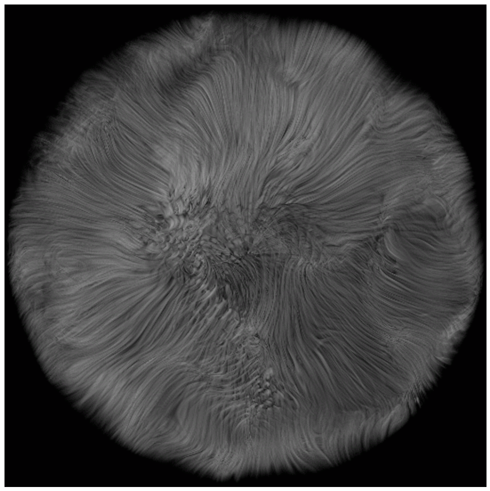
CS148: Introduction to Computer Graphics and Imaging

Interpolation and Basis Fns



Topics

Today

- Interpolation
  - Linear and bilinear interpolation
  - Barycentric interpolation
- Basis functions
  - Square, triangle, …,
  - Hermite cubic interpolation
  - Interpolating random numbers to make noise

Thursday

- Splines and curves
  - Catmull-Rom splines
  - Bezier curves

Interpolation

Fill in between values

Convert discrete (finite) to continuous (infinite)

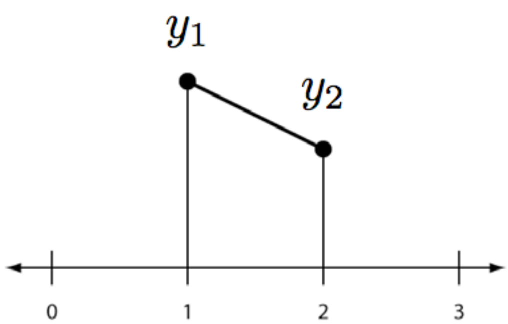
Examples:

- Interpolating across a triangle
  - Interpolating between vertices
- Filtering and reconstructing images
  - Interpolating between pixels/texels
- Creating random functions
  - Noise
- Generating motion
  - Interpolating “inbetween” frames from “keyframes”
- Curves and surfaces
  - Interpolating between control points

Interpolation

Linear Interpolation

$$y(t) = (1-t)\, y_1 + t\, y_2$$



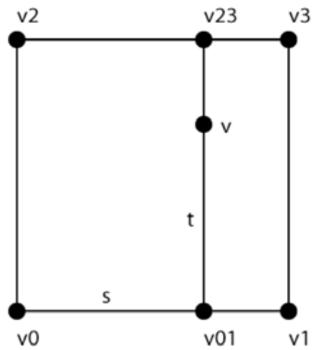
Linear Interpolation - `lerp`

One of the most useful functions in computer graphics

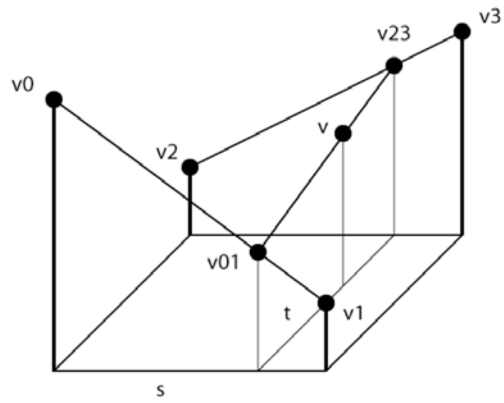
```
lerp(t, v0, v1) {
   return (1-t)*v0 + t*v1;
}

lerp(t, v0, v1) {
   return v0 + t*(v1-v0);
}
```

Bilinear Interpolation

```
bilerp(s, t, v0, v1, v2, v3) {
    v01 = lerp(s, v0, v1);
    v23 = lerp(s, v2, v3);
    v = lerp(t, v01, v23);
    return v;
}
```



Bilinear Interpolation



"Ruled" Surface

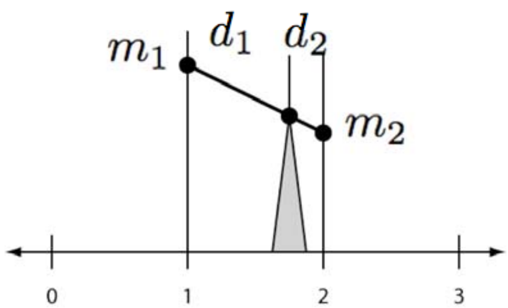
Barycentric Coordinates

Given masses: m1, m2; Given distances: d1, d2

Balance condition (torques equal): m1 × d1 = m2 × d2

Therefore: d1 ∝ m2 and d2 ∝ m1

Alternatively: d1 = m2 / (m1 + m2); d2 = m1 / (m1 + m2)

