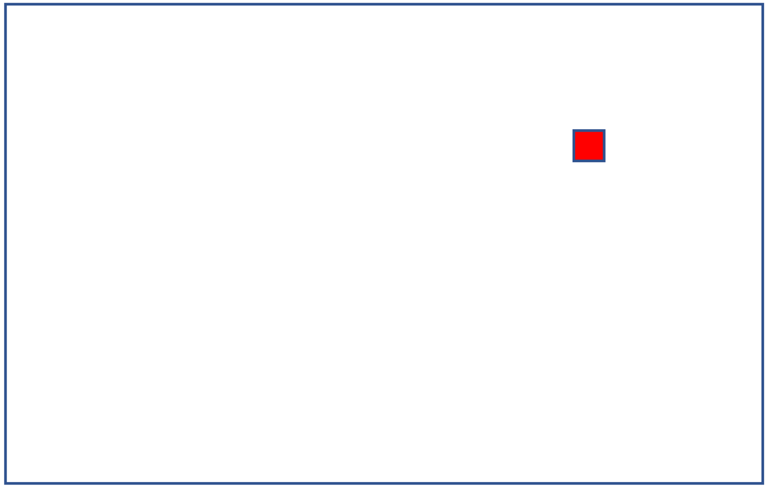
CMSC427 Drawing and parametric curves

Today's topics

- Where drawing happens: CPU vs. GPU
- Drawing a line

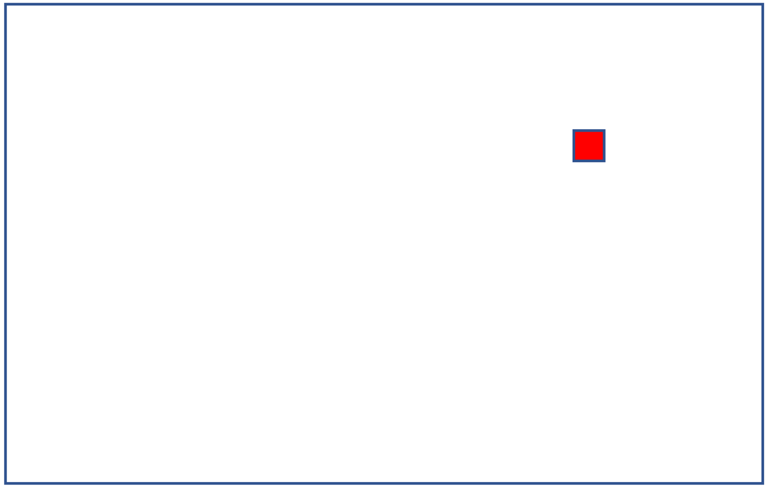
Basics of drawing: putting a pixel

PUTPIXEL(X,Y,R,G,B,A)



Basics of drawing: putting a pixel

PUTPIXEL(X,Y,R,G,B,A)



How much data to transfer?

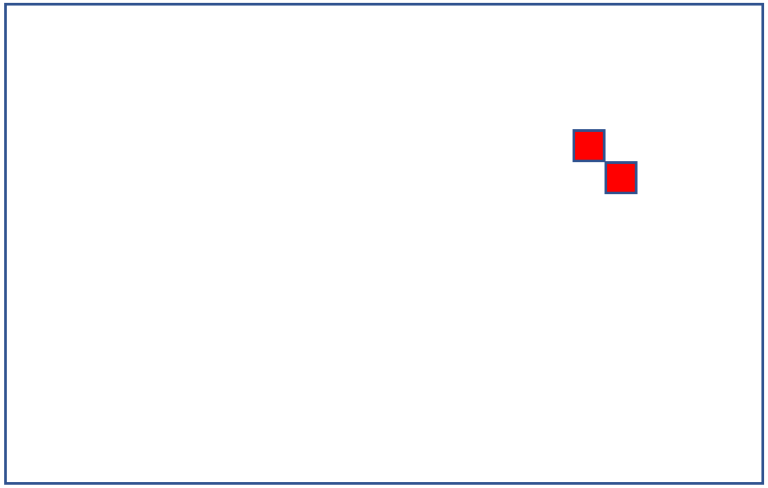
Position X,Y – 16 bits each
Color – 8 bits each

- Red
- Green
- Blue
- Alpha (transparency)

Total: 8 bytes x # of pixels

Retaining state

SETCOLOR(R,G,B,A)
PUTPIXEL(X,Y)
PUTPIXEL(X,Y)



SETCOLOR() sets state of graphics card, doesn't draw

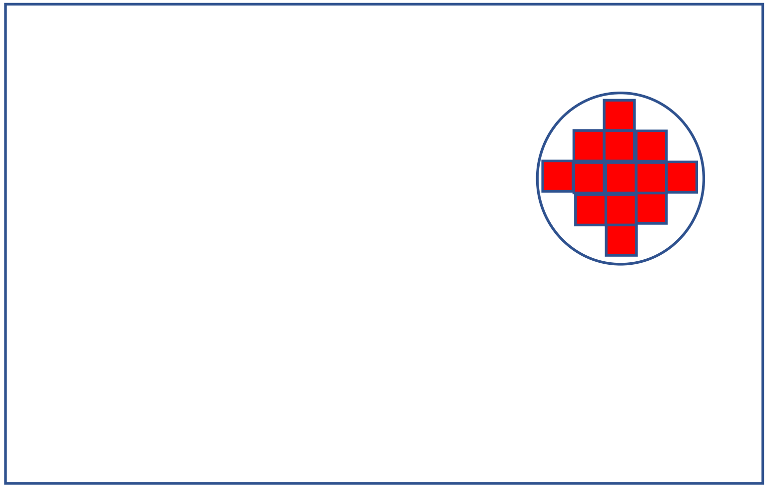
PUTPIXEL() draws

Transfer color bits less often

More we can "preload" on graphics card, less we need to do on CPU and then transfer

Delegating computation

SETCOLOR(R,G,B,A)
DRAWCIRCLE(X,Y,R)

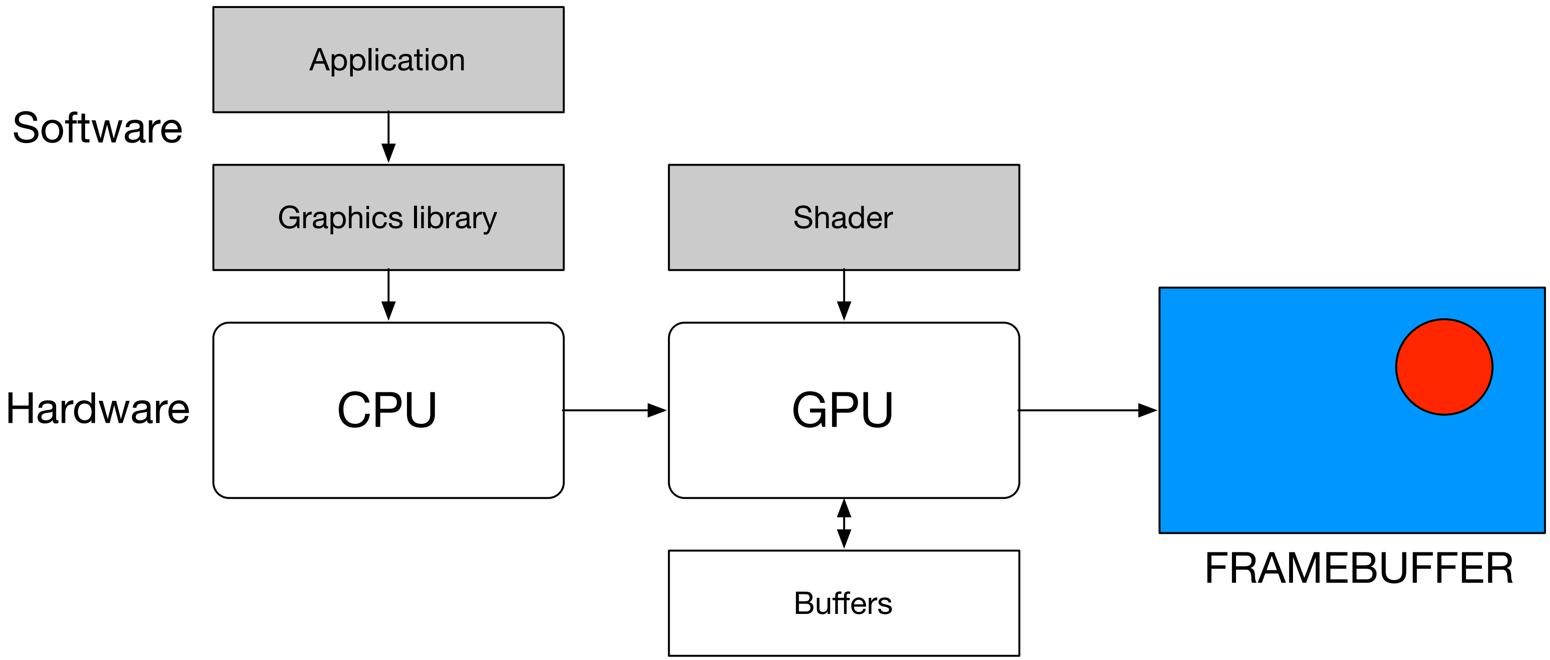


SETCOLOR() sets state of graphics card, doesn't draw

DRAWCIRCLE() draws by invoking routine on graphics card

Less data transferred. Transfer only values x,y,R but get all pixels in circle filled

