RECONSTRUCTION OF FREEFORM SURFACE BY SUPPORT VECTOR REGRESSION

Ling Jing and Ling Zhen
Science of College, China Agriculture University, Beijing,100083,China

Abstract: Reconstruction and manufacturing of existing freeform surfaces are of paramount importance for reverse engineering. The paper presents support vector regression (SVR) to reconstruction of computer models for existing freeform surfaces. Through examples, the effective is compared among different methods, and the influence of kernels to the precision is discussed. The results show that SVR is better than the algorithms given in [3] and [4], when it is used to reconstruct freeform surface.

Key words: reverse engineering, freeform surface, support vector regression, kernel

1. INTRODUCTION

One of the main issues in reverse engineering is to generate models or design representations based on the existing products[1-2]. There are a variety of engineering applications. For example, in the automobile, shipbuilding, or aircraft industries, designers may create a physical model based on the functional requirements and analysis. The model does not have a mathematical representation. For simple geometry, it is not difficult to generate such a representation in terms of drawings although it is very time consuming. If the model has very complex surface such as freeform surface, the development of such design representations or models becomes difficult.

A neural network approach is presented in [3]. To evaluate the effectiveness of the approach, a known non-uniform B-spline surface was

Supported by the National Nature Science Foundation of China (No. 10371131)

used for generating a number of samples for training the networks. There 4-layered neural networks were designed and trained using a modified back propagation algorithm. Taking advantage of the global minimum property of Simulated Annealing Procedure, a new technique[4] is proposed to accept a temporally failed training result in accordance to probability. This method is able to jump out of the local minimum and converge to the global minimum in some cases, but easily leads to the problem of over-fitting .

Support vector machines have become a subject of intensive study. They have been applied successfully to classification and also to regression. In this paper we use support vector machines in the field of the reconstruction of freeform surfaces and we find that they show an excellent performance.

In the following sections we will introduce the basic principles of ε-support vector regression (ε-SVR) and ν-support vector regression (ν-SVR) in reconstruction of freeform surfaces. The experimental section considers a comparison of ε-SVR, ν-SVR and other methods. Furthermore we compare the influence of different types of kernel function. A brief discussion concludes the paper.

2. ε-SVR AND ν-SVR

In order to model and machine an existing freeform surface, the freeform surface is first digitized by sampling a number of points. Our task is to find a regression function based on the digitized sample points—the training points.

Suppose the training data are given by a training set S of labeled inputs

$$S = \{(U_1, P_1), (U_2, P_2), \cdots, (U_l, P_l)\} \tag{1}$$

For $i = 1, \cdots, l$, the input $U_i = (u_i, v_i) \in R^2$ is the parameter of digitized point $P_i = (x_i, y_i, z_i) \in R^3$ of the freeform surface.

A parametric surface $p(u,v)$ can be decomposed into three surfaces $x(u,v), y(u,v)$ and $z(u,v)$ if the surface is a function of the parametric variables u and v. To reconstruct the surface in terms of points, we are interested in the relationships between the parametric variables u, v and the coordinate values of x,y and z. Thus, SVR process is designed as three separate process corresponding to the three functions $x(u,v), y(u,v)$ and $z(u,v)$ .We will only describe the process to find $x(u,v)$ below.

The input $U = (u,v)$ is first mapped to a feature space by a function Φ. Assume that regression function $x(u, v)$ takes the following form:

$$f(U) = (w \cdot \Phi(U)) + b \tag{2}$$

where $\Phi : R^2 \to H$, H is a Hilbert space, $w \in H, b \in R$. Let the loss function is the ε-insensitive loss function

$$c(U,x,f(U))=\tilde{c}(f(U)-x)=|x-f(U)|_{\varepsilon}=\max\{0,|x-f(U)|-\varepsilon\} \quad (3)$$

Where $\varepsilon>0$. This function ignores errors that are within a certain distance of the true value.

