PAT 204/504 (Fall 2024)

Creative Coding

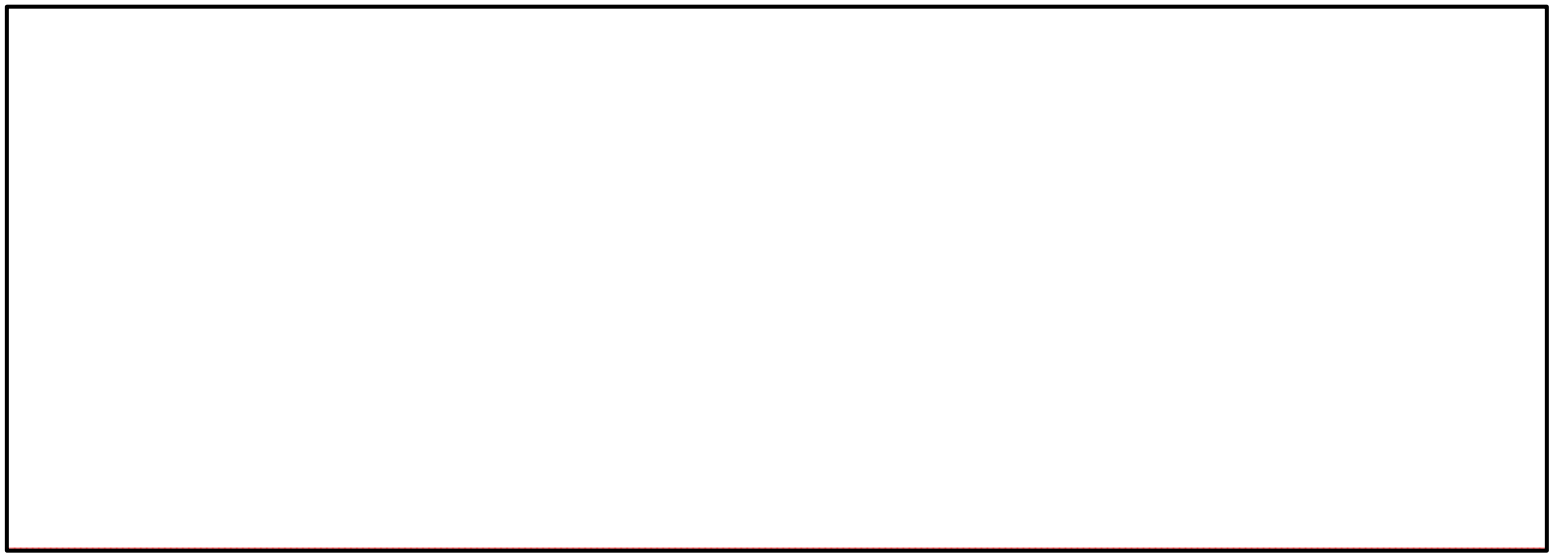
Lecture 8: Images & Videos

Instructor: Hao-Wen Dong

SCHOOL OF MUSIC, THEATRE & DANCE
PERFORMING ARTS TECHNOLOGY
UNIVERSITY OF MICHIGAN

Homework 3: Spectrum Visualizer

- Modify the template code to implement a spectrum visualizer
- Instructions will be released on Gradescope
- Due at **11:59pm ET** on **September 23**
- Late submissions: **1 point deducted per day**



(Recap) Amplitude Class

```
import processing.sound.*;

Amplitude amp = new Amplitude(this);
AudioIn in = new AudioIn(this, 0);
float a;

void setup() {
  size(400, 400);

  in.start();
  amp.input(in);
}

void draw() {
  background(0);
  a = amp.analyze();
  circle(200, 200, a * 400);
}
```

Initialize an Amplitude object

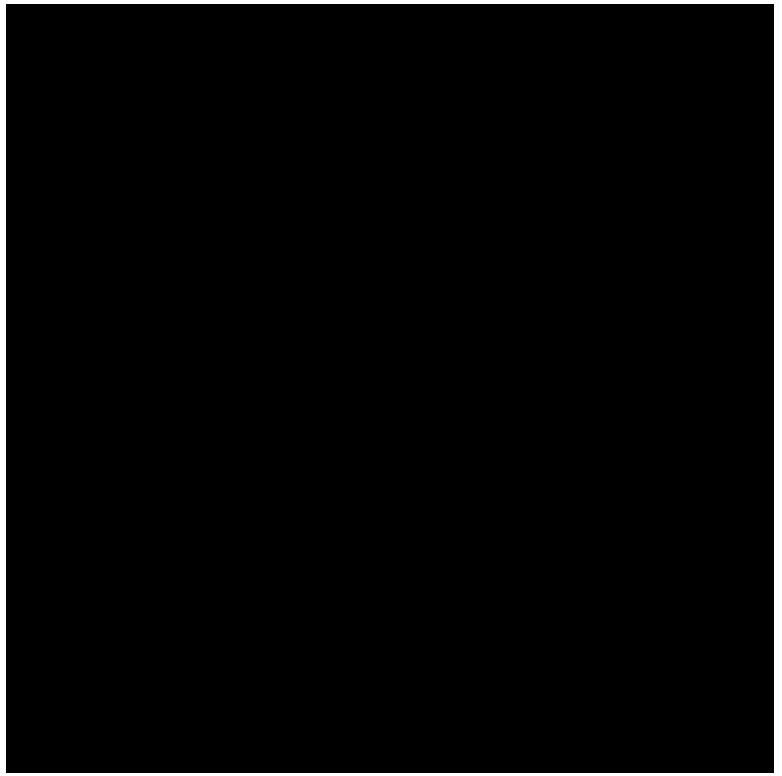
Initialize an AudioIn object

Start taking audio input

Route the audio input to the amplitude meter

Measure the amplitude

Normalized to [0, 1]



