Notes on tweening

Readings: Hill, Computer Graphics with OpenGL, sections 4.5.3 and 4.5.4.

Tweening is an important operation that gets wide use in graphics. Tweening is a blending of two shapes to get intermediate versions, as in this diagram. Here the point R is blended into point S.

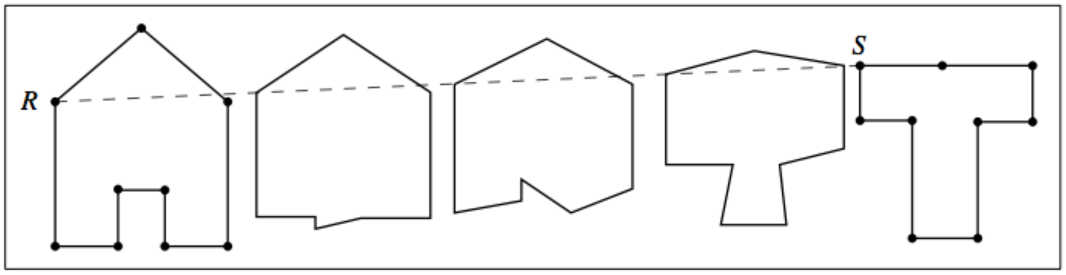


FIGURE 4.22 Tweening a *T* into a house.

The book presents linear tweening of polylines that is based on the blending equation

$$P(t) = (1-t)A + tB$$

Assume the first polyline is A, and the second polyline is B, and both have n points. Then the basic algorithm for tweening becomes

```
$for(\text{int}\, i = 0; i < n; i++)\{$
$P_i(t) = (1-t)A_i + tB_i$
$vertex(P_i(t))$
$\}$
```

Most graphics system provide a library routine for handling blending. In Processing there is the lerp routine, which is the same as the lerp code presented in Hill on page 160. A complete tweening program in Processing is presented at the end of these notes.

There are many applications of tweening that you see very frequently in animation and special effects. Logos fly in commercials by tweening; the ETrade baby's mouth is manipulated by morphing, in which tweening is applied to an image; and both hand and machine animated films use keyframe animation for efficiency.

a) *Shape tweening.* This is what the book emphasizes – the points in two shapes are interpolated to get intermediate shapes.

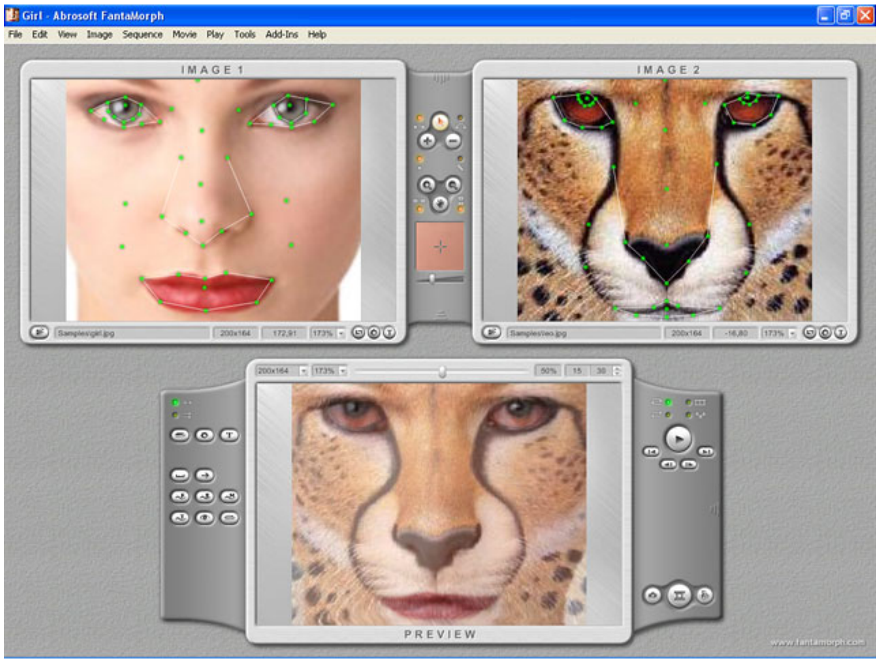
b) *Position animation.* In this, a shape is not changed, but moves along a path defined by a linear interpolation.

c) *Keyframe animation*. Keyframe animation is a combination of shape and position animation where an animator, instead of drawing each of 30 frames a second, draws a keyframe every several frame, and then uses tweening to create the intermediate frame. This can be for hand drawn or machine animation.

d) *Color tweening.* In color tweening or blending, two RGB colors are treated as points and tweened to get intermediate colors. In Processing:

```
float r = lerp( r1, r2, t);
float g = lerp( b1, b2, t);
float b = lerp( g1, g2, t);
fill( r, g, b );
```

e) *Morphing*. In morphing, matched points between two images are tweened in the standard way. In the image below the green points were selected by hand. The regions between the points are distorted along with the matched points.


Girl - Abrosoft FantaMorph
File Edit View Image Sequence Movie Play Tools Add-Ins Help
IMAGE 1
IMAGE 2
PREVIEW
www.fantamorph.com

```
// Tweening example
// Tweens between car and tower polyline
//

// Car outline
float [] x1={69.0,70.0,150.0,186.0,262.0,283.0,354.0,357.0,295.0,280.0,
      247.0,234.0,173.0,147.0,118.0,104.0};
float [] y1 =
{228.0,148.0,146.0,96.0,96.0,144.0,150.0,234.0,234.0,258.0,
259.0,239.0,233.0,261.0,261.0,239.0};
// Tower outline
float [] x2 =
{138.0,154.0,119.0,42.0,80.0,192.0,294.0,309.0,231.0,211.0,234.0,277.0,
221.0,156.0,100.0,129.0};
float [] y2 =
{276.0,184.0,145.0,129.0,78.0,59.0,93.0,149.0,150.0,188.0,273.0,313.0,3
40.0,346.0,313.0,285.0};

void setup() {
  size(400,400);
  strokeWeight(5);
}

void draw() {
  background(128);

  // Scale t to be between 0 and 1
  float t = mouseX/(float)width;

  beginShape();
  // use LERP function to interpolate between individual pts
  for (int i = 0; i < x1.length; i++) {
    float x = lerp( x1[i], x2[i], t );
    float y = lerp( y1[i], y2[i], t );
    vertex(x,y);
  }
  endShape(CLOSE);

}
```