Joseph G. Roberts

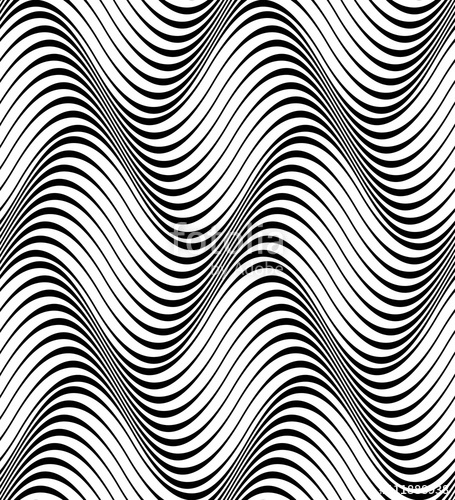
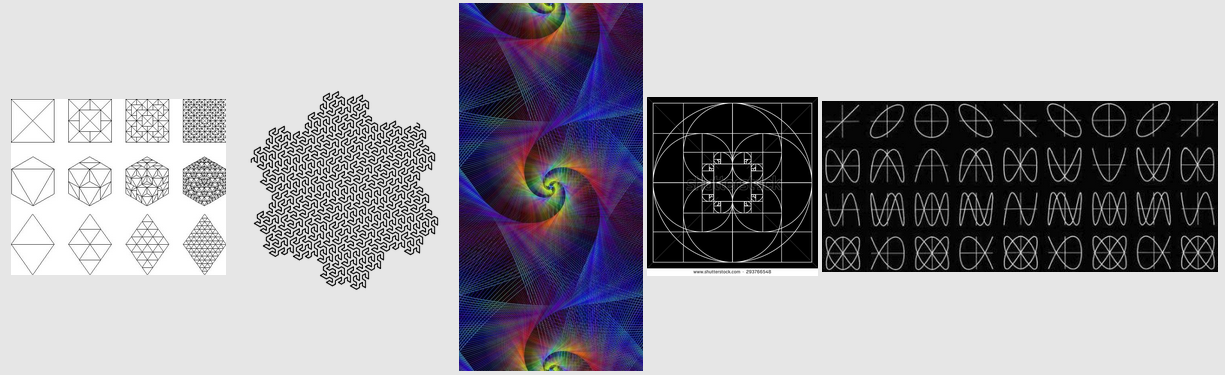
ITGM 220-OL1

