point, x, $\hat{p}(t;\, x)$ assumes the value of the constant term, $\hat{g}(x) = \hat{\beta}_0(x)$ (which is the value of $\hat{p}(\cdot)$ for $t = x$). The parameters $\{\hat{\beta}_j(x)\}$ are computed minimizing:

$$\sum_i (Y_i - \hat{p}(X_i - x;\, x))^2\, K(\|X_i - x\|/h) \tag{7}$$

The number of observation used in the estimation cannot be less than the number of estimated parameters, which depends on n and the number of variables of the polynomial (3 for LP1 and 6 for LP2).

The GM estimator has the form:

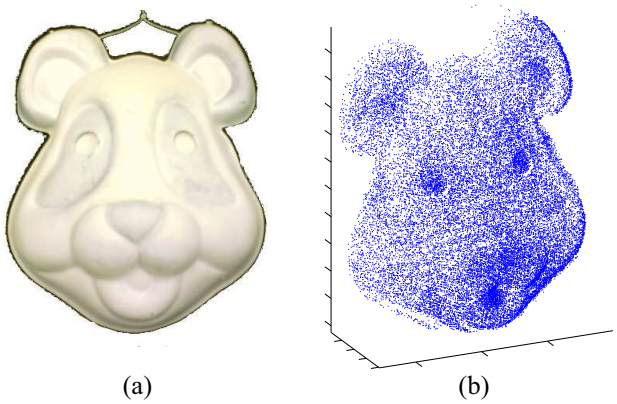
$$\hat{g}(x) = \sum_i \int_{s_{i-1}}^{s_i} K(\|x - u\|/h)\mathrm{d}u\, Y_i \tag{8}$$

where $s_i = (X_i + X_{i+1})/2$, ($s_0 = 0$, and $s_n = \infty$), and the observations are sorted.

For each of these estimators we considered the following kernels with unitary integral over their domains:

$$K_1(u) = \frac{3}{2}(1 - u^2)\, I_{[0,\,1]}(u) \tag{9a}$$

$$K_2(u) = \frac{15}{8}(1 - u^2)^2\, I_{[0,\,1]}(u) \tag{9b}$$



(a) (b)

Fig. 1. The real object (a) and the acquired dataset (b) used for the experiments.

$$K_3(u) = \frac{\pi}{2} cos\left(\frac{\pi}{2}u\right)\, I_{[0,\,1]}(u) \tag{9c}$$

$$K_4(u) = \frac{1}{2\sqrt{2\pi}} e^{-\frac{u^2}{2}}\, I_{[0,\,\infty)}(u) \tag{9d}$$

where I_A is the indicator function of the set A.

It should be noticed that kernels K_1, K_2, and K_3 use only observations that belong to the h-spherical neighborhood of the point x, while the support of the kernel K_4 is not finite.

The statistical properties of the considered kernel regression techniques have been widely studied [13][14]: in the asymptotic case, they all converge to the expected value, $\hat{g}(x) = E(Y|X = x)$, independently by the kernel, $K(\cdot)$, used. However, in practical applications, the choice of the kernel and especially the bandwidth parameter, h, turn out to be very critical, both for the accuracy and the computational resources required by the estimate [11]. Moreover, the effectiveness of the considered estimation technique can depend on the characteristics of the observed data set (e.g., measurement noise, sampling density).

Although in the original formulation of the HRBF algorithm the Nadaraya-Watson regressor (6) with the K_4 kernel (9d) was used, this choice could be questioned. For this scope, several experiments have been carried out, as reported in the next section, to assess if other regressors may produce better results.

IV. EXPERIMENTAL RESULTS

In order to assess the effectiveness of kernel regression and HRBF with different regressors for the reconstruction of the function $\tilde{S}(\cdot)$ in (3), we applied them to the problem of surface reconstruction from points clouds.

We used two datasets, “Panda”, reported in Fig. 1, and “Doll”, acquired by a 3D scanner [15][16]. From these datasets we derived other two datasets, “Panda_noise” and “Doll_noise”, obtained by injecting additional random noise

(Gaussian isotropic white noise, 1 mm of amplitude) to "Panda" and "Doll" data, respectively.

