

PAT 204/504 (Fall 2026)

Creative Coding

Lecture 8: Images & Videos

Instructor: Hao-Wen (Herman) Dong

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

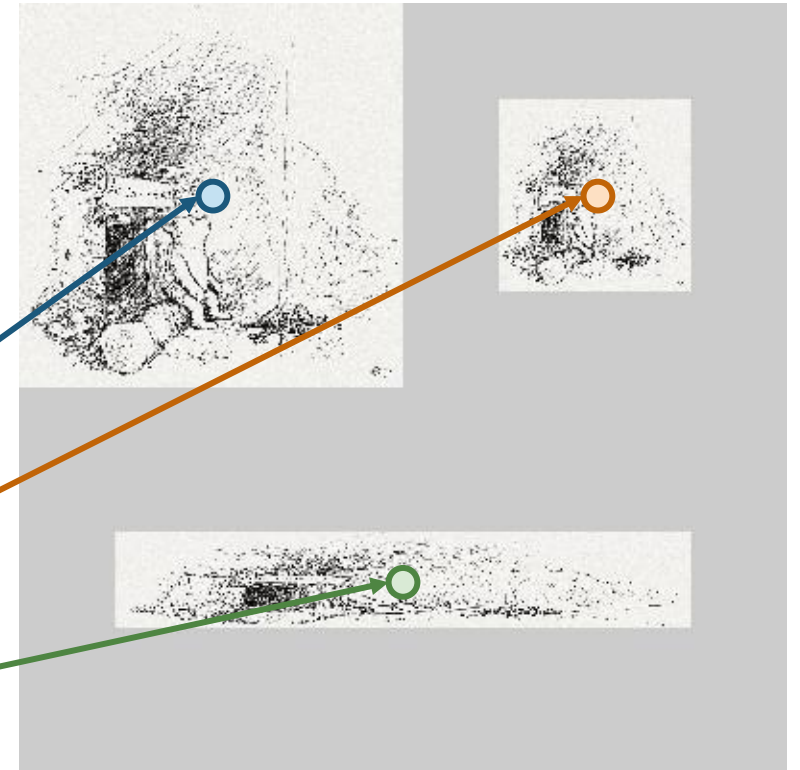
```
PImage img;
```

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

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

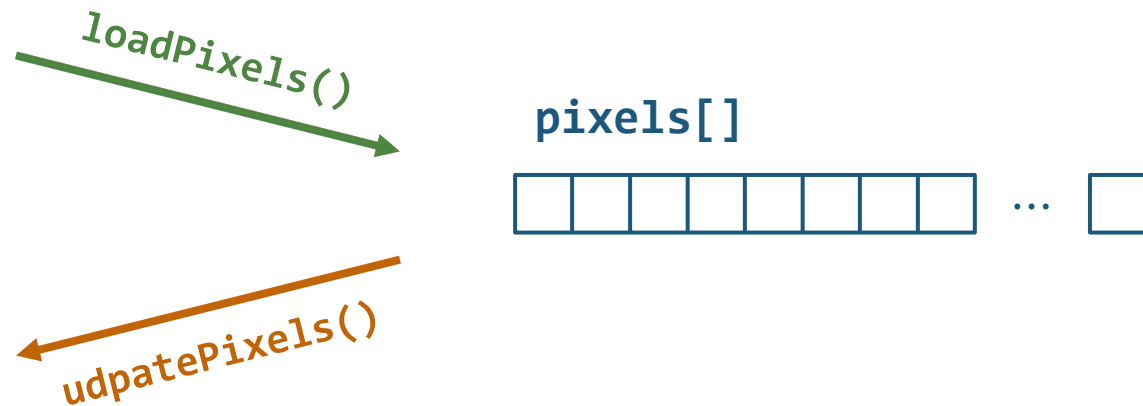
Load the image

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



Loading the Pixels

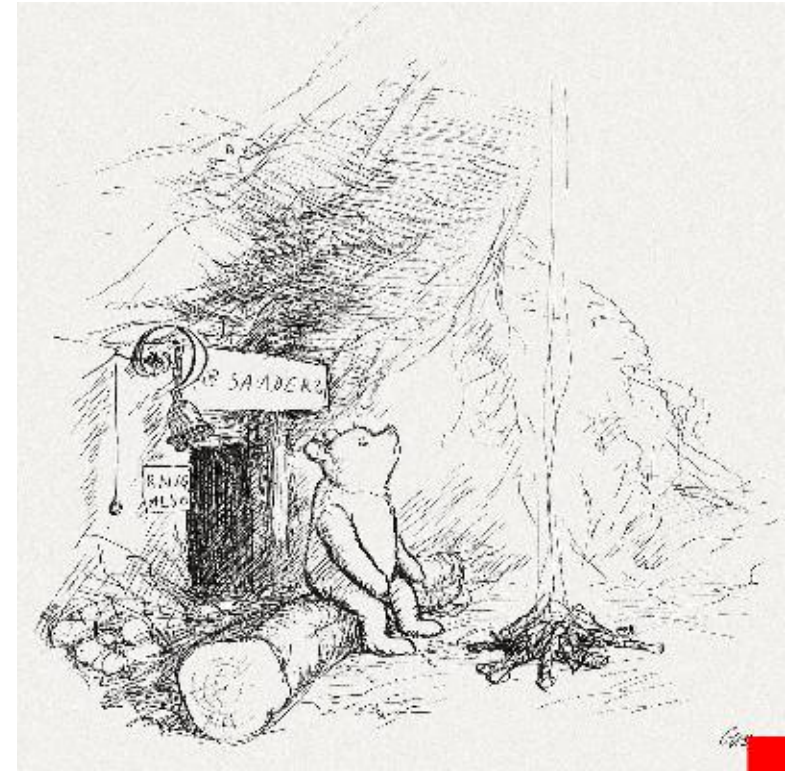
- We can directly interact with the pixels of an image
 - `Image.pixels[]` Array of all the pixels in the image
 - `Image.loadPixels()` Load the image content to `Image.pixels[]`
 - `Image.updatePixels()` Update the image content with `Image.pixels[]`



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"); Load the image  
  img.loadPixels();  
}  
  Load the image content to pixels[]  
  
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;  
    }  
    Update the pixel values  
  }  
  img.updatePixels(); Update the image content with pixels[]  
  image(img, 0, 0, 400, 400);  
}
```



🤔 Exercise: Pointillism

- Repeat for N times
 - Pick a random pixel of the image
 - Find the color of that pixel
 - Draw a circle of that color
- You'll likely need
 - `red(img.pixels[loc])` → to get the red value
 - `green(img.pixels[loc])` → to get the green value
 - `blue(img.pixels[loc])` → to get the blue value



🤔 Exercise: 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;
```

