CS4803/8803CAe-2017—Project 3: Animation

Dibyendu Mondal **Sarang Joshi**

ABSTRACT

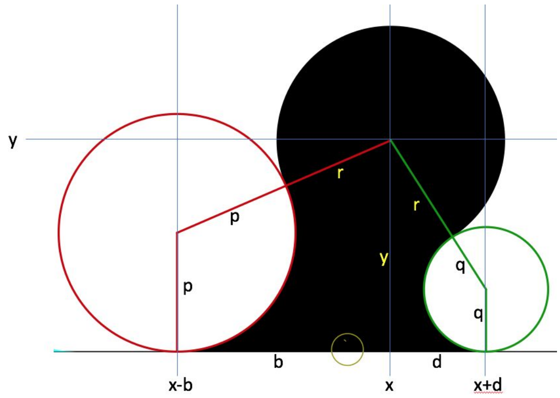
The first goal of this project is to create and implement principles and algorithms that create smooth, interpolating, expressive, and aesthetically pleasing animations of a simple and parameterized, smooth and planar shape that is bounded by three circular arcs. We present (1) the shape parameterization, (2) the math for computing automatically its blending arcs, (3) the broad principles for creating animations that are aesthetically pleasing, (4) the broad principles for creating animations that express specific emotions, (5) noteworthy details of our implementation, (6) experimental results, (7) strengths and limitations of our solution, and (8) suggestions for experimental validations and for further experimental research. We further present a UI which allows the user to create and edit keyframes, and we can create an animation by interpolating between them.

1 Shape parameterization

The shape, S, is defined as the union of a "**head**" represented by a disk with center (x,y) and radius r and a "**neck**" that smoothly connects the head to the floor and is hence delimited by two "**blends**" (circular arcs), each tangent to the head perimeter and to the floor. The blending arcs are tangent to the floor at contact points with coordinates x-b and x+d.

In the typical configuration (below), S is the figure shaded in black. It is bounded by the "**base**" (a straight horizontal edge that slides on the floor) and by three smoothly connecting circular arcs (one for the head and two for the neck).

The shape parameters that are set, at each frame of the animation, by the animation controller are: x, y, r, b, and d. Intuitively, x and y control the head position, and r controls its size. Parameters b and d control the possibly asymmetric thickness of the neck at the base and hence may be used to convey the impression that the neck and head are tilted.



The blend radius parameters, p and q, are computed automatically, as explained below.

2 Computing the blending arcs

The centers of the blending circles (shown in red for left and green for right) have coordinates (x-b,p) and (x+d,q).

We provide below the derivation of the expression for the value of p from values of b, y and r.

Let the center of the head be A, center of left blending circle be B.

Let's consider the triangle formed by A, B and point at (x,p) (let's say this is C). Notice that this triangle is a right angled triangle. Applying Pythagoras theorem,

$AC^2 + BC^2 = AB^2$

$\Rightarrow (y-p)^2 + b^2 = (p+r)^2$

$\Rightarrow y^2 - 2py + b^2 = r^2 + 2pr$

Hence $p = (y^2 + b^2 - r^2)/(2*(y+r))$

We implemented this computation by a call to

```
p=blendRadius(b,y,r);
```

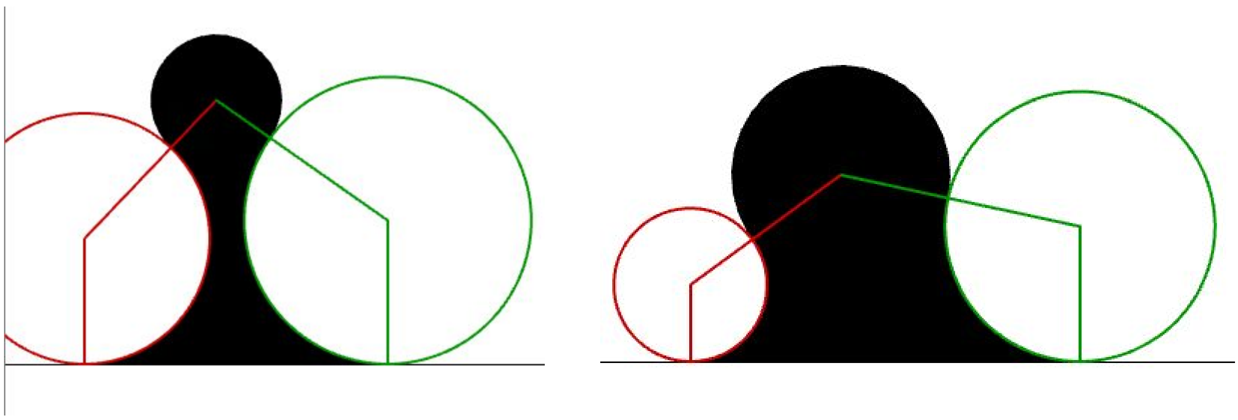
which we have implemented as follows :

```
float blendRadius(float b, float y, float r)  { return  (y*y + b*b - r*r)/(2*(y+r));  }
```

By symmetry, parameter q is computed using

```
p=blendRadius(d,y,r);
```

Results are shown below. It can be seen that the transition between the three circles at the "neck" region is smooth.