Drawing on CPU vs GPU

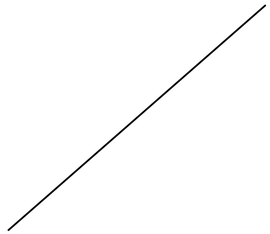


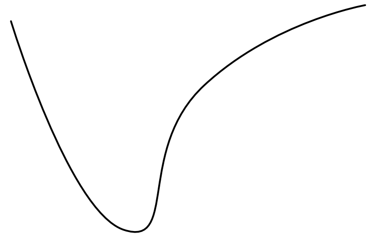
Preloading for online game

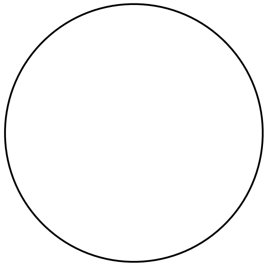
- Download to GPU in advance terrain model, character model, textures, character behaviors
- In gameplay only need to download changes in character state – movement, weapons, etc.
- Significantly reduce network bandwidth

Beyond the pixel - curves, surfaces and solids

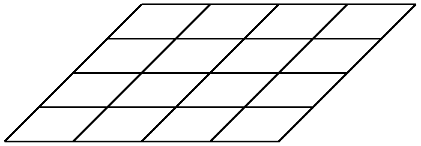
Curves

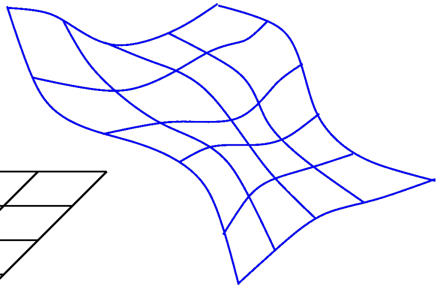


