Keyframe animation
• Process of keyframing

• Keyframe interpolation

• Hermite and Bezier curves

• Splines

• Speed control
Oldest keyframe animation

- Two conditions to make moving images in 19th century
  - at least 10 frames per second
  - a period of blackness between images
2D animation

• Highly skilled animators draw the keyframes
• Less skilled (lower paid) animators draw the in-between frames
• Time consuming process
• Difficult to create physically realistic animation
3D animation

- Animators specify important keyframes in 3D
- Computers generates the in-between frames
- Some dynamic motion can be done by computers (hair, clothes, etc)
- Still time consuming; Pixar spent four years to produce Toy Story
General pipeline

- Story board
- Keyframes
- Inbetweens
- Painting
Storyboards

- The film in outline form
- specify the key scenes
- specify the camera moves and edits
- specify character gross motion
- Typically paper and pencil sketches on individual sheets taped on a wall
“A bug’s life”

The process of keyframing

- Specify the keyframes
- Specify the type of interpolation
  - linear, cubic, parametric curves
- Specify the speed profile of the interpolation
  - constant velocity, ease-in-ease-out, etc
- Computer generates the in-between frames
A keyframe

- In 2D animation, a keyframe is usually a single image
- In 3D animation, each keyframe is defined by a set of parameters
Keyframe parameters

- What are the parameters?
  - position and orientation
  - body deformation
  - facial features
  - hair and clothing
  - lights and cameras
• Process of keyframing
• Keyframe interpolation
• Hermite and Bezier curves
• Splines
• Speed control
In-between frames

- Linear interpolation
- Cubic curve interpolation
Linear interpolation

Linearly interpolate the parameters between keyframes

$$x = x_0 + \frac{t - t_0}{t_1 - t_0} (x_1 - x_0)$$

- **t = 0**
- **t = 5**
- **t = 10**
- **t = 15**
Cubic curve interpolation

We can use three cubic functions to represent a 3D curve:

Each function is defined with the range $0 \leq t \leq 1$

$$Q(t) = [x(t) \ y(t) \ z(t)]$$

or

$$Q_x(t) = a_x t^3 + b_x t^2 + c_x t + d_x$$
$$Q_y(t) = a_y t^3 + b_y t^2 + c_y t + d_y$$
$$Q_z(t) = a_z t^3 + b_z t^2 + c_z t + d_z$$

bold: a vector or a matrix
italic: a scalar

vectors
- $a \cdot b$: inner product
- $a \times b$: cross product
- $ab$: multiplication

matrices
- $A \cdot B$: multiplication
Compact representation

\[ Q(t) = \begin{bmatrix} Q_x(t) & Q_y(t) & Q_z(t) \end{bmatrix} \]

\[ Q_x(t) = a_x t^3 + b_x t^2 + c_x t + d_x \]
\[ Q_y(t) = a_y t^3 + b_y t^2 + c_y t + d_y \]
\[ Q_z(t) = a_z t^3 + b_z t^2 + c_z t + d_z \]

\[ C = \begin{bmatrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \\ d_x & d_y & d_z \end{bmatrix} \quad \quad T = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \]
Compact representation

\[ Q(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \\ d_x & d_y & d_z \end{bmatrix} = TC \]

\[ \dot{Q} = \frac{d}{dt} Q(t) = \begin{bmatrix} 3t^2 & 2t & 1 & 0 \end{bmatrix} C \]
Constraints on the cubics

How many constraints do we need to determine a cubic curve? 4

Redefine $C$ as a product of the basis matrix $M$ and the geometry matrix $G$

$$C = M \cdot G$$

$$Q(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix} \begin{bmatrix} G_{1x} & G_{1y} & G_{1z} \\ G_{2x} & G_{2y} & G_{2z} \\ G_{3x} & G_{3y} & G_{3z} \\ G_{4x} & G_{4y} & G_{4z} \end{bmatrix} = T \cdot M \cdot G$$
• Process of keyframing
• Keyframe interpolation
• Hermite and Bezier curves
• Splines
• Speed control
Hermite curves

- A Hermite curve is determined by
  - endpoints $P_1$ and $P_4$
  - tangent vectors $R_1$ and $R_4$ at the endpoints
- Use these elements to construct geometry matrix

$$G_h = \begin{bmatrix} P_{1x} & P_{1y} & P_{1z} \\ P_{4x} & P_{4y} & P_{4z} \\ R_{1x} & R_{1y} & R_{1z} \\ R_{4x} & R_{4y} & R_{4z} \end{bmatrix}$$
Given desired constraints:

1. endpoints meet $P_1$ and $P_4$

$$Q(0) = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \end{bmatrix} \cdot M_h \cdot G_h = P_1$$
$$Q(1) = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix} \cdot M_h \cdot G_h = P_4$$