For each regressor, the reconstruction has been evaluated in terms of accuracy (mean absolute error), network compactness (number of Gaussians used), and computational time (time required for configuring the network). In order to estimate the accuracy, each dataset has been partitioned in a training (32000 points for "Panda", and 16000 points for "Doll") and a test (1000 points) data set.

Two experiments have been carried out. In the first experiment, 16 HRBF networks are configured using one of the four considered regressors (NW, LP1, LP2, GM), and for each regressor one of the four considered kernel function (K_1, K_2, K_3, K_4). In the second experiment, we configured each layer of the HRBF network by choosing the regressor-kernel combination that achieved the minimal residual error for that considered layer.

As the bandwidth parameter, h, operates as a selector of the sampled points considered for the estimate, it has been set as the receptive field size for the first three kernels, while it has been set as one third of this value for the fourth kernel, K_4. This value has been chosen for uniformity with the other three kernels: with this set up, K_4 assumes a value close to zero for points distant h from the estimate point, as the other kernels. Besides, this value is quite in accordance with the regressor used in the traditional HRBF setup.

The configuration algorithms have been implemented in Matlab (R2007a) and the experiments run on an Intel Pentium 4 CPU, 2.40 GHz, 512 MiB RAM. The computational time measurements reported in this paper have been obtained by the `cputime` function of Matlab. To limit variability, time has been measured as the average of 5 runs.

The performance of the network configured in the two experiments in which the different regressors were used, has been compared with that obtained through kernel regression. This allows evaluating the convenience of using HRBF networks for this kind of problems. Besides, it allows evaluating the behavior of each regressor when applied to the considered datasets.

As mentioned above, one of the main problem in kernel regression is the selection of a suitable value for h. For overcoming the problem of the bandwidth selection, we implemented a simple algorithm of variable bandwidth. We started with a size of h equal to the receptive field size of the last layer of the HRBF network, and doubling it until the number of points that belong to the considered neighborhood is large enough for a good estimate (6 for the LP1, 12 for LP2, and 3 for the other regressors).

In Table I and Table II, results of the first experiment are reported. As the error threshold, ϵ, is set to 0.4 mm for "Panda" and 0.9 mm for "Panda_noise", the HRBF network with all the regressors achieves the required reconstruction accuracy, although the LP2 regressor allows a reconstruction error lower than the others (about 5% lower than NW for "Panda", 1% for "Panda_noise"). . More interesting is the number of Gaussians

TABLE I. Reconstruction with single regressor HRBF

	Panda			Panda with noise		
Regressor	E [mm]	#gauss	t [s]	E [mm]	#gauss	t [s]
NW K_1	0.410	10827	70.9	0.907	10671	70.6
NW K_2	0.399	10683	70.4	0.903	10496	70.1
NW K_3	0.408	10806	71.4	0.906	10649	71.0
NW K_4	0.394	10572	70.9	0.901	10403	70.6
LP1 K_1	0.410	10681	77.9	0.907	10484	77.2
LP1 K_2	0.399	10563	77.4	0.902	10325	77.0
LP1 K_3	0.408	10664	78.3	0.906	10460	77.8
LP1 K_4	0.394	10471	78.0	0.900	10225	77.4
LP2 K_1	0.376	10111	80.1	0.894	9855	79.5
LP2 K_2	0.375	10085	80.1	0.894	9838	79.4
LP2 K_3	0.376	10107	80.7	0.894	9848	80.0
LP2 K_4	0.375	10062	80.6	0.894	9815	79.9
GM K_1	0.392	10500	141.0	0.901	10329	141.0
GM K_2	0.387	10386	158.0	0.899	10205	158.0
GM K_3	0.391	10478	157.0	0.901	10308	157.0
GM K_4	0.383	10296	308.0	0.898	10118	308.0