Joshua Lomelino

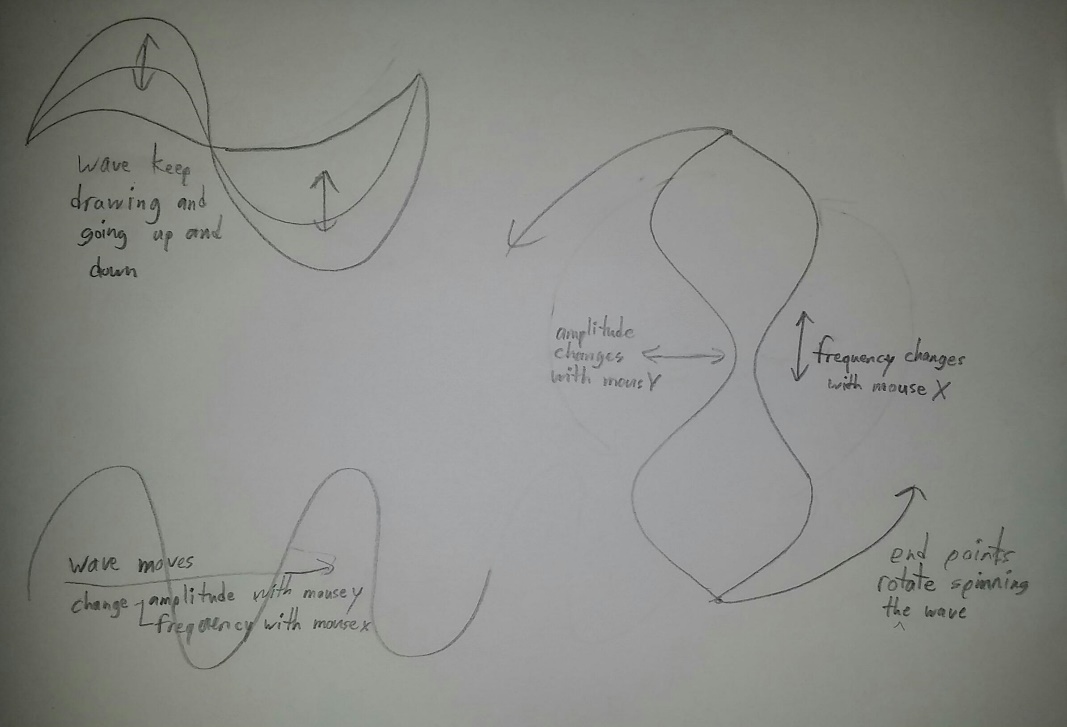
07/21/2017

Animated Patterns

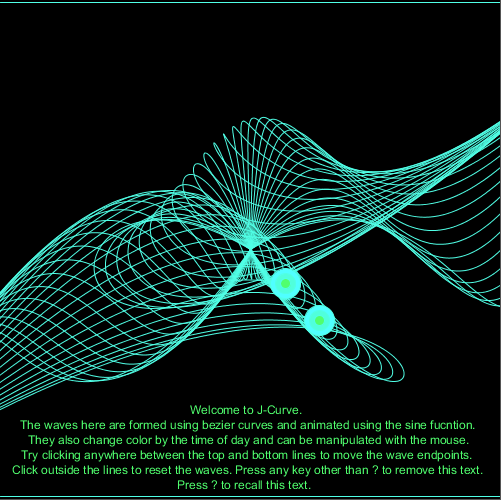
# eference Images



# Sketches



# Screen Capture



## RobertsJ\_animatedpattern

int xspacing = 16; // How far apart should each horizontal location be spaced

int w; // Width of entire wave

int a=1; // variable a

int b=2; // variable b

int n1=0; // number 1

int n2=0; // number 2

int s = second(); // Values from 0 - 59

int m = minute(); // Values from 0 - 59

int h = hour(); // Values from 0 - 23

int mouseXcor=0; // stored mouse X cordinates at time of left click

int mouseYcor=0; // stored mouse Y cordinates at time of left click

float theta = 0.0; // Start angle at 0

float theta\_vel = .5 ;

float amplitude = 100.0; // Height of wave

float period = 500.0; // How many pixels before the wave repeats

float dx; // Value for incrementing X, a function of period and xspacing

float r;

float[] yvalues; // Using an array to store height values for the wave

StringList menu;

void **setup**() {

  size(500, 500); // canvas size

  w = width+16;

  dx = (TWO\_PI / period) \* xspacing;

  yvalues = new float[w/xspacing];

  key=&apos;?&apos;;

  /\*menu= new StringList ();

  menu.append ("OPTIONS (type number to turn option on or off)");

  menu.append ("1 ROTATION-ON | 2 ROTATION-OFF");                    //code I plan on using later

  menu.append ("3 TIME-ON | 4 TIME-OFF");

  \*/

  textAlign(CENTER);

}

/\* ----------------\/ draw \/-------------------\*/

void **draw**() {

  s = second(); // updates second variable

  m = minute(); // updates minute variable

  h = hour(); // updates hour variable

  background(0); // !IMPORTANT! clears the canvas for waves to be redrawn

  calcWave();

  line(0,3,500,3);

  line(0,497,500,497);

  renderWave();

  help();

  bullseye(30,30);

}

/\* --------------\/ functions \/------------------\*/

void calcWave() {

  theta += 0.03;                                  // Increment theta (try different values for &apos;angular velocity&apos; here

  float x = theta;

  for (int i = 0; i < yvalues.length; i++) { // loop that calculates a y value with sine function for every x value

    yvalues[i] = sin(x)\*amplitude;

    x+=dx;