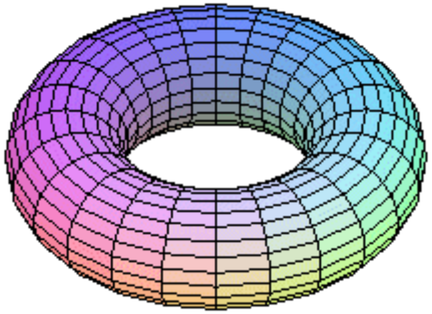




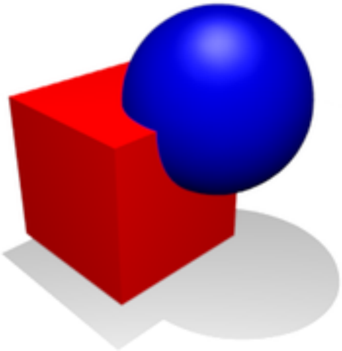
Surfaces



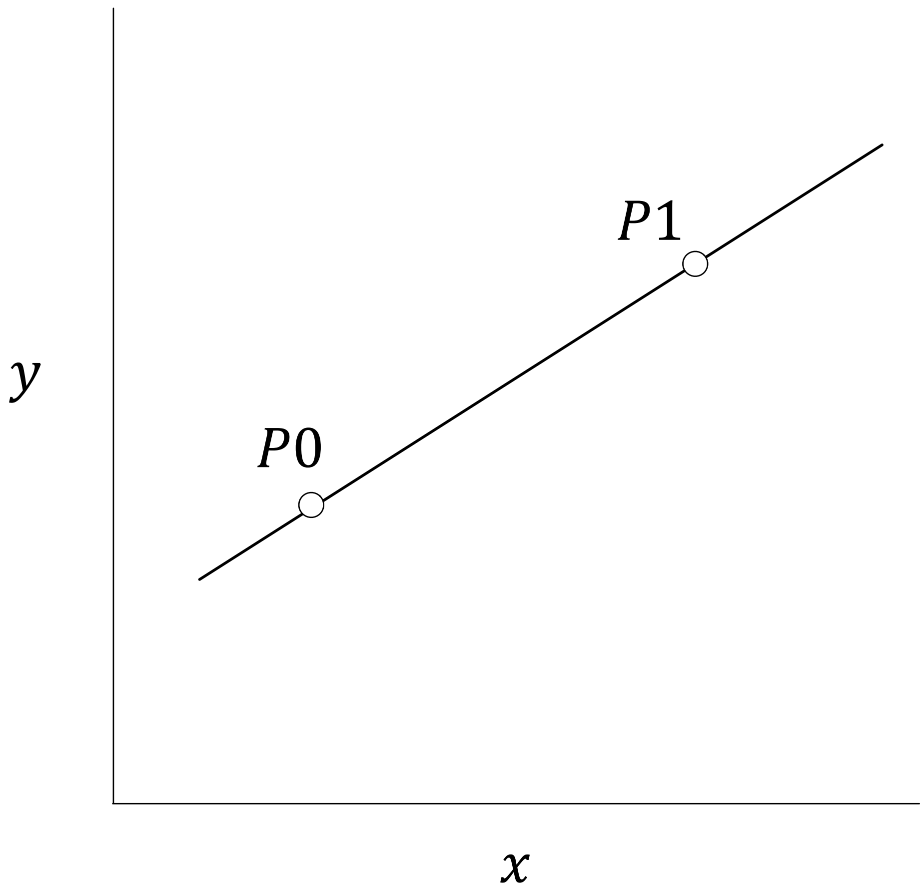




Solids



Drawing a line segment



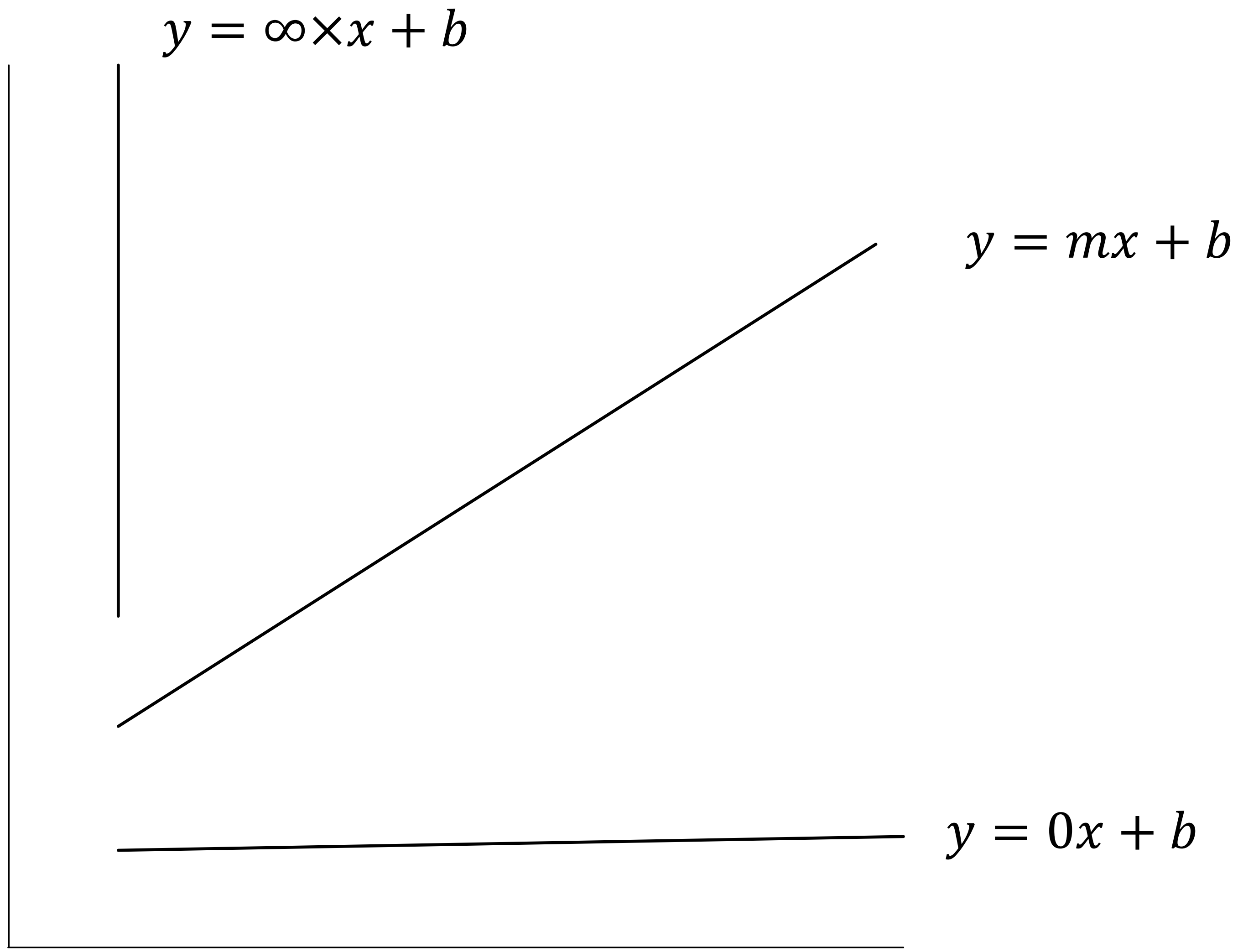
$$y = mx + b$$

$$P0 = (x0, y0)$$

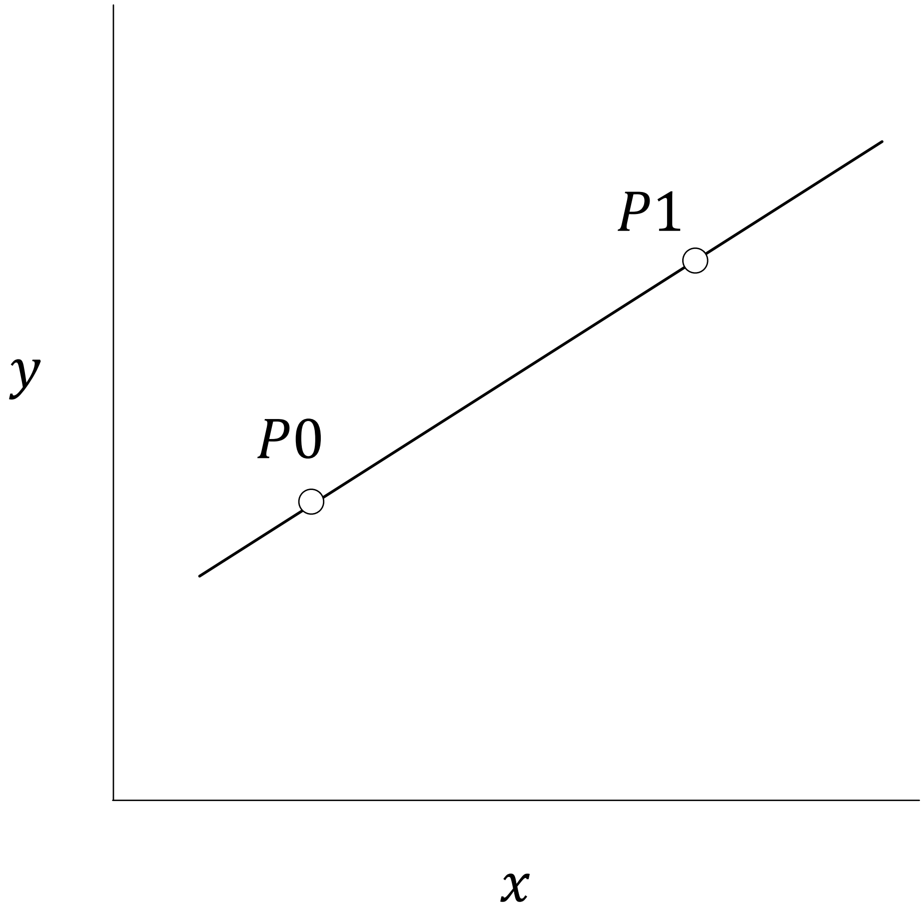
$$P1 = (x1, y1)$$

```
for x = x0 to x1
  y = mx+b
  putpixel(x,y)
```

Drawbacks of standard formula: special cases



Solution: parametric form



$$x = t\,dx + px$$
$$y = t\,dy + py$$

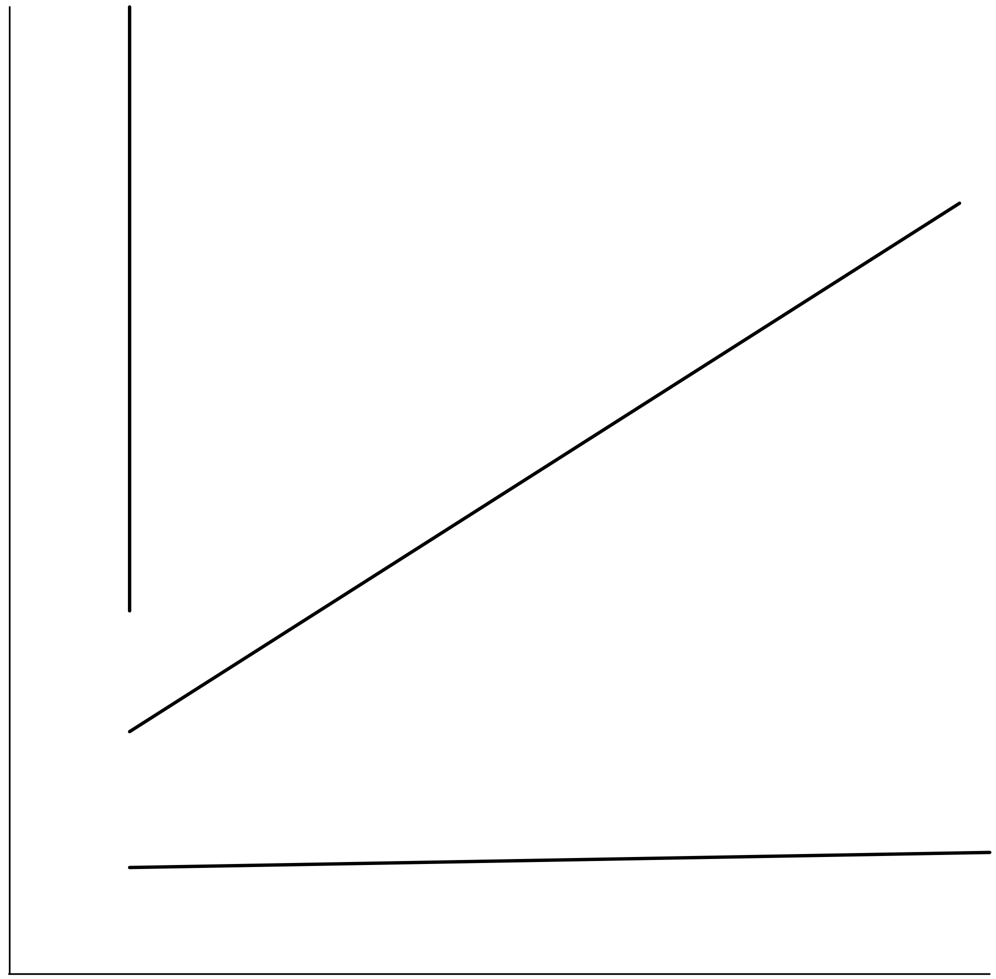
$$m = \frac{dy}{dx} = \frac{y1 - y0}{x1 - x0}$$

