B-Spline Volumes

B-Spline Volumes are a simple extension of B-Splines to 3 Dimensions. This is a straightforward adaption of the 2-Dimensional version.

Nomenclature

x, y, z denote space-coordinates,
u, v, w denote spline-coordinates (Between 0-1),
P_{ijk} denote the control-Points on the control-Polygon,
$N_{i,d,\tau}(u)$ denote the value of the underlying Basis-Functions at value u using the i-th Basis-Function of degree d in range τ.

For our case we only care about degree-3 splines, so we omit the d furtheron. τ is defined statically (in each direction) with each P as Position on the whole surface/volume and within [0,1]. For a regular Control-Grid this defaults to $\tau_i = i/n$

Given n, m, o control points in x, y, z-direction each Point on the curve is defined by

$$C(u,v,w) = \sum_{i=0}^{n-d-2} \sum_{j=0}^{m-d-2} \sum_{k=0}^{o-d-2} P_{ijk} N_i(u) N_j(v) N_k(w)$$

Calculate u, v, w

Given a target-point $\mathbf{p}^*$ and an initial guess $\mathbf{p} = C(u,v,w)$ we define the error-function as:

$$Err(u,v,w,\mathbf{p}^*) = \|\mathbf{p}^* - C(u,v,w)\|_2^2$$

As the error is just the sum of the components

$$Err(u,v,w,\mathbf{p}^*) = Err_x(u,v,w,\mathbf{p}^*) + Err_y(u,v,w,\mathbf{p}^*) + Err_z(u,v,w,\mathbf{p}^*)$$

we just take one axis into account, as the others are nearly identical. So we yield

$$Err_x(u,v,w,\mathbf{p}^*) = \left(p_x^* - \sum_{i=0}^{n-d-2} \sum_{j=0}^{m-d-2} \sum_{k=0}^{o-d-2} P_{ijk_x} N_i(u) N_j(v) N_k(w)\right)^2$$

To solve this we derive:

$$\begin{aligned}
\frac{\partial}{\partial u} & \left(p_x^* - \sum_{i=0}^{n-d-2}\sum_{j=0}^{m-d-2}\sum_{k=0}^{o-d-2} P_{ijk_x} N_i(u) N_j(v) N_k(w)\right)^2 \\
= & \sum_{i=0}^{n-d-2}\sum_{j=0}^{m-d-2}\sum_{k=0}^{o-d-2} P_{ijk_x} N_i'(u) N_j(v) N_k(w) \\
& \cdot \left(p_x^* - \sum_{i=0}^{n-d-2}\sum_{j=0}^{m-d-2}\sum_{k=0}^{o-d-2} P_{ijk_x} N_i(u) N_j(v) N_k(w)\right) \\
& \cdot 2
\end{aligned}$$

The other partial derivatives follow the same pattern.

Basis-Splines and Derivatives

The previously mentioned $N_{i,d,\tau}$ are defined recursively:

$$N_{i,0,\tau}(u) = \begin{cases} 1, & u \in [\tau_i, \tau_{i+1}[\\ 0, & \text{otherwise} \end{cases}$$

and

$$N_{i,d,\tau}(u) = \frac{u - \tau_i}{\tau_{i+d}} N_{i,d-1,\tau}(u) + \frac{\tau_{i+d+1} - u}{\tau_{i+d+1} - \tau_{i+1}} N_{i+1,d-1,\tau}(u)$$

This fact can be exploited to get the derivative for an arbitrary N:

$$\frac{d}{du} N_{i,d,r}(u) = \frac{d}{\tau_{i+d} - \tau_i} N_{i,d-1,\tau}(u) - \frac{d}{\tau_{i+d+1} - \tau_{i+1}} N_{i+1,d-1,\tau}(u)$$

Warning: in the case of $d = 1$ the recursion-formula yields a 0 denominator, but N is also 0. The right solution for this case is a derivative of 0