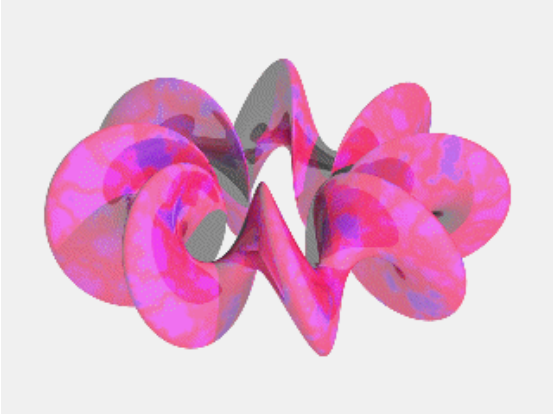
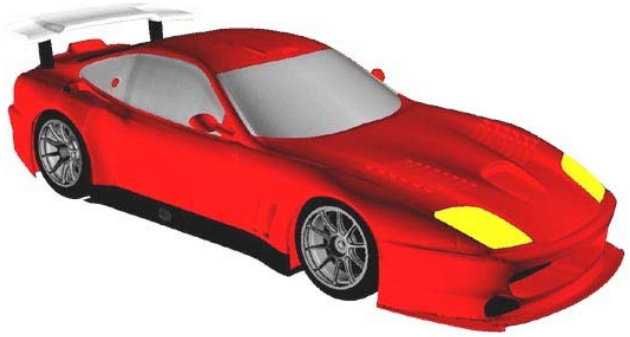
Curves and Surfaces

Many applications of computer graphics require (in addition to simple elements) also the possibility to model and display arbitrary surfaces (*free-form surfaces*). First, due to easier understanding of the mathematics used, the principles of free-form curves will be explained. The examined methods can then be extended to surfaces easily.





Curves

General curves can be defined either by a *formula* (analytical representation) or by specifying some *control points*, which define the shape of the curve. Since an appropriate analytical formula for a specific curve is usually rather difficult to find, control points are used most of the time.

The following properties characterize the different types of curves:

- *interpolating* (curve passes through the control points) versus *approximating* (control points lie close to the curve)
- *degree of continuity* at the connection of curve segments
- *global* influence (all points influence all curve points) versus *local* influence (points only influence close parts of the curve)
- *axis-dependent* representation (rotation of the coordinate system changes the curve) versus *axis-independent* representation (rotation of the coordinate system does not change the curve)
- tendency to *damping* versus to *oscillation* at the end points
- possible forms of curves, limitations, double points, closed curves, etc.

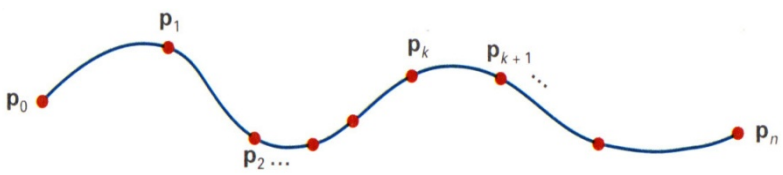
Curves, which are defined by control-points, are called *Splines*. In the following, some simple but common spline-variants will be described.

Cubic Spline-Interpolation

Given are n+1 control-points $\mathbf{p_k} = (x_k, y_k, z_k)$, k=0..n. An interpolating curve which consists of cubic polynomials between two control points is called a cubic spline. Between control points $\mathbf{p_k}$ and $\mathbf{p_{k+1}}$ a parameter u describes the curve:

$$\mathbf{P_k}(u) = \mathbf{a_k}u^3 + \mathbf{b_k}u^2 + \mathbf{c_k}u + \mathbf{d_k}$$

where k = 0, 1, 2, …, n−1 and $0 \le u \le 1$



(attention: = $\mathbf{a_k}$, $\mathbf{b_k}$, $\mathbf{c_k}$, $\mathbf{d_k}$ are vectors).

