3.1.9 Curve Interpolation using Uniform Cubic B-Spline Curves

- Give a set of data points $D_i = (x_i, y_i)$, $i = 0, 1, \cdots, n$, ($n \geq 2$), how can a cubic B-spline curve that interpolates these points be constructed?

- The cubic B-spline curve has $n$ segments $C_1(t)$, $C_2(t)$, ..., $C_n(t)$ with $D_{i-1}$ and $D_i$ being the start and end points of $C_i(t)$. 
An analysis of the problem:

- To get the curve constructed, how many knots are needed? Consider the following case:

So, to interpolate \((n + 1)\) data points, one needs \((n + 7)\) knots, \(t_0, t_1, \ldots, t_{n+6}\), for a uniform cubic B-spline interpolating curve.
3.1.9 Curve Interpolation using Uniform Cubic B-Spline Curves

To get the curve constructed, how many control points are needed? Consider the following case:

So, to interpolate \((n + 1)\) points, one needs \((n + 3)\) control points, \(P_0, P_1, \ldots, P_{n+2}\), for a uniform cubic B-spline interpolating curve.
3.1.9 Curve Interpolation using Uniform Cubic B-Spline Curves

To make things easier, we shall assume that

\[ t_i = i - 3, \quad i = 3, 4, \ldots, n + 3 \]

with \( t_0 = t_1 = t_2 = t_3 \) and \( t_{n+3} = t_{n+4} = t_{n+5} = t_{n+6} \).

Consequently, we have \( P_0 = D_0 \) and \( P_{n+2} = D_n \).

Still, we need to find \( P_1 = P_2 = \ldots = P_{n+1} \).

How?
3.1.9 Curve Interpolation using Uniform Cubic B-Spline Curves

The interpolating curve to be constructed must be of the following form:

\[ C(t) = \sum_{i=0}^{n+2} N_{i,3}(t) P_i, \quad t \in [0, n] \]

and satisfies the following conditions:

\[ C(i) = \sum_{i=0}^{n+2} N_{i,3}(i) P_i = D_i, \quad i = 0, 1, \ldots, n \]

\[ (*) \]
3.1.9 Curve Interpolation using Uniform Cubic B-Spline Curves

Note that at each knot there are at most 3 cubic B-spline basis functions which are non-zero. Therefore, equations in (*) are of the following form:

\[
N_{i,3}(i)P_i + N_{i+1,3}(i)P_{i+1} + N_{i+2,3}(i)P_{i+2} = D_i, \quad i=0,1,\ldots,n
\]

or

\[
\begin{align*}
P_0 &= D_0 \\
\frac{1}{4}P_1 + \frac{7}{12}P_2 + \frac{1}{6}P_3 &= D_1 \\
\frac{1}{6}P_2 + \frac{2}{3}P_3 + \frac{1}{6}P_4 &= D_2 \\
\frac{1}{6}P_{n-2} + \frac{2}{3}P_{n-1} + \frac{1}{6}P_n &= D_{n-2} \\
\frac{1}{6}P_{n-1} + \frac{7}{12}P_n + \frac{1}{4}P_{n+1} &= D_{n-1} \\
P_{n+2} &= D_n
\end{align*}
\]
So, actually, only $P_1, P_2, \ldots, P_{n+1}$ are unknown. By ignoring the 1st and the last equations, we have a system of $n - 1$ equations (those marked with "#") in $n + 1$ unknowns. We need two extra conditions to get this system solved.

One option is to set the second derivative of the curve at the start and end points to zero:

\[
C''(0) = N_{0,3}''(0)P_0 + N_{1,3}''(0)P_1 + N_{2,3}''(0)P_2 = 0
\]

\[
C''(n) = N_{n,3}''(n)P_n + N_{n+1,3}''(n)P_{n+1} + N_{n+2,3}''(n)P_{n+2} = 0
\]
3.1.9 Curve Interpolation using Uniform Cubic B-Spline Curves

\[ 6P_0 - 9P_1 + 3P_2 = 0 \]

\[ 3P_n - 9P_{n+1} + 6P_{n+2} = 0 \]