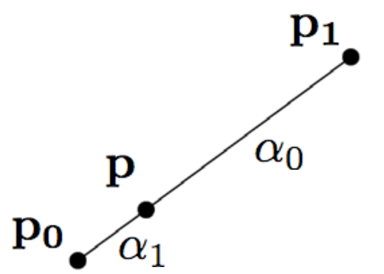


Barycentric Interpolation

Edge

$$\mathbf{p} = \alpha_0 \mathbf{p_0} + \alpha_1 \mathbf{p_1}$$

$$\alpha_0 + \alpha_1 = 1$$

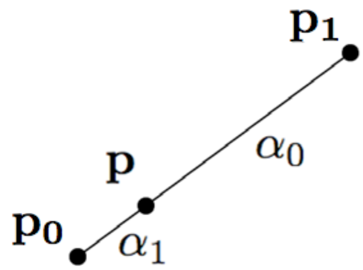


Barycentric Interpolation

Edge

$$\mathbf{p} = \alpha_0 \mathbf{p_0} + \alpha_1 \mathbf{p_1}$$

$$\alpha_0 + \alpha_1 = 1$$

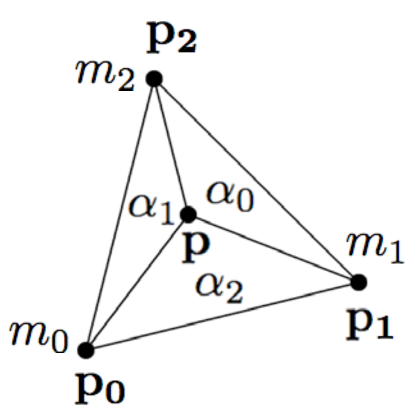


Triangle

$$\mathbf{p} = \alpha_0 \, \mathbf{p_0} + \alpha_1 \, \mathbf{p_1} + \alpha_2 \, \mathbf{p_2}$$

$$\alpha_0 + \alpha_1 + \alpha_2 = 1$$

Barycentric Interpolation - Triangle



$$\alpha_0 = \frac{\mathrm{area}(\mathbf{p}\,\mathbf{p_1}\,\mathbf{p_2})}{\mathrm{area}(\mathbf{p_0}\,\mathbf{p_1}\,\mathbf{p_2})}$$

$$\alpha_1 = \frac{\mathrm{area}(\mathbf{p_0}\,\mathbf{p}\,\mathbf{p_2})}{\mathrm{area}(\mathbf{p_0}\,\mathbf{p_1}\,\mathbf{p_2})}$$

$$\alpha_2 = \frac{\mathrm{area}(\mathbf{p_0}\,\mathbf{p_1}\,\mathbf{p})}{\mathrm{area}(\mathbf{p_0}\,\mathbf{p_1}\,\mathbf{p_2})}$$

Center of mass: $\mathbf{p}$

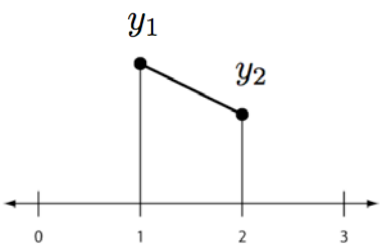
Triangle

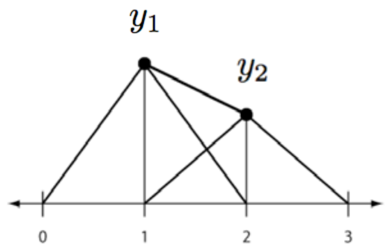
Barycentric interpolation

- Precisely defined
- Parameters define points inside the triangle
- If all parameters positive, then inside
- Generalizes to 3D
- Can be used to interpolate colors
- Can be used to interpolate textures
- Example of homogenous coordinates

