# DEVELOPABLE SURFACE MODELLING BY NEURAL NETWORK

MIKLÓS HOFFMANN AND EMŐD KOVÁCS

*Dedicated to Professor Mátyás Arató and László Varga on the occasion of their 70th birthday.*

Abstract. In this paper the construction of developable NURBS surface for a set of tangent planes is presented. The problem is handled by the concept of duality of projective spaces. Using a special distance function a curve modelling is solved by a Kohonen neural network approach.

## 1. Introduction

Ruled surfaces and especially developable surfaces are well-known and widely used in computer aided design and manufacture. Since these fields apply B-spline or NURBS surfaces as de facto standard description methods, it is highly desired to use these methods to construct ruled and developable surfaces from any type of data. These data can be scattered points or given lines or a set of tangent planes, as well. Since these special surfaces possess a wide range of applications, e.g., from ship hulls to sheet metal forming processes, one can find several algorithms solving this problem. Some of the previous methods give special derive conditions for these surfaces [1-3], but most of the recent methods formulate the problem in a projective space depending on the input data [4-8]. Using famous classical results of projective geometry one can consider, e.g., the developable surface as a dual B-spline curve. In this paper, we will follow this latter concept.

On the other hand, a neural network based approximation method has been developed for scattered data resulting general type of B-spline surfaces [9-11]. In this method the Kohonen neural network [12], a self-organizing artificial intelligence tool helps us to handle the given data in a preprocessing step, after which the standard approximation methods can be used. During the preprocessing step the neural network produces a topologically quadrilateral grid for surfaces or a polygon for curves based on the given data, the vertices of which will be the input control points of the surface or the curve. The main goal of this paper is the application of this algorithm for developable surfaces using the above mentioned results of projective geometry. Since some of our problems can be formulated in spaces of geometrical objects instead of point-spaces, we apply some recently defined measuring methods and general concepts for approximation in these spaces.

*Key words and phrases.* Developable surface, NURBS, Kohonen neural network, Dual projective space.

This research was supported by the Hungarian National Foundation for Scientific Reseach Grant No. F032679 and Hungarian Ministry of Education (Grant No. FKP 2001/0027).

## 2. Basic geometrical concepts

For the sake of completeness we begin our discussion with some basic definitions.

**Definition 1.** *A surface in $E^3$ is called ruled surface, if through every point of the surface there passes a line which lies entirely on the surface. These lines are called rulings of the surface.*

This means, that a ruled surface is covered by a one parameter set of lines, the rulings. These lines have the following property: a ruling line lies in the tangent plane of the surface at every point of the ruling. This property is important in terms of the special ruled surfaces, called developable surfaces. The original definition of these surfaces is the following.

**Definition 2.** *A surface in $E^3$ is called developable surface, if it can be isometrically mapped (i.e., developed) onto a plane.*

Considering the property of the ruled surfaces mentioned above and this latter definition, one can easily see the following, well-known result.

**Theorem 1.** *A ruled surface is developable, if and only if the tangent planes at all points of an arbitrary ruling coincide.*

If the tangent planes vary along the ruling (i.e., they are different from each other), the surface is called general (non developable) ruled surface. This means that considering the tangent planes of a surface, the developable surface can be described as an envelope of a one parameter family of planes, while the set of tangent planes of a ruled surface has a second parameter along the rulings.

For our approach the concept of duality in the three dimensional projective space $P^3$ is also relevant. In this space points have 4 homogeneous coordinates $(x_0, x_1, x_2, x_3)$, while a plane can be represented by an equation $v_0x_0 + v_1x_1 + v_2x_2 + v_3x_3 = 0$, that is by the four-tuple $(v_0, v_1, v_2, v_3)$, the "coordinates" of the plane, hence analytically a point and a plane has the same representation in $P^3$. If they have the same coordinates, the point and the plane are called *duals*.

The principle of duality can be applied to describe developable NURBS surfaces as dual spatial NURBS curves, as we can see from the following definition.