TABLE II. Reconstruction with single regressor HRBF

	Doll			Doll with noise		
Regressor	E [mm]	#gauss	t [s]	E [mm]	#gauss	t [s]
NW K_1	0.304	7243	38.4	0.849	7598	38.6
NW K_2	0.295	6863	38.0	0.845	7325	38.5
NW K_3	0.303	7176	38.7	0.848	7557	39.1
NW K_4	0.292	6688	38.3	0.843	7185	38.6
LP1 K_1	0.297	6541	41.7	0.845	7091	42.2
LP1 K_2	0.291	6302	41.4	0.842	6935	42.0
LP1 K_3	0.296	6502	42.0	0.844	7063	42.5
LP1 K_4	0.288	6192	41.7	0.840	6830	42.2
LP2 K_1	0.496	5855	42.5	2.709	6956	43.8
LP2 K_2	0.497	5867	42.7	2.704	7014	44.0
LP2 K_3	0.495	5851	42.9	2.708	6952	44.1
LP2 K_4	0.494	5795	42.8	2.692	6954	44.1
GM K_1	0.290	6380	70.6	0.842	7049	71.2
GM K_2	0.285	6128	77.2	0.839	6877	78.2
GM K_3	0.289	6336	77.7	0.841	7020	78.3
GM K_4	0.283	5990	142.0	0.838	6769	143.0

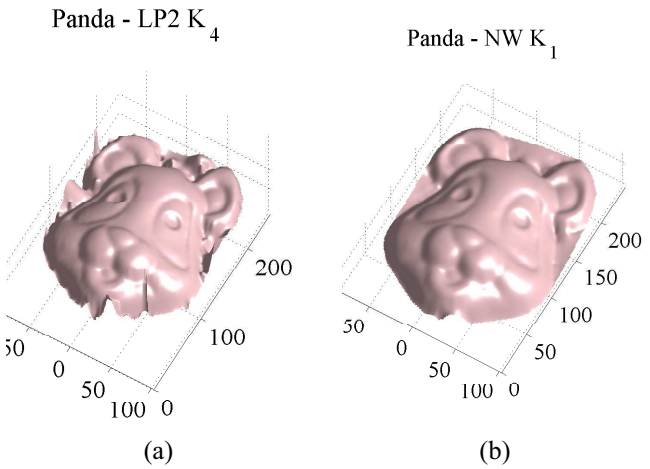


Fig. 2. Reconstructions of the "Panda" dataset performed by the LP2 K_4 (a) and NW K_1 (b) HRBF networks.

TABLE III. Reconstruction with a layer-wise regressor choice of Panda datasets

	Panda			Panda with noise		
l	#gauss	E [mm]	Regr.	#gauss	E [mm]	Regr.
1	1	43.9	LP2 K_4	1	43.9	LP2 K_4
2	4	26.4	NW K_4	4	26.4	NW K_4
3	16	9.55	NW K_3	16	9.62	NW K_3
4	62	5.31	GM K_4	62	5.39	GM K_4
5	204	2.50	GM K_4	204	2.65	GM K_4
6	678	1.38	GM K_4	678	1.61	GM K_4
7	2339	0.640	GM K_4	2347	1.04	GM K_4
8	6995	0.383	GM K_4	6810	0.898	GM K_4
	#gauss: 10298 conf. time: 658 s rec. time: 0.462 s			#gauss: 10122 conf. time: 655 s rec. time: 0.460 s		

used, as the LP2 regressor allows saving about 5.6% Gaussians with respect to the other regressors for the "Panda" dataset, and 6.3% in case of the "Panda_noise" dataset.

However, as the LP2 regressor requires a high number of points for a reliable estimate, when the dataset is not very dense, as the "Doll" dataset, its performance dramatically decrease. As can be noticed in Table II, when challenged on the "Doll" dataset, the HRBF networks using LP2 regressors have less Gaussians than the other networks. This can be explained as, in particular in the last layers where the neighborhoods are small, the LP2 regressor is unable to operate due to the scarcity of the data points. In the case of "Doll_noise", the estimate tends to overfit to noisy data, causing a large increment in the reconstruction error.

Qualitatively, there is only a slight difference between the surface reconstructed by the HRBF with the LP2 K_4 and the NW K_1 regressors (which achieved the lower and the higher reconstruction error, respectively), as shown in Fig. 2. Besides, it is apparent from Fig. 2a that the LP2 regressor may produce an unsatisfactory reconstruction in the regions close to the border, where the data scarcity induce overfitting. However, in the internal regions, the reconstruction is good for all the considered regressors.

The computational time required to configure the HRBF with the GM regressors nearly doubles the time required by the LP and NW regressors.

These results have been confirmed by using different datasets.

