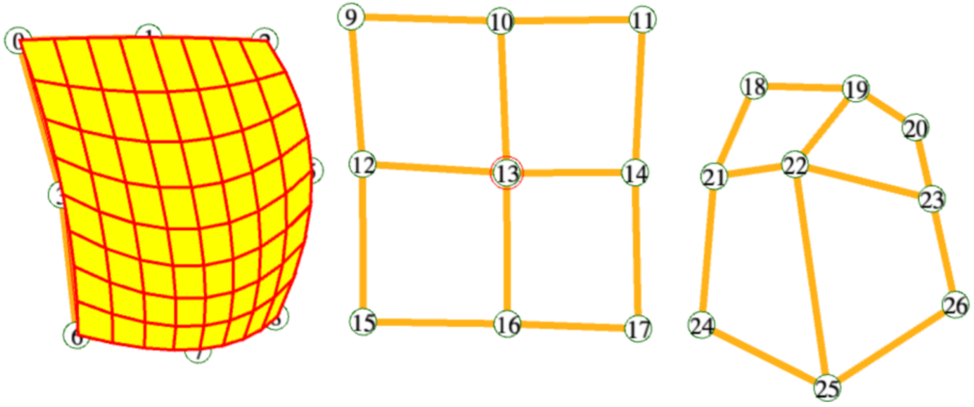
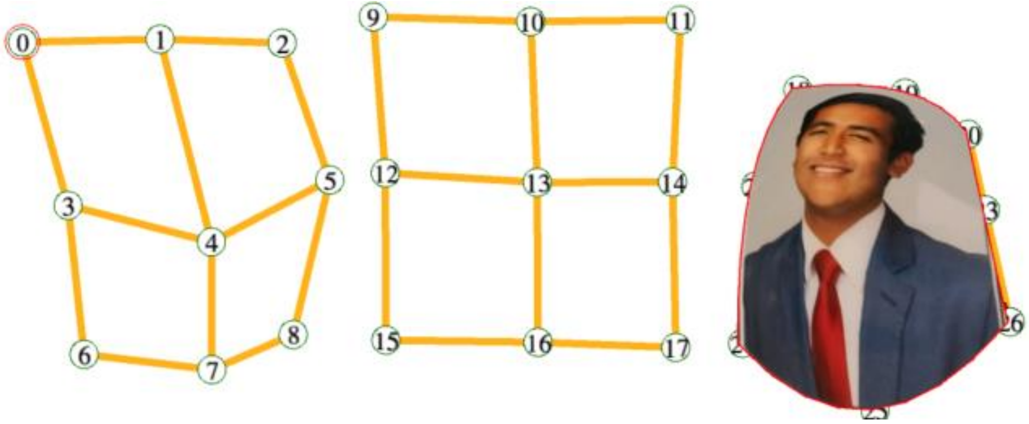
ANIMATED IMAGE WARP AND MORPH





CS3451 FALL 2020 PROJECT 3

Saumya JAIN

Phase A: PROBLEM = Tri-Quadratic Warp Animation

Given an array of control points, define a continuous and smooth point-valued function $F_c (t , u , v)$ that interpolates 27 constraints: $F_c (i/2 , j/2 , k/2) = C[i][j][k]$ with $i, j, k \in \{0,1,2\}$. Implement this using parabolic interpolants. Define the mapping of (u, v) onto the control grid to track the mouse movement over time.

COMMENTS:

In order to accurately implement F_c using parabolic interpolants, we assume the following:

i. PNT C[3][3][3] is the array of control points.