**Definition 3.** *A nonuniform rational B-spline, i.e., NURBS curve $\mathbf{s}(t)$ can be written in the following homogeneous form*

$$\mathbf{s}(t) = \sum_{i=0}^{n} N_i^k(t)\mathbf{p}_i,$$

*where $N_i^k(t)$ are the normalized B-spline functions of order $k$ over a knot vector $t_0 = t_1 = ... = t_k < t_{k+1} < ... < t_n < t_{n+1} = ... = t_{n+k+1}$, while $\mathbf{p}_i(w_i, w_ix_i, w_iy_i, w_iz_i)$ are the control points of the curve.*

*The dual form of this curve is*

$$\mathbf{U}(t) = \sum_{i=0}^{n} N_i^k(t)\mathbf{U}_i,$$

*with planes $\mathbf{U}_i$ $(w_i, w_ix_i, w_iy_i, w_iz_i)$.*

This dual form can be interpreted as an envelope of a one parameter set of planes, i.e., this is a developable NURBS surface. One can easily transform this dual form to the more convenient tensor product form of the surface (see [7]), but for our purpose this dual form will be preferred. Detailed description and additional properties of this dual form can be found in [4, 7, 8].

The above mentioned definition yields, that if a set of planes are given one can find the developable NURBS surface as an envelope of these planes with a curve approximation technique of scattered points in the dual space. In this space, however, first an appropriate measure has to be found, which can produce a distance-like value for planes in an area of interest. This is necessary for us because in the preprocessing step the neural network has to compare the given data with its own data by their distance.

## 3. Distance function in the space of planes

The idea described in this section has been recently developed by Pottmann et al., and the detailed description can be found, e.g., in [8]. Euclidean distance and invariants of planes, based on the angle of the two planes are inappropriate for our purpose, because we have to measure the difference between two planes only in an area of interest, and this difference can be arbitrarily large meanwhile the angle between the planes is arbitrarily small.

The dual space of planes $P^*$ is isomorphic to $P^3$. To introduce a Euclidean metric we need at most an affine space, hence we have to remove a plane from $P^3$, a hyperplane from the dual space $P^*$. This hyperplane is a bundle of planes passing through a point at infinity, e.g., the ideal point of the z-axis. Hence, if we restrict our consideration for planes not parallel to the z-axis, the space of these planes is an affine space $A^*$. The analytical representation of these planes is given by

$$z = u_0 + u_1 x + u_2 y,$$

while the homogeneous coordinates are $(u_0, u_1, u_2, -1)$ which can be considered as affine coordinates $(u_0, u_1, u_2)$ of a plane in $A^*$. The planes of this space are not parallel to the $z$-axis, hence the area of interest can be an area $D$ in the $xy$-plane. Now, we can define the special distance of two planes.

**Definition 4.** *If* $\mathbf{U}_1\,(u_{0,1}, u_{1,1}, u_{2,1})$ *and* $\mathbf{U}_2\,(u_{0,2}, u_{1,2}, u_{2,2})$ *are two planes in* $A^*$*, then their distance over* $D$ *is the following*

$$d_\mu(\mathbf{U}_1, \mathbf{U}_2) = \|(u_{0,1} - u_{0,2}) + (u_{1,1} - u_{1,2})x + (u_{2,1} - u_{2,2})y\|_{L^2(\mu)},$$

*i.e., the* $L^2(\mu)$*-distance of the linear functions whose graphs are* $\mathbf{U}_1$ *and* $\mathbf{U}_2$*, where* $\mu$ *is a positive measure in* $\mathbb{R}^2$*.(These linear functions are supposed to be always in* $L^2(\mu)$*).*

The measure $\mu$ can be defined in different ways (see also [8]). For our purpose the simple form

$$d_\mu(\mathbf{U}_1, \mathbf{U}_2)^2 = \sum_i ((u_{0,1} - u_{0,2}) + (u_{1,1} - u_{1,2})x_i + (u_{2,1} - u_{2,2})y_i)^2,$$

has been chosen, i.e., the sum of some point masses at points $(x_i, y_i)$.

