Today

- Review of Bézier curves
- Piecewise-cubic Bézier curves
- B-splines

Curves

- Construct a function $x(t)$
  - Moves a point from start to end of curve as $t$ goes from 0 to 1
  - Tangent vector given by derivative $x'(t)$
- Use a few control points to intuitively describe a curve
- Bézier curves

![Linear, Quadratic, Cubic Bézier curves](image)

Linear interpolation

- Given two points $p_0$ and $p_1$
- “Curve” is line segment between them
- Three ways to write linear interpolation
  - Weighted average $x(t) = p_0(1-t) + p_1t$
  - Polynomial $x(t) = (p_1 - p_0)t + p_0$
  - Matrix form $x(t) = \begin{bmatrix} p_0 & p_1 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} t \\ 1 \end{bmatrix}$

Cubic Bézier curve

- Four control points $p_0, p_1, p_2, p_3$
  - Interpolates the endpoints
  - Intermediate points determine tangents at endpoints
- Recursive geometric construction
  - de Casteljau algorithm
- Three ways to express curve
  - Weighted average of the control points, Bernstein polynomials
  - Polynomial in $t$
  - Matrix form

Cubic Bernstein polynomials

$$x(t) = B_0(t)p_0 + B_1(t)p_1 + B_2(t)p_2 + B_3(t)p_3.$$  

The cubic Bernstein polynomials:

$$
\begin{align*}
B_0(t) &= -t^3 + 3t^2 - 3t + 1 \\
B_1(t) &= 3t^3 - 6t^2 + 3t \\
B_2(t) &= -3t^3 + 3t^2 \\
B_3(t) &= t^3
\end{align*}
$$

$$\sum B_i(t) = 1$$

- Partition of unity, weights always add to 1
- Endpoint interpolation, $B_0$ and $B_3$ go to 1
Bézier curves properties
- Convex hull property
- Variation diminishing property
- Affine invariance

Convex hull property
Variation diminishing property

Tangent
- The derivative of a curve represents the tangent vector to the curve at some point

Tangent of polynomial curves
- Easy to compute
  - Example: cubic curve
    \[
    \begin{align*}
    x(t) &= \begin{bmatrix} a & b & c & d \end{bmatrix} \begin{bmatrix} t^3 \\ t^2 \\ t \\ 1 \end{bmatrix} \\
    x'(t) &= \frac{dx}{dt}(t) = \begin{bmatrix} 3a & 2b & c \end{bmatrix} \begin{bmatrix} t^2 \\ t \\ 1 \end{bmatrix}
    \end{align*}
    \]
  - Notice \( x'(t) \) is a vector

Tangent of cubic curve
- Note that
  \[
  x'(0) = c = 3(p_1 - p_0) \\
  x'(1) = 3a + 2b + c = 3(p_3 - p_2)
  \]
  - Tangents at endpoints are parallel to section of control polyline

Drawing Bézier curves
- Generally no low-level support for drawing curves
  - Draw line segments or individual pixels
  - Approximate the curve as a series of line segments (tessellation)
    - Uniform sampling
    - Adaptive sampling
    - Recursive subdivision
**Uniform sampling**

- Approximate curve with \( N \) straight segments
  - \( N \) chosen in advance
  - Evaluate \( x_i = x(t_i) \) where \( t_i = \frac{i}{N} \) for \( i = 0, 1, \ldots, N \)
  - \( x_i = a \frac{i^3}{N^3} + b \frac{i^2}{N^2} + c \frac{i}{N} + d \)
  - Connect the points with lines
- Too few points?
  - Bad approximation
  - “Curve” is faceted
- Too many points?
  - Slow to draw too many line segments
  - Segments may draw on top of each other

**Adaptive Sampling**

- Use only as many line segments as you need
  - Fewer segments where curve is mostly flat
  - More segments where curve bends
  - Segments never smaller than a pixel
- Various schemes for sampling, checking results, deciding whether to sample more