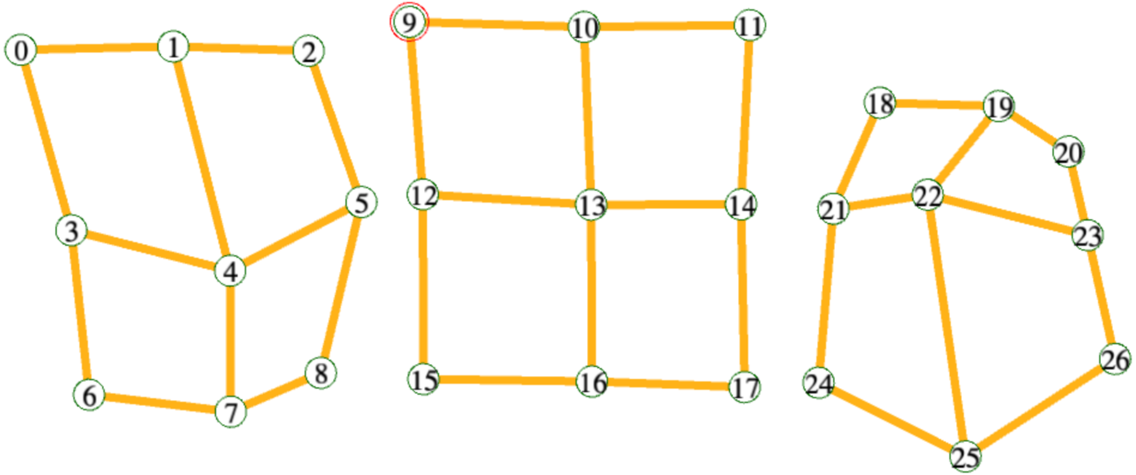
ii. The knots for the parabolic interpolants are 0, ½, and 1

Here, there is no ambiguity.

PHSE 1: Solution outline

We can compute the required function and mapping using interpolation.

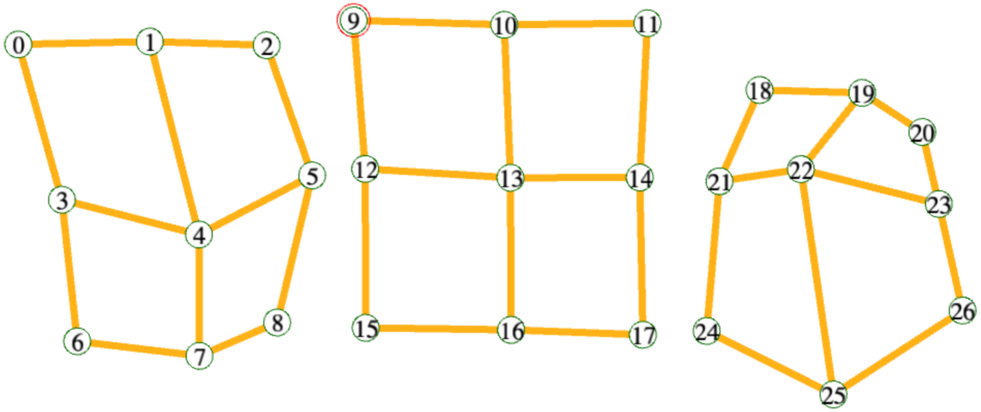
- In the 2-D control grid, for each point in the first subarray (C[0]), interpolate between the corresponding points in the second and third subarray (C[1] and C[2])
- If given a function L that linearly interpolates between two points, we can use the quadratic Neville interpolation L(0, L(0, C[0][j][k], 0.5, C[1][j][k], t), 1, L(0.5, C[1][j][k], 1, C[2][j][k], t), t)



PHSE 1: Solution outline

- Now, given the normalized values of (x, u) and (y, v), we need to interpolate between C[i][j][k] for k = 0, 1, 2 with time u. After this, we again need to interpolate between the resulting points and time v. This is essentially the mapping of (u, v) onto the control grid.