Pick a random pixel

```
img.loadPixels();
```

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

Find the color of the pixel

```
noStroke();
```

```
fill(r, g, b, 100);
```

Set the color of the circle

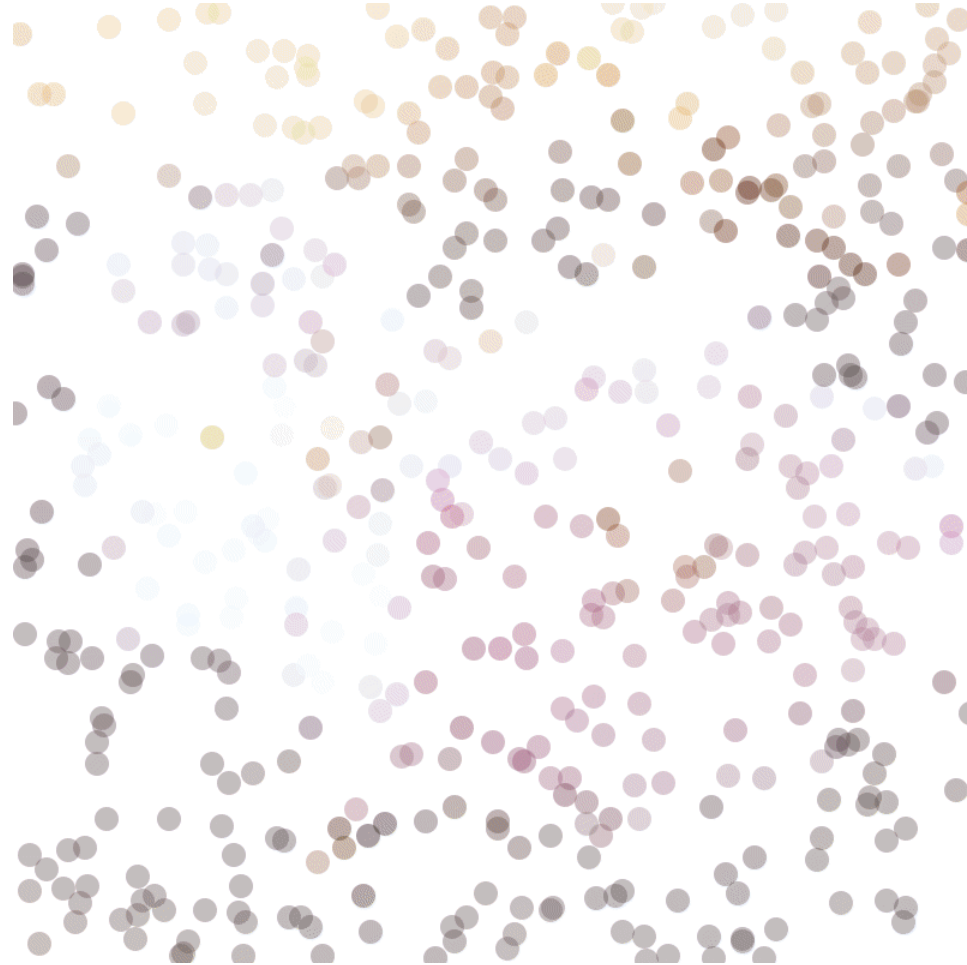
```
circle(x, y, 20);
```

Draw the circle

```
}  
}
```



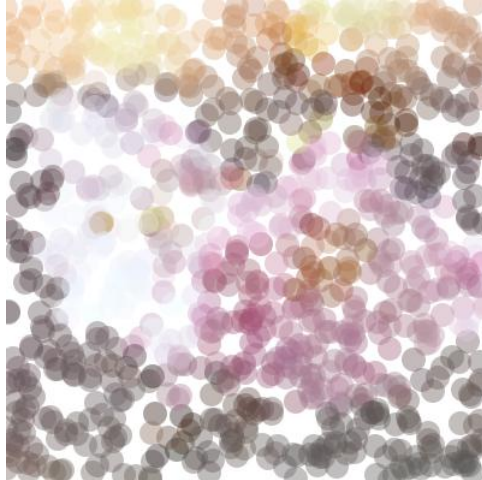
| Pointillism



Pointillism

pointSize = 20

1,000 points



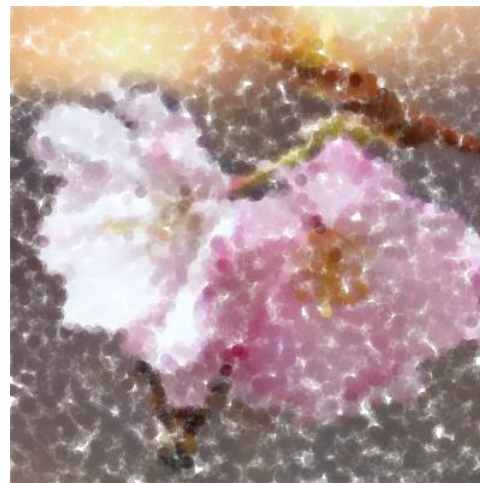
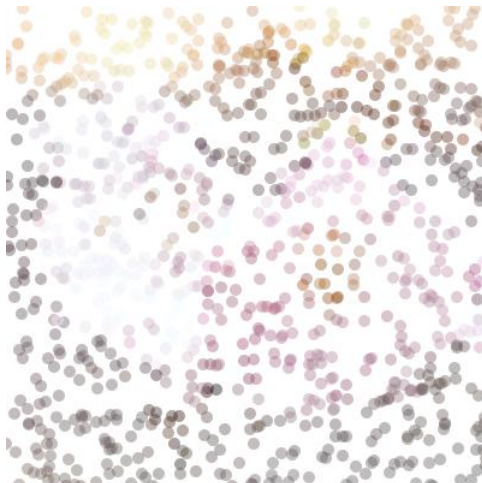
10,000 points