(Recap) Array vs List

Array

```
float[] pos = new float[3];
```
Declaration

```
void setup() {
  size(400, 400);

  pos[0] = 100;
  pos[1] = 200;
  pos[2] = 300;
}

void draw() {
  for (int i = 0; i < pos.length; i++) {
    circle(pos[i], 200, 50);
  }
}
```
Initialization

Length of the array

List

```
FloatList pos = new IntList();
```
Declaration

```
void setup() {
  size(400, 400);

  pos.append(100);
  pos.append(200);
  pos.append(300);
}

void draw() {
  for (int i = 0; i < pos.size(); i++) {
    circle(pos.get(i), 200, 50);
  }
}
```
Initialization

Length of the list

(Recap) Array vs List

	Array	List
Size	Fixed	Dynamic
Item data type	Same	Can be different
Access speed	Faster	Slower
Memory requirement	Low	High
Multi-dimensional	Possible	Not supported

(Recap) Sorting an Array vs Sorting a List

```
int[] arr = {3, 2, 1};
sort(arr);
println(arr);
```

```
[0] 3
[1] 2
[2] 1
```

`sort(arr)` returns a new sorted array

```
int[] arr = {3, 2, 1};
arr = sort(arr);
println(arr);
```

```
[0] 1
[1] 2
[2] 3
```

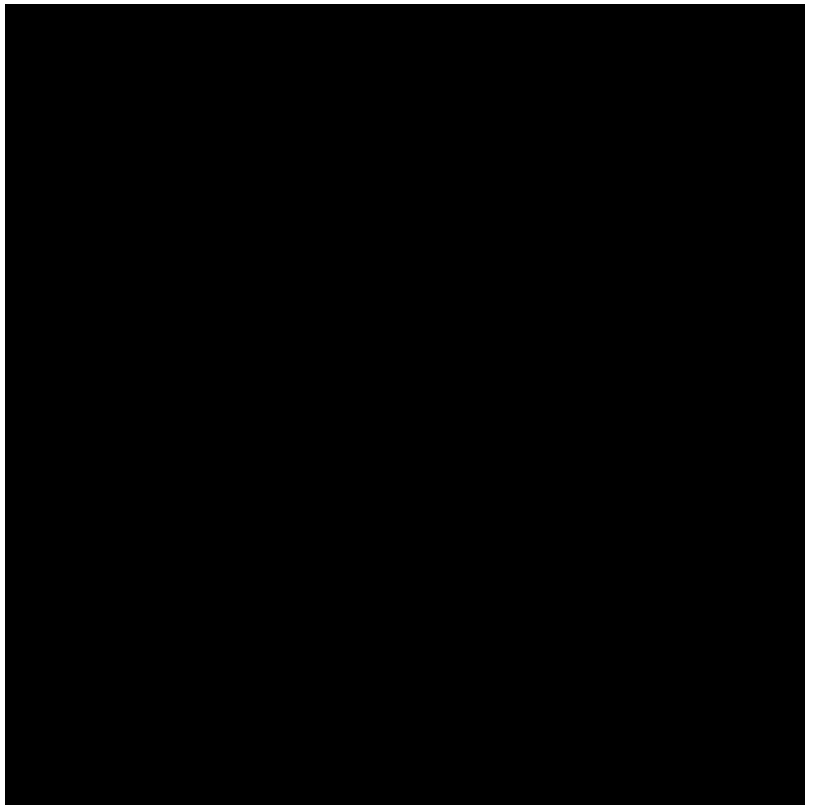
```
IntList li = new IntList();
li.append(3);
li.append(2);
li.append(1);
li.sort();
println(li);
```

```
IntList size=3 [ 1, 2, 3 ]
```

`List.sort()` returns the original list, sorted

Exercise: Bouncing Balls

- **Add a ball when the mouse is clicked**, where the new ball **starts from where the mouse is**
- Two approaches
  - **Array of objects** `Ball[] balls = new Ball[20];`
  - **ArrayList** `ArrayList<Ball> balls = new ArrayList<Ball>()`