2. tangent vectors meet $R_1$ and $R_4$

$$\dot{Q}(0) = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \cdot M_h \cdot G_h = R_1$$
$$\dot{Q}(1) = \begin{bmatrix} 3 & 2 & 1 & 0 \end{bmatrix} \cdot M_h \cdot G_h = R_4$$
Hermite basis matrix

We can solve for basis matrix $M_h$

$$
\begin{bmatrix}
P_1 \\
P_4 \\
R_1 \\
R_4 \\
\end{bmatrix}
= 
G_h =
\begin{bmatrix}
0 & 0 & 0 & 1 \\
1 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 \\
3 & 2 & 1 & 0 \\
\end{bmatrix}
\cdot
M_h
\cdot
G_h
$$

$$
M_h =
\begin{bmatrix}
0 & 0 & 0 & 1 \\
1 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 \\
3 & 2 & 1 & 0 \\
\end{bmatrix}
^{-1}
= 
\begin{bmatrix}
2 & -2 & 1 & 1 \\
-3 & 3 & -2 & -1 \\
0 & 0 & 1 & 0 \\
1 & 0 & 0 & 0 \\
\end{bmatrix}
$$
Hermite Blending functions

Let’s define \( B \) as a product of \( T \) and \( M \)

\[
B_h(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}
\]

\( B_h(t) \) indicates the weight of each element in \( G_h \)

\[
Q(t) = B_h(t) \cdot \begin{bmatrix} P_1 \\ P_4 \\ R_1 \\ R_4 \end{bmatrix}
\]
Bézier curves

Indirectly specify tangent vectors by specifying two intermediate points

\[ \mathbf{R}_1 = 3(\mathbf{P}_2 - \mathbf{P}_1) \]

\[ \mathbf{R}_4 = 3(\mathbf{P}_4 - \mathbf{P}_3) \]

\[ \mathbf{G}_b = \begin{bmatrix} \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \\ \mathbf{P}_4 \end{bmatrix} \]
Bézier basis matrix

Establish the relation between Hermite and Bezier geometry vectors

\[
G_h = \begin{bmatrix}
P_1 \\
P_4 \\
R_1 \\
R_4
\end{bmatrix} = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 \\
-3 & 3 & 0 & 0 \\
0 & 0 & -3 & 3
\end{bmatrix} \begin{bmatrix}
P_1 \\
P_2 \\
P_3 \\
P_4
\end{bmatrix} = M_{hb} \cdot G_b
\]
Bézier basis matrix

\[ Q(t) = T \cdot M_h \cdot G_h = T \cdot M_h \cdot (M_{hb} \cdot G_b) \]
\[ = T \cdot (M_h \cdot M_{hb}) \cdot G_b = T \cdot M_b \cdot G_b \]

\[ M_b = M_h M_{hb} = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \]

http://www.math.ucla.edu/~baker/java/hoefer/Bezier.htm
Bézier blending functions

Bezicer blending functions are also called Bernstein polynomials

\[ B_b(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \]

\[ Q(t) = B_b(t) \cdot \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix} \]
What if we want to model a curve that passes through these points?

Problem with higher order polynomials:

- Wiggly curves
- No local control
• Process of keyframing
• Keyframe interpolation
• Hermite and Bezier curves
• Splines
• Speed control
Splines

- A piecewise polynomial that has locally very simple form, yet be globally flexible and smooth
- There are three nice properties of splines we’d like to have
  - Continuity
  - Local control
  - Interpolation
Continuity

- Cubic curves are continuous and differentiable
- We only need to worry about the derivatives at the endpoints when two curves meet
Continuity

$C^0$: points coincide, velocities don’t

$C^1$: points and velocities coincide

What’s $C^2$?

points, velocities and accelerations coincide
Local control

- We’d like to have each control point on the spline only affect some well-defined neighborhood around that point.
- Bezier and Hermite curves don’t have local control; moving a single control point affects the whole curve.
Interpolation

- We’d like to have a spline interpolating the control points so that the spline always passes through every control points
- Bezier curves do not necessarily pass through all the control points
B-splines

- We can join multiple Bezier curves to create B-splines
- Ensure $C^2$ continuity when two curves meet
Continuity in B-splines

Suppose we want to join two Bezier curves \((V_0, V_1, V_2, V_3)\) and \((W_0, W_1, W_2, W_3)\) so that \(C^2\) continuity is met at the joint.

\[
\begin{align*}
Q_v(1) &= Q_w(0) \\
\dot{Q}_v(1) &= \dot{Q}_w(0) \\
\ddot{Q}_v(1) &= \ddot{Q}_w(0)
\end{align*}
\]

\[
\begin{align*}
V_3 &= W_0 \\
V_3 - V_2 &= W_1 - W_0 \\
V_1 - 2V_2 + V_3 &= W_0 - 2W_1 + W_2 \\
W_2 &= V_1 + 4V_3 - 4V_2
\end{align*}
\]
Continuity in B-splines

What does this derived equation mean geometrically?

\[ W_2 = V_1 + 4V_3 - 4V_2 \]

What is the relationship between \( a, b \) and \( c \), if \( a = 2b - c \)?