Note that \( P_0 \) and \( P_{n+2} \) are known to us (\( P_0 = D_0 \) and \( P_{n+2} = D_n \)). Hence, the above equations can be written as:

\[ 3P_1 - P_2 = 2D_0 \]

\[ -P_n + 3P_{n+1} = 2D_n \]
By combining these two equations with the equations on page 6 marked with #, we have a system of $n + 1$ equations in $n + 1$ unknowns:

\[3P_1 - P_2 = 2D_0\]
\[\frac{1}{4}P_1 + \frac{7}{12}P_2 + \frac{1}{6}P_3 = D_1\]
\[\frac{1}{6}P_2 + \frac{2}{3}P_3 + \frac{1}{6}P_4 = D_2\]
\[\frac{1}{6}P_{n-2} + \frac{2}{3}P_{n-1} + \frac{1}{6}P_n = D_{n-2}\]
\[\frac{1}{6}P_{n-1} + \frac{7}{12}P_n + \frac{1}{4}P_{n+1} = D_{n-1}\]
\[-P_n + 3P_{n+1} = 2D_n\]
3.1.9 Curve Interpolation using Uniform Cubic B-Spline Curves

The matrix form of this system is:

\[
\begin{bmatrix}
3 & -1 & \ & \ & \ \\
1/4 & 7/12 & 1/6 & \ & \ \\
1/6 & 2/3 & 1/6 & \ & \ \\
1/6 & 2/3 & 1/6 & & \\
1/6 & 7/12 & 1/4 & & \\
-1 & & & 3 & \\
\end{bmatrix}
\begin{bmatrix}
P_1 \\
P_2 \\
P_3 \\
P_{n-1} \\
P_n \\
P_{n+1} \\
\end{bmatrix}
= 
\begin{bmatrix}
2D_0 \\
D_1 \\
D_2 \\
D_{n-2} \\
D_{n-1} \\
2D_n \\
\end{bmatrix}
\]

can be solved using Gaussian elimination without pivoting. Most of the curve drawing programs in commercial packages (such as xfig, ...) are implemented using this approach.
Question:

What should we do to generate smooth, closed interpolating curves?

For instance, give a set of data points $D_i = (x_i, y_i)$, $i = 0, 1, \ldots, n$, with $D_0 = D_n$, how can a closed, smooth ($C^2$-continuous) cubic B-spline curve that interpolates these points be constructed?

The closed cubic B-spline curve has $n$ segments $C_1(t), C_2(t), \ldots, C_n(t)$, with $D_{i-1}$ and $D_i$ being the start and end points of $C_i(t)$. 
From previous discussion, we know such a curve must have \((n + 3)\) control points: \(P_0, P_1, \ldots, P_{n+2}\).

To guarantee \(C_2\) continuity at \(C_0 = C_n\), control points must satisfy the following conditions:

\[
P_n = P_0, \quad P_{n+1} = P_1, \quad P_{n+2} = P_2
\]
3.1.9 Curve Interpolation using Uniform Cubic B-Spline Curves

- Such a curve needs \((n + 7)\) knots: \(t_0, t_1, \ldots, t_{n+6}\)
- To make things easier, we shall assume that \(t_i = i - 3\), \(i = 0, 1, \ldots, n + 6\)
- Such a (cyclic) curve can be defined as follows

\[
C(t) = \sum_{i=0}^{n+2} N_{i,3}(t) P_{(i \mod n)} \quad t \in [t_3, t_{n+3}] = [0, n]
\]

such that

\[
C(t_{i+3}) = C(i) = \sum_{i=0}^{n+2} N_{i,3}(i) P_{(i \mod n)} = D_i, \quad (C1)
\]

\[
i = 0, 1, \ldots, n
\]
There are $n$ unknows and $n$ equations in the above system (C1):

\[
\frac{1}{6}P_0 + \frac{2}{3}P_1 + \frac{1}{6}P_2 = D_0
\]
\[
\frac{1}{6}P_1 + \frac{2}{3}P_2 + \frac{1}{6}P_3 = D_1
\]

\[
\frac{1}{6}P_{n-2} + \frac{2}{3}P_{n-1} + \frac{1}{6}P_n = D_{n-2}
\]
\[
\frac{1}{6}P_{n-1} + \frac{2}{3}P_n + \frac{1}{6}P_{n+1} = D_{n-1}
\]
\[
\frac{1}{6}P_n + \frac{2}{3}P_{n+1} + \frac{1}{6}P_{n+2} = D_n
\]

(The last equation is the same as the first equation and, hence, can be ignored.)
3.1.9 Curve Interpolation using Uniform Cubic B-Spline Curves