- We can calculate the needed points like so:
  - A = L(0, C[0][0][0], 0.5, C[0][0][1], 1, C[0][0][2], u)
  - B = L(0, C[0][1][0], 0.5, C[0][1][1], 1, C[0][1][2], u)
  - C = L(0, C[0][2][0], 0.5, C[0][2][1], 1, C[0][2][2], u)
  - X = L(0, A, 0.5, B, 1, C, v)

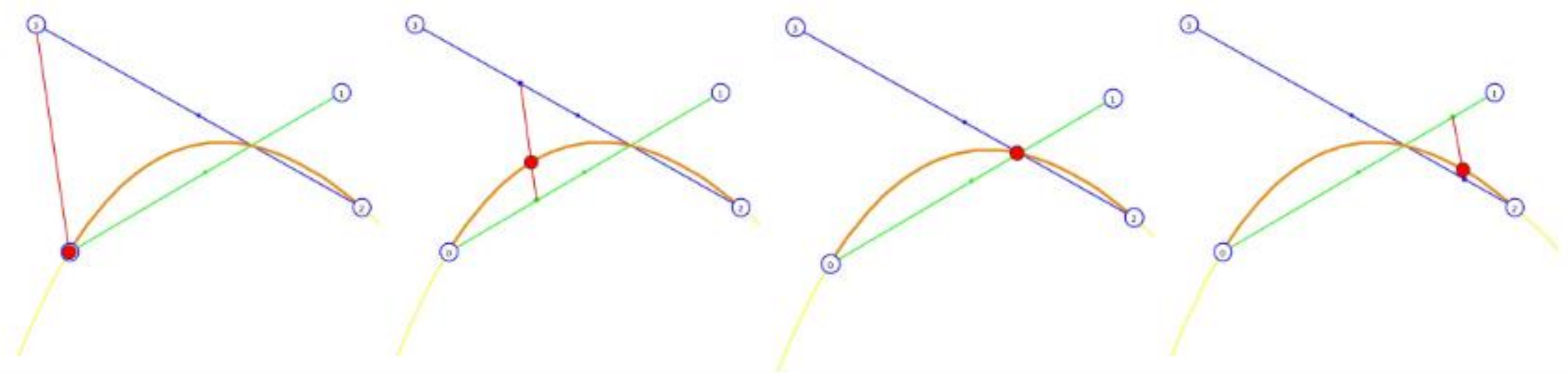


Phase A: Solution math

We can compute if the created interpolation is the required parabolic interpolation between points A, B, and C.

- The interpolation is given by L(a, L(a, A, b, B, T), c, L(b, B, c, C, t), t) in its general form, where a, b, and c are knots.
- Let’s say that by linearly interpolating between the points A and B, we create point M. Then, interpolating between B and C, we create point N. This results in two linear paths.
- So, by linearly interpolating between M and N using the same variable of time, we get a second degree function as a result.
- This is the required parabola.

Phase A: Solution math



Phase A: Solution math

We can compute if the interpolations calculated result in the mapping of (u, v) onto the control grid.

- The series of interpolations are:
  - A = L(0, C[0][0][0], 0.5, C[0][0][1], 1, C[0][0][2], u)
  - B = L(0, C[0][1][0], 0.5, C[0][1][1], 1, C[0][1][2], u)
  - C = L(0, C[0][2][0], 0.5, C[0][2][1], 1, C[0][2][2], u)
  - X = L(0, A, 0.5, B, 1, C, v)