- \( W_0 = V_3 \)
- \( W_1 = 2V_3 - V_2 \)
- \( W_2 = V_1 + 4V_3 - 4V_2 = 2(2V_3 - V_2) - (2V_2 - V_1) = 2W_1 - B_1 \)

What is \( B_2 \)?
de Boor points

Instead of specifying the Bezier control points, let’s specify the corners of the frames that form a B-spline.

These points are called de Boor points and the frames are called A-frames.
de Boor points

What is the relationship between Bezier control points and de Boor points?

\[ V_0 = \frac{1}{2} \left( B_0 + \frac{2}{3}(B_1 - B_0) + B_1 + \frac{1}{3}(B_2 - B_1) \right) \]

\[ V_1 = B_1 + \frac{1}{3}(B_2 - B_1) \]

\[ V_2 = B_1 + \frac{2}{3}(B_2 - B_1) \]

\[ V_3 = \frac{1}{2} \left( B_1 + \frac{2}{3}(B_2 - B_1) + B_2 + \frac{1}{3}(B_3 - B_2) \right) \]

Verify this by yourself
What about the next set of Bezier control points, $W_0$, $W_1$, $W_2$, and $W_3$? What de Boor points do they depend on?

B1, B2, B3 and B4.

Verify it by yourself
B-spline basis matrix

\[ Q(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} M_{bs} \begin{bmatrix} B_1 \\ B_2 \\ B_3 \\ B_4 \end{bmatrix} \]

\[ M_{bs} = \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \]

http://www.siggraph.org/education/materials/HyperGraph/modeling/splines/demoprog/curve.html
B-Spline properties

- $C^2$ continuity?
- Local control?
- Interpolation?
B-Spline properties

- C² continuity?
- Local control?
- Interpolation?
Catmull-Rom splines

- If we are willing to sacrifice $C^2$ continuity, we can get interpolation and local control.
- If we set each derivative to be a constant multiple of the vector between the previous and the next controls, we get a Catmull-Rom spline.
Catmull-Rom splines

\[
D_0 = C_1 - C_0
\]
\[
D_1 = \frac{1}{2} (C_2 - C_0)
\]
\[
D_2 = \frac{1}{2} (C_3 - C_1)
\]
\[\vdots\]
\[
D_n = C_n - C_{n-1}
\]
Catmull-Rom Basis matrix

\[
Q(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \frac{1}{2} \begin{bmatrix}
-1 & 3 & -3 & 1 \\
2 & -5 & 4 & -1 \\
-1 & 0 & 1 & 0 \\
0 & 2 & 0 & 0
\end{bmatrix} \begin{bmatrix}
P_1 \\
P_2 \\
P_3 \\
P_4
\end{bmatrix}
\]

Derive Catmull-Rom basis matrix by yourself
Catmull-Rom properties

- $C^2$ continuity?
- Local control?
- Interpolation?
Catmull-Rom properties

- C^2 continuity?
- Local control?
- Interpolation?
How can we keep the $C^2$ continuity of B-splines but get interpolation property as well?

Suppose we have a set of Bezier control points, our goal is to find a $C^2$ spline that passes through all the points.
We know the control points $C$’s, but we don’t know the tangents $D$’s.

If we want to create a Bezier curve between each pair of these points, what are the $V$’s and $W$’s control points in terms of $C$’s and $D$’s?
Derivatives of splines

\[ V_0 = C_0 \]
\[ V_1 = C_0 + \frac{1}{3}D_0 \]
\[ V_2 = C_1 - \frac{1}{3}D_1 \]
\[ V_3 = C_1 \]

\[ W_0 = C_1 \]
\[ W_1 = C_1 + \frac{1}{3}D_1 \]
\[ W_2 = C_2 - \frac{1}{3}D_2 \]
\[ W_3 = C_2 \]

To solve for \( D \)'s we apply \( C^2 \) continuity requirement

\[ 6(V_1 - 2V_2 + V_3) = 6(W_0 - 2W_1 + W_2) \]
\[ \downarrow \]
\[ D_0 + 4D_1 + D_2 = 3(C_2 - C_0) \]
Derivatives of splines

\[ D_0 + 4D_1 + D_2 = 3(C_2 - C_0) \]
\[ D_1 + 4D_2 + D_3 = 3(C_3 - C_1) \]
\[ \vdots \]
\[ D_{m-2} + 4D_{m-1} + D_m = 3(C_m - C_{m-2}) \]

How many equations do we have?