Basis Functions

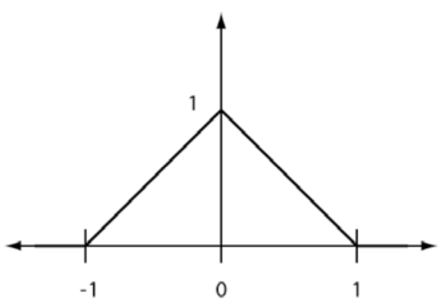
Linear Interpolation = Triangle Basis

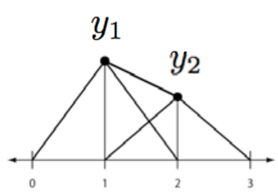




$$y(t) = (1-t)\,y_1 + t\,y_2 \qquad y(t) = y_1\,T_1(t) + y_2\,T_2(t)$$

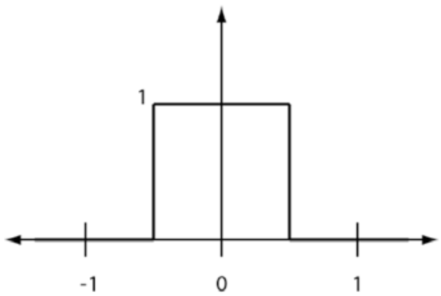
Linear Interpolation = Triangle Basis

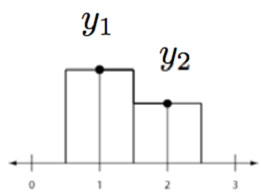




$$T(t) = \begin{cases} 0 & t < -1 \\ 1+t & -1 < t < 0 \\ 1-t & 0 < t < 1 \\ 0 & t > 1 \end{cases}$$

Constant Interpolation = Square Basis





Nearest-Neighbor Interpolation

$$\Pi(t) = \begin{cases} 0 & t < -0.5 \\ 1 & -0.5 < t < 0.5 \\ 0 & t > 0.5 \end{cases}$$

Basis Functions

Basic formula

$$y(t) = \sum_{i=0}^{n} y_i \, B_i(t)$$

Basis functions

$$B_i(t)$$

Often i'th functions are shifted versions of 0'th

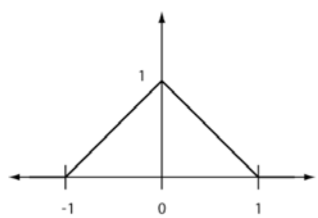
Interpolating Function

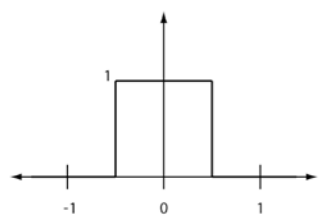
Necessary conditions:

$$B_i(0) = 1$$

$$B_i(k) = 0$$

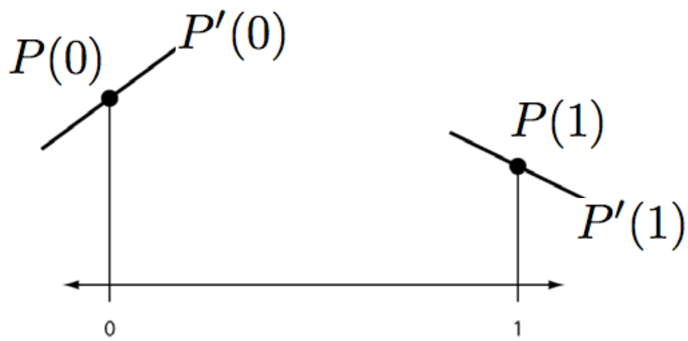
True for triangle and square basis functions





Cubic Hermite Interpolation

Cubic Hermite Interpolation



Given: values and derivatives at 2 points

Hermite Basis Function Formulation

$$h_0 = P(0)$$
$$h_1 = P(1)$$
$$h_2 = P'(0)$$
$$h_3 = P'(1)$$

$$P(t) = \sum_{i=0}^{3} h_i \, H_i(t)$$

Cubic Hermite Interpolation

Assume cubic polynomial

$$P(t) = a\,t^3 + b\,t^2 + c\,t + d$$

Why? 4 coefficients need 4 degrees of freedom

Cubic Hermite Interpolation

Assume cubic polynomial

$$P(t) = a\,t^3 + b\,t^2 + c\,t + d$$
$$P'(t) = 3a\,t^2 + 2b\,t + c$$

Solve for coefficients:

$$P(0) = h_0 = d$$
$$P(1) = h_1 = a + b + c + d$$
$$P'(0) = h_2 = c$$
$$P'(1) = h_3 = 3a + 2b + c$$

