ESANN'2006 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 26-28 April 2006, d-side publi., ISBN 2-930307-06-4.

Freeform Surface Induction from Projected Planar Curves via Neural Networks*

Usman Khan, Abdelaziz Terchi, Sungwoo Lim, David Wright, Sheng-Feng Qin

Brunel University, School of Engineering and Design, Kingston Lane, Uxbridge, UB8 3PH
United Kingdom

Abstract. We propose a novel intelligent approach into 2D to 3D of on-line sketching in conceptual design. A Multilayer Perceptron (MLP) neural network is employed to construct 3D freeform surfaces from 2D freehand curves. Planar curves were used to represent the boundary strokes of a freeform surface patch and varied iteratively to produce a training set. Sampled curves were used to train and test the network. The results obtained demonstrate that the network successfully leaned the inverse-projection map and correctly inferred respective surfaces from curves previously unencountered.

1 Introduction

At the early stages of conceptual product design, a designer always tries to covert his new ideas into sketches as soon as possible. It can be argued that sketching is an essential activity for creative design because it permits a designer to explore and to evaluate ideas quickly [1]. It also assists the designer's short-term memory and facilitates communication with other people. When designers sketch shapes on a sheet of paper, they start with a fuzzy concept, which is then progressively refined. While numerous iterations are undertaken, the salient properties of the original idea are usually maintained. Recently, the desire to automate the early phase of the conceptual product design have given impetus to the development of intelligent tools to simulate the natural way of sketching, [2-4]. However, most existing approaches are restricted to two-dimensional and polygonal shapes, [5]. The generation of complex free-form surfaces is a challenging process, which surprisingly has received little attention in the literature.

2 Theoretical Framework

A planar sketch is produced by a projection of a 3D object onto an arbitrary 2D plane. Reconstruction attempts to get the 3D geometry from the 2D sketch, i.e., to recover the lost depth information from the inherently planar sketch. This process can be regarded as the inverse process of such projection. Obviously, inferring a 3D surface from a 2D curve is mathematically indeterminate, so the reconstruction should need additional information such as general assumptions or experiences. Humans seem to be able to accomplish this task with minimal difficulty and most observers of a sketch will agree on a particular interpretation. We demonstrate how an MLP neural network can be trained with backpropagation with momentum on a database of 2D-3D

* This work is supported by the British EPSRC (GR/S01712/01).

dependencies to approximate the inverse map in a compact and computationally efficient form.

A neural network has the ability to be trained to learn to solve a certain pattern recognition problem. It does this by trying to mime the way that a human identifies an object without using any sort of rules or recipes, [6]. The reason that neural networks are chosen over other means is that they have a remarkable ability to derive meaning from complicated or imprecise data. They can also be used to extract patterns or discover trends that may be too complicated to be recognised by eye or other computational techniques.

2.1 **Back Propagation with momentum**

In the backpropagation algorithm learning the weight changes are proportional to the gradient of the error. The larger the learning rate is, the larger the weight changes on each iteration, and the quicker the network learns. However, the size of the learning rate can also influence whether the network achieves a stable solution. If the learning rate is too large, then the weight changes n o longer approximate a gradient descent procedure and hence oscillation of the weights would often result.

The backpropagation learning algorithm is in essence a gradient descent optimisation strategy of a multidimensional energy surface in the weight space. Such strategy has inherently slow convergence. This trait becomes more pronounced when the eigenvalues of the corresponding Hessian matrix exhibit large spread. In such cases, the change in the cost function between successive iteration becomes oscillatory leading thus to slow convergence. One way to circumvent this problem is to add a momentum term. The momentum term has the following effects: 1) it smoothes the weight changes, that it smoothes the oscillations across the error valley; 2) when all the weights change in the same direction the momentum amplifies the learning rate thereby causing a faster convergence; and 3) enables the algorithm to escape from small local minima.

The momentum term introduces a kind of ‘inertia’ in the movement of the weight vector. Once the weights start moving in a particular direction in the weight space, they tend to continue moving along same direction. If the weight vector has sufficient momentum, it will be able to bypass local minima continue down the hill. This increases the movement speed along the ravine, and helps to prevent oscillations across it. This effect can also be regarded as a linear low-pass filtering of the gradient delta. The effect becomes more pronounced as the momentum term approaches 1. However, one has to be conservative in the choice of momentum because of an adverse effect of the momentum term: the ravines are normally curved, and in a bend the weight movement may be jump over a ravine wall, if too much momentum has been previously acquired.

The momentum parameter has to be appropriately selected for each problem. Typical values of the momentum are in the range 0.05 to 0.95. Values below 0.05 usually contribute little improvement relative to the backpropagation without momentum, while values above 0.95 often tend to cause divergence at bends. The momentum technique may be used both in batch and on-line training modes. In this paper the batch version is used.