The results of the second experiment are reported in Table III. The use of the best approximator for each layer does not improve the performance with respect to the networks configured using the LP2 regressors (neither in accuracy, nor in compactness), despite the computational time spent (2 to 9 times the time required by the single regressor HRBFs). Similar results have been obtained for "Doll" and "Doll_noise" datasets.

It should be noticed that for the considered datasets the best regressor for each layer match those of the noisy versions dataset. In particular, the GM regressor tend to be used in the last layer.

TABLE IV. Results of the reconstruction operated by the variable bandwidth kernels on Panda datasets

	Panda		Panda with noise	
Regressor	E [mm]	time [s]	E [mm]	time [s]
NW K_1	0.352	2.96	0.893	2.95
NW K_2	0.346	2.95	0.901	2.95
NW K_3	0.351	2.97	0.894	2.98
NW K_4	0.343	2.96	0.907	2.97
LP1 K_1	0.335	3.35	0.890	3.37
LP1 K_2	0.330	3.34	0.898	3.37
LP1 K_3	0.334	3.35	0.891	3.39
LP1 K_4	0.328	3.36	0.904	3.38
LP2 K_1	0.337	3.44	0.948	3.46
LP2 K_2	0.344	3.43	0.971	3.47
LP2 K_3	0.338	3.44	0.950	3.49
LP2 K_4	0.341	3.45	0.972	3.48
GM K_1	0.354	3.98	0.933	4.01
GM K_2	0.358	4.15	0.954	4.19
GM K_3	0.355	4.12	0.936	4.15
GM K_4	0.364	5.75	0.978	5.81

TABLE V. Results of the reconstruction operated by the variable bandwidth kernels on Panda datasets

	Doll		Doll with noise	
Regressor	E [mm]	time [s]	E [mm]	time [s]
NW K_1	0.279	1.57	0.871	1.57
NW K_2	0.272	1.57	0.882	1.57
NW K_3	0.277	1.57	0.871	1.58
NW K_4	0.267	1.57	0.890	1.57
LP1 K_1	0.225	1.96	0.858	1.95
LP1 K_2	0.222	1.95	0.876	1.94
LP1 K_3	0.224	1.95	0.860	1.95
LP1 K_4	0.221	1.96	0.881	1.96
LP2 K_1	0.229	2.06	0.954	2.07
LP2 K_2	0.236	2.06	0.988	2.06
LP2 K_3	0.229	2.07	0.957	2.08
LP2 K_4	0.235	2.08	0.984	2.07
GM K_1	0.293	2.24	0.924	2.24
GM K_2	0.299	2.34	0.954	2.34
GM K_3	0.293	2.33	0.929	2.33
GM K_4	0.306	3.18	0.981	3.18

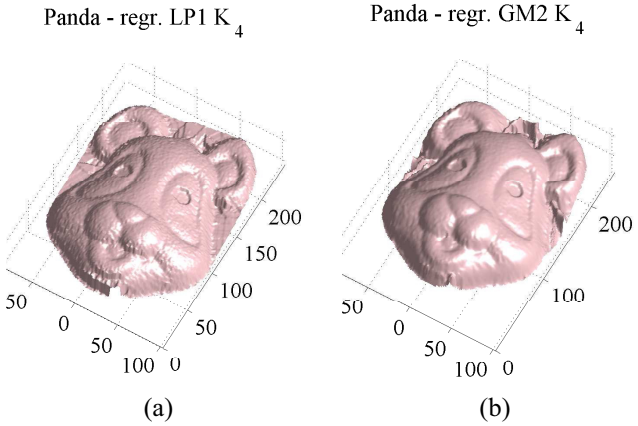


Fig. 3. Reconstruction of the "Panda" dataset performed by the kernel regression using LP1 with kernel K_4 (a) and using GM with kernel K_4 (b).

In Tables IV and V, the results for the reconstruction obtained by different kernel regression techniques are reported. Here the best results in terms of accuracy are produces by kernel regression employing LP1, while the other two regressors families achieve a quite small error. It can be noticed that, in case of noise, the LP2 regressors tend to perform worse than the other regressors.

However, as shown in Fig. 3, the reconstruction achieved is not smooth in any case. Besides, the computational time required for processing the 1000 points of the test set is more than 6 times the time required by the HRBF networks. Although the HRBF network needs nearly 80 s for being configured, when it is required to densely resample the surface, the use of the HRBF can be convenient with respect to kernel regression (not to mention the quality of the reconstruction, as in Fig. 2).