Suppose that ε is given in priori, the primal problem is[5-6]:

$$\begin{aligned} &\min_{w,b,\xi,\xi^*} \quad \tau(w,b,\xi_i,\xi_i^*)=\frac{1}{2}\|w\|^2+C\cdot\frac{1}{l}\sum_{i=1}^{l}(\xi_i+\xi_i^*) \\ &s.t. \quad \begin{cases}(w\cdot\Phi(U_i))+b-x_i\le\varepsilon+\xi_i \\ x_i-(w\cdot\Phi(U_i))-b\le\varepsilon+\xi_i^* \quad (i=1,...,l) \\ \xi_i,\xi_i^* \quad \ge 0\end{cases} \end{aligned} \quad (4)$$

where $\xi=(\xi_1,\xi_2,\cdots,\xi_l)^{\mathrm{T}},\xi^*=(\xi_1^*,\xi_2^*,\cdots,\xi_l^*)^{\mathrm{T}}$. $C\geqslant 0$, $\varepsilon>0$ are constants . In order to solve problem (4), SVR introduces its dual problem

$$\begin{aligned} &\min_{\alpha_i,\alpha_i^*} \quad \frac{1}{2}\sum_{i,j=1}^{l}(\alpha_i^*-\alpha_i)(\alpha_j^*-\alpha_j)K(U_i,U_j)+\varepsilon\sum_{i=1}^{l}(\alpha_i^*+\alpha_i)-\sum_{i=1}^{l}(\alpha_i-\alpha_i^*)x_i \\ &s.t. \quad \begin{cases}\sum_{i=1}^{l}(\alpha_i-\alpha_i^*)=0 \\ \alpha_i,\alpha_i^*\in[0,C/l], \quad (i=1,...,l)\end{cases} \end{aligned} \quad (5)$$

Where $\alpha=(\alpha_1,\alpha_2,\cdots,\alpha_l)^{\mathrm{T}},\alpha^*=(\alpha_1^*,\alpha_2^*,\cdots,\alpha_l^*)^{\mathrm{T}},K(U_i,U_j)=(\Phi(U_i),\Phi(U_j))$ is the kernel function. After getting the solution α and α^* of problem (5), we obtain the decision function as

$$f(U)=\sum_{i=1}^{l}(\alpha_i^*-\alpha_i)K(U_i,U)+b=\sum_{SV}(\alpha_i^*-\alpha_i)K(U_i,U)+b \quad (6)$$

Where b is determined by KKT conditions. SV means the support vectors.

It must be pointed out that if U_i satisfies to $|f(U_i)-x_i|<\varepsilon$, α_i, α_i^* must be zero by the KKT conditions. This means w in (2) relies only on the U_i with $\alpha_i-\alpha_i^*\neq 0$. We call these points the support vectors, this property makes SVR be applied in pre-handle the data in reverse engineering.

Although the parameter ε does control the sparseness of the solution, it does this only in a rather indirect way. Lacking a priori information about the accuracy of the x-values, it can be difficult to come up with a reasonable value of ε a priori. Instead, one would rather specify the degree of sparseness and let the algorithm compute ε from the data. This is the idea of the ν-SVM[5-6], a modification of the original ε-SVM. In the ν-SVM the size of ε is not defined a priori but is itself a variable. Its value is traded off against model complexity and slack variables via a constant ν :

$$\min_{w,b,\xi,\xi^*,\varepsilon} \quad \tau(w,b,\xi_i,\xi_i^*,\varepsilon) = \frac{1}{2}\|w\|^2 + C(\nu\varepsilon + \frac{1}{l}\sum_{i=1}^{l}(\xi_i + \xi_i^*))$$

$$s.t. \quad \begin{cases} (w\cdot U_i) + b - x_i \le \varepsilon + \xi_i \\ x_i - (w\cdot U_i) - b_i \le \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0 \quad , \quad \varepsilon \quad \ge 0 \end{cases} \tag{7}$$

where $C > 0$, $0 \le \nu \le 1$ are constants. The dual problem of problem (7) is

$$\min_{\alpha_i,\alpha_i^*} \quad \frac{1}{2}\sum_{i,j=1}^{l}(\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j)K(U_i,U_j) - \sum_{i=1}^{l}(\alpha_i^* - \alpha_i)x_i$$

$$s.t. \quad \begin{cases} \sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0 \\ \alpha_i, \alpha_i^* \in [0, C/l] \\ \sum_{i=1}^{l}(\alpha_i + \alpha_i^*) \le C\cdot\nu \end{cases} \tag{8}$$

We will use the term errors to refer to training points lying outside of the ε -tube, and the term fraction of errors/SVs to denote the relative numbers of errors/SVs, i.e. divided by l. This has been proved that ν can be used to control the fraction of support vectors (and hence the sparsity of the solution's expansion) and the fraction of outliers (i.e. the amount of confidence that we have in the data).