$$0 \le t \le 1$$

or

$$t \in [0,1]$$

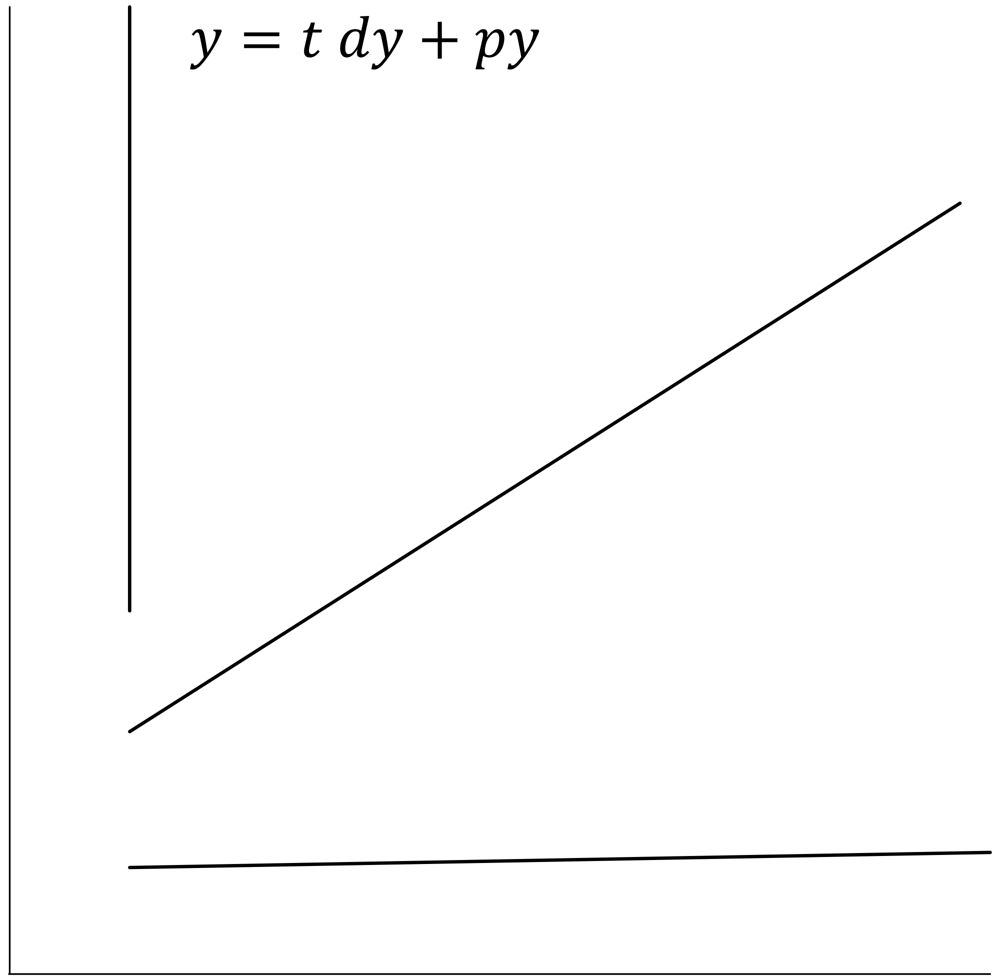
Work for special cases?



$$x = t\ dx + px$$
$$y = t\ dy + py$$

Work for special cases?

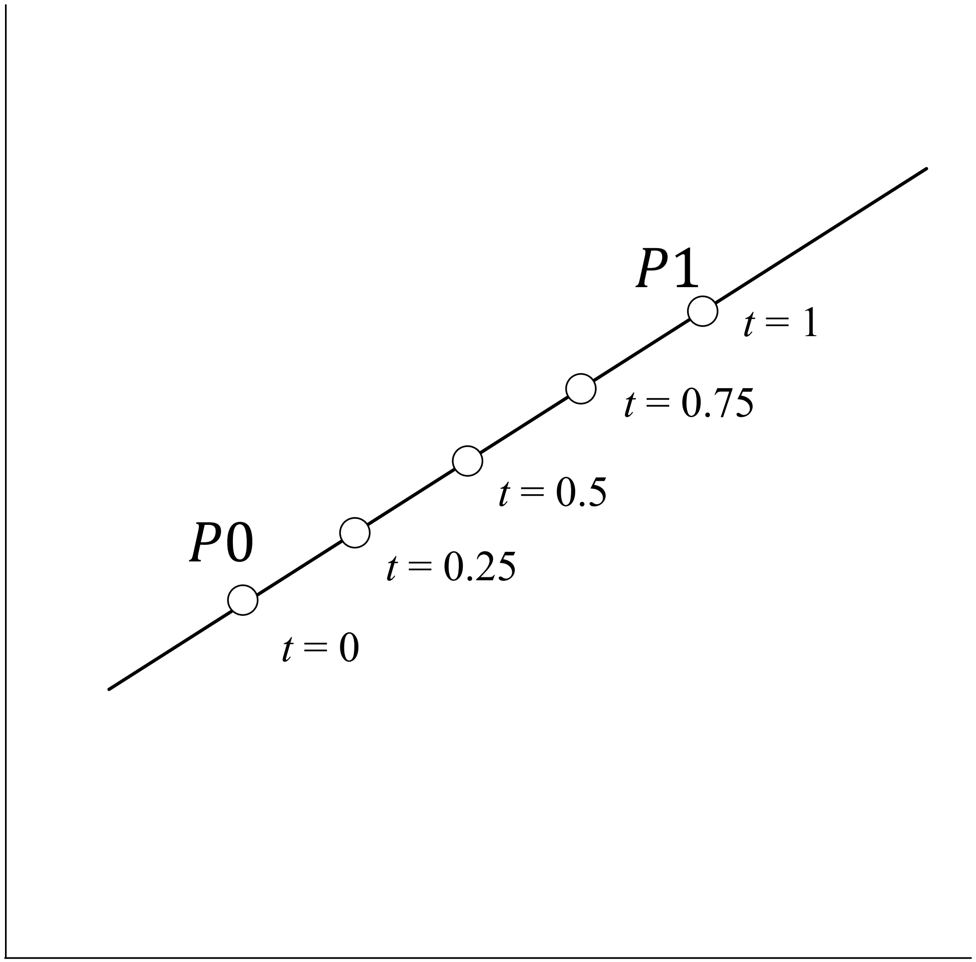
$$x = t \times \mathbf{0} + px$$
$$y = t\,dy + py$$



$$x = t\,dx + px$$
$$y = t\,dy + py$$

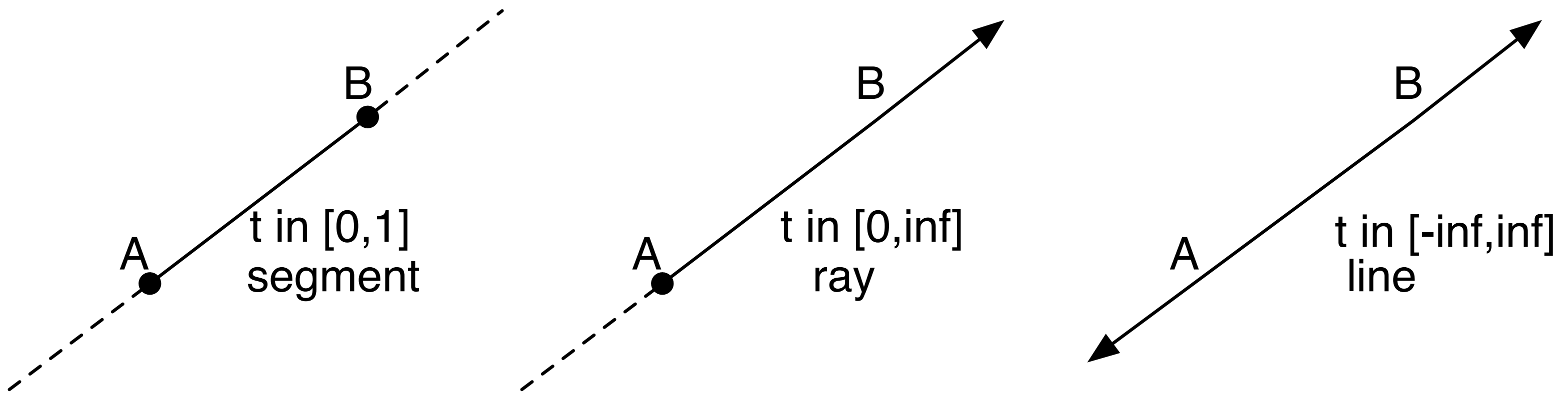
$$x = t\,dx + px$$
$$y = t \times \mathbf{0} + py$$

Using t for proportional placement (midpoint, etc)