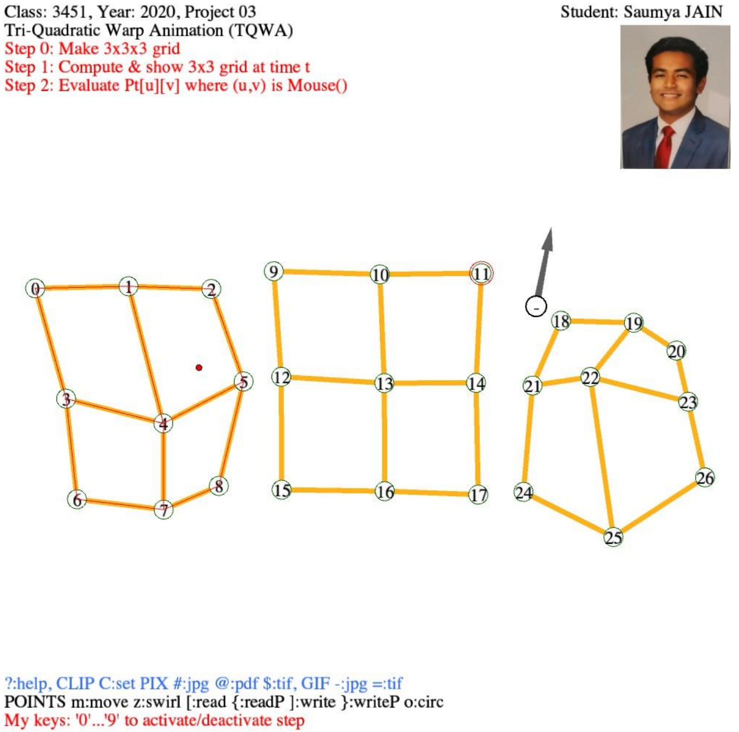
- Here, each of A, B and C is on the horizontal edge of the control grid. The knots here are 0, 0.5 and 1 respectively. This means that the interpolation scales uniformly across A, B, C (between 0 and 1).

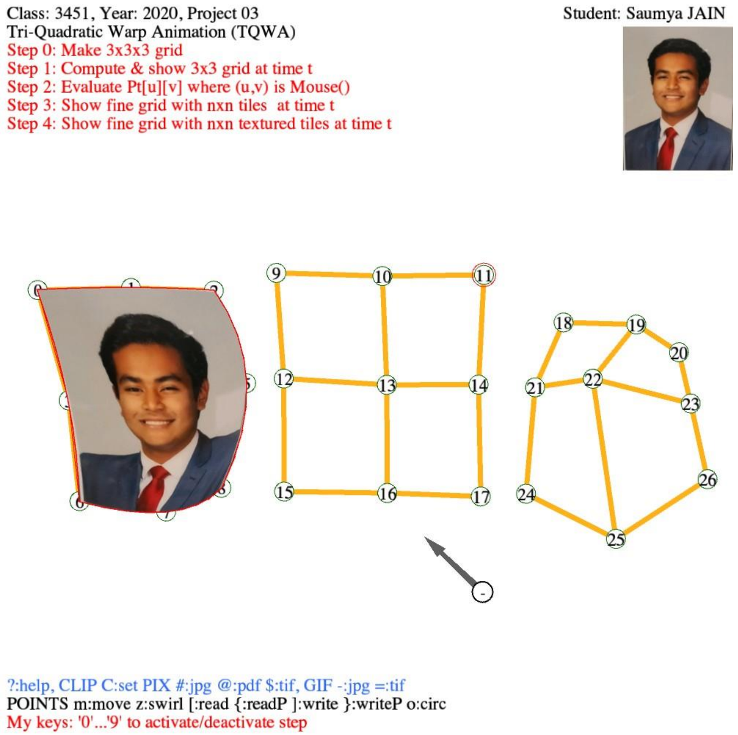
- We are given the relative position of u on each edge when we interpolate up to time u across the horizontal edges. This is because u is the normalized value of x for a certain position.

- So, when we interpolate across A, B, and C, we get the warped y-axis by x = u on the grid.

- Hence, interpolating this to a time v across A, B, and C is the mapping of (u, v) onto the grid.

Phase A: Results and limitations of your implementation





DISCLAIMER, LIMITATIONS:

This solution always works.