3. RECONSTRUCTION OF SURFACE

The example in [3-4] is now used to evaluate the effectiveness of SVR. Applying $\varepsilon - SVR$ and $\nu - SVR$, the reconstruction of the existing surface is realized by first training a series of points from a mathematically known surface in [3], the results were severed as models to generate all the points that are needed. The input is the two parametric variables $U = (u, v)$ of sample point. The output is the point of reconstruction surface. The expected output is the accurate point calculated using the NURBS equations. Then, we calculated a number of testing points using other set of $U = (u, v)$, the points and the differences between the points output from the predicting and the points generated by the NURBS equations indicated the performance of the different ways.

In learning process, the kernel form is chosen as:

$$K(U_i, U_j) = ((U_i \cdot U_j) + 1)^5$$

For the sake of simplicity, only the x-coordinate result is generated by different methods in *Table 1*. There x_1 is accurate value in the known surface, x_2 and x_3 are the results in [3] and [4], x_4 is given by $\varepsilon - SVR$ with the capacity constant $C = 225$ and $\varepsilon = 0.01$, and x_5 is obtained by $\nu - SVR$ with the capacity constant $C = 225$ and $\nu = 0.9$.

Table 1. x-coordinate generated by different methods

u	v	x_1	x_2	x_3	x_4	x_5
1.000000	1.000000	11.500000	11.447000	11.487403	11.5035	11.5099
1.000000	1.100000	12.264167	12.275300	12.266978	12.2589	12.2529
1.000000	1.200000	13.053333	13.102600	13.042305	13.0473	13.0398
1.000000	1.300000	13.862500	13.928400	13.870028	13.8597	13.8562
1.000000	1.400000	14.686677	14.752000	14.706395	14.688	14.6902
1.000000	1.500000	15.520833	15.572800	15.547921	15.5252	15.5322
1.000000	1.600000	16.360000	16.390100	16.391150	16.365	16.3744
1.000000	1.700000	17.199167	17.203300	17.232766	17.202	17.2113
1.000000	1.800000	18.033333	18.011800	18.048636	18.0321	18.0386
1.000000	1.900000	18.857500	18.815000	18.875552	18.8518	18.8539
1.000000	2.000000	19.666667	19.612400	19.717749	19.6586	19.6558
1.000000	2.100000	20.457167	20.403300	20.427167	20.4511	20.444
1.000000	2.200000	21.230667	21.187200	21.215274	21.2287	21.2192
1.000000	2.300000	21.990167	21.963600	21.989935	21.9918	21.9826
1.000000	2.400000	22.738677	22.732100	22.736344	22.7418	22.736
1.000000	2.500000	23.479167	23.492200	23.483185	23.481	23.4815
1.000000	2.600000	24.214667	24.243400	24.299485	24.2126	24.2212
1.000000	2.700000	24.948167	24.985300	24.964167	24.9409	24.9571
1.000000	2.800000	25.682667	25.717500	25.667085	25.671	25.6911
1.000000	2.900000	26.421167	26.439700	26.417486	26.4092	26.4242
1.000000	3.000000	27.166667	27.151500	26.945260	27.1625	27.1571
1.100000	1.000000	11.500000	11.446900	11.430027	11.5029	11.5105
1.100000	1.100000	12.264167	12.275200	12.269364	12.2588	12.2529
1.100000	1.200000	13.053333	13.102600	13.044518	13.0475	13.0394
1.100000	1.300000	13.862500	13.928400	13.871958	13.8603	13.8555
1.100000	1.400000	14.686677	14.752000	14.708116	14.6889	14.6893
1.100000	1.500000	15.520833	15.572800	15.549450	15.5262	15.5312
1.100000	1.600000	16.360000	16.390200	16.392507	16.3661	16.3735
1.100000	1.700000	17.199167	17.203400	17.233973	17.2032	17.2104
1.100000	1.800000	18.033333	18.011900	18.030717	18.0334	18.0379
1.100000	1.900000	18.857500	18.815200	18.899829	18.8531	18.8534
1.100000	2.000000	19.666667	19.612500	19.718640	19.66	19.6555
1.100000	2.100000	20.457167	20.403500	20.424773	20.4526	20.444
1.100000	2.200000	21.230667	21.187400	21.216083	21.2303	21.2194
1.100000	2.300000	21.990167	21.963900	21.990731	21.9935	21.983
1.100000	2.400000	22.738677	22.732400	22.747147	22.7436	22.7367
1.100000	2.500000	23.479167	23.492500	23.514025	23.4829	23.4824
1.100000	2.600000	24.214667	24.243700	24.300312	24.2147	24.2224