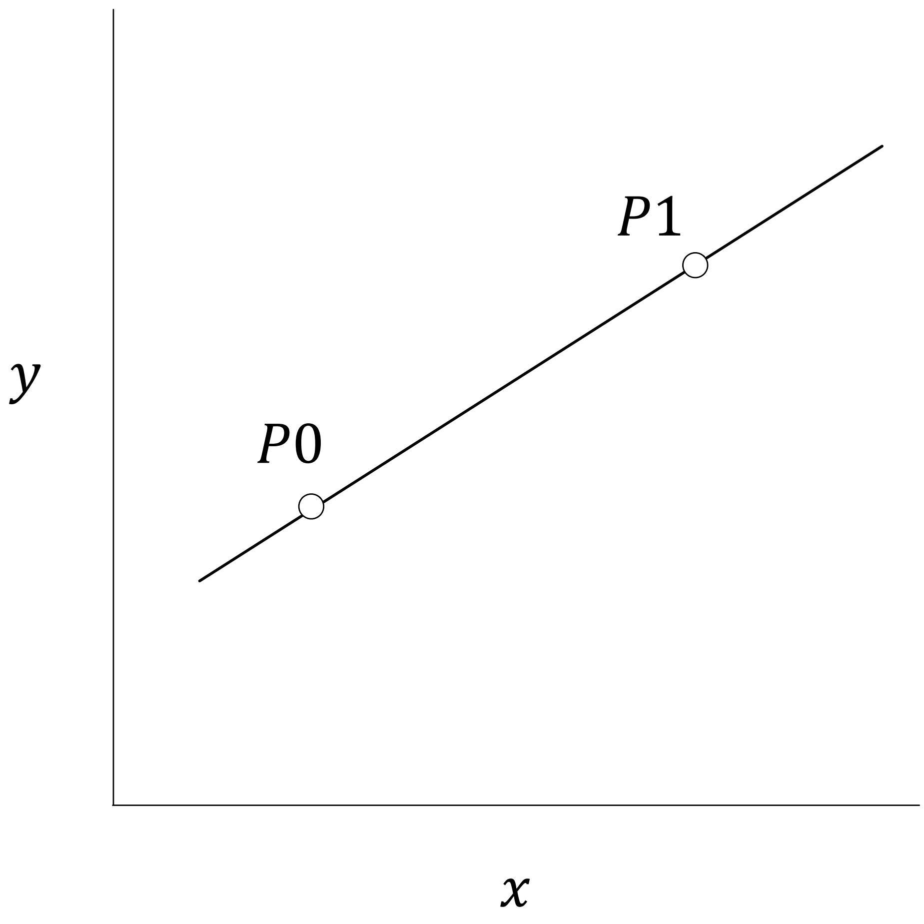


$$x = t\ dx + px$$
$$y = t\ dy + py$$

Varying the range of t: line, line segment and ray



Must a parametric line be linear?



$$x = t^2\, dx + px$$

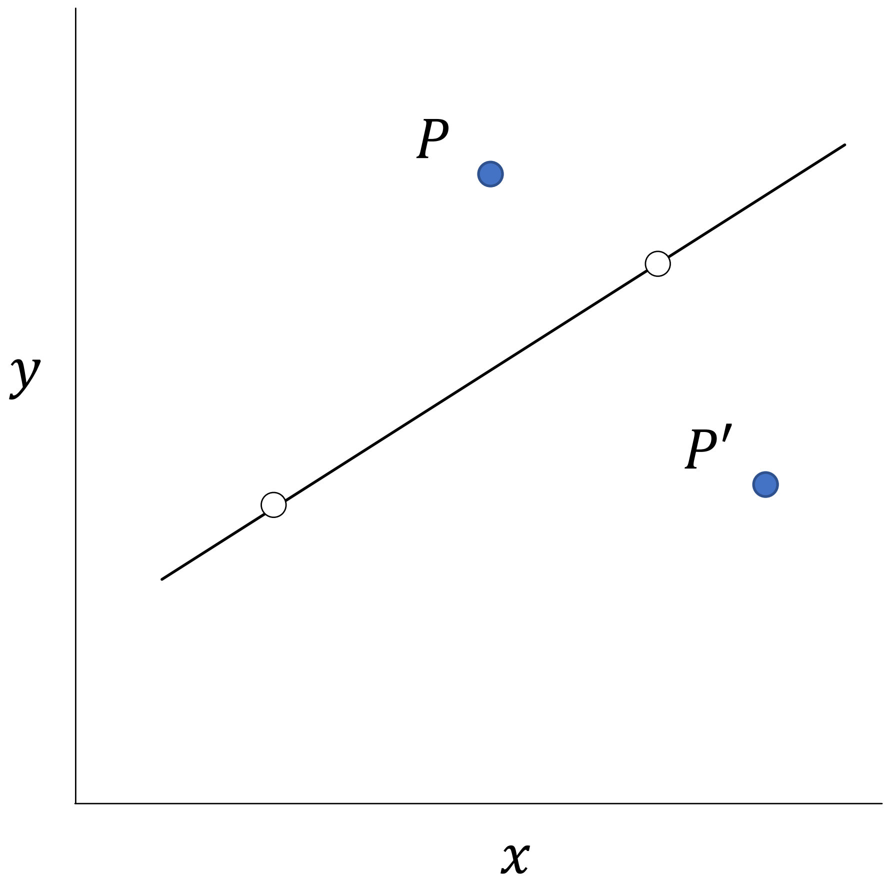
$$y = t^2\, dy + py$$

$$0 \le t \le 1$$

or

$$t \in [0,1]$$

Any use to y=mx+b?

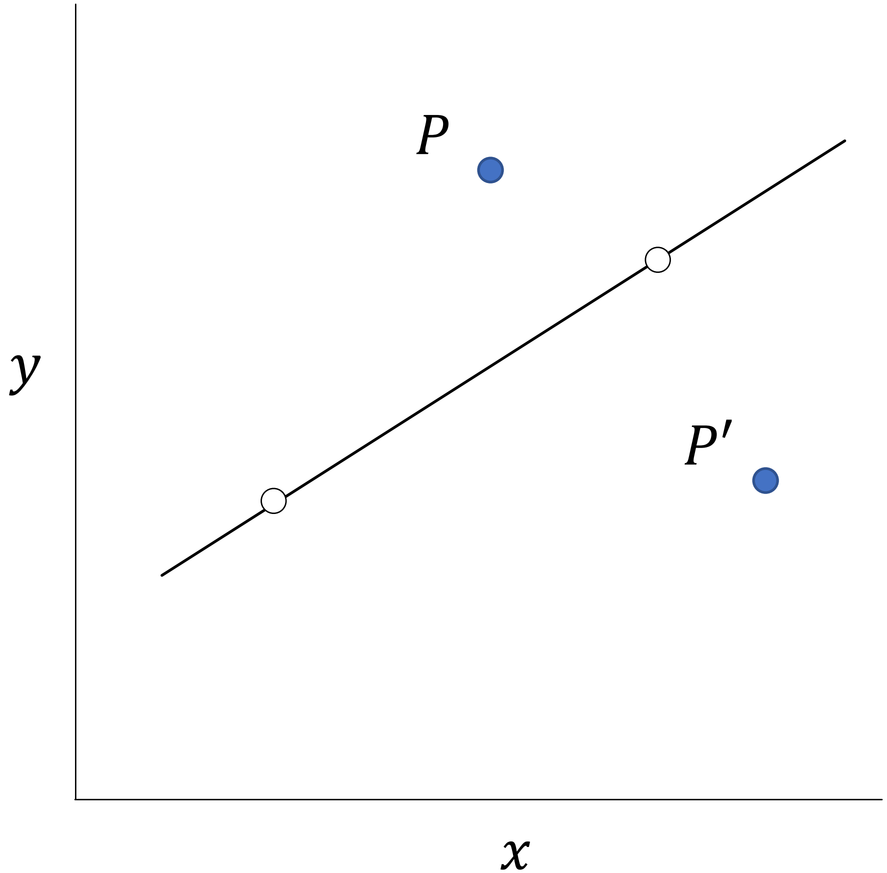


Functional line equation

$$y = mx + b$$

Are P and P' above or below the line?

Any use to mx+b?



Functional line equation

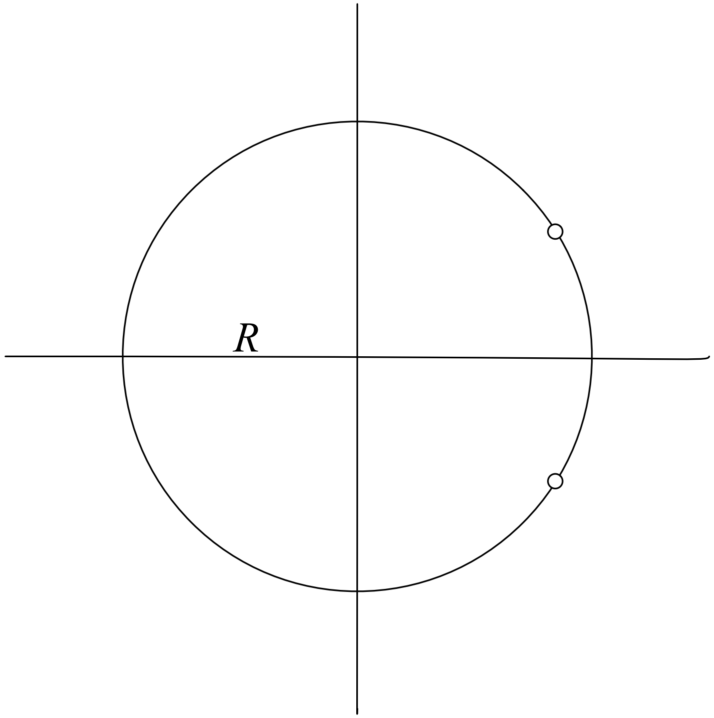
$$y = mx + b$$

Are P and P' above or below the line?

