Technical Review 1

Interactive Calculus

March, Matthew, Judy

Software Design Spring 2016

Agenda

-Background/Introduction - 3 mins

-Presenting current code (in pseudo-code) - 3 mins

-Technical questions (3) - 4*3 = 12 mins

-UX ideation - 7 mins

Background/Introduction

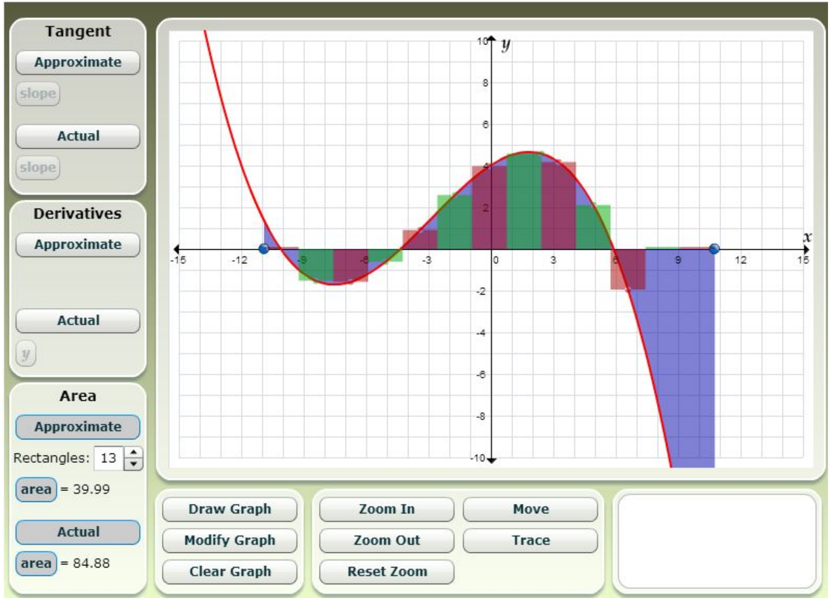
-Interactive math app

-Fundamental Theorem of Calculus

-Purely visual and geometric

-Eliminates equations

-Allows the user to build a geometrically intuitive understanding of the theorem



Rectangles: 13

area = 39.99

area = 84.88

Already created example - but lacks purely geometric functionality

Pseudo-Code

Curve:
self.points is the raw data given by view
Self.smooth is the interpolated points using self.points
Self.derivative is, well, the derivative of the smooth points
Self.integral is pretty self explanatory

self.update() uses self.points and generates the other attributes.
self.smoothen() is the method that smooths the points from self.points to self.smooth.
Self.move_point moves a specified point from self.point and moves it an appropriate distance, while also moving the points around it.

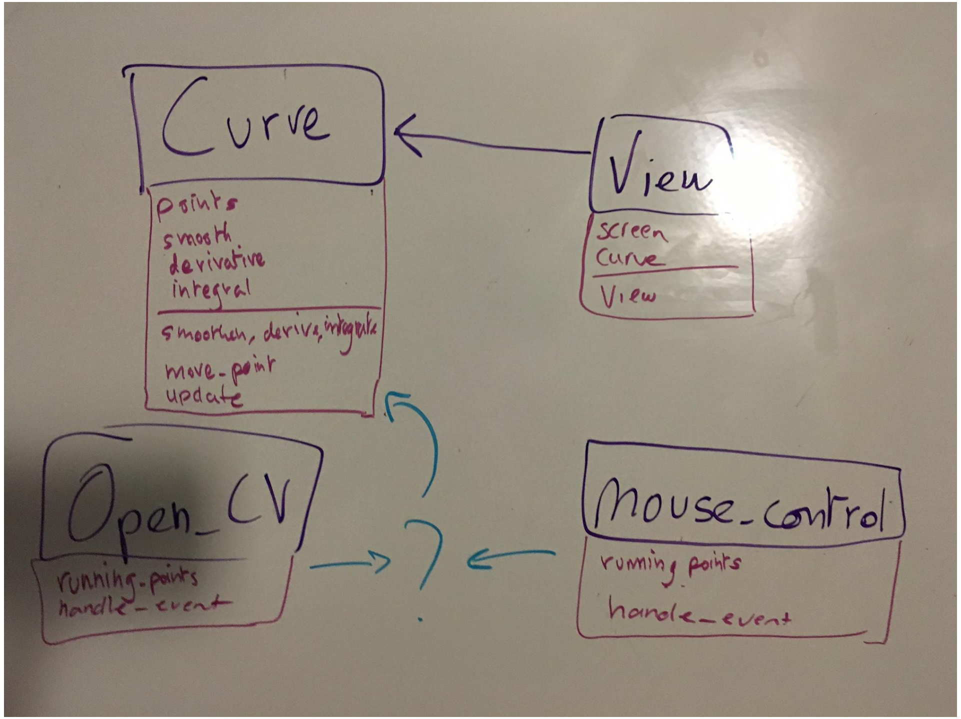
Mouse_control:
Currently, its main function is to record a new curve when the user clicks to draw.