3 Principles for creating aesthetic animations

[3] mentions some of the principles that are usually considered in creating aesthetic animations, such as: motion being *smooth*, *squashing and stretching* an object based on its acceleration, making the user *anticipate* the next action, timing each part of the motion appropriately (i.e. having slow in and slow out motions). In section 5, we discuss how we implement some of these.

4 Principles for creating expressive animations

The authors of [1] talk about the influence of different physical characteristics of body movements on the perception of emotions. Studies employing static pictures show that emotion recognition is influenced by body posture. They mention some of the important posture features as **head inclination**, which is typical for sadness, or elbow flexion, which observers associate with the expression of anger. Further, they mention that the perception of emotions from body expressions is influenced by **kinematics**. Typically, velocity, acceleration and jerk have been considered as interesting parameters, and all three have been shown to affect emotional classifications of expressive movements. Based on this, we propose the following principles for creating expressive motions:

- Jumping up and down to show *excitement*
- Head bent down to show *depression*
- Slow motion with the head moving back and forth to signify *laziness*.
- Back and forth motion to signify shuffling or dragging the feet, also signifying laziness.

5 Noteworthy details of our implementation

We first provide below a high level description of our approach. Then, we discuss several noteworthy details, explaining what they accomplish, showing we implemented them, and providing the mathematical derivation of the formulae used.

Outline:

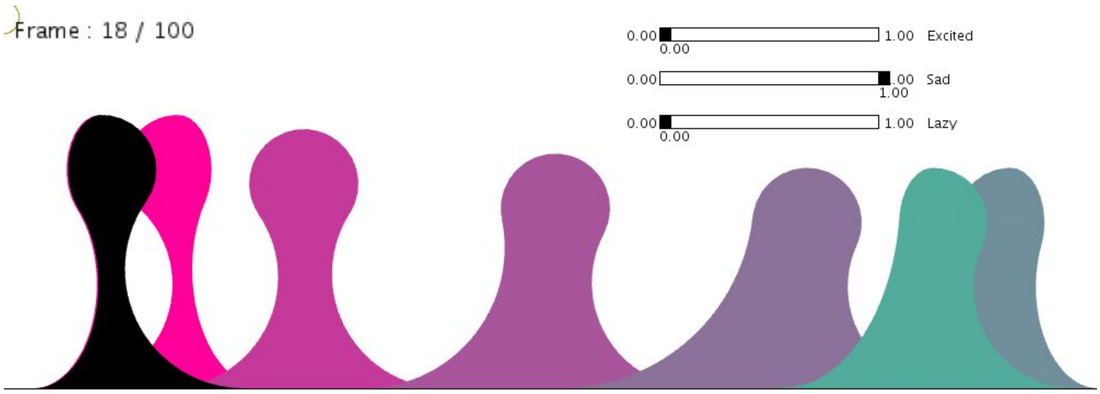
We are given the initial and final shapes (keyframes), at t=0 and at t=1.Their x-coordinates are given by values x_0 and x_1 implying that $x(0) = x_0$ and $x(1) = x_1$. We assume that the shape is stationary at the beginning (t=0) and at the end (t=1) of the animation. By "stationary", we mean that all derivatives of all control parameters are 0 at these times. Hence, the shape will accelerate at the beginning and decelerate at the end.

We break our time interval into three chunks: an initial phase where the object accelerates from rest, the middle phase where it performs the main action, and the final phase where it decelerates. For the initial and final phases, we use some of the principles for creating aesthetic animations mentioned above. In the middle phase, we try to express our emotion. This is done by modelling the animation corresponding to that emotion parametrically, as a function of *t*. We ensure continuity at the boundaries

Details:

Generating Aesthetic Emotions:

Squash and Stretch: We try to mimic the squashing and stretching effect in [3] by scaling our shape along x, so that the circles become ellipses. This is done by taking each point on our shape boundary, finding its distance from the axis of symmetry and then scaling that distance. We further add different values of scaling along the left and the right side, so that one side is squashed and the other is stretched. Further, the squashing and stretching magnitude is proportional to the velocity, which means the faster the object is, the more it deforms. This is only done in the initial and final phases of the motion, when it is accelerating or decelerating.