$y > mx + b$ above

$y < mx + b$ below

Drawing a circle



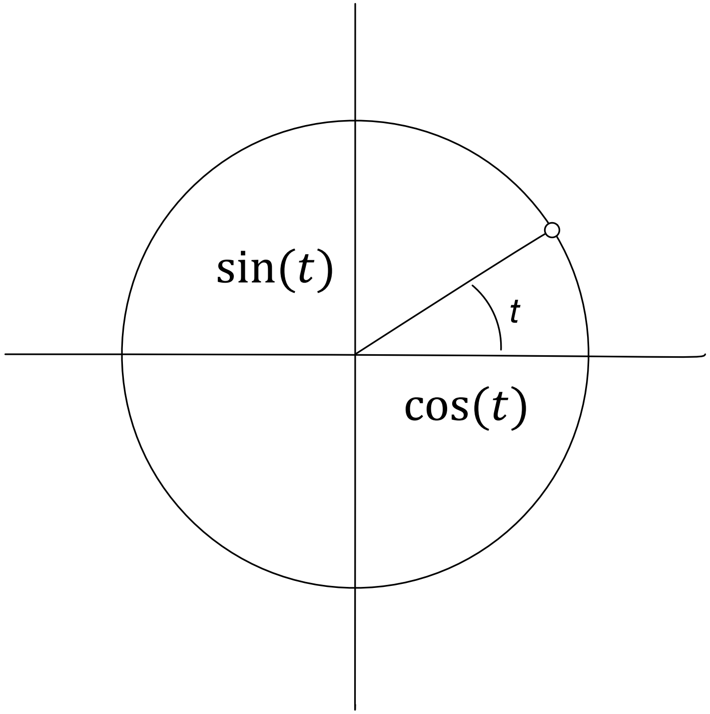
Implicit equation

$$x^2 + y^2 = R^2$$

“Functional” equation
(multi-valued)

