Report on Go.Js and P5.js utility

Nonlinear Learning Team

Introduction:

One of the main focuses of our non-linear learning project was to incorporate mind maps for nonlinear learning, and use p5.js for videos with interactive elements. We spent over 2 months exploring the different aspects of the Go.Js library (mind map) and the p5.js library (interactive videos), and this report will highlight our findings and opinions.

Incorporating Go.JS for Mind Maps

We found the Go.Js library to be very useful in our implementation of a mind map. The mind map incorporates JSON (JavaScript Object Notation) objects, which is easily user-changeable and easy for machines to parse/generate. JSON allows for each node of the mind map to be stored and exhibit variables, which can be interchangeable

go.TreeModel:

This object was the backbone of our mind map. Allows for introduction of elements that define a node, such as color, text, title, text size, link, link color, parent node, child node etc. Saved in JSON format.

Node objects in Go.JS Library:

The node objects had several useful functions such as layoutTree(), which flattens the diagram and aligns each node to its parent. dir() determines each node's direction on the tree, which is useful for organization. Nodes are part of the diagram class, which contains other useful elements such as TextBlocks, which are used for right click functions on each node. These functions include the ability to create links for each node, which we used to link our node to the videos. Other functions include the ability to change font size and bold the text, which is part of the adorn class.

Other useful tools in Go.Js:

There are many layout examples on Go.Js, which are not limited to just Tree Diagrams (which our mindmap is based off of). There are also flowcharts, pyramid charts, and various data visualization elements that could be useful for other projects.

P5.js and Interactive Videos

P5.js is a javascript library that builds off of processing, and allows for canvassing on html, and contains many of the same functions that processing has. We decided to make use of the video canvassing option for P5.js, part of the DOM class. The DOM class is very extensive, and contains many useful elements such as buttons, event listeners, and video players. One of the main ideas was to use p5.js to create a duplicate image of the video player, and then hide the video player so we can freely manipulate the canvas.

DOM Class Buttons:

For our video overlay, we used the DOM button function, which creates various buttons on the video canvas so we can create user interactivity. We used the mouseY and mouseX variables which determine the position of the cursor so we can determine what elements the user is trying to interact with. For example, in the video, the instructor can pull up a multiple choice problem. We can then map the points of the multiple choice answer boxes, and then prompt a message to appear when the user clicks a right (or wrong!) position on the canvas.

DOM Video Element:

We used the createVideo function to create a video element inside the canvas. The video is then hidden, through the use of the hide() function. From this point, we used the draw() function to duplicate the video through the use of the image() function. We used the p5vid.time() function to keep track of the video time, so we could assign a variable to it to have certain events (such as a pop up question) appear at certain times of the video. We also created a slider, which manipulates the time of the video. There is a showControl() option of the video element, but we hid the original video to avoid having two videos on the screen at the same time (the video canvas is simply a clone of the video player).

Conclusion:

Both go.js and p5.js are very useful tools for incorporating mind maps and nonlinear learning. For p5.js, we feel like a lot more can be done through the use of the different classes it offers. We brainstormed a lot of different ways user interactivity can be accomplished through videos, but some ideas such as using physics and boids were too flashy and not very useful for educational purposes. We decided the main usefulness of p5.js for interactivity was the radio buttons, mouse positioning functions, video canvassing, and input boxes.