100,000 points

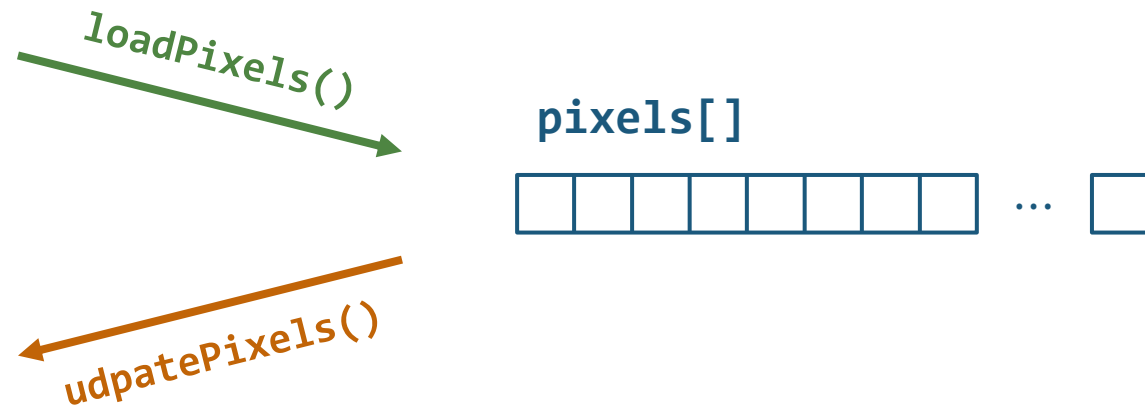


pointSize = 10



The Processing Window

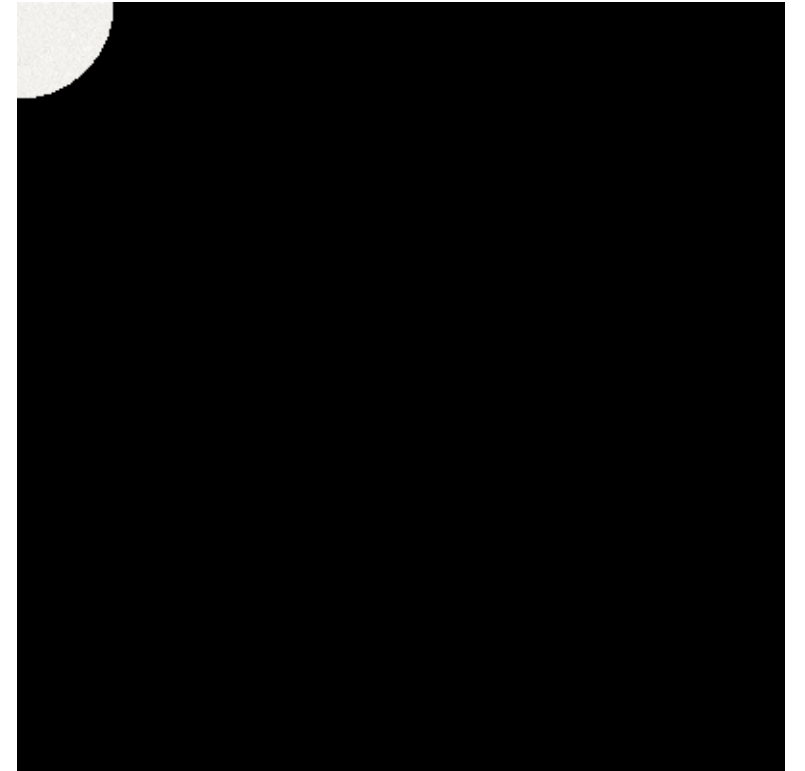
- The processing display window works like an Image object!
 - **pixels[]** Array of all the pixels in the image
 - **loadPixels()** Load the image content to pixels[]
 - **updatePixels()** Update the image content with pixels[]



🤔 Exercise: Reveal Effect

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

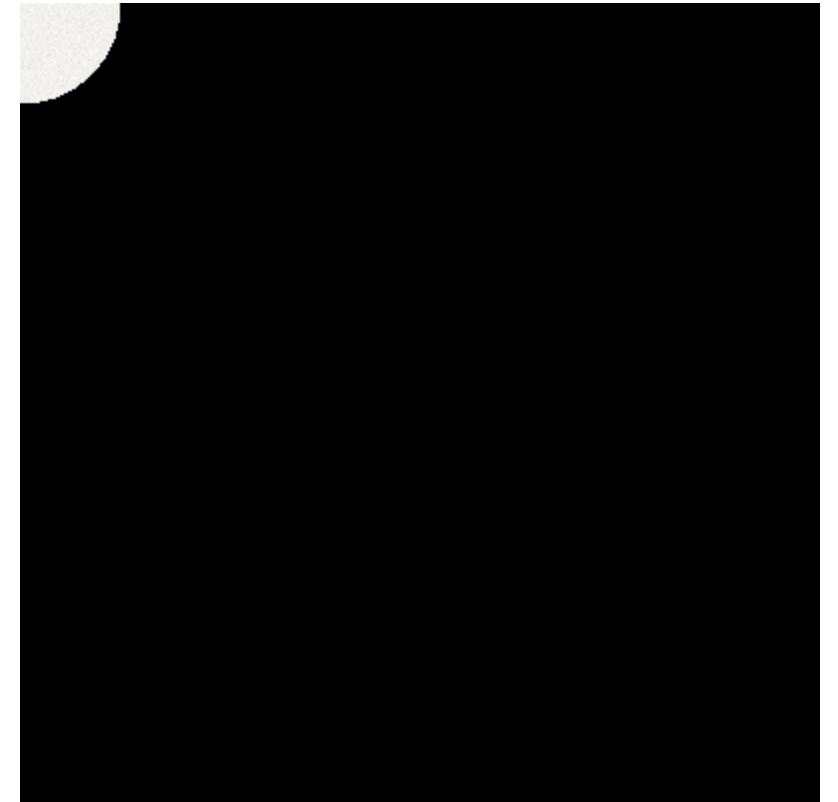
Example image



🤔 Exercise: Reveal Effect

```
PImage img;  
int[] org;  
  
void setup() {  
  size(400, 400);  
  img = loadImage("pooh.jpg"); Load the image  
  image(img, 0, 0, 400, 400);  
  loadPixels(); Load the image content to pixels[]  
  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(); Update the image content with pixels[]  
}
```

Why again?