Anticipation: We start the animation by backing off a little from the starting position, in the direction opposite to the intended motion, and end our motion by moving beyond our target and coming back. This makes the motion look more aesthetically pleasing.

This is done by using a *quadratic Neville interpolation* between t=0 and t=0.25, which starts and ends at the starting point, and its third point is in the direction opposite to the motion. We then execute our emotion between the start and end point, making sure it reaches the endpoint at t=0.75, and then we overshoot in a manner symmetric to our initial backing off.

Generating Expressive Emotions:

Excitement: We model the locus of the center of the head as a smooth curve which is basically a spiral with a constant radius. The mathematical formulae for this are as follows:

The radius of the spiral has an x coordinate which varies linearly with time and a constant y coordinate.

cx = x0 + t1*(x1-x0);

cy = y0 - r0;

The c and y coordinates of the curve vary as follows:

x = cx + r0*sin(8*PI*t1);

y = cy + r0*cos(8*PI*t1);

Sadness: We model the locus of the center of the head as a smooth curve in which we give an effect of bending to show depression, the x coordinate varies linearly and the y coordinate in a sin curve. The mathematical formulae is as follows:

x = x0 + t1*(x1-x0);

y = y0 - r0*sin(PI*t1/2);

b = b0 + 1.5*t1*b0;

Laziness: We model the locus of the center of the head as a spiral with constant radius which is almost similar to the excited animation but we change the amplitude or radius of the spiral and also decrease the time increment to show slow movement. The mathematical formulae for this is as follows:

cx = x0 + t1*(x1-x0);

cy = y0 - r0/2;

x = cx + r0*sin(10*PI*t1);

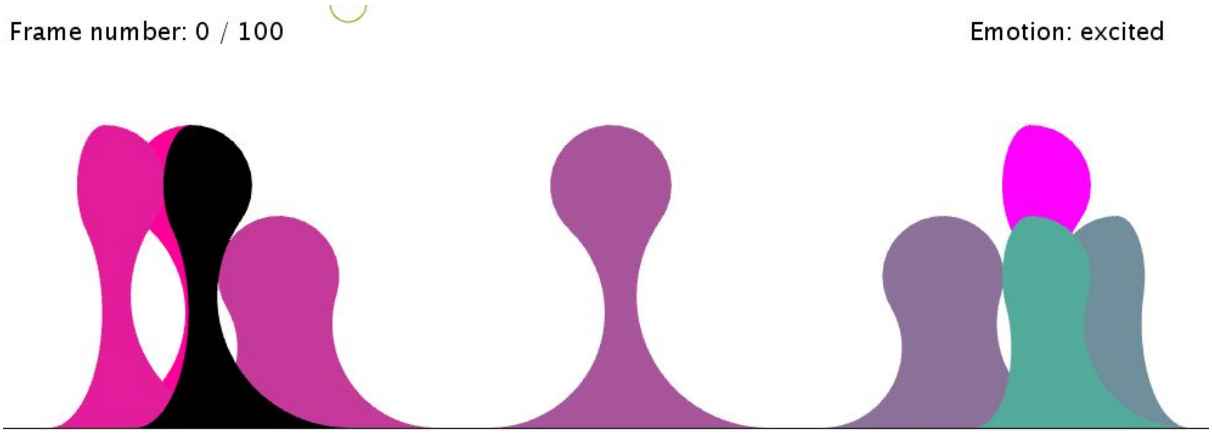
y = cy + r0/2*cos(10*PI*t1);

where cx and cy are the coordinates of the center of the spiral and x and y are the coordinates of the spiral.