V. DISCUSSION

Kernel regression techniques have been widely investigated as a tool for function estimate [13][14][12]. One of the main open problems is the selection of the bandwidth parameter.

Naive strategies for selecting the best local bandwidth, like the one here implemented, give inaccurated estimation and are sensitive to noise and data sparsity: a too large bandwidth wipe out details, while a too small bandwidth produce overfitting. This is particularly true for LP2 regressors (or polynomial of higher order) that are very sensitive to data scarcity.

The same problem can occur in RBF networks, where several strategies have been studied for adapting the size of the Gaussian units during the learning phase [9][17]. HRBF make use of a kernel regression technique for a first estimate of the surface height in the center of the Gaussians, but thanks to its hierarchical structure, is able to overcome the problem of the bandwidth selection.

In the experiments we described in section IV, the best regressors are usually those that belongs to the LP family. This can be explained because they performs well when a high number of observations are available, as it happens in the first layers, and the filtering action of the Gaussians allows to temper possible local bad estimates. However, it should be noticed that NW regressor require less computational time.

Computing more than one network per layer, as in the second experiment in section IV, is computationally expensive. However, the cost is not linear in the number of regressors, as the regressors have different computational cost and they can share a considerable amount of operations. Hence, the method can be improved selecting few regressors that features complementary characteristics. In that case, when one regressor fails the estimate the surface (e.g., when few points are available), a second one (e.g., one which requires less points for a reliable estimate) may provide an alternative estimate. This strategy requires a longer configuration time, but may worth the effort.

VI. CONCLUSIONS

In this paper, the use of different regressors for HRBF configuration has been investigated. Four kernel regression techniques and four kernels have been combined for being used in HRBF networks, and have been extensively challenged in the problem of surface reconstruction.

Results show that, although the HRBF performance is robust with respect to the regressor used, the use of LP2 regressors may generally improve the accuracy and the compactness of the network with respect to the traditional NW regressor.

As the performance of each regressor depends by the observation set, the use of more regressors with complementary characteristics may grant an advantage with respect the use of a single one. Future works will be focused on the selection of few regressors that may used to recover each other in case of bad estimation.

REFERENCES

[1] H. Hoppe, "Progressive meshes," *Proceedings of Siggraph 96*, pp. 99–108, 1996.

[2] M. Lounsbery, T. DeRose, and J. Warren, "Multiresolution analysis for surfaces of arbitrary topological type," *ACM Trans. on Graphics*, vol. 16, no. 1, pp. 34–73, 1997.

[3] I. Guskov, K. Vidimče, W. Sweldens, and P. Schröeder, "Normal meshes," *Proceedings of Siggraph 2000*, pp. 95–102, 2000.

[4] I. Friedel, P. Schröeder, and A. Khodakovsky, "Variational normal meshes," *ACM Trans. on Graphics*, vol. 23, no. 4, pp. 1061–1073, 2004.

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

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

[7] 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.

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

[9] F. Girosi, M. Jones, and T. Poggio, "Regularization theory and neural networks architectures," *Neural Computation*, vol. 7, no. 2, pp. 219–269, 1995.

[10] C. M. Bishop, *Pattern Recognition and Machine Learning*, ser. Information Science and Statistics. Springer, 2006.

[11] D. F. Specht, "Probabilistic neural networks and general regression neural networks," pp. 301–344, 1996.

[12] J. Fan and I. Gijbels, "Data-driven bandwidth selection in local polynomial fitting: Variable bandwidth and spatial adaptation," *J. R. Statist.Soc. Series B (Methodological)*, vol. 57, no. 2, pp. 371–394, 1995.

[13] T. Hastie and C. Loader, "Local regression: Automatic kernel carpentry," *Statistical Science*, vol. 8, pp. 120–143, 1993.

[14] C. Stone, "Consistent nonparametric regression," *Ann. Statist.*, vol. 5, no. 4, pp. 595–620, 1977.

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

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

[17] B. Fritzke, "Growing cell structures — A self-organizing network for unsupervised and supervised learning," *Neural Networks*, vol. 7, no. 9, pp. 1441–1460, 1994.