Images

`PImage` Class

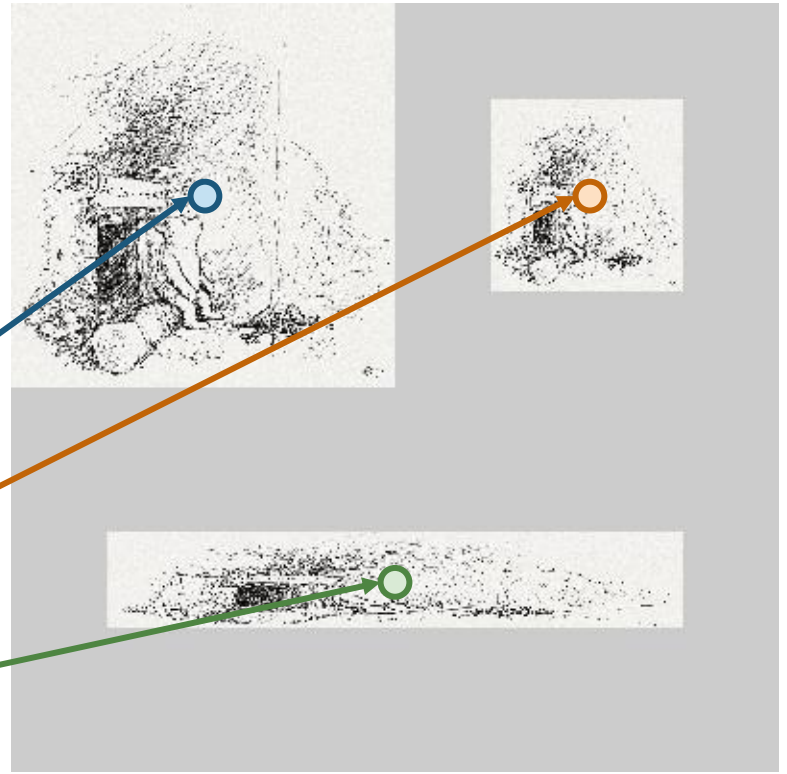
- A class for storing images
  - Processing natively supports **GIF**, **JPG**, **TGA** and **PNG** files
  - Usually created by loading an image using **`loadImage`**
  - You may also create a new image from scratch using **`createImage`**

- Fields
  - **`pixels[]`** Array of all the pixels in the image **Note that this is a 1D array!**
  - **`width`** Width of the image (in pixels)
  - **`height`** Height of the image (in pixels)

Example: Displaying Images

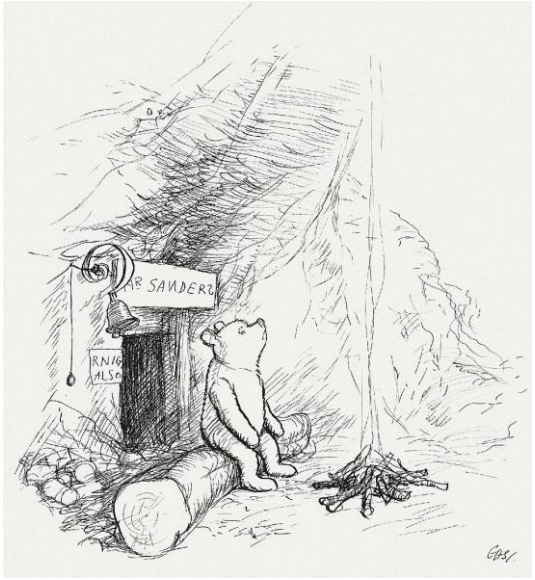
```
PImage img;

void setup() {
  size(400, 400);
  noLoop();

  img = loadImage("pooh.jpg");
}

void draw() {
  imageMode(CENTER);
  image(img, 100, 100, 200, 200);
  image(img, 300, 100, 100, 100);
  image(img, 200, 300, 300, 50);
}
```

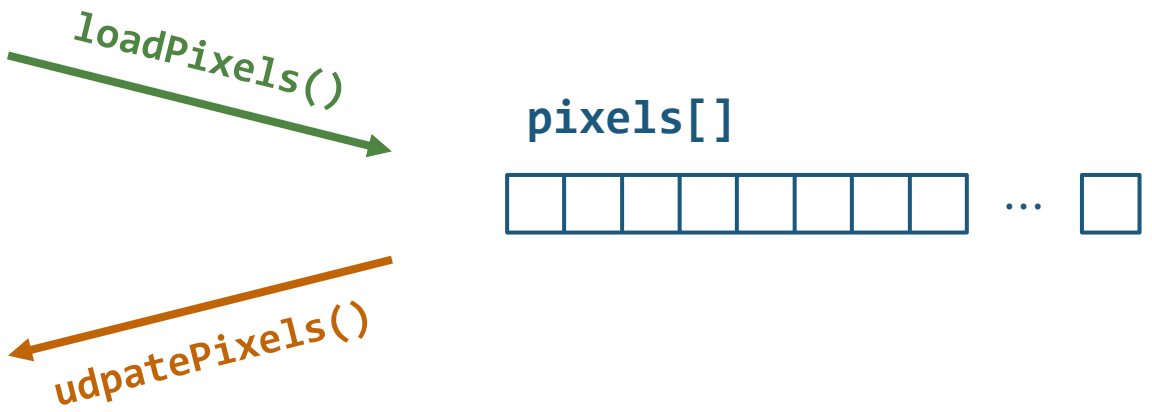
Load the image



Loading the Pixels

- We can directly interact with the pixels of an image
  - `Image.pixels[]` Array of all the pixels in the image
  - `Image.loadPixels()` Load the image content to `Image.pixels[]`
  - `Image.updatePixels()` Update the image content with `Image.pixels[]`





Example: Manipulating Pixels

- Modify **`image.pixels`** directly
- Add a small red box at the bottom right

```
void setup() {
  size(400, 400);
  img = loadImage("pooh.jpg");
  img.loadPixels();
}

void draw() {
  for (int i = img.width - 50; i < img.width; i++) {
    for (int j = img.height - 50; j < img.height; j++) {
      img.pixels[j * img.width + i] = #ff0000;
    }
  }
  img.updatePixels();
  image(img, 0, 0, 400, 400);
}
```