## 4. The approximation by Kohonen-network

Artificial neural networks are biologically inspired models, based on the functions and structure of neurons in biology. A neural network consists of numerous computational elements (neurons or nodes), highly interconnected to each other. A weight is associated to every connection. Normally, nodes are arranged into layers. During a training procedure input vectors are presented to the input layer with or without specifying the desired output. According to this difference neural networks can be classified as supervised or unsupervised (self-organizing) neural nets. Networks can also be classified according to the input values (binary or continuous). The learning procedure itself contains three main steps, the presentation of the input sample, the calculation of the output and the modification of the weights by specified training rules. These steps are repeated several times, until the network is said to be trained. For details and survey of artificial neural networks see, e.g., [13, 14].

The Kohonen network, a two-layer unsupervised continuous valued neural network, described first in [12], is an excellent tool for ordering any kind of scattered data. The network has a strong self-organizing ability, which is used practically for dragging a predefined structure - a polygon for curve modelling and a quadrilateral grid for surface modelling - towards the given points. After the so-called training procedure this predefined grid will follow the structure and distribution of the given points, or - if the number of points are relatively small - will pass through every given point. The main advantage of this technique is the unique way of handling problems defined by a few given points or a large cloud of points. For the detailed description of the standard method see [9]. The following steps give the training algorithm in the standard curve approximation case in $E^3$:

Let the given scattered points $\mathbf{p}_i(x_{1,i}, x_{2,i}, x_{3,i}), (i = 1, ..., m)$ be given. The Kohonen network has an input layer with 3 neurons (for the three coordinates) and an output layer with $n(> m)$ neurons. The $i^{th}$ neuron of the input layer is connected to the $j^{th}$ neuron of the output layer and the weight $w_{ij}$ is associated to this connection. These weights can be considered as coordinates of the output points $\mathbf{q}_j(w_{1j}, w_{2j}, w_{3j})$,$(j = 1, ..., n)$, which will form the result polygon after the training process:

1. Initialize the weights $w_{sj}, (s = 1, 2, 3, j = 1, ..., n)$ as small random values around the average of the coordinates of the input points. Let the training time $t = 1$.
2. Present new input values $(x_{1,i_0}, x_{2,i_0}, x_{3,i_0})$, as the coordinates of a randomly selected input point $\mathbf{p}_{i_0}$.
3. Compute the distance $d_j(\mathbf{p}_{i_0}, \mathbf{q}_j), (j = 1...n)$ of all output nodes to the input point.
4. Find the winning unit $\mathbf{q}_{j_0}$ as the node which has the minimum distance to the input point, i.e. $d_{j_0} = min(d_j)$.
5. Compute the neighborhood $N(t) = (j_0, j_1, ..., j_k)$.
6. Update the weights (i.e., the coordinates) of the nodes in the neighborhood by the following equation:

$$w_{sj}(t + 1) = w_{sj}(t) + \eta(t)(x_{s,i_0} - w_{sj}(t)), \forall j \in N(t),$$

   where $\eta(t)$ is a so called gain term, a Gaussian function decreasing in time, e.g.,

$$\eta(t) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}t^2}, t \geq 0.$$

7. Let the training time $t = t + 1$. Repeat step 2-7. until the network is trained.

For a relatively small number of points the network is said to be trained if all the input points are on the polygon, that is for all the input points $\mathbf{p}_i (i = 1, ..., m)$ there is an output vector $\mathbf{q}_j$ such that after a time the distance of $\mathbf{q}_j$ and $\mathbf{p}_i$ is smaller than a predefined limit. A stronger convergence can be obtained if we require that the output vectors which do not converge to an input vector be on the line determined by its two neighboring output points. This stronger convergence is important especially in terms of the smoothness of the future curve. If the number of input points is large or infinite (given by a distribution) the network is trained if the changes of the output polygon fall under a predefined limit.