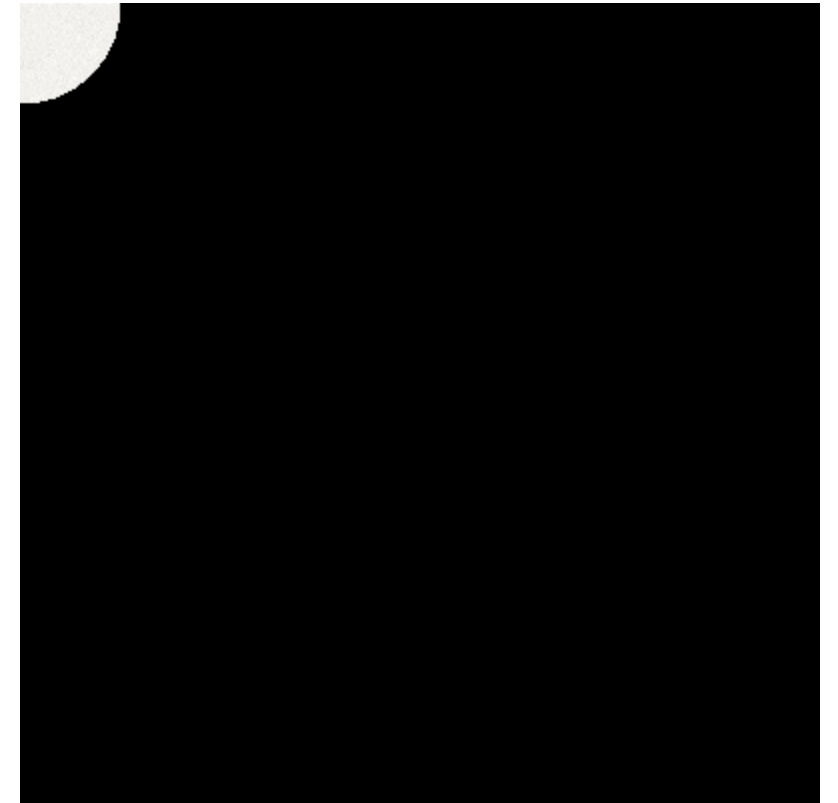


🤔 Exercise: Reveal Effect

```
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

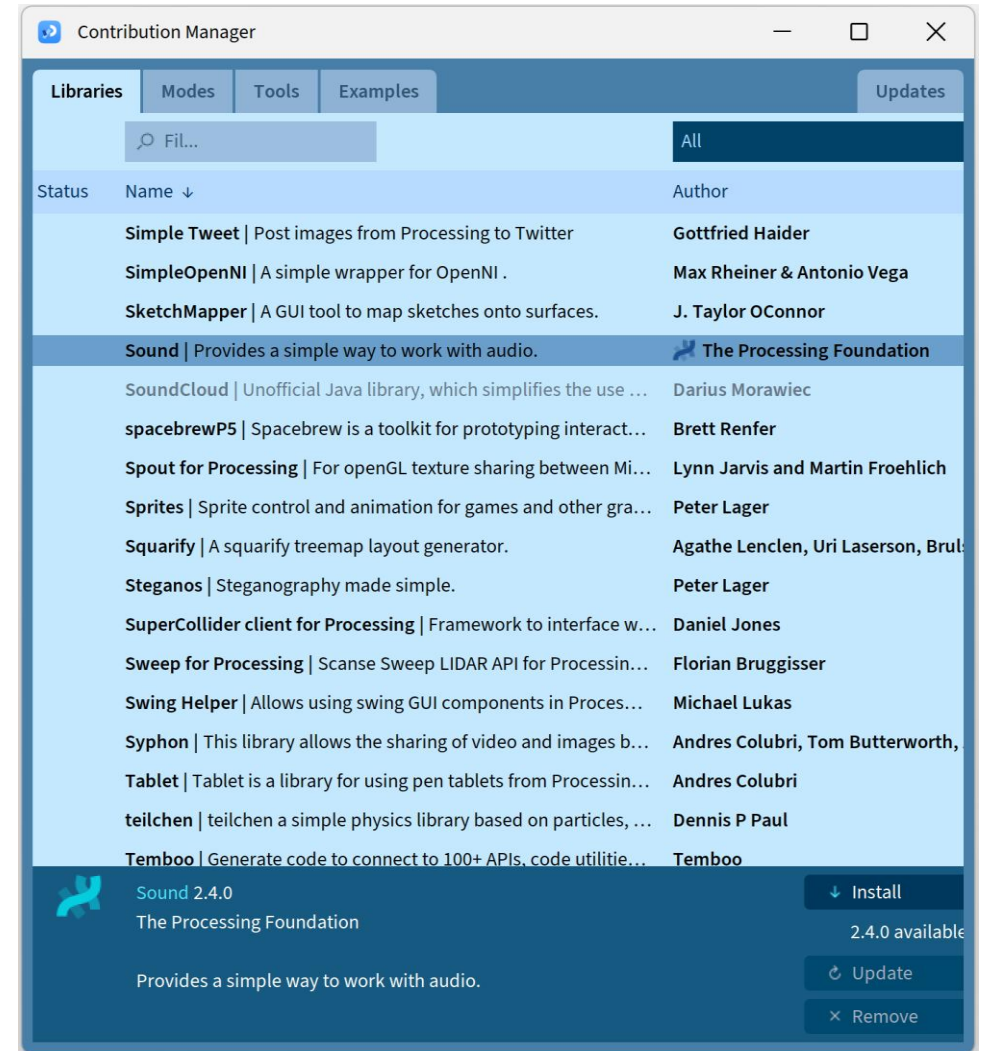
```
int[][] arr = {{0, 1, 2, 3}, {3, 2, 1, 0}, {3, 5, 6, 1}, {3, 8, 3, 4}};
```

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

Videos

Library Manager

- Official Libraries maintained by the **Processing Foundation**
 - Sound
 - **Video**
 - Hardware I/O
 - JavaFX
- Many other third-part libraries
 - Networking
 - GUI
 - Animation
 - DeepVision



Movie Class

- 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.*;
```

Import video library

```
Movie myMovie;
```

```
void setup() {  
  size(640, 360);
```

```
  myMovie = new Movie(this, "movie.mov");  
  myMovie.loop();  
}
```

Initialize the movie object

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

Called whenever a new frame is available to read

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

Capture Class

- 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.*;
```

Import video library

```
Capture cam;
```

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

```
  println(Capture.list());
```

Print the webcam list

```
  cam = new Capture(this, 640, 480);  
  cam.start();
```

Initialize the Capture object

```
}
```

```
void draw() {
```

```
  if (cam.available() == true) {  
    cam.read();  
  }
```

Read the frame
whenever it's available

```
  image(cam, 0, 0);
```

```
}
```

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();  
}
```

Get the webcam list

Use a specific webcam

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

| Mirroring Webcam Capture

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



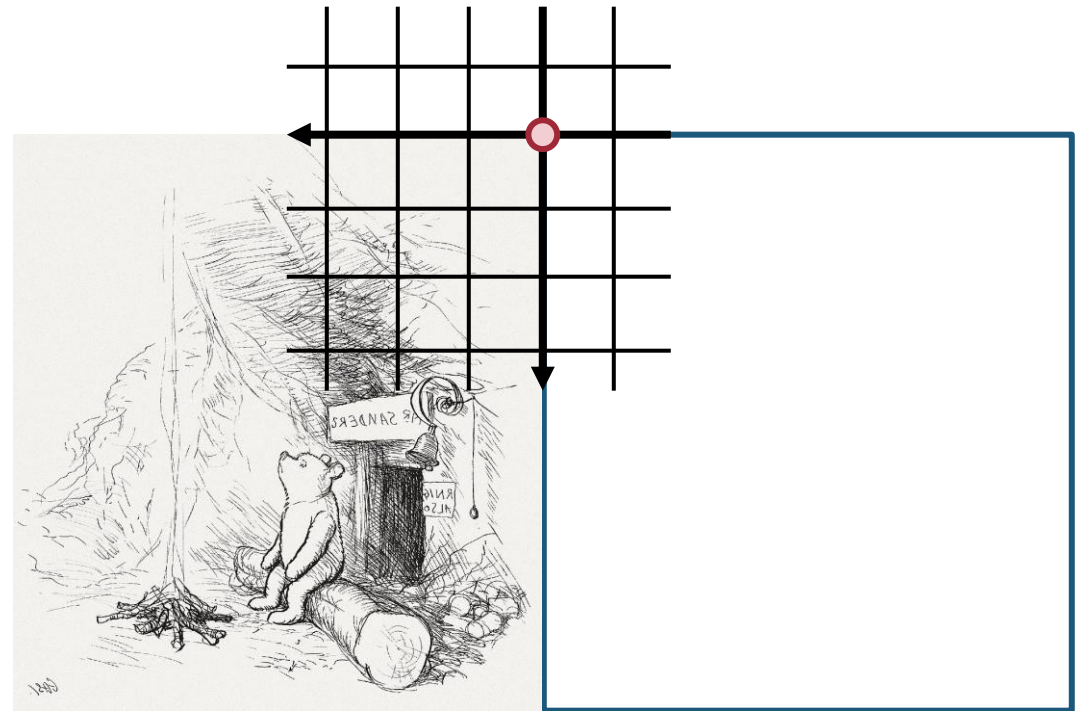
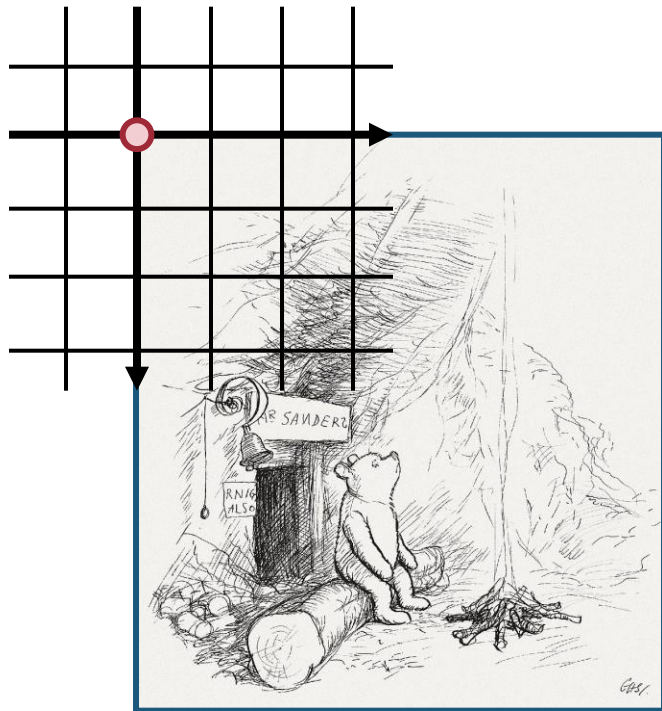
```
void draw() {  
  scale(-1, 1);  
  image(video, -video.width, 0);  
}
```