Load the image

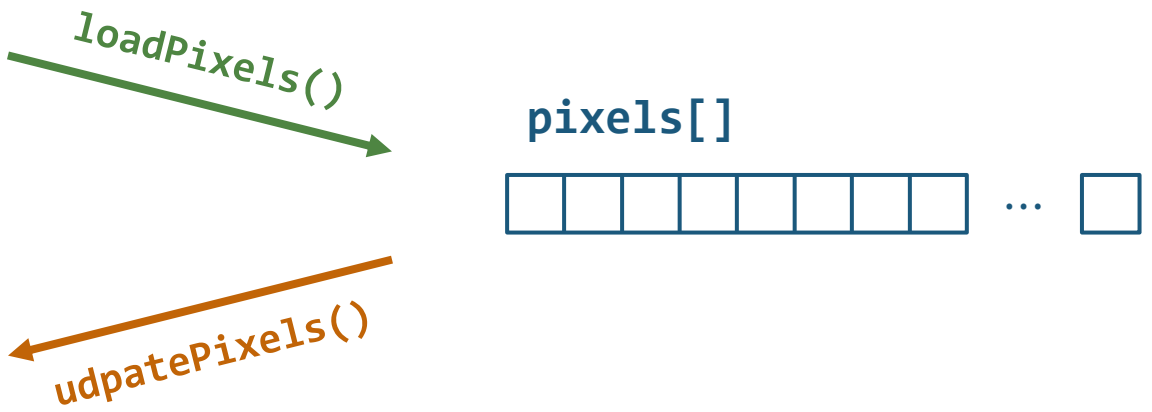
Load the image content to `pixels[]`

Update the pixel values

Update the image content with `pixels[]`

The Processing Window

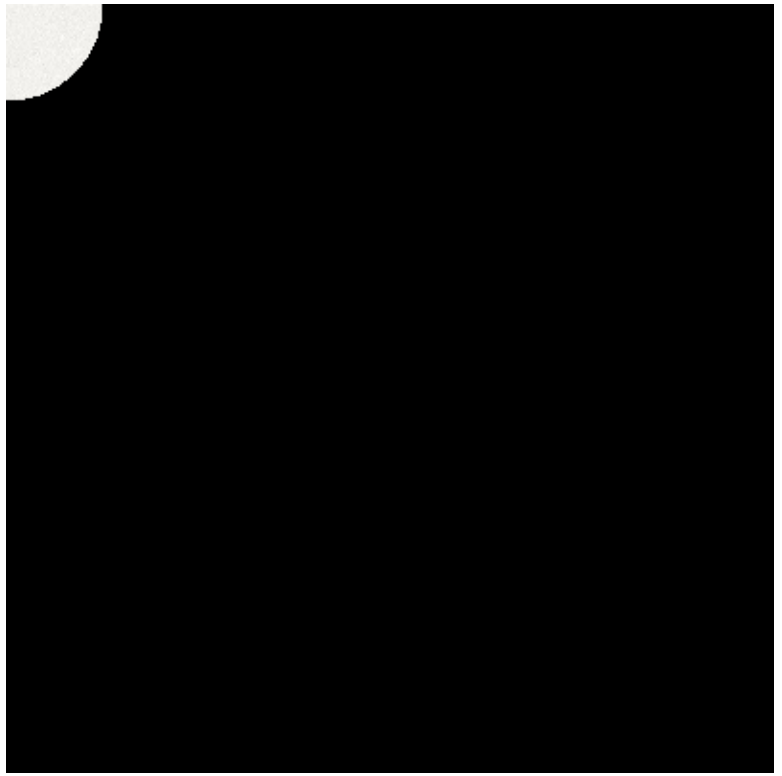
- The processing display window works like an `Image` object!
  - **`pixels[]`** Array of all the pixels in the image
  - **`loadPixels()`** Load the image content to `pixels[]`
  - **`updatePixels()`** Update the image content with `pixels[]`



Exercise: The Reveal Effect

- Implement the “reveal” effect by using
  - `pixels[]`
  - `loadPixels()`
  - `updatePixels()`