3 Data Generation

The neural network requires training with a record of input and output data. This permits to set the network parameters (i.e. synaptic weights and biases). Training the net is accomplished through a learning algorithm that iteratively adjusts the network parameters until the mean squared error (MSE) between the predicted output and the desired output reaches a minimum.

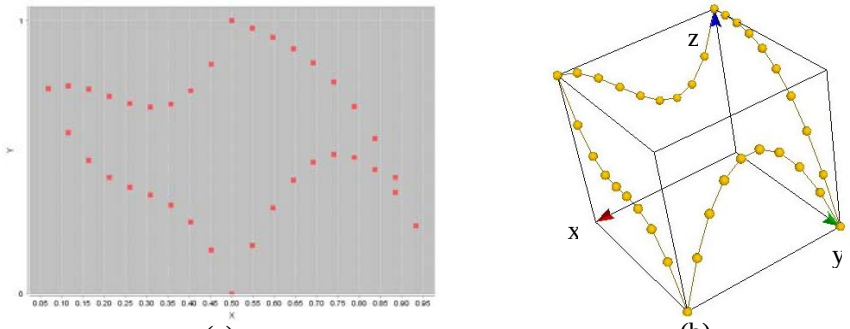
A training set was generated from a family of freeform surfaces whose boundaries that consist of four planar curves are arranged orthogonally. Each curve is governed by four independent control points and is represented as a Non Uniform Rational B-Spline (NURBS). Two control points are associated with the ends of the curve whereas the remaining ones control its general shape. It is worth noting that the control points need not intersect the curve and can be positioned anywhere in the 3D space. The parametric curve is uniformly sampled and the coordinates of the sample points form the input features for the neural network.

The curves were placed in the x-z plane or the y-z plane and their control points were only moved in the z-direction. Such action maintains their planar property. Along each boundary curve 10 sample points were selected. Hence a surface, whether in 2D or 3D, is represented by 40 sample points. A point in the 3D surface is represented by the coordinates x, y and z. whereas in 2D, is represented by the x, y coordinates. Therefore a 3D surface is represented by 120 features and its respective 2D with 80 features.

An iterative algorithm was used to vary the position of the control points to produce a class of unique freeform surfaces. Each surface was then projected onto a plane to produce the respective 2D curves. The training set is composed of pattern pairs, each containing a 3D surface and its corresponding 2D shape.

Most applications of neural networks include normalisation of the data sets in order to ensure that the values lie within the characteristic bounds of the activation functions. Therefore after the data set was generated, the maximum and minimum points were identified and the whole set was scaled accordingly by subtracting every point from the minimum value in that set and then dividing by the maximum value. Therefore the input 3D pattern would fit within a unit cube and its respective 2D pattern within the unit square.

Figure 1 shows an example of a typical normalised pattern pair that was used to train the neural network. A 2D input pattern is depicted in Figure 1 (a) whereas its corresponding 3D output pattern can be seen in Figure 1 (b). It can be seen that the boundary of the surface is described by a series of sample points and fits within a unit square for 2D and unit cube for 3D.



(a) (b)

Figure 1: Example of 2D input pattern and corresponding 3D output pattern

Notice that the camera is positioned for the 3D pattern according to view of the 2D input pattern where the axes are positioned in a way that it resembles the 2D view of the 3D surface.

The entire data set cannot be used to train the network as a whole because there would be no data left to test the network's ability to generalise into fresh inputs. Therefore the data set was randomly split, using a proportion of 70%, 20% and 10% into three subsets that are used for training, validation and testing. The size of the whole data set contained 4096 patterns and was generated as mentioned using the procedure above. Accordingly, the number of training, validation and testing patterns pairs were therefore 2867, 819 and 410 respectively. The generated set covered a wide range of the possible forms that the surface could take and provided good results.

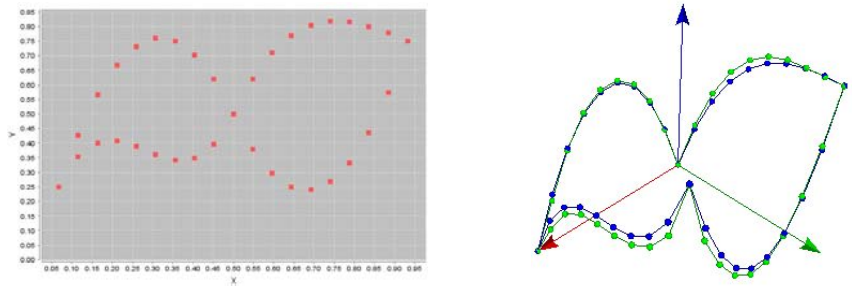
3.1 **Experimental Results**

From the above information regarding the dimensions of the input patterns the number of neurons used in the input and output layers were set to correspond to the dataset used to train it. A three-layer MLP network was employed in our research. The input and output layer dimensions of the neural network were determined from the number of features in the data set. The input layer consists of 80 nodes and while the output layer consists of 120 nodes. The number of nodes in the hidden layer is freely adjustable and results in different network performance depending on the number of hidden nodes used.

