Image Morphing

CS 510 Lecture #10
22u2215202

Morphing

- Given two images, create a series of intermediate images that blend them.

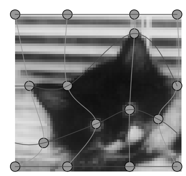
From *The Computer Image* by A. Watt & F. Policarpo

Morphing: How?

- Objects being morphed may vary in
  - Shape
  - (Apparent) color
- We will interpolate both properties
  - A grid of matching "control points" will dictate how shape is interpolated
    - We will use a two-pass technique
  - Pixel values will be linearly interpolated between frames
    - Extension: add a photometric "knot vector" to dictate the rate of pixel interpolation

Representing Shape

- Lay a 4x4 grid of control points over source image



- Fit a Bezier curve to each column of points (green)
- Later, we will Fit a Bezier curve to each row of points (red)

Shape (II)

- Do the same thing for the target image
  - Control points should match across images
  - Movement of control points between images determines the shape deformation
- General idea is to
  - interpolate control points between images
  - Use control point splines to interpolate other pixels

(For the moment, ignore photometric changes)

2-Spline Mesh Warping

- Goal:
  - Create N intermediate frames between image A and image B, with
  - 16 control points in each: $C_pA_{i,j}$ & $C_pB_{i,j}$
    - $1 \le i,j \le 4$
- Step 1: place control points in intermediate image
  - High end movies: place by hand
  - Otherwise: linearly interpolate

2-Spline (II)

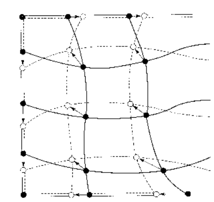
- Step 1 (cont.):
  - For each control point, compute position in new image:

$$M_{i,j} = C_p A_{i,j} + \frac{F}{N+1}\left(C_p B_{i,j} - C_p A_{i,j}\right)$$

    - F is frame number; N is number of (intermediate) frames
  - This creates an intermediate grid of control points
- Step 2: Fit 4 Bezier curves (or BSplines) through the columns of control points in A & F

2-Spline Illustration

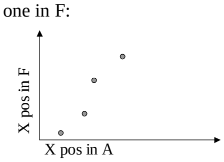
- At this point, you should have two grids with Bezier curves:



From *The Computer Image* by A. Watt & F. Policarpo

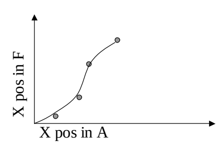
2-Spline (III)

- Step 3: For every row Y
  - Calculate scan-line intersections with the four Bezier curves in A
  - Do the same for the curves in F
  - Every control point now has two x-intercepts, one in A and one in F:



2-Spline (IV)

- For every row Y (cont.):
  - Fit a curve to the 4 points in this graph:



  - This curve now maps x positions in A to x positions in B (and vice-versa) for row Y

2-Spline (V)

- Step 4: Create an Image $F_{A,x}$ by:
  - For every row Y, create X mapping function as above
  - For every pixel in row Y, use mapping function to find (real-valued) x coordinates in A
  - Set value from A using 1D interpolation.
- Step 5: Create an Image $F_{A,xy}$ by repeating steps 1-4, except:
  - Use $F_{A,x}$ instead of A
    - Requires finding control points in $F_{A,x}$
  - Swap role of X and Y coordinates

2-Spline (VI)

- Create $F_{B,xy}$ by repeating steps 1-5, except:
  - Swap B for A
- Create F by doing a pixelwise weighted average of $F_{A,xy}$ and $F_{B,xy}$

$$F(x,y) = \frac{F}{N+1} F_{A,xy}(x,y) + \left[1 - \frac{F}{N+1}\right] F_{B,xy}(x,y)$$

Bezier Curves (review)

- Bezier curves are 3rd order curves:

$$x(t) = a_x t^3 + b_x t^2 + c_x t + d_y$$
$$y(t) = a_y t^3 + b_y t^2 + c_y t + d_y$$

- Notice that nothing changes in 2D (vs 3D)

Fitting Bezier's

- Bezier's are fit to data using the Basis matrix:

$$Q(t) = T\begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} P1_x & P1_y \\ P2_x & P2_y \\ P3_x & P3_y \\ P4_x & P4_y \end{bmatrix}$$

- So, for example,

$$a_x = -P1_x + 3P2_x - 3P3_x + P4_x$$

Applying Bezier's

- How do you find scan-line intersections with a Bezier curve?
  - By definition, y(t) is known (it's the scan line)
  - So are all a's, b's, c's and d's.
  - Plug y(t) into Bezier equation:

$$y(t) = a_y t^3 + b_y t^2 + c_y t + d_y$$

  - Solve for t (fortunately, we know how to solve 3rd order equations!)
  - Calculate x(t)!

The Assignment

- Write a program that takes as input:
  - Source image A + a file of 16 control points
  - Target image B + a file of 16 control points
  - N, the number of intermediate frames
- It should generate N intermediate images
- Possible Extensions:
  - Allow different sized input & output (requires filtering)
  - Allow bicubic as well as bilinear interpolation
  - Use BSplines instead of Bezier curves
  - Add perspective transformations