Example image



Exercise: The Reveal Effect

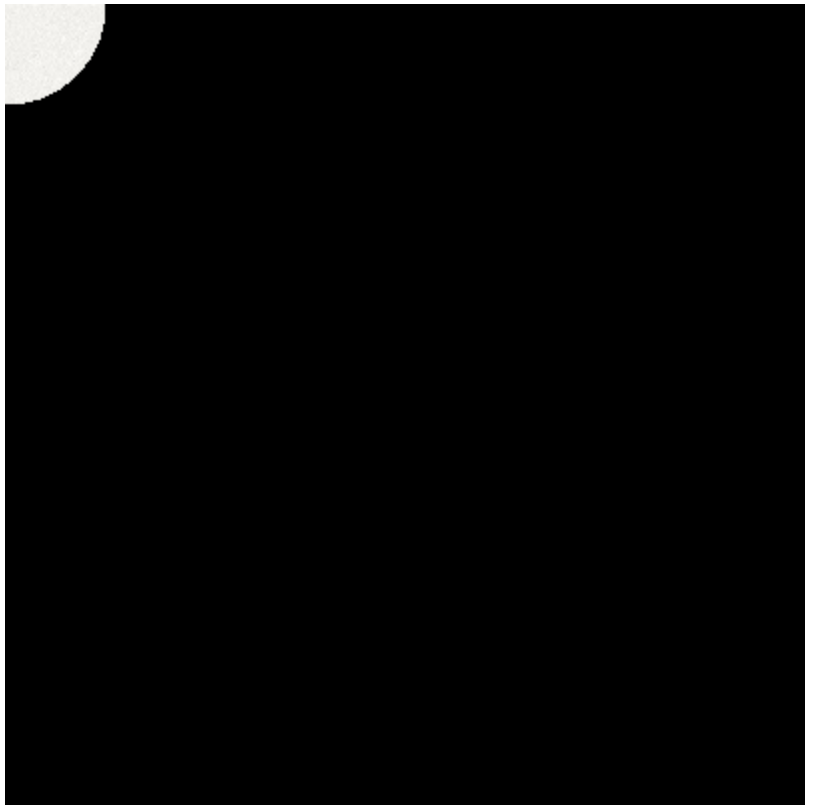
```
PImage img;
int[] org;

void setup() {
  size(400, 400);
  img = loadImage("pooh.jpg");
  image(img, 0, 0, 400, 400);
  loadPixels();
  org = pixels.clone();
  background(0);
  loadPixels();
}

void draw() {
  for (int x = 0; x < width; x++) {
    for (int y = 0; y < height; y++) {

      // YOUR CODE HERE

    }
  }
  updatePixels();
}
```

Load the image

Load the image content to `pixels[]`

Why again?

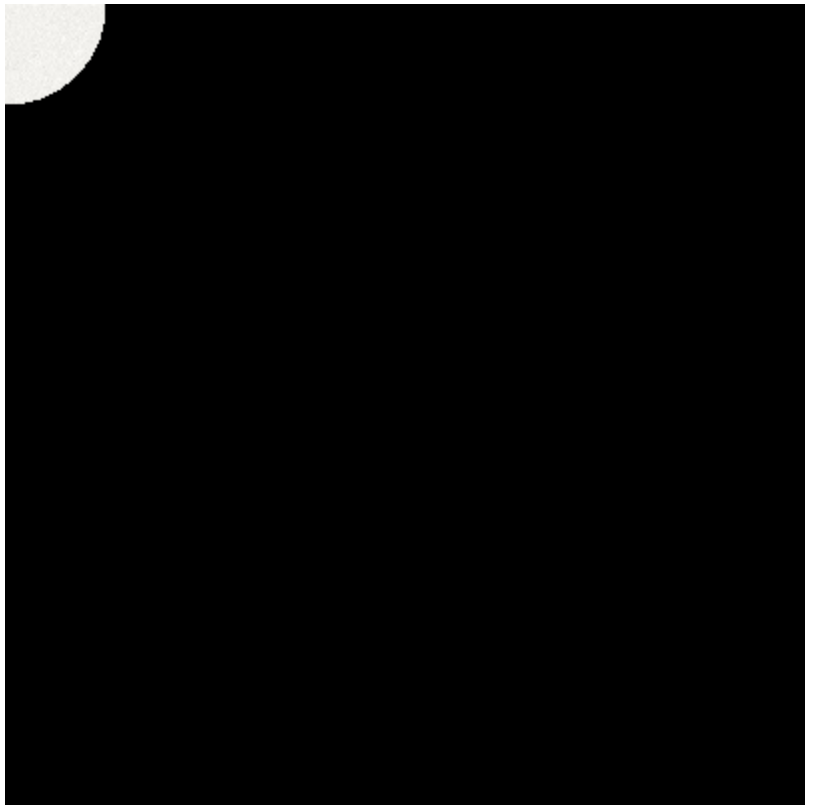
Update the image content with `pixels[]`



Exercise: The Reveal Effect

```
void setup() {
  size(400, 400);
  img = loadImage("pooh.jpg");
  image(img, 0, 0, 400, 400);
  loadPixels();
  org = pixels.clone();
  background(0);
  loadPixels();
}

void draw() {
  for (int x = 0; x < width; x++) {
    for (int y = 0; y < height; y++) {
      int loc = x + y * width;
      float d = dist(x, y, mouseX, mouseY);
      if (d < 50) {
        pixels[loc] = org[loc];
      }
    }
  }
  updatePixels();
}
```

Update the pixel values



2D Arrays

```
int[][] arr = {{0, 1, 2, 3}, {3, 2, 1, 0}, {3, 5, 6, 1}, {3, 8, 3, 4}};
```

```
int[][] arr = {{0, 1, 2, 3},
               {3, 2, 1, 0},
               {3, 5, 6, 1},
               {3, 8, 3, 4}};
```

Example: Pointillism

```
PImage img;

void setup() {
  size(400, 400);
  img = loadImage("sakura.jpg");
  background(255);
  noLoop();
}

void draw() {
  for (int i = 0; i < 10000; i++) {
    int x = int(random(img.width));
    int y = int(random(img.height));
    int loc = x + y * img.width;

    img.loadPixels();
    float r = red(img.pixels[loc]);
    float g = green(img.pixels[loc]);
    float b = blue(img.pixels[loc]);

    noStroke();
    fill(r, g, b, 100);
    circle(x, y, 20);
  }
}
```