Phase A: Code and whole screen shot of result

```
//========================================== PART 1
PNT[][] Pt = new PNT [3][3];
void doStep1(PNTS R) //
  {
  titleOfStep[1] = "Compute & show 3x3 grid at time t";
  // compute 3x3 control grid at currentTime
   for (int i = 0; i < 3; i++) {
     for (int j = 0; j < 3; j++) {
       Pt[i][j] = L(0, P[0][i][j], 0.5, P[1][i][j], 1, P[2][i][j], currentTime);
     }
   }
   show3x3grid(Pt);

  guide="My keys: '0'...'9' to activate/deactivate step, 'a' to animate";
  }
```

Phase A: Code and whole screen shot of result

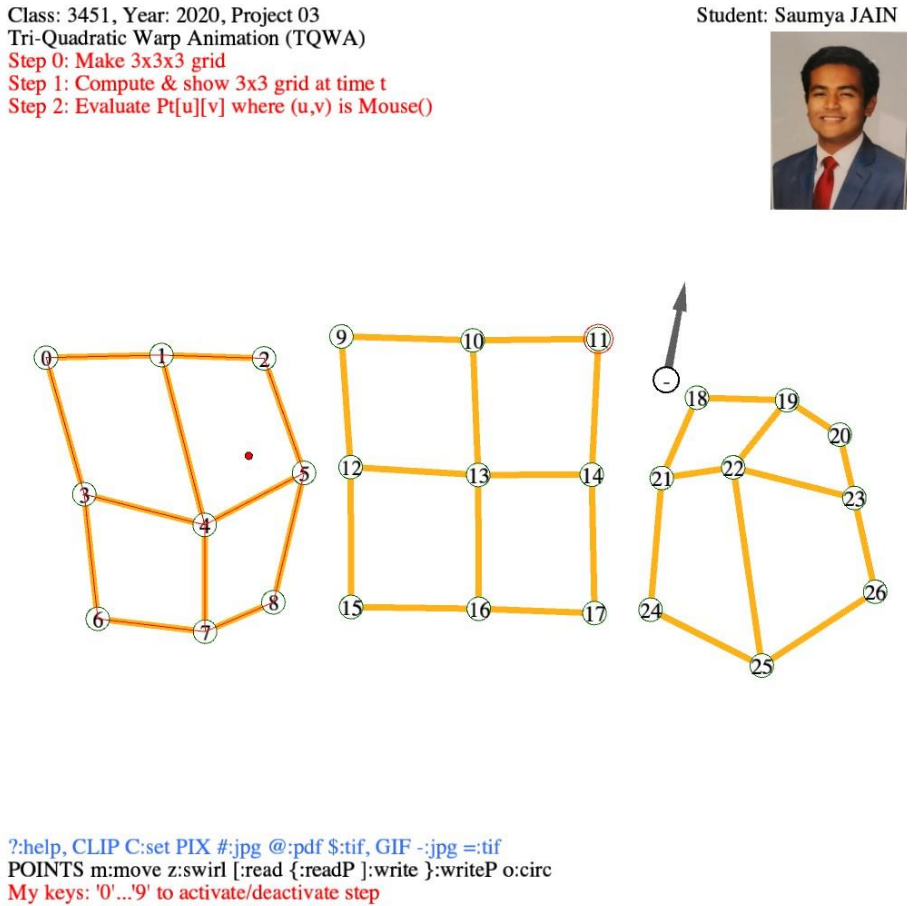
```
//=========================================== PART 2
void doStep2(PNTS MySites) //
  {
  titleOfStep[2] = "Evaluate Pt[u][v] where (u,v) is Mouse()";
  PNT M= Mouse();
  PNT X = Evaluate(M.x/width,M.y/height,Pt);
  cwf(black,1,dred); show(X,4);
  guide="My keys: '0'...'9' to activate/deactivate step";
  }
PNT Evaluate(float u, float v, PNT Q[][])
  {
    PNT A = L(0, Q[0][0], 0.5, Q[0][1], 1, Q[0][2], u);
    PNT B = L(0, Q[1][0], 0.5, Q[1][1], 1, Q[1][2], u);
    PNT C = L(0, Q[2][0], 0.5, Q[2][1], 1, Q[2][2], u);

    PNT X = L(0, A, 0.5, B, 1, C, v);
    return X;
    //return P(); // replace this!
  }

```

Phase A: Code and whole screen shot of result

The code from the previous slides outputs the following (Grid + Mouse location) :