The number of hidden nodes indicates the network complexity and governs how accurately it learns the mapping from the input patterns to the outputs. It also affects the time it takes the network to perform each training cycle. The higher the number of hidden nodes the more computation is required and hence a longer training time. The same number of hidden nodes as the output layer were used resulting in the network architecture having 80 input nodes, 120 hidden nodes and 120 output nodes. The network was trained and validated for a fixed number of training cycles using the back propagation learning algorithm with momentum and validated at regular intervals. The learning rate and the momentum term are 0.7 and 0.4 respectively. At the end of 250 training cycles the net was saved the test set was applied to the network.

The purpose of validation is to check the network ability to generalise with new patterns that are not in the training set. This is an important activity to prevent the problem of overfitting, [7]. This was achieved by creating a copy of the neural network and freezing its parameters. The validation set was then applied and the validation error recorded. The original network allowed to train further until the specified training cycles were reached. The final training error was 0.067 and validation error was 0.012.

After the training was completed, the test set was applied to the neural network. The obtained results show that the neural network was able to infer the 3D shape of a freeform surface from its respective 2D input pattern.



(a) (b)

Figure 2: Test input pattern with predicted and desired output

The test pattern that was applied to the trained network is shown in Figure 2 (a). The predicted pattern and the expected pattern that corresponds to the 2D surface from Figure 2 (a) are shown in Figure 2 (b). The predicted pattern is depicted in green whereas the desired pattern is in blue. It can be noticed from Figure 2 (b) that the two surfaces are almost identical and hence that the neural network has inferred the correct shape that was desired. However, small deviations in the predicted pattern can be observed. They relate to the network's ability to predict the desired surface. The distribution of error presented in Figure 3. This shows the Euclidean distance between each point from the predicted surface and its corresponding point on the desired surface. The RMS error for this pattern was 1.97%.

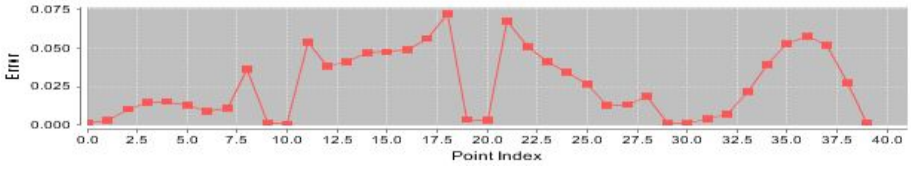


Figure 3: Distribution of squared errors between predicted output and expected output

3.2 **Conclusions**

In this paper a neural network approach for the inference of 3D freeform surfaces from 2D data has been presented. A dataset was generated by iteratively adjusting control points of freeform surface boundary curves and sampled uniformly along the curves. The dataset was normalised and randomly split into subsets. An MLP was employed and trained with a representative family of 2D and 3D pattern pairs and applied to another subset of patterns it had not encountered before. Results showed that the network was able to train well and reproduce the desired freeform surface from a 2D input pattern. Future work will involve extending sketch-based recognition techniques to more complex shapes and skinning over the inferred surface.

References

[1] Lim S, Lee B, Duffy A. Incremental modelling of ambiguous geometric ideas (I-MAGI): representation and maintenance of vague geometry. Artificial Intelligence in Engineering 2001;15:93-108.

[2] Karpenko O, Hughes J, Raskar R. Free-form Sketching with variational implicit surfaces. Eurographics 2002;21 (3):585-94.

[3] Igarashi T, Matsuoka S, Tanaka H. Teddy: A Sketching Interface for 3D Freeform Design. 26th International Conference on Computer Graphics and Interactive Techniques, 1999. pp. 409-16.

[4] Alexe A, Gaildrat V, Barth L. Interactive Modelling from Sketches using Spherical Implicit Functions. Computer Graphics, Virtual Reality, Visulisation and Interaction in Africa. Proceedings of the 3rd International Conference on Computer Graphics, Virtual Reality, Visulisation and Interaction in Africa. Africa, 2004. pp. 25-34.

[5] Lipson H, Shpitalni M. Optimization-based reconstruction of a 3D object from a single freehand line drawing. Computer Aided Design 1996;28 (8):651-63.

[6] Nezis K, Vosniakos G. Recognizing 2 1/2D shape features using a neural network and heuristics. Computer-Aided Design 1997;29 (7):523-39.

[7] Swingler K. Applying Neural Networks, A Practical Approach: Academic Press Limited, 1996.