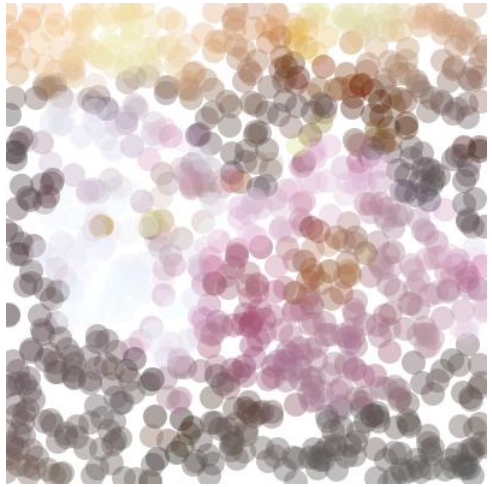
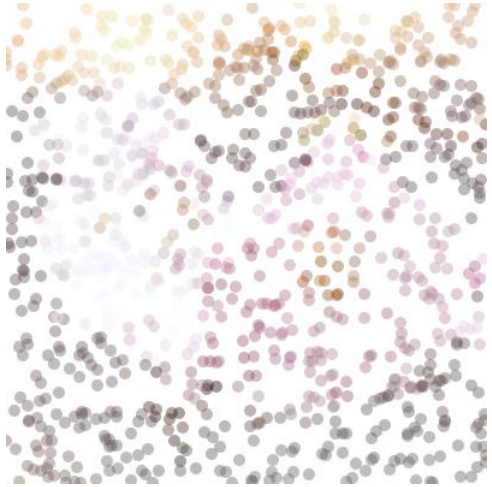
Pick a random pixel

Find the color of the pixel

Set the color of the circle

Draw the circle

Pointillism

	1,000 points	10,000 points	100,000 points
pointSize = 20			
pointSize = 10			

Videos

Movie

- A class for storing and playing **movies**
  - Processing natively supports **MOV** and **MP4** files
- Methods
  - `frameRate()` Set the frame rate of the movie
  - `speed()` Set the speed of the movie
  - `play()` Play the movie one time
  - `loop()` Play the movie non-stop
  - `pause()` Pause the movie playback
  - `stop()` Stop the movie playback
  - `jump()` Jump to a specific location

Example: Loading a Movie

```
import processing.video.*;

Movie myMovie;

void setup() {
  size(640, 360);
  myMovie = new Movie(this, "movie.mov");
  myMovie.loop();
}

void movieEvent(Movie m) {
  m.read();
}

void draw() {
  image(myMovie, 0, 0);
}
```

Import video library

Initialize the `movie` object

Called whenever a new frame is available to read

Capture

- A class for working with **capture device** such as **webcams**
- Methods
  - `frameRate()` Set the frame rate of the capture
  - `start()` Start capturing frames
  - `stop()` Stop capturing frames
  - `read()` Read the current frame

Example: Webcam Capture

```
import processing.video.*;

Capture cam;

void setup() {
  size(640, 480);

  println(Capture.list());
  cam = new Capture(this, 640, 480);
  cam.start();
}

void draw() {
  if (cam.available() == true) {
    cam.read();
  }
  image(cam, 0, 0);
}
```

Import video library

Print the webcam list

Initialize the `Capture` object

Read the frame whenever it's available

Example: Webcam Capture

```
import processing.video.*;

Capture cam;

void setup() {
  size(640, 480);

  String[] cameras = Capture.list();
  if (cameras.length == 0) {
    println("No cameras available for capture");
    exit();
  }
  cam = new Capture(this, cameras[0]);
  cam.start();
}

void draw() {
  if (cam.available() == true) cam.read();
  image(cam, 0, 0);
}
```

Get the webcam list

Use a specific webcam

Capture.available() vs captureEvent()

```
import processing.video.*;

Capture cam;

void setup() {
  size(640, 480);

  String[] cameras = Capture.list();
  if (cameras.length == 0) {
    println("No cameras available for capture");
    exit();
  }
  cam = new Capture(this, cameras[0]);
  cam.start();
}

void draw() {
  if (cam.available() == true) cam.read();
  image(cam, 0, 0);
}
```

```
import processing.video.*;

Capture cam;

void setup() {
  size(640, 480);

  String[] cameras = Capture.list();
  if (cameras.length == 0) {
    println("No cameras available for capture");
    exit();
  }
  cam = new Capture(this, cameras[0]);
  cam.start();
}

void captureEvent(Capture c) {
  c.read();
}

void draw() {
  image(cam, 0, 0);
}
```

Examples

- Pixelating Capture
- Brightness mirror
- Motion Detection
- Motion Sensor

Example: Mirroring Capture

```
void draw() {
  image(video, 0, 0);
}
```



```
void draw() {
  scale(-1, 1);
  image(video, -video.width, 0);
}
```