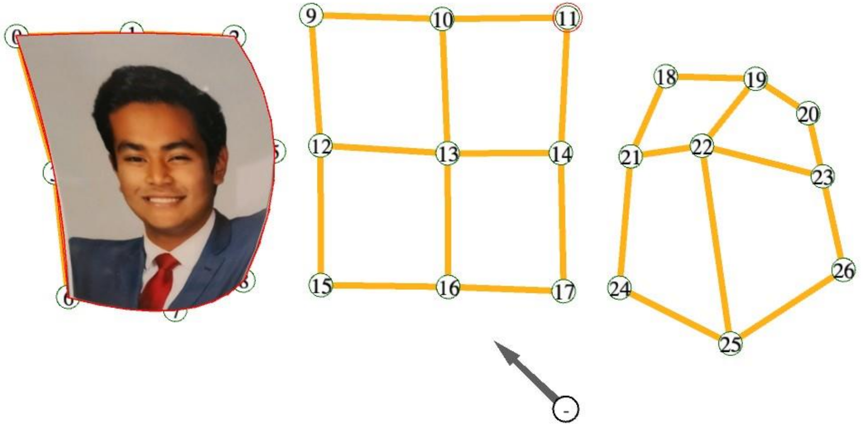


Phase A: Code and whole screen shot of result

We can further enable Steps 3 and 4 for an animated face warp representation

Class: 3451, Year: 2020, Project 03
Tri-Quadratic Warp Animation (TQWA)
Step 0: Make 3x3x3 grid
Step 1: Compute & show 3x3 grid at time t
Step 2: Evaluate Pt[u][v] where (u,v) is Mouse()
Step 3: Show fine grid with nxn tiles at time t
Step 4: Show fine grid with nxn textured tiles at time t

Student: Saumya JAIN

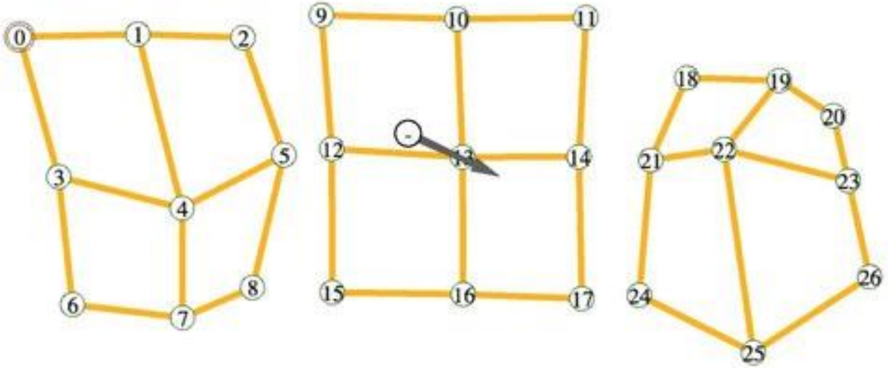


?:help, CLIP C:set PIX #:jpg @:pdf $:tif, GIF -:jpg =:tif
POINTS m:move z:swirl [:read {:readP]:write }:writeP o:circ
My keys: '0'...'9' to activate/deactivate step

Phase A: GIF

Class: 3451, Year: 2020, Project 03
Tri-Quadratic Warp Animation (TQWA)
Step 0: Make 3x3x3 grid

Student: Saumya JAIN



?:help, CLIP C:set PIX #:jpg @:pdf $:tif, GIF -:jpg =:tif
POINTS m:move z:swirl [:read {:readP]:write }:writeP o:circ
My keys: '0'...'9' to activate/deactivate step, 'I' to hide/show Ids

Phase A: Sources

I had no preexisting knowledge on parabolic interpolations.

I learned about the topic from:

- http://fourier.eng.hmc.edu/e176/lectures/NM/node25.html
- https://en.wikipedia.org/wiki/Successive_parabolic_interpolation
- https://www.youtube.com/watch?v=noczK51tOgE
- Lecture