$$y = \pm\sqrt{x^2 + R^2}$$

Circle with trig



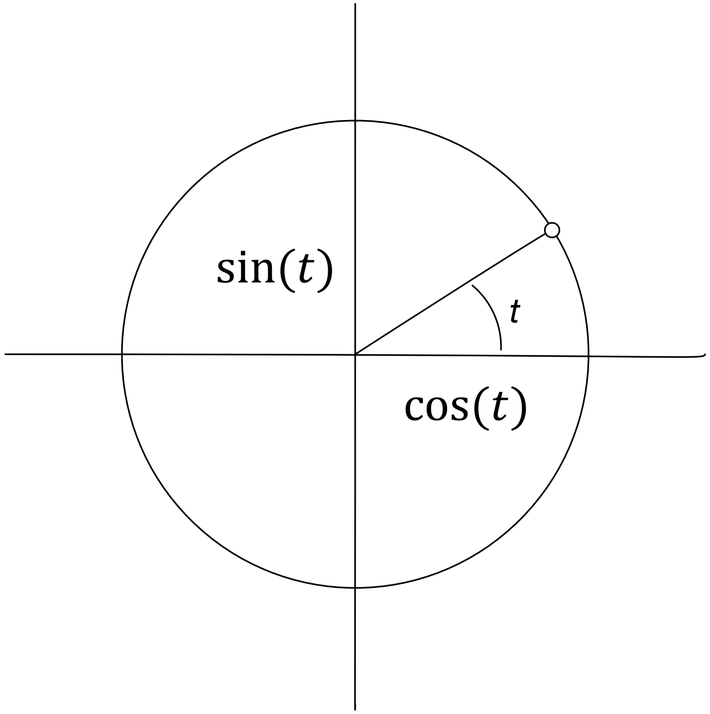
Parametric equation

$$x = R\cos(t)$$

$$y = R\sin(t)$$

$$0 \leq t \leq ??$$

Circle with trig



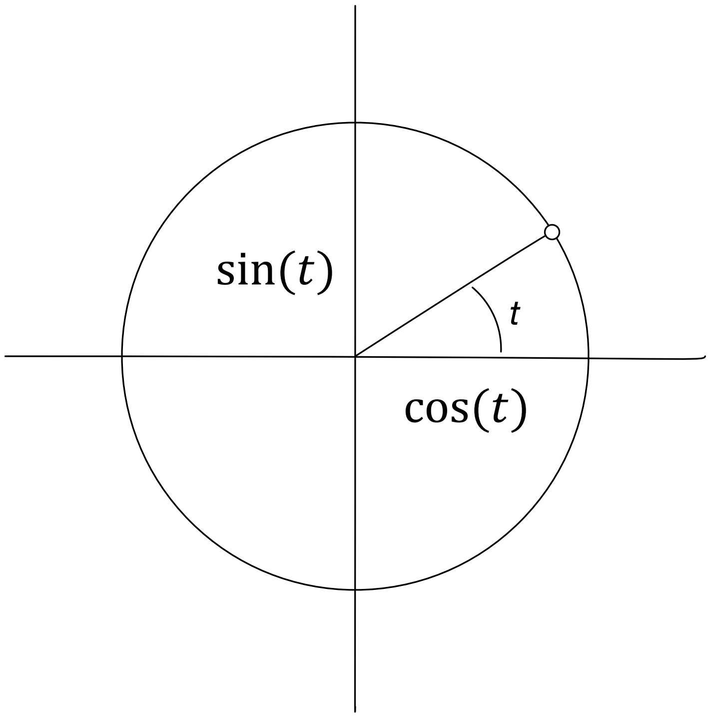
Parametric equation

$$x = R\cos(t)$$

$$y = R\sin(t)$$

$$0 \leq t \leq 2\pi$$

Validating parametric equations



Substitute parametric into implicit

$$x = R\cos(t)$$

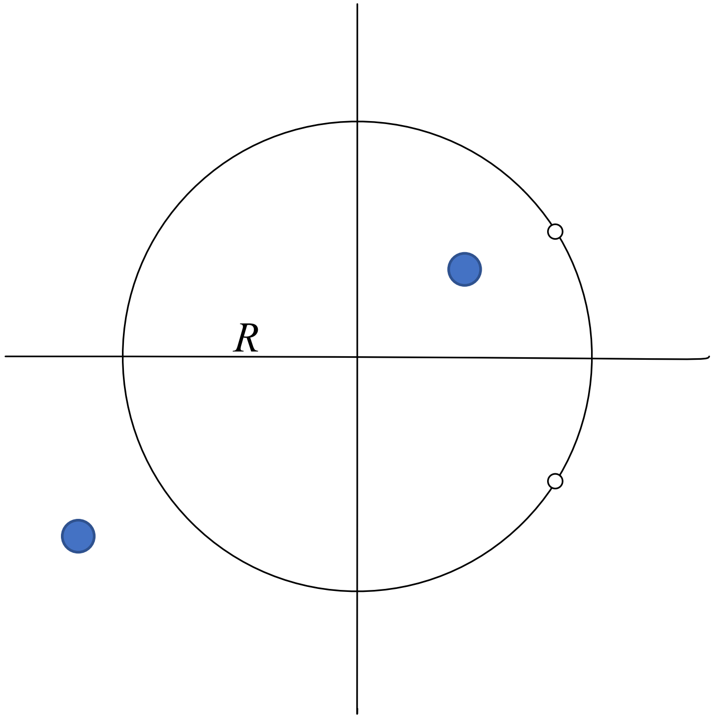
$$y = R\sin(t)$$

$$(R\cos(t))^2 + (R\sin(t))^2 = R^2$$

$$R^2(cos^2(t) + sin^2(t)) = R^2$$

$$R^2(1) = R^2$$

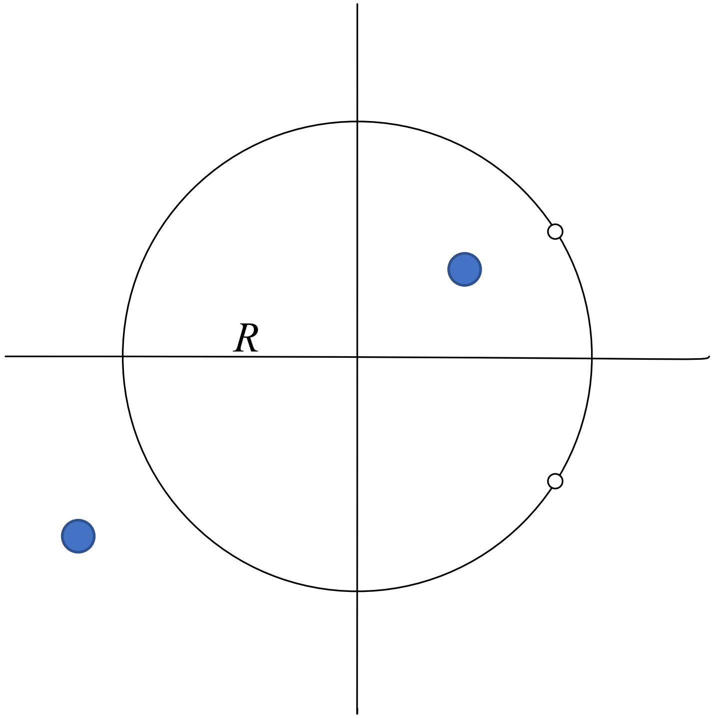
Inside or outside?



Implicit equation

$$x^2 + y^2 = R^2$$

Inside or outside?



Implicit equation

$$x^2 + y^2 = R^2$$

$x^2 + y^2 > R^2$ out

$x^2 + y^2 < R^2$ in

Summary of curve equations

- Implicit $f(x,y) = 0$
  - Inside/Outside tests
- Parametric $x = fx(t)$ $y = fy(t)$
  - Drawing/Intersection tests
- Functional $y = f(x)$
  - Dual purpose: drawing, testing

Coding parametric curves

Circle

```
for t = 0 to 2π by step 0.1
    x = R*cos(t)
    y = R*sin(t)
    putpixel(x,y)
```

Generic

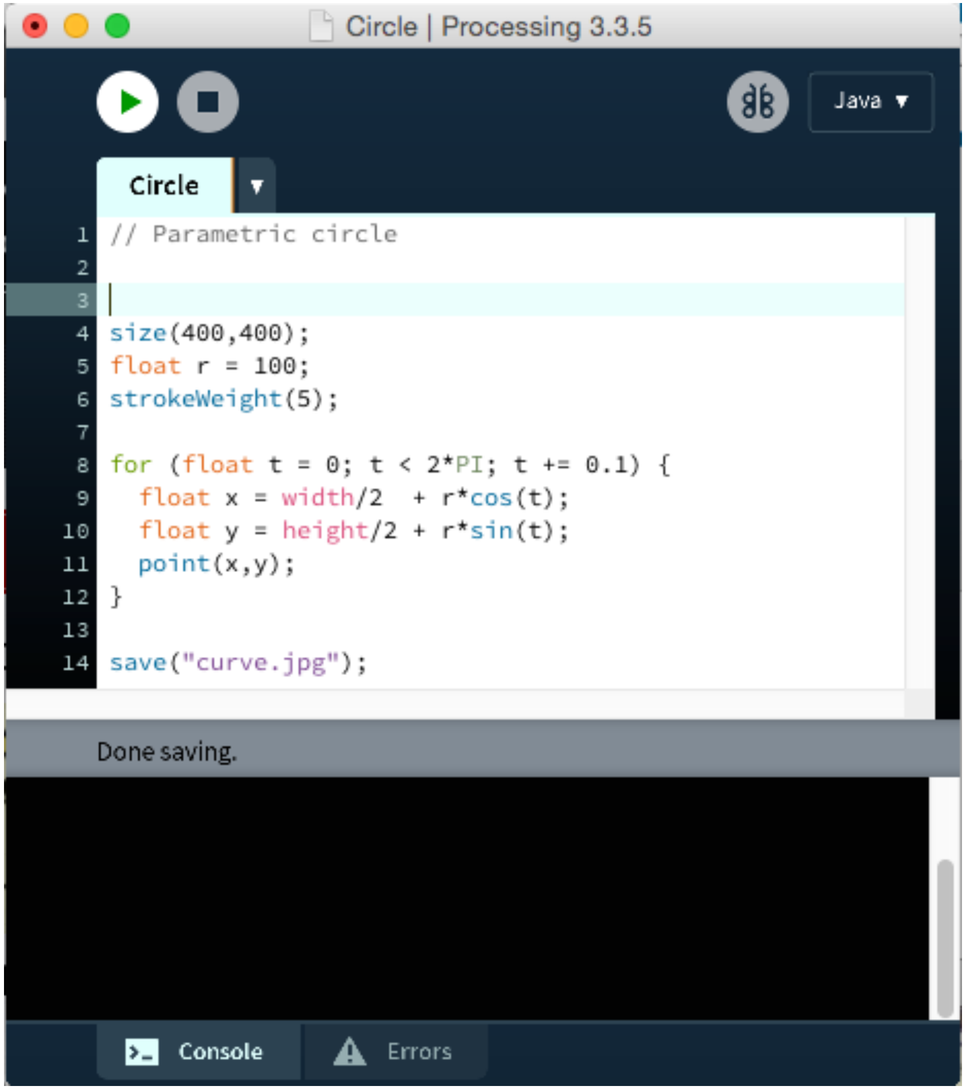
given

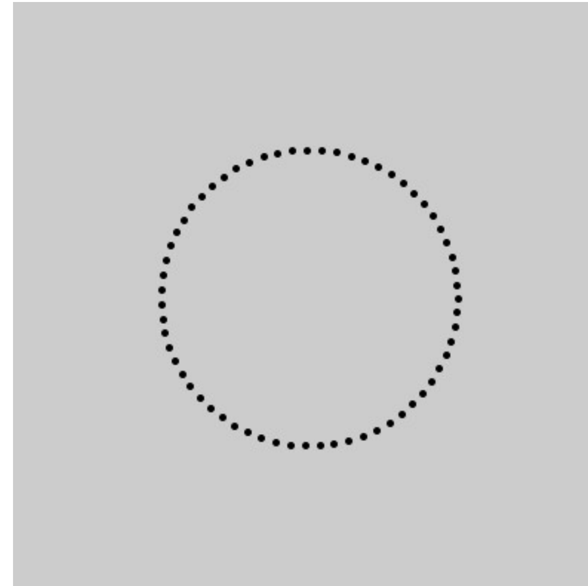
$$x = f_x(t)$$

$$y = f_y(t)$$

```
for t = t0 to t1 by step dt
    x = fx(t)
    y = fy(t)
    putpixel(x,y)
```

Processing.org





```
size(400,400);
float r = 100;
strokeWeight(5); // Set point size in pixels

for (float t = 0; t < 2*PI; t += 0.1) {
   float x = width/2  + r*cos(t);
   float y = height/2 + r*sin(t);
   point(x,y);
   }

save("curve.jpg");
```

Playing with the code …

```
for (float t = 0; t < 2*PI; t += 0.3) { // Change increment

for (float t = 0; t < PI; t += 0.3) { // Change limits

float x = width/2  + 1.5*r*cos(t);   // Unequal r
float y = height/2 + r*sin(t);
```

strokeWeight(random(1,10)); point(x,y);

```
float x = width/2  + 10*t*cos(t);   // Varying r
float y = height/2 + 10*t*sin(t);


strokeWeight(random(1,10)); // Random point size
point(x,y);
```

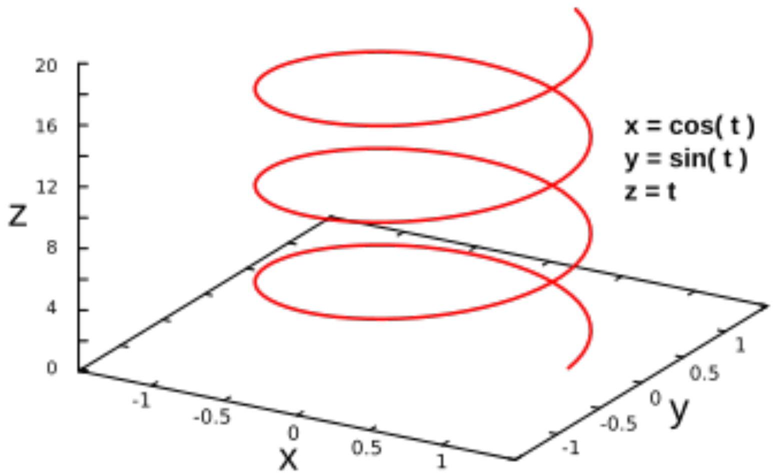
Animate

```
// Infinity Rainbow
// from Jason Labbe (jasonlabbe3d.com)
void setup() {
    size(600,600);
    colorMode(HSB,255); // Hue/Saturation/Brightness
    background(255);
    noStroke();
}

void draw() {
    fill(frameCount % 255,255,255);
    float x = (width/2)+100*sin(frameCount/20.0);
    float y = (height/2)+200*sin(frameCount/10.0);
    ellipse(x,y,20,20);
    }
```