Mirroring an Image

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



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

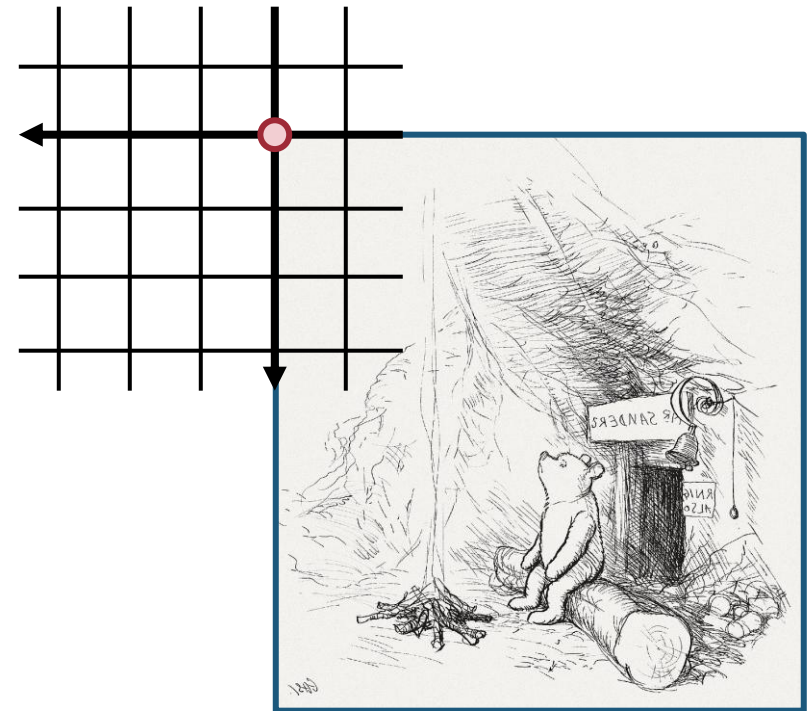
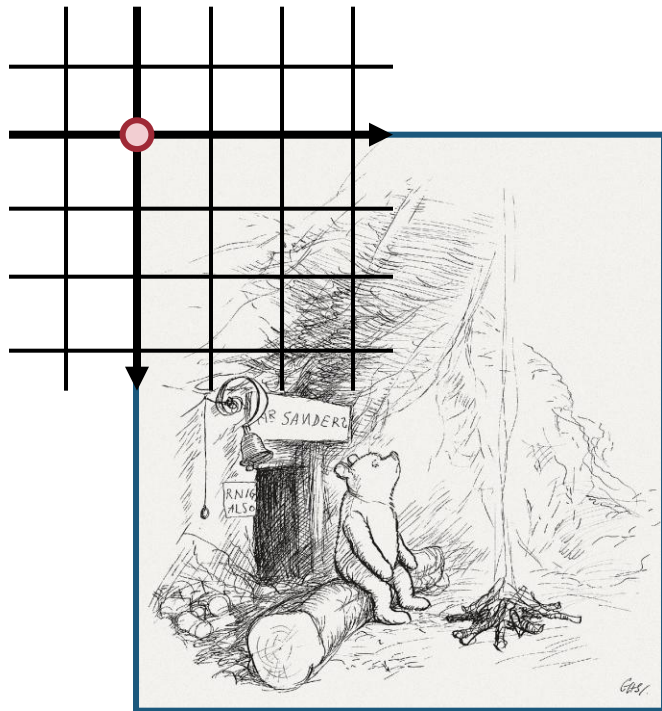