To calculate the part of a curve between 2 control points, 4 conditions have to be available. If the definition at the control points is chosen in a way, that the cubic polynomials at those points are C^1-continuous (differentiable) as well as C^2-continuous (2x differentiable, i.e. same curvature) when connected, such

curves are called *natural cubic splines*. These are obtained by solving a system of 4n equations with 4n variables and setting constraints at the two end points, e.g. that the curvature in the beginning and end of the curve is zero. Cubic splines have the disadvantage that each control point influences the whole curve, i.e. each point has global influence.

Hermite-Interpolation

A special form of cubic splines is the Hermite-Interpolation. Here, in addition to the control-points p_k, also the derivations $\mathbf{Dp_k}$ at the control points are given. The cubic interpolation polynomial $P_k(u)$, $0 \le u \le 1$, between points $\mathbf{p_k}$ and $\mathbf{p_{k+1}}$, can then uniquely be calculated from these 4 elements of determination:

$$\mathbf{P_k}(0) = \mathbf{p_k}, \quad \mathbf{P_k}(1) = \mathbf{p_{k+1}}, \quad \mathbf{P'_k}(0) = \mathbf{Dp_k}, \quad \mathbf{P'_k}(1) = \mathbf{Dp_{k+1}} \qquad \text{with } k = 0,\ldots,n-1$$

$\mathbf{P_k}(u) = \mathbf{a_k}u^3 + \mathbf{b_k}u^2 + \mathbf{c_k}u + \mathbf{d_k}$ can also be written in matrix-form, and also the first derivation of the curve $\mathbf{P'_k}(u) = 3\mathbf{a_k}u^2 + 2\mathbf{b_k}u + \mathbf{c_k}$. Using this, the determination elements $\mathbf{p_k}$, $\mathbf{p_{k+1}}$, $\mathbf{Dp_k}$, $\mathbf{Dp_{k+1}}$ can be formulated as:

$$\mathbf{P_k}(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{a_k} \\ \mathbf{b_k} \\ \mathbf{c_k} \\ \mathbf{d_k} \end{bmatrix} \qquad \mathbf{P'_k}(u) = \begin{bmatrix} 3u^2 & 2u & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{a_k} \\ \mathbf{b_k} \\ \mathbf{c_k} \\ \mathbf{d_k} \end{bmatrix} \qquad \begin{bmatrix} \mathbf{p_k} \\ \mathbf{p_{k+1}} \\ \mathbf{Dp_k} \\ \mathbf{Dp_{k+1}} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{a_k} \\ \mathbf{b_k} \\ \mathbf{c_k} \\ \mathbf{d_k} \end{bmatrix}$$

To calculate the coefficients $\mathbf{a_k}$, $\mathbf{b_k}$, $\mathbf{c_k}$, $\mathbf{d_k}$ in $\mathbf{a_k}u^3 + \mathbf{b_k}u^2 + \mathbf{c_k}u + \mathbf{d_k}$, the matrix must be inverted. The resulting matrix is called Hermite-matrix M_H:

$$\begin{bmatrix} \mathbf{a_k} \\ \mathbf{b_k} \\ \mathbf{c_k} \\ \mathbf{d_k} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix}^{-1} \cdot \begin{bmatrix} \mathbf{p_k} \\ \mathbf{p_{k+1}} \\ \mathbf{Dp_k} \\ \mathbf{Dp_{k+1}} \end{bmatrix} = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{p_k} \\ \mathbf{p_{k+1}} \\ \mathbf{Dp_k} \\ \mathbf{Dp_{k+1}} \end{bmatrix} \qquad \mathbf{P_k}(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \cdot M_H \cdot \begin{bmatrix} \mathbf{p_k} \\ \mathbf{p_{k+1}} \\ \mathbf{Dp_k} \\ \mathbf{Dp_{k+1}} \end{bmatrix}$$