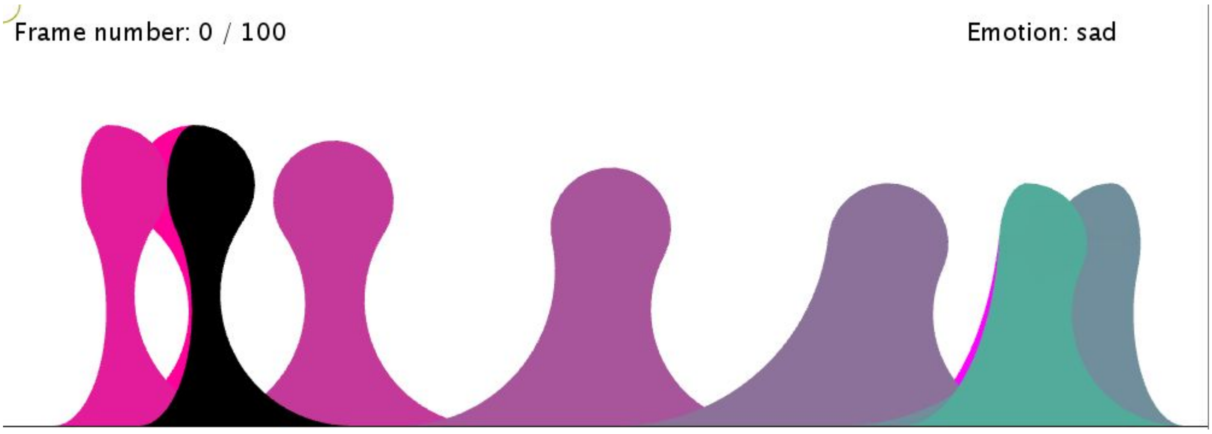
6 Experimental results

We show results for several emotional targets using a stroboscopic superposition of consecutive frames. For each, we comment on how the principles described above have been used to enhance aesthetics and emotion. We encourage the reader to check the corresponding short video segments included with the project.

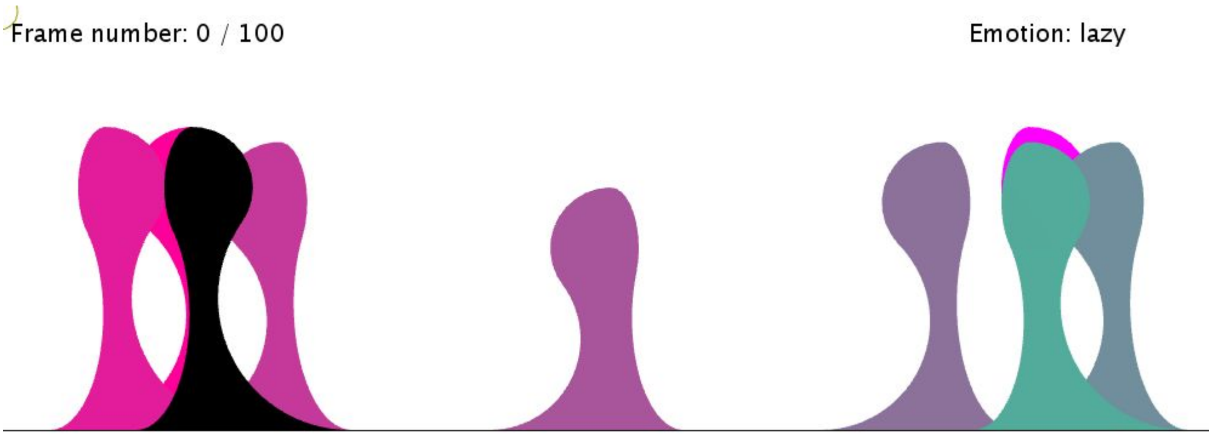
Excited:



Sad:



Lazy:



7 Strengths and limitations

From our experimental results, we conclude that the strengths of our approach include the following:

- There is no need to specify keyframes manually.
- The user can model his own emotions parametrically, and then test out the animation.

As key drawbacks, we observed the following:

- It may not always be trivial to come up with a parametric equation describing even a simple motion. Also, parameter tuning may be involved.
- Our quashing and stretching only works correctly when the figure is symmetric about the Y-axis.

8 Suggestions for further research

Our main drawback is that manually defining the entire motion is not visually intuitive. To improve upon that, we have explored two possible extensions:

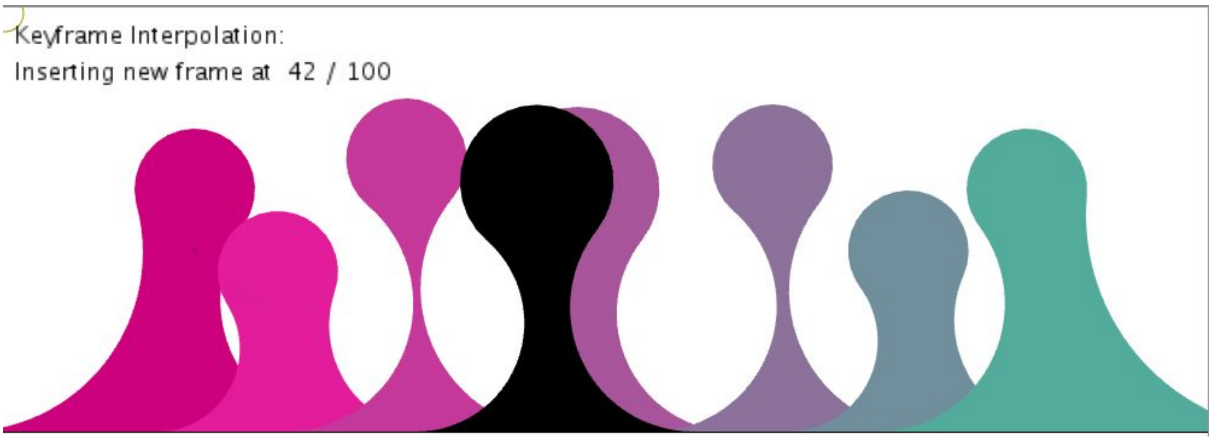
(i) allowing the user to define keyframes and interpolating between them, and

(ii) creating new emotions by mixing or interpolating between the three existing ones.

These are described in the section below.

9 Extension for graduate students

Keyframe Design: To allow the user to define motions visually, we have made an interface where they can define and edit keyframes. For this, we define a new class called keyframe which stores the parameters x, y, d, b, r corresponding to our shape, and the *frame number* at which this frame should appear. We start with two keyframes -- the start and end frame -- and allow the user to insert new keyframes based on the mouse position. *When the user presses the key i, we insert a keyframe at that mouse position. The user can also change the values of x, y, r, d and b by pressing the keys x, r, d and b as was given in the base code.* The motion/animation at time *t* is then obtained by doing a LERP between the *i*-th and (*i*+1)-th keyframe, where *i* is the index of the last keyframe which appears before time *t*.

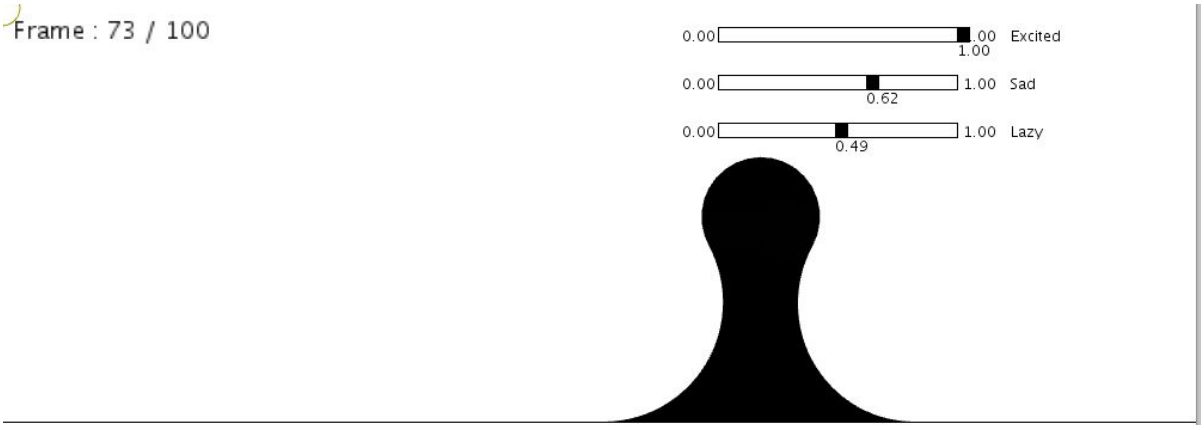


Interpolating Emotions: This is done by assigning weights to each of the three emotions, whose values can be controlled by the user with the help of the sliders in the UI. The figure below shows one such example. We then compute each parameter for our interpolated emotion as follows

$$x = \frac{x_1 w_1 + x_2 w_2 + x_3 w_3}{w_1 + w_2 + w_3}$$ where x_1 is the x parameter for the *i*-th emotion

w_i is the weight for the i-th emotion

We then do the same for the other parameters: y, d, b and r to essentially obtain a new emotion which is a *weighted average* of the existing ones.



10 References cited

[1] Critical features for the perception of emotion from gait, Claire L. Roether; Lars Omlor; Andrea Christensen; Martin A. Giese

http://jov.arvojournals.org/article.aspx?articleid=2204009

[2] Decomposing biological motion: A framework for analysis and synthesis of human gait patterns, Nikolaus F. Troje

http://jov.arvojournals.org/article.aspx?articleid=2192503

[3] The Illustration of Life

https://vimeo.com/93206523