Deep Vision Library

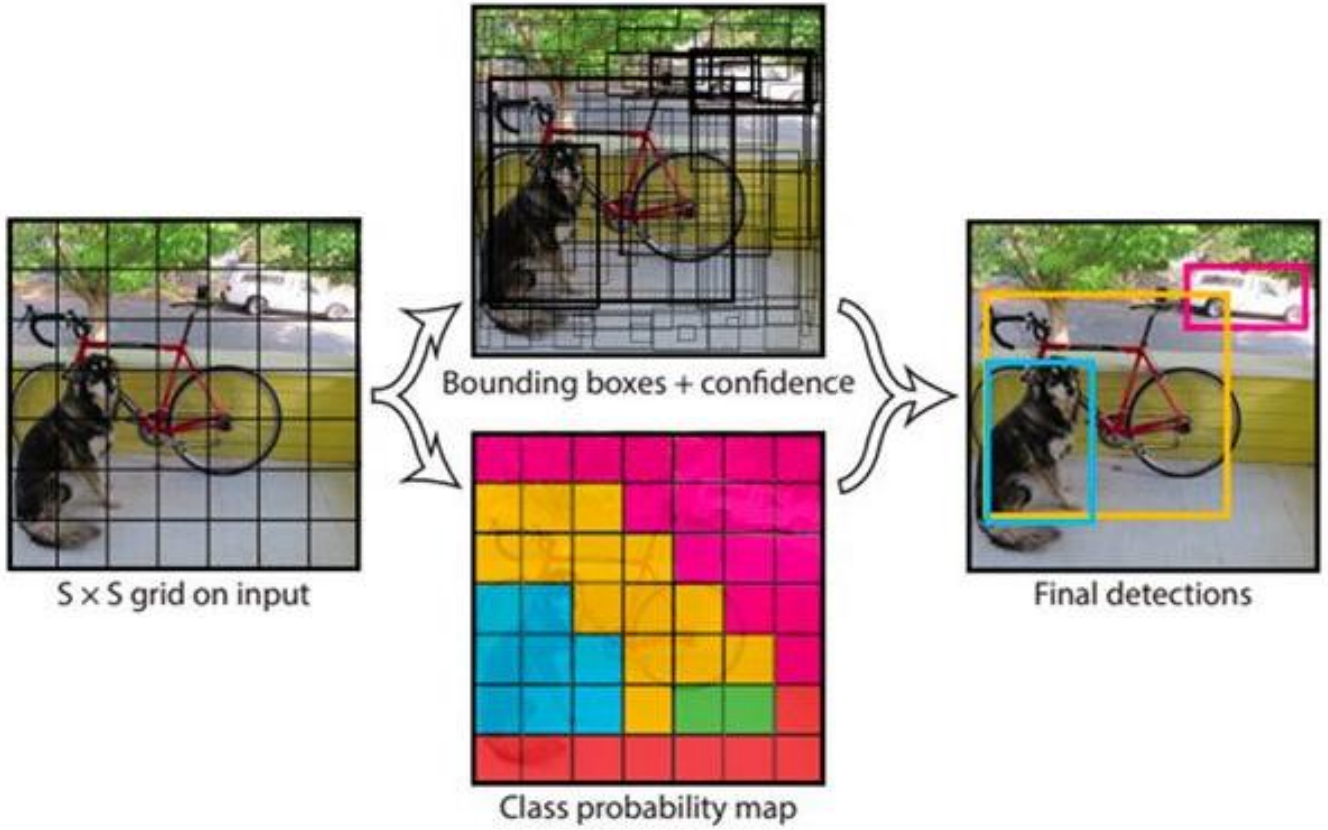
Deep Vision Library

- Deep learning-powered computer vision library that supports
  - Object **detection**
  - Object **recognition**
  - Object **segmentation**
  - **Keypoint detection**
  - **Depth estimation**
  - **Style transfer**
  - **Super-resolution**