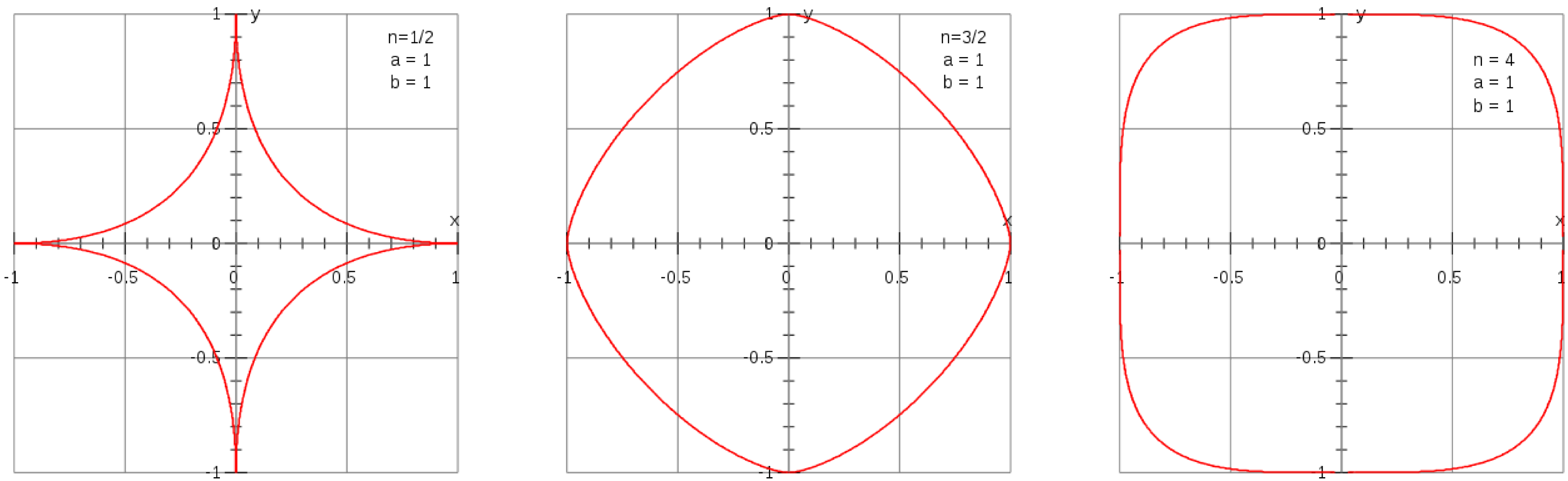
Next steps

- Animate
- Collect more parametric curves
  - Cartesian
  - Polar
- Add third dimension
- Move to surfaces



Superellipses

Generalized ellipse that can take different shapes based on exponent n



$$|x|^n + |y|^n = R^2$$

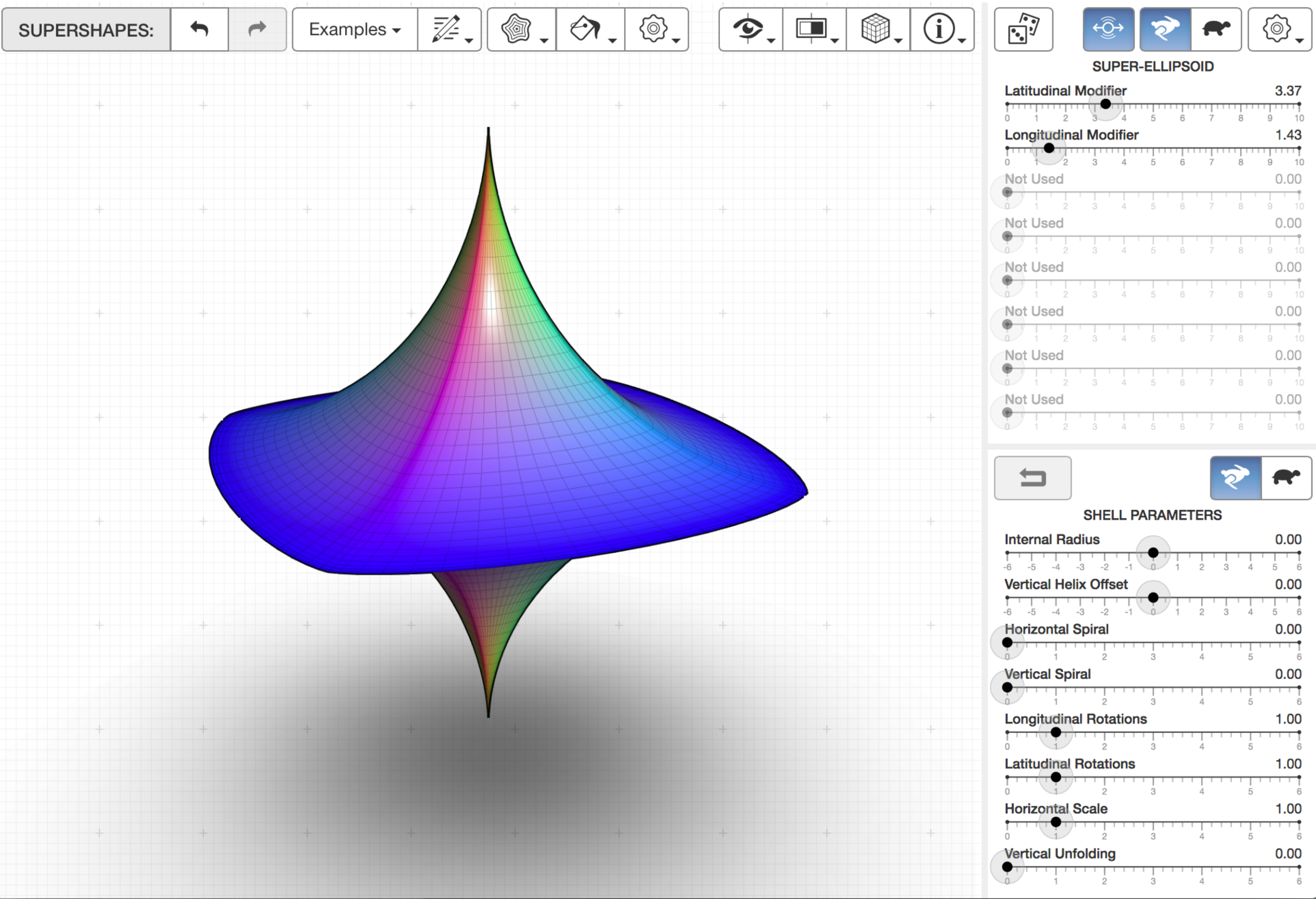
$$x = Rcos^{\frac{2}{n}}(t)sgn(\cos(t))$$

$$y = Rsin^{\frac{2}{n}}(t)sgn(\sin(t))$$

https://www.desmos.com/calculator

https://en.wikipedia.org/wiki/Superellipse

Andrew March supershapes



What you should know after today

1. Where drawing happens – software rendering on CPU, hardware rendering on GPU, preloading
2. Why functional equations are problematic
3. How to draw a line and circle with parametric equation
4. How to use ranges of t for segments, rays and lines
5. Using implicit and functional equations for shape inside/outside tests
6. How to validate parametric equations
7. Using powers of fractions to bias curves
8. Using parametric equations for curves, shapes, animations and other purposes

Today's resources

- Desmos
  - https://www.desmos.com/calculator
  - 2D graphing with parametric curves

- Andrew Marsh's supershapes
  - http://andrewmarsh.com/apps/staging/supershapes.html
  - Super ellipsoids and similar shapes

- Processing
  - https://processing.org
  - Resource for quick program "sketches", concepts
  - Sketches Circle.pde, InfinityRainbow.pde, HelixFly.pde