How many variables are we trying to solve?
Boundary conditions

We can impose more conditions on the spline to solve the two extra degrees of freedom

Natural $C^2$ interpolating splines require second derivative to be zero at the endpoints

$$6(V_0 - 2V_1 + V_2) = 0$$
Collect \( m+1 \) equations into a linear system

\[
\begin{bmatrix}
2 & 1 & & & \\
1 & 4 & 1 & & \\
1 & 4 & 1 & & \\
\vdots & & & & \\
1 & 4 & 1 & & \\
1 & 2 & & & \\
\end{bmatrix}
\begin{bmatrix}
D_0 \\
D_1 \\
D_2 \\
\vdots \\
D_{m-1} \\
D_m \\
\end{bmatrix}
= 
\begin{bmatrix}
3(C_1 - C_0) \\
3(C_2 - C_0) \\
3(C_3 - C_1) \\
\vdots \\
3(C_m - C_{m-2}) \\
3(C_m - C_{m-1}) \\
\end{bmatrix}
\]

Use forward elimination to zero out every thing below the diagonal, then back substitute to compute \( D \)'s
## Choice of Splines

<table>
<thead>
<tr>
<th>Spline types</th>
<th>Continuity</th>
<th>Interpolation</th>
<th>Local control</th>
</tr>
</thead>
<tbody>
<tr>
<td>B-Splines</td>
<td>$C^2$</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Catmull-Rom Splines</td>
<td>$C^1$</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>$C^2$ interpolating splines</td>
<td>$C^2$</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>
de Casteljau’s algorithm

For each sample of $t$ from 0 to 1, use de Casteljau’s algorithm to compute $Q(t)$

Where is $Q(0)$?
Where is $Q(1)$?
Where is $Q(\frac{1}{3})$?

What is the equation for $v_0^1$?
de Casteljau’s algorithm

\[ V_0^1 = (1 - t)V_0 + tV_1 \]
\[ V_1^1 = (1 - t)V_1 + tV_2 \]
\[ V_2^1 = (1 - t)V_2 + tV_3 \]

\[ V_0^2 = (1 - t)V_0^1 + tV_1^1 \]
\[ V_1^2 = (1 - t)V_1^1 + tV_2^1 \]

\[ Q(t) = (1 - t)V_0^2 + tV_1^2 \]
\[ = (1 - t)[(1 - t)V_0^1 + tV_1^1] + t[(1 - t)V_1^1 + tV_2^1] \]
\[ = (1 - t)^3V_0 + 3t(1 - t)^2V_1 + 3t^2(1 - t)V_2 + t^3V_3 \]
\[ = \sum_{i=0}^{n} \binom{n}{i} t^i (1 - t)^{n-i} V_i \]
Displaying Bezier curves

DisplayBezier(V0, V1, V2, V3)
begin
    if (FlatEnough(V0, V1, V2, V3))
        Line(V0, V3)
    else
        do something

It would be nice if we had an adaptive algorithm that would take into account flatness.
```
DisplayBezior(V_0, V_1, V_2, V_3)
begin
  if (FlatEnough(V_0, V_1, V_2, V_3))
    Line(V_0, V_3)
  else
    Subdivide(V) -> L, R
    DisplayBezior(L_0, L_1, L_2, L_3)
    DisplayBezior(R_0, R_1, R_2, R_3)
```
Flatness Test

Compare total length of control polygon to length of line connecting endpoints:

\[
\frac{|V_0 - V_1| + |V_1 - V_2| + |V_2 - V_3|}{|V_0 - V_3|} < 1 + \epsilon
\]
Endpoints of B-splines

- We can see that B-splines don’t interpolate the de Boor points
- It would be nice if we could at least control the endpoints of the splines explicitly
- There’s a trick to make the spline begin and end at the de Boor points by repeating them
Endpoints of B-splines

How many $B_0$'s need to be repeated?

3 times. See slide 41.
Wrapping the curves

• Wrapping is an important feature that makes the animation restart smoothly when looping back to the beginning

• Create “phantom” control points before and after the first and the last control points
• Process of keyframing
• Keyframe interpolation
• Hermite and Bezier curves
• Splines
• Speed control
Speed control

- Simplest form is to have constant velocity along the path
Ease-in Ease-out curve

- Assume that the motion slows down at the beginning and end of the motion curve
Issues

• What kind of bad things can occur from interpolation? How do we prevent them?
  • Invalid configurations (pass through walls)
  • Unnatural motions (painful twists)
What’s next?
• **What about rotation?**

• **Can we interpolate rotations using the these same techniques?**