1.100000	2.700000	24.948167	24.985600	24.995192	24.9432	24.9585
1.100000	2.800000	25.682667	25.717900	25.668068	25.6736	25.6926
1.100000	2.900000	26.421167	26.440100	26.418542	26.412	26.4259
1.100000	3.000000	27.166667	27.152000	26.946396	27.1655	27.1589
max-error			0.0659	0.221407	0.011967	0.0144
min-error			0.004133	0.000232	3.3e-5	0.001967
average-error			0.034621	0.033148	0.004391	0.008415

The last three rows in *Table 1* indicate the maximum error, minimum error and the average error between the predicted value and the true value. The results show that the SVR approach is more effective than the methods used in [3] and [4].

4. DISCUSSION THE KERNEL

The use of the kernel function is the key of SVR to the reconstruction of the freeform surface. In order to observe the influence of different forms of kernel to approximate surface, different kernels are used , and the related results are showed in *Table 2*.Here we use $\varepsilon - SVR$ with the capacity constant $C = 225$ and $\varepsilon = 0.01$.

Table 2. the influence of different kernels

kernel function		max-error	min-error	average-error
dot kernel		0.371167	0.0005	0.179011
polynomial kernel	d = 2	0.1468	0.002	0.054185
	d = 5	0.018367	0	0.007824
radial kernel	$\gamma = 2$	0.792433	0.0033	0.185621
	$\gamma = 4$	0.634733	0.001267	0.142206
hyperbolic kernel	$a = 1, b = 1$	251.4227	2.1898	94.74889
	$a = 0.0001, b = 1$	0.294233	0.008967	0.110213
	$a = 0.0001, b = 0.0001$	0.283133	0.012667	0.107197

5. CONCLUSION

One of challenges in reverse engineering is to reconstruct computer representations of existing surfaces, based on digitized points. The problem becomes even more difficult when a surface is partially damaged or worn. Surface fitting is an obvious technique, however, to fit an existing surface, appropriate surface equations are first selected, and then the equations are evaluated to determine whether the approximation is acceptable. Selecting and evaluating the models remain difficult tasks. It will be difficult to use the surface fitting technique if the surface is partially damaged or worn or incompleted.

In this work we show that SVR can be used to reconstruction the freeform surfaces in reverse engineering. It not only gives the formula of the freeform surface but also predicts the position of input parameters which aren't be solved in [3-4].

The parameters ε and ν provide the user to construct freeform surface and control the errors at the same time. How to given the appropriate forms of kernel function is worth studying for the future.

REFERENCES

[1] Bolle R M, Vemuri B C. On three-dimensional surface reconstruction methods. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1991, 13 (1): 1~13

[2] Varady T, Martin R R, and Cox J. Reverse engineering of geometric models-an introduction. Computer Aided Design, 1997, 29 (4): 255~268

[3] Gu P, Yan X. Neural network approach to the reconstruction of freeform surface for reverse engineering. Computer-Aided Design, 1995, 27: 59~64

[4] Wang K, Zhang C M, Neural network method to reconstruct the freeform surface. Journal of Computer-Aided Design & Computer Graphics, 1998, 10 (3): 193~199

[5] Nello Cristianini and John Shawe-Taylor. An introduction to Support vector machines. Cambridge University Press,Cambridge,UK,2000

[6] Deng N Y, Tian Y J. The new method in data mining— Support Vector Machine. Science Press, Beijing, China, 2004