github.com/cansik/deep-vision-processing

YOLO

- A deep-learning based **object detector**



superannotate.com/blog/yolo-object-detection

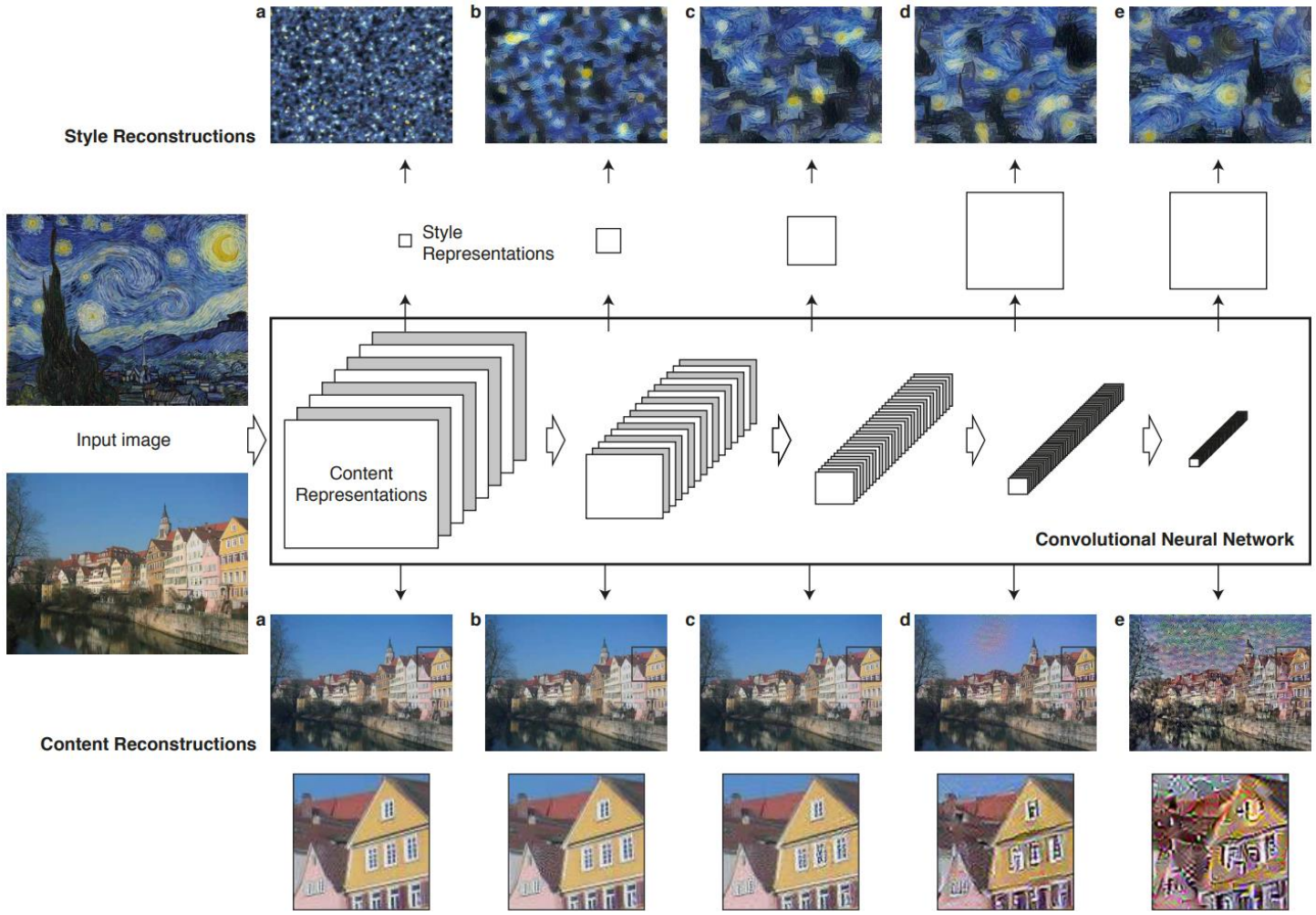
Examples

- YOLO Object Detector
- YOLO Hand Detector

Examples

- Face Detector
- Face & Emotion Detection
- Facial Landmarks Detection

Neural Style Transfer



Leon A. Gatys, Alexander S. Ecker and Matthias Bethge, "Image Style Transfer Using Convolutional Neural Networks," *CVPR*, 2016

Neural Style Transfer – Examples

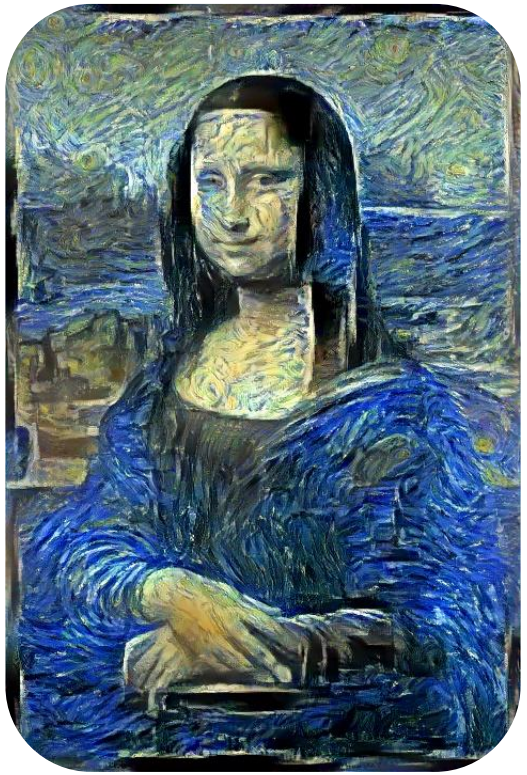
Content

+

Style

→

Output



github.com/PJ-Finlay/Neural-Style-Transfer-Images

Neural Style Transfer – Examples

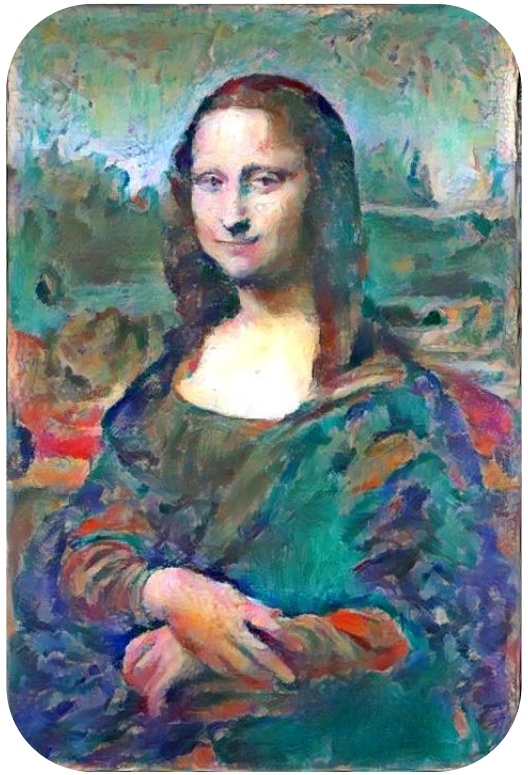
Content

+

Style

→

Output



github.com/PJ-Finlay/Neural-Style-Transfer-Images

Neural Style Transfer – Examples

Content

+

Style

→

Output

github.com/PJ-Finlay/Neural-Style-Transfer-Images

Example

- Neural Style Transfer