Mirroring an Image

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



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



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);
}
```

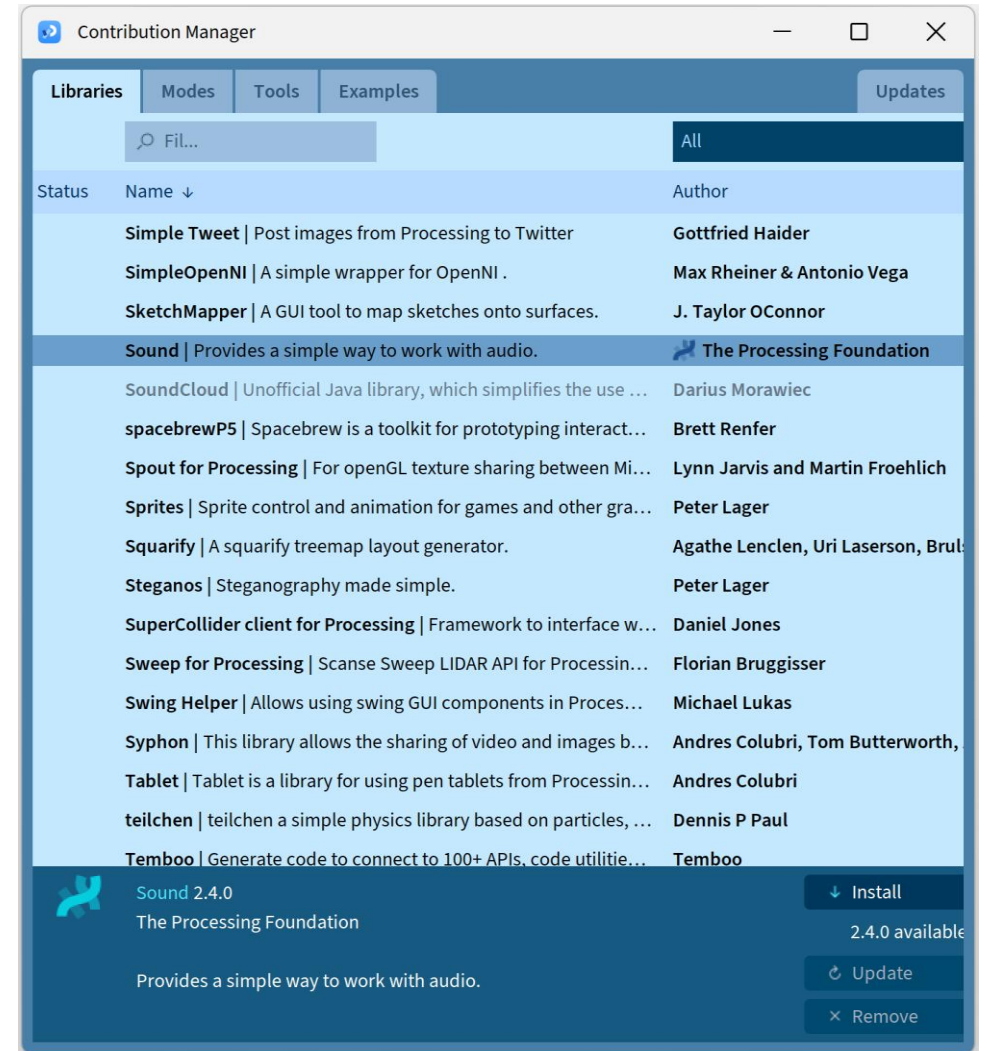
| Examples

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

Deep Vision Library

Library Manager

- Official Libraries maintained by the **Processing Foundation**
 - Sound
 - Video
 - Hardware I/O
 - JavaFX
- Many other third-part libraries
 - Networking
 - GUI
 - Animation
 - **DeepVision**



Deep Vision Library

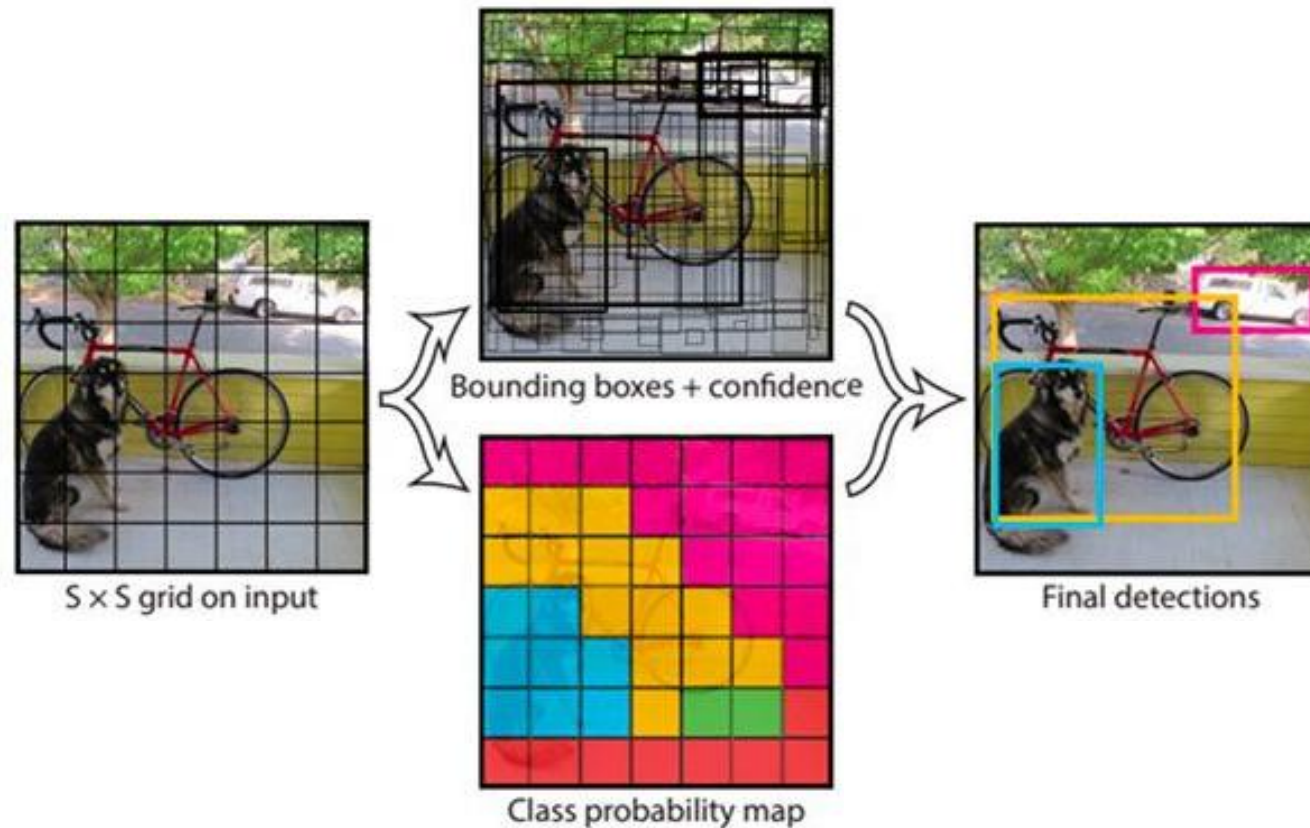
- Deep learning-powered computer vision library that supports
 - Object **detection**
 - Object **recognition**
 - Object **segmentation**
 - **Keypoint detection**
 - **Depth estimation**
 - **Style transfer**
 - **Super-resolution**

[github.com/cansik/
deep-vision-processing](https://github.com/cansik/deep-vision-processing)



Real-Time Object Detection with YOLO

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



(Source: superannotate.com)

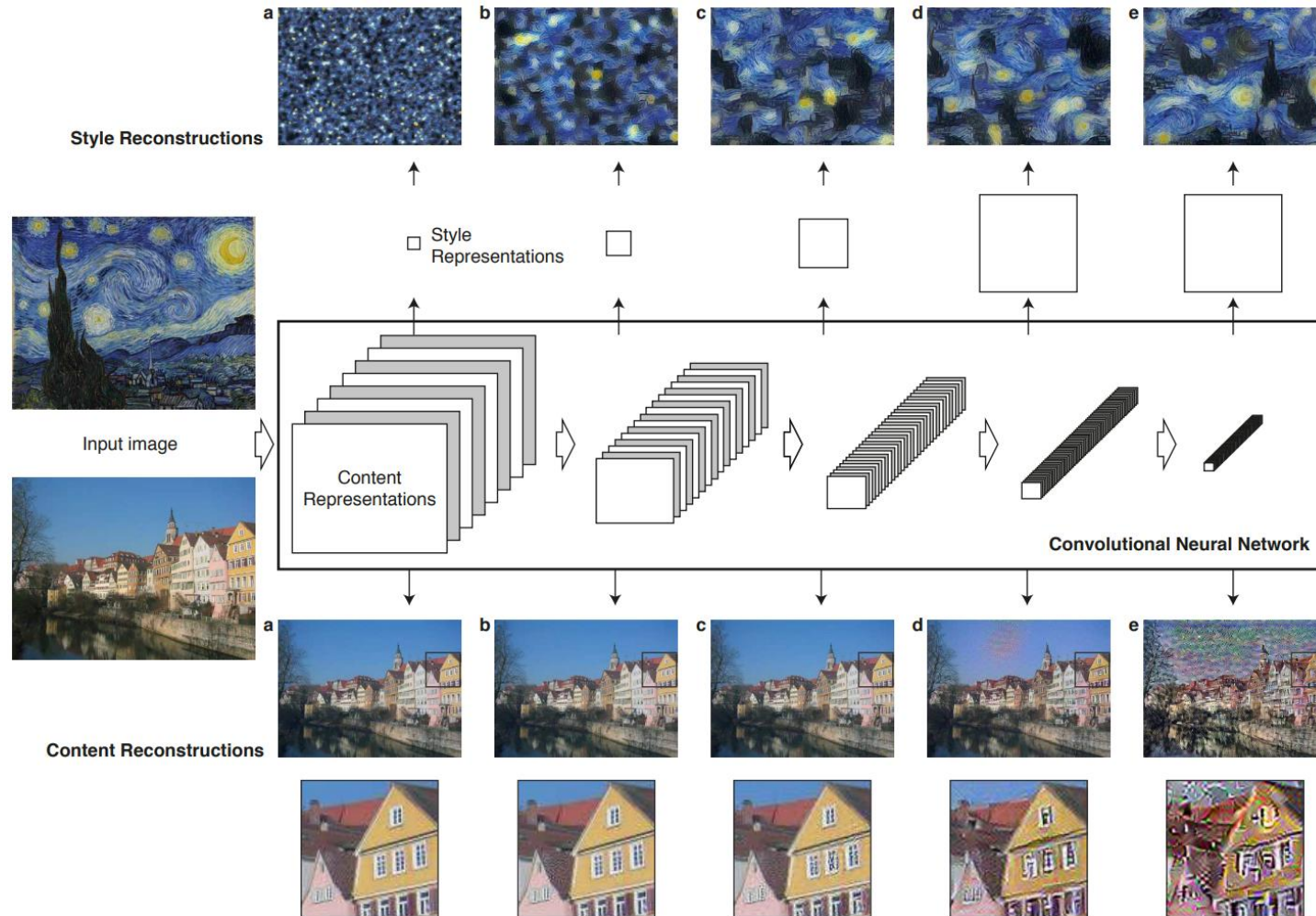
Examples

- YOLO **object** detector
- YOLO **hand** detector
- **Face** detector
- **Face** & **emotion** detection