  }

}

void renderWave() { // function to draw (render) waves

  stroke(s\*5,m\*5,h\*10); // chages stroke color according to time of day

  noFill();

  // A simple way to draw the wave with an ellipse at each location

  if(mouseYcor<=3||mouseYcor>=497) { // loop to check for mouse click; can be reset by clicking before the top line or below the bottom line in the canvas window

    for (int x = 0; x < yvalues.length; x++) { // loop to draw wave(s) BEFORE mouse click

      beginShape(); // bezier curve aka wave (right)

        vertex(0-mouseX,250+mouseY);

        bezierVertex(x\*xspacing,height/2+yvalues[x]-250,x\*xspacing,height/2+yvalues[x]+250,250,250);

      endShape();

      beginShape(); //bezier curve aka wave (left)

        vertex(250,250);

        bezierVertex(x\*xspacing,height/2+yvalues[x]-250,x\*xspacing,height/2+yvalues[x]+250,500+mouseX,250-mouseY);

      endShape();

    }

  }

  else {

    for (int x = 0; x < yvalues.length; x++) { // loop to draw wave(s) AFTER mouse click

      beginShape(); // bezier curve aka wave (right)

        vertex(mouseX,mouseY);

        bezierVertex(x\*xspacing,height/2+yvalues[x]-250,x\*xspacing,height/2+yvalues[x]+250,mouseXcor,mouseYcor);

      endShape();

      beginShape(); // bezier curve aka wave (left)

        vertex(mouseXcor,mouseYcor);

        bezierVertex(x\*xspacing,height/2+yvalues[x],x\*xspacing,height/2+yvalues[x],mouseX,mouseY);

      endShape();

     }

  }

}

void help(){

  if (key==&apos;?&apos;||key==&apos;/&apos;){

    width=500;

    height=500;

    /\*textAlign (CENTER);

    text(menu.get(0), 250, 20);

    text(menu.get(1), 250, 35);          //code I plan on using later

    text(menu.get(2), 250, 50);

    \*/

    text("Welcome to J-Curve.", 250, 415);

    text("The waves here are formed using bezier curves and animated using the sine fucntion.", 270, 430);

    text("They also change color by the time of day and can be manipulated with the mouse.", 270, 445);

    text("Try clicking anywhere between the top and bottom lines to move the wave endpoints.", 270, 460);

    text("Click outside the lines to reset the waves. Press any key other than ? to remove this text.", 270, 475);

    text("Press ? to recall this text.", 250, 490);

  }

}

void bullseye(float w, float h){ //my custom function that draws my custom bullseye shape

  translate(width/2, height/2);

  r = height \* 0.1;

  float x1 = r \* cos(theta\*theta\_vel);

  float y1 = r \* sin(theta\*theta\_vel);

  float x2 = r / cos(theta\*theta\_vel);

  float y2 = r / sin(theta\*theta\_vel);

  if(mouseYcor<=3||mouseYcor>=497) {

    while(w>=1&&h>=1) //draws ellipses over each other to create bullseye effect

    {

      float fillRed=s\*5; //fill red value

      float fillBlue=m\*5; //fill blue value

      float fillGreen=h\*11; //fill green value

      beginShape();

        fill(fillRed,fillBlue,fillGreen); //sets fill for ellipses

        ellipse(x1,y1,w,h); //draws ellipse

        ellipse(x2,y2,w,h); //draws ellipse

      endShape();

      w-=10;                   //updates w next for next ellipse

      h-=10;                   //updates h next for next ellipse

    }

  }

}

void **mouseClicked**() { // stores (in console window only) and displays the mouse X and Y coordinates whenever the mouse is left clicked

  mouseXcor=mouseX;

  mouseYcor=mouseY;

  println("X="+mouseX+",Y="+mouseY);

}

void **keyPressed**() {

}