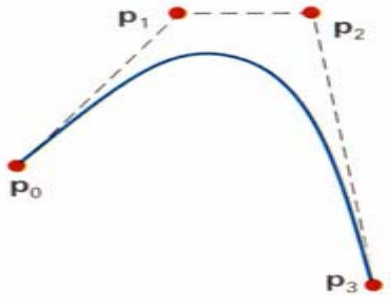
Bezier-Curves

Pierre Bezier developed an *approximating* curve in 1960 at Renault to model and describe car-bodies. In these curves so called Bernstein-polynomials ($BEZ_{k,n}$) are used as weighting functions for the control points. Every point on the curve is the weighted average of all control-points:

$$\mathbf{P}(u) = \sum_{k=0}^{n} \mathbf{p_k} BEZ_{k,n}(u) \qquad 0 \le u \le 1 \qquad \text{with} \qquad BEZ_{k,n}(u) = \binom{n}{k} u^k (1-u)^{n-k}$$

E.g., for 4 control-points (thus n=3) we get:

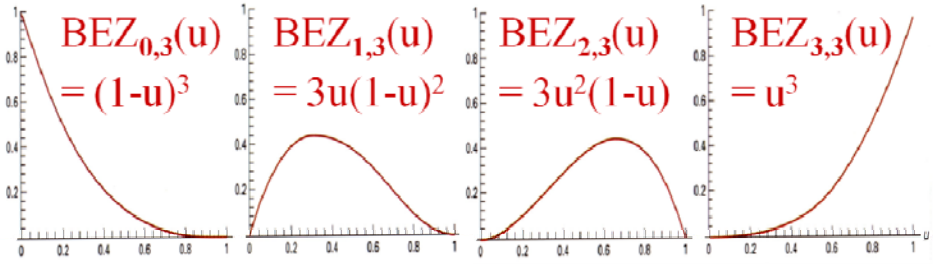
$$\mathbf{P}(u) = (1-u)^3.\mathbf{p_0} + 3u(1-u)^2.\mathbf{p_1} + 3u^2(1-u).\mathbf{p_2} + u^3.\mathbf{p_3}$$



Properties of Bezier-curves:

- for n+1 control points, the degree of **P**(u) is n
- every control point attracts the curve like a rubber band
- *global influence* (weighting function >0 almost everywhere)
- $\mathbf{p_0}$ and $\mathbf{p_n}$ lie on the curve
- the tangents in $\mathbf{p_0}$ and $\mathbf{p_n}$ are the connections to the next points $\mathbf{p_1}$ and $\mathbf{p_{n-1}}$
- the curve lies completely in the convex hull of the control-points (the convex hull is the smallest convex polygon, which includes all control-points)

Some of these properties can be perceived from the form of the Bernstein-polynomials $BEZ_{k,n}$:



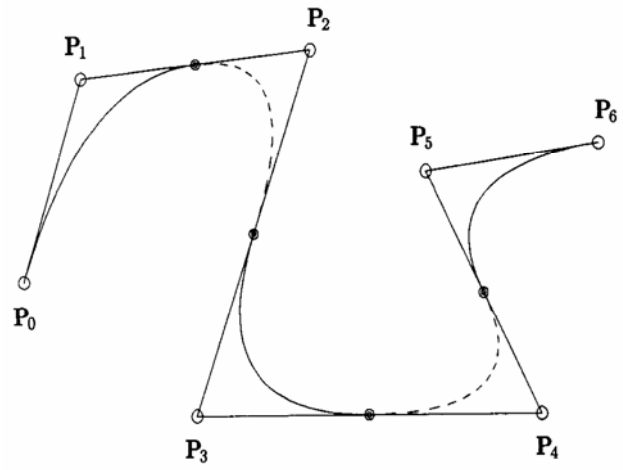
B-Spline-Curves

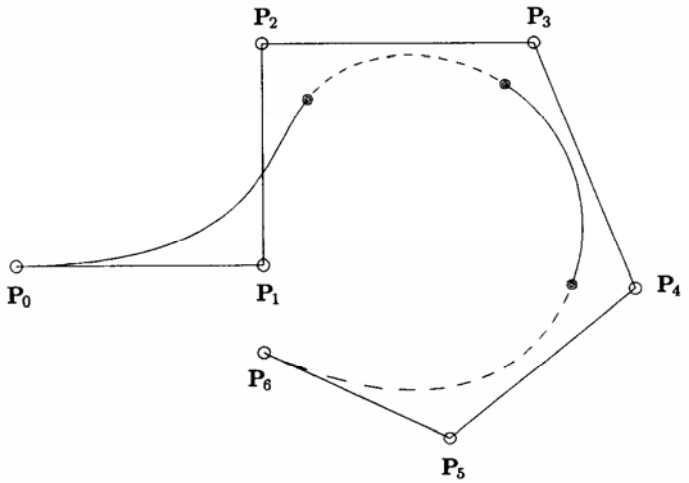
The main disadvantage of the Bezier-curves is the global influence of the control points on the whole curve. This has two major drawbacks: (1) every change of a control point (insert, move, delete) changes the look of the curve in all points of the curve, and (2) the computation time needed for a big set of control points is comparatively high. The reason is the weighting-function form. The so called *B-Splines* are, just as the Bezier-Splines, approximating curves, but the Bernstein-polynomials are replaced by the B-spline-polynomials $B_{k,d}$. These limit the number of control points, which influence any curve point, to d. The computation of $B_{k,d}$ is more complex and is done recursively. However, to understand B-Splines, it is sufficient to look at the form of the B-spline polynomials. It can be observed that each weighting curve is non-zero only in a limited range, so that every control point has only limited influence on the B-spline-curve.

One important property of the B-spline weighting functions is the fact, that for every point on the curve their sum equals 1. Thus every curve-point is a *weighted average* of the control points.

$$\sum_{k=0}^{n} B_{k,d}(u) = 1$$

Examples for B-spline-curves with d=3 (left) and d=4 (right):





If d=n+1 is chosen, the resulting curve is a Bezier-curve, which means that Bezier-curves are a special case of the B-splines.

The main differences to the Bezier-curves are:

- local influence of the control points
- complexity linear in number of control points (instead of quadratic for Bezier-curves)

The most important extensions of these so called uniform B-splines are the *Non-Uniform Rational B-Splines*, better known as *NURBS*. They allow a consistent representation of regular geometric objects, too.

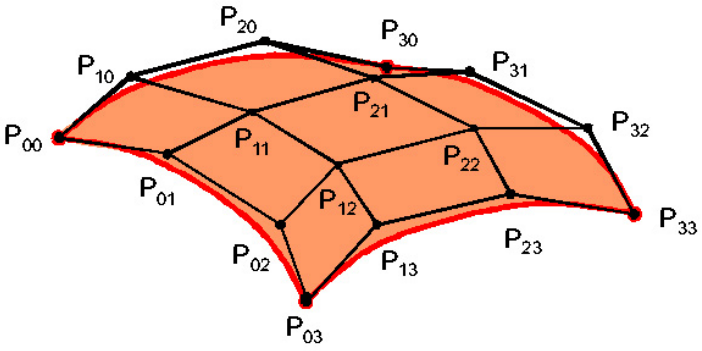
Note, moreover, that all described methods for points are valid in two dimensions as well as in three dimensions. Basically all these splines describe spatial curves in three-dimensional space.

Free-Form-Surfaces

Bezier- and B-Spline-Surfaces

If the Cartesian product is constructed over two arrays of curves, the resulting set of points is a free-form-surface. This is a natural way of constructing free-form-surfaces. Depending on which curves are involved, different types of surfaces are obtained, like Bezier-surfaces from Bezier-curves, B-spline-surfaces from B-spline-curves, etc.

$$\mathbf{P}(u,v) = \sum_{j=0}^{m}\sum_{k=0}^{n} \mathbf{p}_{j,k} BEZ_{j,m}(v) BEZ_{k,n}(u)$$



Each pair of parameters (u, v) corresponds to a point on the constructed surface. The curves along the edges of the surface are of the same type, e.g. in the Bezier-surface example they are Bezier-curves. The other properties of Bezier-curves also hold for the corresponding surfaces.

Similar to the Bezier-surfaces, B-spline-surfaces can be obtained when using B-spline weighting-functions in the formula, or NURBS-surfaces with NURBS-curves.

To draw free-form-surfaces, triangle meshes can be constructed from the surfaces which are then rendered like B-Reps. Alternatively, ray-casting methods can be used. For this, methods have to be implemented to calculate the intersection point between a line and a surface as accurately as possible.