Open CV control:
It’s main function will be to record a new curve when the user clicks to draw

View:
Uses a backend to display Matplotlib with pygame.

Technical question- code architecture

- Suggestions on how the “flow of information” should go.
- Information about the mouse position goes into a controller’s update method which calls appropriate update methods for curve. Thoughts?



Technical question - openCV

How to use OpenCV to trace a marker and convert into position of points?

Current code:

```
# Define ’BGR’ values range of color
lower_red = np.array([0, 0, 120])
upper_red = np.array([80, 80, 255])

# use cv2.inRange to find pixels where BGR is in range-returns ndarray

# use np.where to get the indices of the pixels in color range. ---A LOT OF POINTS.. WHICH ONE TO USE FOR POSITION of marker?

# If detects red marker, find the middle point in the red pixel list and add that to running_points.
```

- Uses cv2.inRange() to detect a certain color.
    - Disadvantages: color range can be easily disturbed by similar colors (skin color, clothing, etc..) and background light
- Some other object detection method using OpenCV? (potentially a pattern?)

Technical question - points

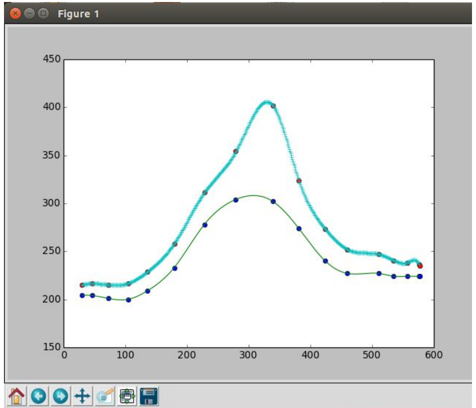
Absolute: Make the distance incremented proportional to the index away

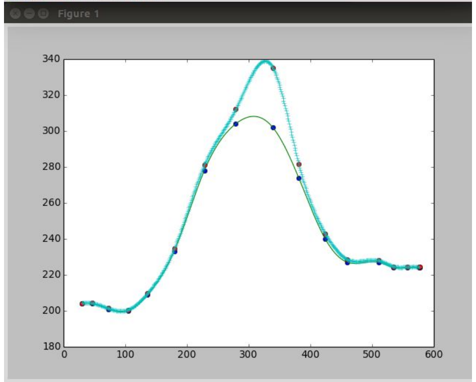
Relative: Same, but instead of adding the distance, make it a percent increase.

Q:

- Thoughts on which is better?
- Better idea to keep the curve’s feel?
- Move raw data points or smoothed points?

Absolute

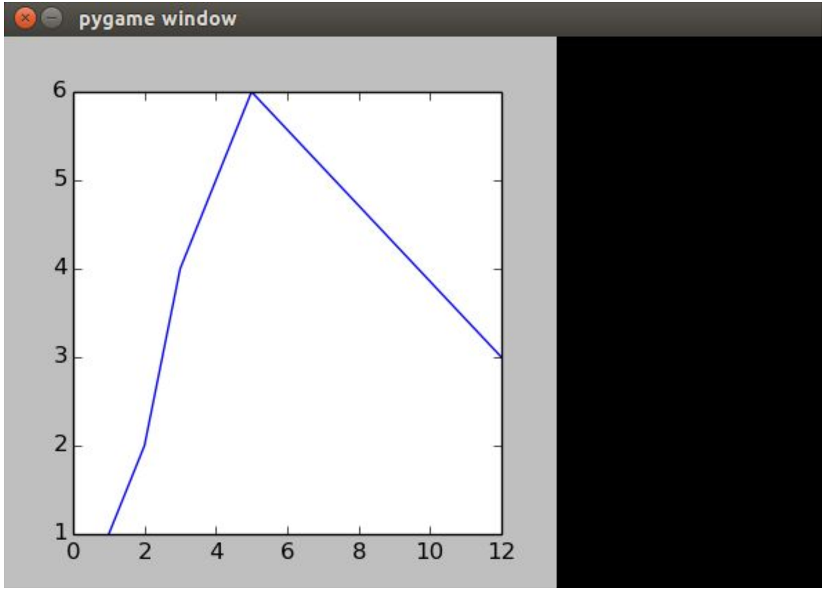




Relative

UX Questions

-What kind of interaction would others find useful/good?
-Is there a main point or functionality that would be helpful to add?
-How would you like to interact with the UX?



Current UX - could definitely be prettier!