**Recursive Subdivision**

- Any cubic curve segment can be expressed as a Bézier curve
- Any piece of a cubic curve is itself a cubic curve
- Therefore:
  - Any Bézier curve can be broken up into smaller Bézier curves

**de Casteljau subdivision**

- de Casteljau construction points are the control points of two Bézier sub-segments

**Adaptive subdivision algorithm**

- Use de Casteljau construction to split Bézier segment
- For each half
  - If flat enough: draw line segment
  - Else: recurse
- Curve is flat enough if hull is flat enough
- Test how far the handles are from a straight segment
  - If it’s about a pixel, the hull is flat

**Outline for today**

- Summary of Bézier curves
- Piecewise-cubic Bézier curves
- B-splines
- Surface patches
More control points

- Cubic Bézier curve limited to 4 control points
  - Cubic curve can only have one inflection
  - Need more control points for more complex curves
- K+1 order Bézier curve with k control points
  - Continuous by construction
  - N+1 points define curve

Hard to control and hard to work with
- Intermediate points don’t have obvious effect on shape
- Changing any control point changes the whole curve
- Want local support: each control point only influences nearby portion of curve

Continuity

- Want smooth curves
- C0 continuity
  - No gaps
  - Segments match at the endpoints
- C1 continuity: first derivative is well defined
  - No corners
  - Tangents/normals are C1 continuous (no jumps)
- C2 continuity: second derivative is well defined
  - Important for high quality reflections

Piecewise-linear curve

- Given N+1 points p0, p1, ..., pN
- Define curve

\[ x(u) = \sum_{i=0}^{N} \lambda_i (u, p_{i-1}, p_i), \quad i \leq u \leq i+1 \]

\[ x(u) = (1 - u + i/p, u - i/p), \quad i = [u] \]

- N+1 points define N linear segments
- C0 continuous by construction
- C1 at p, when p_{i-1} = p_{i+1} / 2

Piecewise curves

- Sequence of simple (low-order) curves, end-to-end
  - Known as a piecewise polynomial curve
- Sequence of line segments
  - Piecewise linear curve
- Sequence of cubic curve segments
  - Piecewise cubic curve (here piecewise Bézier)

Global parameterization

- Given N curve segments x0(t), x1(t), ..., xN(t)
- Each is parameterized for t from 0 to 1
- Define a piecewise curve
  - Global parameter u from 0 to N

\[ x(u) = \begin{cases} 
  x_i(u), & 0 \leq u \leq 1 \\
  x_{i+1}(u-1), & 1 \leq u \leq 2 \\
  \vdots & \\
  x_{i-N+1}(u-(N-1)), & N-1 \leq u \leq N \\
  x_{i-N}(N-1), & u \geq N 
\end{cases} \]

Alternate: u also goes from 0 to 1

\[ x(u) = x_i(Nu - i), \quad \text{where } i = \lfloor Nu \rfloor \]

Piecewise Bézier curve

- Given 3N+1 points p_{0,0}, p_{0,1}, ..., p_{N,N}
- Define N Bézier segments:

\[ x_i(t) = B_i(t)p_0 + B_i(t)p_1 + B_i(t)p_2 + B_i(t)p_3 \]

\[ x_i(t) = B_i(t)p_1 + B_i(t)p_2 + B_i(t)p_3 + B_i(t)p_4 \]

\[ \vdots \]

\[ x_{N-1}(t) = B_i(t)p_3N-3 + B_i(t)p_3N-2 + B_i(t)p_3N-1 + B_i(t)p_{3N} \]

- Piecewise Bézier curve (here piecewise Bézier)
Piecewise Bézier curve

- Parameter in \(0 \leq u \leq 3N\)

\[
x(u) = \begin{cases} 
  x_i (\frac{u}{3}), & 0 \leq u \leq 3 \\
  x_i (\frac{u-1}{3}), & 3 \leq u \leq 6 \\
  \vdots \\
  x_{N-1}(\frac{u-(N-1)}{3}), & 3N - 3 \leq u \leq 3N 
\end{cases}
\]

\[x(u) = x_i \left( \frac{u}{3} - i \right), \text{ where } i = \left\lfloor \frac{u}{3} \right\rfloor\]