- **Facial landmarks** detection

[github.com/cansik/
deep-vision-processing](https://github.com/cansik/deep-vision-processing)



Neural Style Transfer



(Source: Gatys et al., 2016)

Neural Style Transfer: Examples

Content

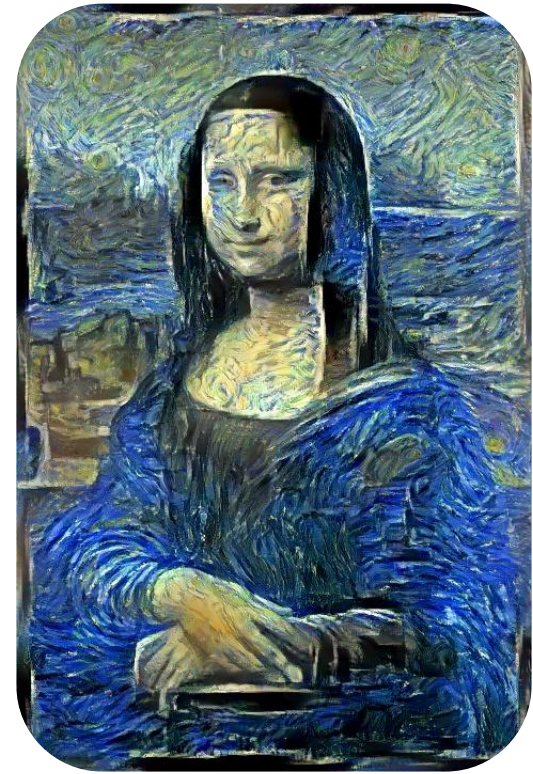


+

Style



Output



Neural Style Transfer: Examples

Content



+

Style



Output



Neural Style Transfer: Examples

Content



+

Style



Output



Recap

Example: Displaying Images

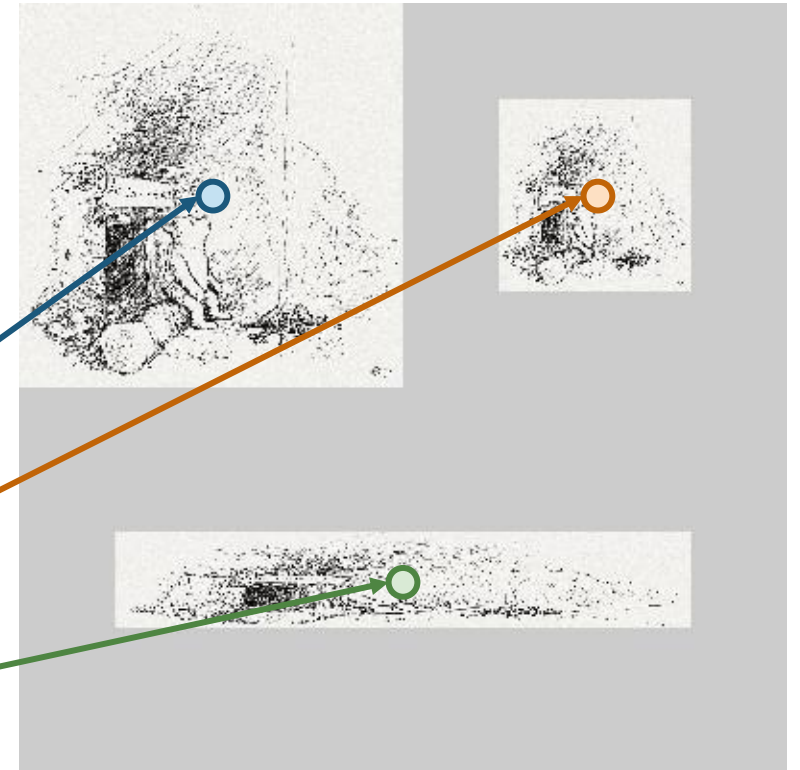
```
PImage img;
```

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

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

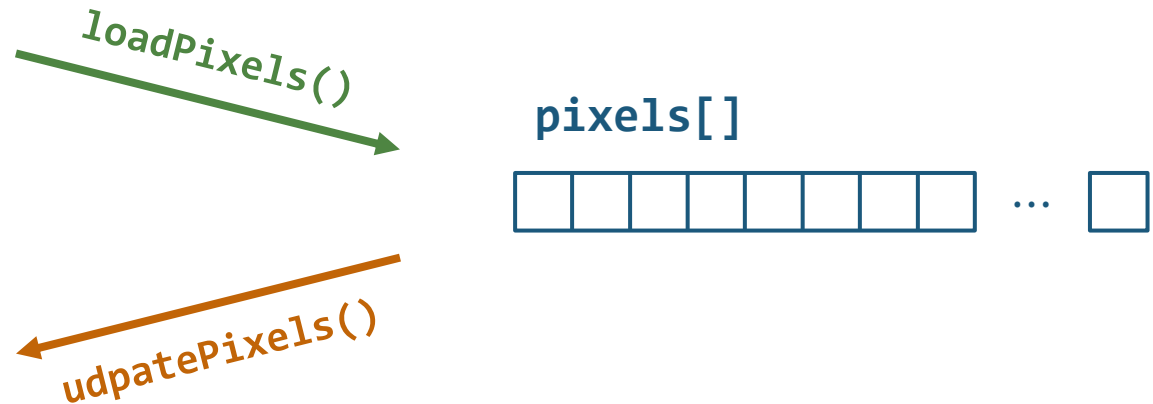
Load the image

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



Loading the Pixels

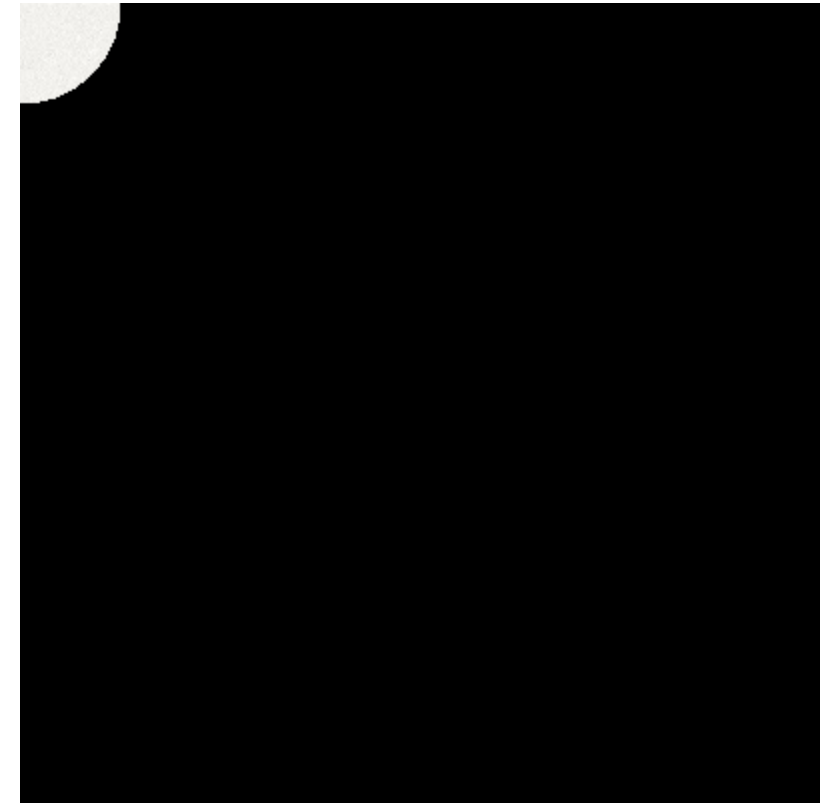
- We can directly interact with the pixels of an image
 - `Image.pixels[]` Array of all the pixels in the image
 - `Image.loadPixels()` Load the image content to `Image.pixels[]`
 - `Image.updatePixels()` Update the image content with `Image.pixels[]`



🤔 Exercise: 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



🤔 Exercise: 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;
```