Determine Polynomial Coefficients

$$P(0) = h_0 = d$$

$$P(1) = h_1 = a + c + c + d$$

$$P'(0) = h_2 = c$$

$$P'(1) = h_3 = 3a + 2b + c$$

Solve

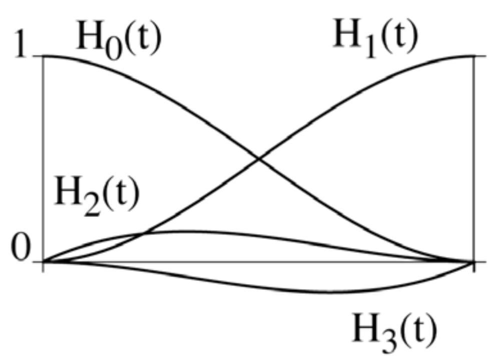
$$a = 2h_0 - 2h_1 + h_2 + h_3$$

$$b = -3h_0 + 3h_1 - 2h_2 - h_3$$

$$c = h_2$$

$$d = h_0$$

Hermite Basis Functions



$$H_0(t) = 2t^3 - 3t^2 + 1$$

$$H_1(t) = -2t^3 + 3t^2$$

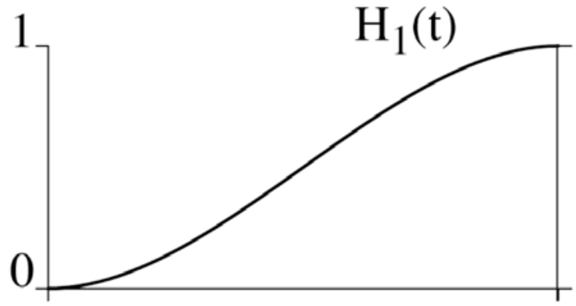
$$H_2(t) = t^3 - 2t^2 + t$$

$$H_3(t) = t^3 - t^2$$

Ease

A *very useful* function

In animation, start and stop slowly (zero velocity)



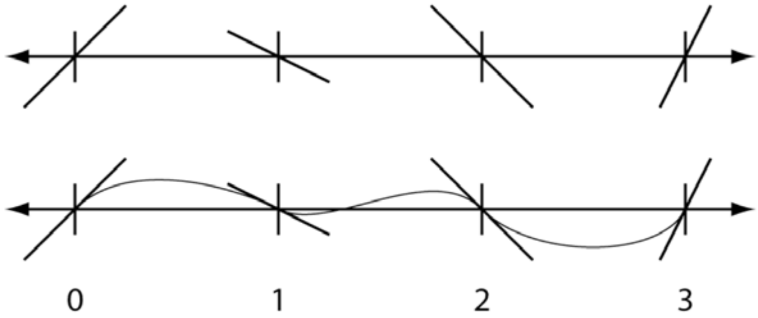
$$H_1(t) = -2t^3 + 3t^2 = t^2(3 - 2t)$$

CS148 Lecture 7 Pat Hanrahan, Winter 2009

Fractal Interpolation

Ken Perlin Noise

Idea: Interpolate random slopes

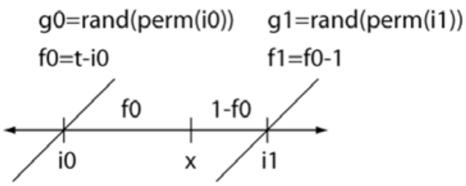


CS148 Lecture 7 Pat Hanrahan, Winter 2009

Code

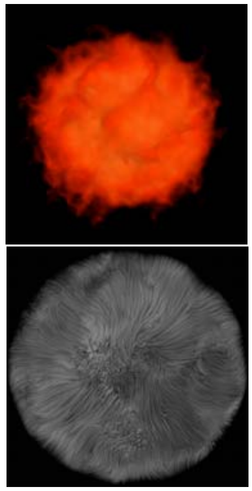
http://mrl.nyu.edu/~perlin/doc/oscar.html

```
double noise1(double x)
{
    double t = x + N;
    // compute integer locations
    int i0 = (int)t % BITSN;
    int i1 = (i0+1) % BITSN;
    // compute fractional parts
    double f = t - (int)t;
    double f0 = f;
    double f1 = f - 1.;

    double g0 = rand[ perm[ i0 ] ];
    double g1 = rand[ perm[ i1 ] ];
    double u = f0 * g0;
    double v = f1 * g1;

    double s = f0 * f0 * (3. - 2. * f0); // hermite spline
    return(lerp(s, u, v));
```



CS148 Lecture 7 Pat Hanrahan, Winter 2009

Noise and Turbulence Functions



Ken Perlin, An Image Synthesizer

Things to Remember

Interpolation

- Widely used in graphics: image, texture, noise, animation, curves and surfaces
- Nearest neighbor, bilinear, cubic interpolation

Basis functions

- Square
- Triangle
- Hermite
- Noise
- Many others: sines, cosines, sinc, wavelets, …