Piecewise Bézier curves

- Used often in 2D drawing programs
- Inconveniences
  - Must have 4 or 7 or 10 or 13 or ... (1 plus a multiple of 3) control points
  - Some points interpolate, others approximate
  - Need to impose constraints on control points to obtain \(C^1\) continuity
  - \(C^2\) continuity more difficult
- Solutions
  - User interface using “Bézier handles”
  - Generalization to B-splines

Outline for today

- Summary of Bézier curves
- Piecewise-cubic Bézier curves
- B-splines
- Surface patches

Generalization to B-splines

- Evaluate piecewise Bézier curve using sliding window

\[
x(u) = \begin{bmatrix} x_0 & x_1 & \cdots & x_{3k-1} \
  3 & 3 & 2 & 2 & 1 & 1 & 0 & 0 
\end{bmatrix} \begin{bmatrix} \frac{u}{3} \\
  \frac{u}{3} - 1 \\
  \vdots \\
  \frac{u}{3} - (3k-1) \end{bmatrix}
\]

Where \(i = \left\lfloor \frac{u}{3} \right\rfloor\) and \(r = \{u/3\}\)
Generalization to B-splines

- Still sliding window, but same weight function at each point (B-spline blending function)
- Blending functions have local support
- Cubic curve: at each \( u \) only 4 weighting functions are non-zero
- Shift “window” by 1, not by 3

B-splines formulations

- Weighted average of control points using B-spline blending functions \( b_i(u) \) (no details here) \( x(u) = \sum b_i(u) p_i \)
- Positive, partition of unity => convex hull property
- Matrix form

\[
x(u) = \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \begin{bmatrix} p_i \\ p_{i+1} \\ p_{i+2} \\ p_{i+3} \end{bmatrix}
\]

where \( i = u - i \) and \( i = \lfloor u \rfloor \)

B-splines properties

- \( n \)-th order b-spline is \( C^{n-1} \) continuous
- Widely used for curve and surface modeling
  - Intuitive behavior
  - Local support: curve only affected by nearby control points
  - Techniques for inserting new points, joining curves, etc.
- Doesn’t interpolate endpoints
  - Not a problem for closed curves
  - Can be generalized to interpolate any point along the curve

Further generalizations

- Rational B-splines
- Non-uniform B-splines
- Non-uniform rational B-splines (NURBS)

Rational curves

- Big drawback of all cubic curves: can’t make circles!
  - Nor ellipses, nor arcs. I.e. can’t make conic sections
- Rational B-spline
  - Add a weight to each control point
  - Homogeneous point: use \( w \)

\[
x(u) = \sum b_i(u) p_i \\
x(u) = \frac{\sum b_i(u) w_i p_i}{\sum b_i(u) w_i}
\]

Rational curves

- Weight causes point to “pull” more (or less)
- With proper points & weights, can do circles
### Rational curves
- Can generate curves for circles etc. with appropriate weights
- Need extra user interface to adjust the weights
- Often, hand-drawn curves are unweighted

### Non-uniform rational B-splines (NURBS)
- Non-uniform: don’t assume that control points are equidistant in $u$
- Introduce knot vector
  - Describes the distribution of the control points
- More flexibility in defining blending functions

<table>
<thead>
<tr>
<th>Uniform knot vector</th>
<th>Nonuniform knot vector</th>
</tr>
</thead>
</table>

### NURBS
- Can make corners ($C^1$ discontinuity)
- Certain knot values turn out to give a Bézier segment
- Allows mixing interpolating (e.g. at endpoints) and approximating

| Bézier curve as B-spline with nonuniform knot vector |

### NURBS
- Math is more complicated
- Very widely used for curve and surface modeling
  - Supported by virtually all 3D modeling tools
  - Open source modeling tool: www.blender.org
- Techniques for cutting, inserting, merging, revolving, etc...
- Applets
  - [http://ibiblio.org/e-notes/Splines/Intro.htm](http://ibiblio.org/e-notes/Splines/Intro.htm)