Pick a random pixel

```
img.loadPixels();
```

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

Find the color of the pixel

```
noStroke();
```

```
fill(r, g, b, 100);
```

Set the color of the circle

```
circle(x, y, 20);
```

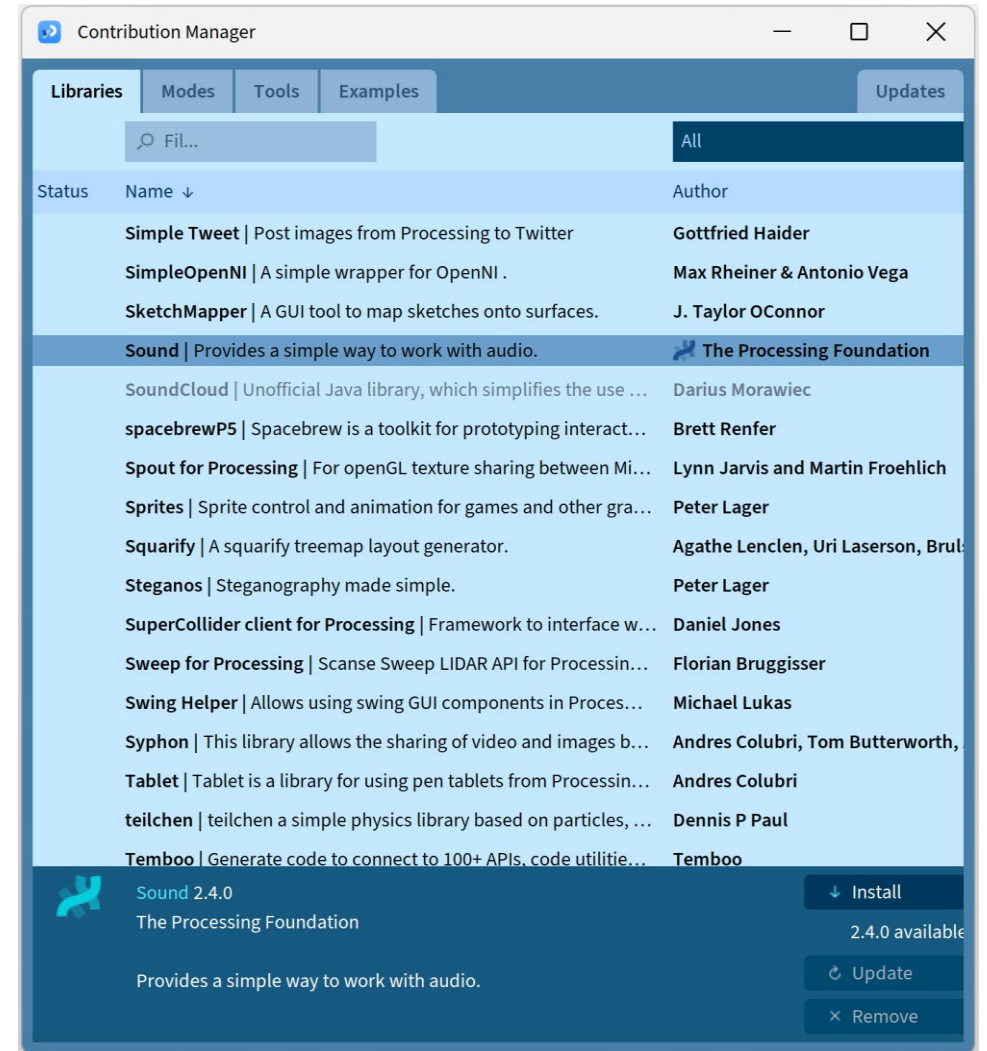
Draw the circle

```
}  
}
```



Library Manager

- Official Libraries maintained by the **Processing Foundation**
 - Sound
 - **Video**
 - Hardware I/O
 - JavaFX
- Many other third-part libraries
 - Networking
 - GUI
 - Animation
 - DeepVision



Example: Webcam Capture

```
import processing.video.*;
```

Import video library

```
Capture cam;
```

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

```
  println(Capture.list());
```

Print the webcam list

```
  cam = new Capture(this, 640, 480);  
  cam.start();
```

Initialize the Capture object

```
}
```

```
void draw() {
```

```
  if (cam.available() == true) {  
    cam.read();  
  }
```

Read the frame
whenever it's available

```
  image(cam, 0, 0);
```

```
}
```

Next Lecture

Sounds