The matrix form of this system is:

\[
\begin{bmatrix}
2/3 & 1/6 & & & & 1/6 \\
1/6 & 2/3 & 1/6 & & & \\
1/6 & 2/3 & 1/6 & & & \\
& & & \ddots & & \\
& & & & 1/6 & 2/3 & 1/6 \\
& & & & 1/6 & 2/3 & \\
1/6 & & & & & 2/3
\end{bmatrix}
\begin{bmatrix}
P_1 \\
P_2 \\
P_3 \\
\vdots \\
P_{n-1} \\
P_n
\end{bmatrix}
=
\begin{bmatrix}
D_0 \\
D_1 \\
D_2 \\
\vdots \\
D_{n-2} \\
D_{n-1}
\end{bmatrix}
\]

This system of equations can be solved using Gaussian elimination without pivoting as well.
Curve Design Procedure:

1. Specify a set of points \( D_0, D_1, \ldots, D_n \) which lie approximately on the desired curve

2. Generate a cubic B-spline or composite Bezier curve that interpolates these points

3. Adjust control points of the interpolating curve to "sculpt" it into a more satisfactory shape.
3.1.10 Non-Uniform B-Spline Surfaces

**Definition:** A B-spline surface of degree $p$ in $u$ direction and degree $q$ in $v$ direction is defined as follows

$$
S(u,v) = \sum_{i=0}^{m} \sum_{j=0}^{n} N_{i,p}(u) N_{j,q}(v) P_{i,j}
$$

where $P_{i,j}$ are 3D control points, $N_{i,p}(u)$ and $N_{j,q}(v)$ are B-spline basis functions of degree $p$ and $q$, respectively, defined with respect to the knot vectors

$$
U=\{ u_0, u_1, \cdots, u_{m+p+1} \},
$$

and

$$
V=\{ v_0, v_2, \cdots, v_{n+q+1} \},
$$

respectively. The parameter space of the surface is

$$
[u_p, u_{m+1}] \times [v_q, v_{n+1}] .
$$
3.1.10 Non-Uniform B-Spline Surfaces

An example of a bicubic B-spline surface:
3.1.10 Non-Uniform B-Spline Surfaces

- The grid defined by the control points of a B-spline surface is called a control net or a control polyhedron.
- The image of each $[u_i, u_{i+1}] \times [v_j, v_{j+1}]$ in the domain of a B-spline surface is called a Bspline patch.
- The B-spline surface defined on page 17 is $C^{p-1}$-continuous in u direction and $C^{q-1}$-continuous in v direction.
- A B-spline surface usually does not interpolate its control points. However, if the knots satisfy the following condition

$$u_0 = u_1 = \cdots = u_p; \quad v_0 = v_1 = \cdots = v_q$$

and

$$u_{m+1} = \cdots = u_{m+p+1}; \quad v_{n+1} = \cdots = v_{n+q+1}$$

then the surface interpolates the corner points of the control net.
3.1.10 Non-Uniform B-Spline Surfaces

- Each patch of the B-spline surface defined on page 17 is contained in the convex hull of \((p+1) \times (q+1)\) control points.

- A B-spline surface can be converted into a composite Bezier surface using a technique similar to the curve case.

- Surface Design, in some cases, follows the same procedure as the curve design process. More complicated shape design techniques will be introduced in the next chapter.

- Sometime it is necessary to remove certain portions of a surface to get a special shape. The resulting surface is called a *trimmed surface*. 

University of Kentucky
3.1.10 Non-Uniform B-Spline Surfaces

**Definition:** A trimmed B-spline surface is a B-spline surface whose actual extent is specified by a set of closed loops defined in the parameter space of the surface.

- An ordinary B-spline surface can be considered as a special case of a trimmed B-spline surface by viewing the boundary of the parameter space as a trimming loop.
- The loops are called trimming loops or trimming curves.
3.1.10 Non-Uniform B-Spline Surfaces

Examples of trimmed NURBS surfaces:
3.1.10 Non-Uniform B-Spline Surfaces

- The trimming loops are typically produced by the intersections between two or more untrimmed surfaces.

- Each of the trimming loops may be defined by one or more components. Each component is defined by a B-spline curve.

- The interior (trimming area) of a trimmed surface is usually defined by the odd winding rule or curve handedness rule.

- Curve handedness rule: the increasing parameter value of a trimming curve must corresponding to counterclock motion around the enclosed (trimming) region.

- The trimming curves are not permitted to intersect each other, nor is a trimming curve permitted to intersect itself.
End of Section 3.1.10