Now, we give the changes of the general algorithm described above in case of the approximation of a set of planes by their envelope, a developable NURBS surface. Here, the input data will be the coordinates of the planes $\mathbf{U}_i(u_{0,i}, u_{1,i}, u_{2,i})$. The main difference, however, appears in step 3. where originally the distance $d_j(\mathbf{p}_{i_0}, \mathbf{q}_j)$ is used. In the general algorithm and applications it is the standard Euclidean distance (see [9]), but in our case, since we are in the dual space $P^*$, the special distance $d_\mu$ is used which has been defined in the previous section. With this change the neural network algorithm automatically produces the "control polygon" of the dual curve, which can be transformed to the normal tensor product surface. Hence, the scattered set of input planes is modelled by a developable standard NURBS surface.

## 5. Conclusion

A special method for constructing developable NURBS surface from a given set of scattered tangent planes has been presented in this paper. The method used a special geometrical principle, the duality of projective geometry, to transform the problem to a curve modelling in the dual space $P^*$. The Kohonen neural network has been ordered the scattered data, but for using this technique a special distance function has been applied. We mention that one of the main advantages of this method is that the neural network has a kind of minimal energy property (see [12]), which yields a fairly smooth control grid and surface without any additional smoothing and fairing techniques. The other advantage, what has already been mentioned, that this method can handle rather few input data as well as infinitely many data given by a distribution.

## 6. References


1. G. Aumann, Approximate development of skew ruled surfaces. *Comput. & Graph.* **13** 361-366 (1989).
2. G. Aumann, Interpolation with developable Bézier patches. *Computer Aided Geom. Des.* **8** 409-420 (1991)
3. J. Lang and O. Röschel, Developable $(1, n)$-Bézier surfaces. *Computer Aided Geom. Des.* **9** 291-298 (1992)
4. R.M.C. Bodduluri and B. Ravani, Design of developable surfaces using duality between plane and point geometries. *Computer-Aided Design* **25** 621-632 (1993).
5. J. Hoschek and H. Pottmann, Interpolation and approximation with developable B-spline surfaces. In: *Mathematical Methods for Curves and Surfaces,* (Edited by M. Dæhlen *et al.*), pp.255-264, Vanderbilt University Press, Nashville (1995).

6. J. Hoschek and M. Schneider, Interpolation and approximation with developable surfaces. In: *Curves and Surfaces with Applications in CAGD*, (Edited by A. Le Méhauté *et al.*), pp.185-202, Vanderbilt University Press, Nashville (1997).
7. H. Pottmann and G. Farin, Developable rational Bezier and B-spline surfaces. *Comput. Aided Geom. Des.* **12** 513-531 (1995)
8. H. Pottmann and J. Wallner, Approximation algorithms for developable surfaces. *Computer Aided Geom. Des.* **16** 539-556 (1999).
9. M. Hoffmann and L. Várady, Free-form surface design by neural networks. *Journal for Geom. and Graph.* **2** 1-6 (1998)
10. L. Várady, M. Hoffmann and E. Kovács, Improved free-form modelling of scattered data by dynamic neural networks. *Journal for Geom. and Graph.* **3** 177-183 (1999)
11. M. Hoffmann, Modified Kohonen neural network for surface reconstruction. *Publ. Math.* **54** Suppl. 857-864, (1999)
12. T. Kohonen, Self-organization and associative memory. 3rd edition, Springer-Verlag (1996)
13. J. Freeman and D. Skapura, Neural Networks; Algorithms, Applications and Programming Techniques. Addison-Wesley (1991)
14. R. Rojas, Neural Networks. A Systematic Introduction. Springer-Verlag (1996).

Institute of Mathematics and Computer Science, Károly Eszterházy College, Leányka str. 4-6., 3300 Eger, Hungary

*E-mail address*: hofi@ektf.hu. emod@ektf.hu

*URL*: http://www.ektf.hu/tanszek/matematika/mateng.html