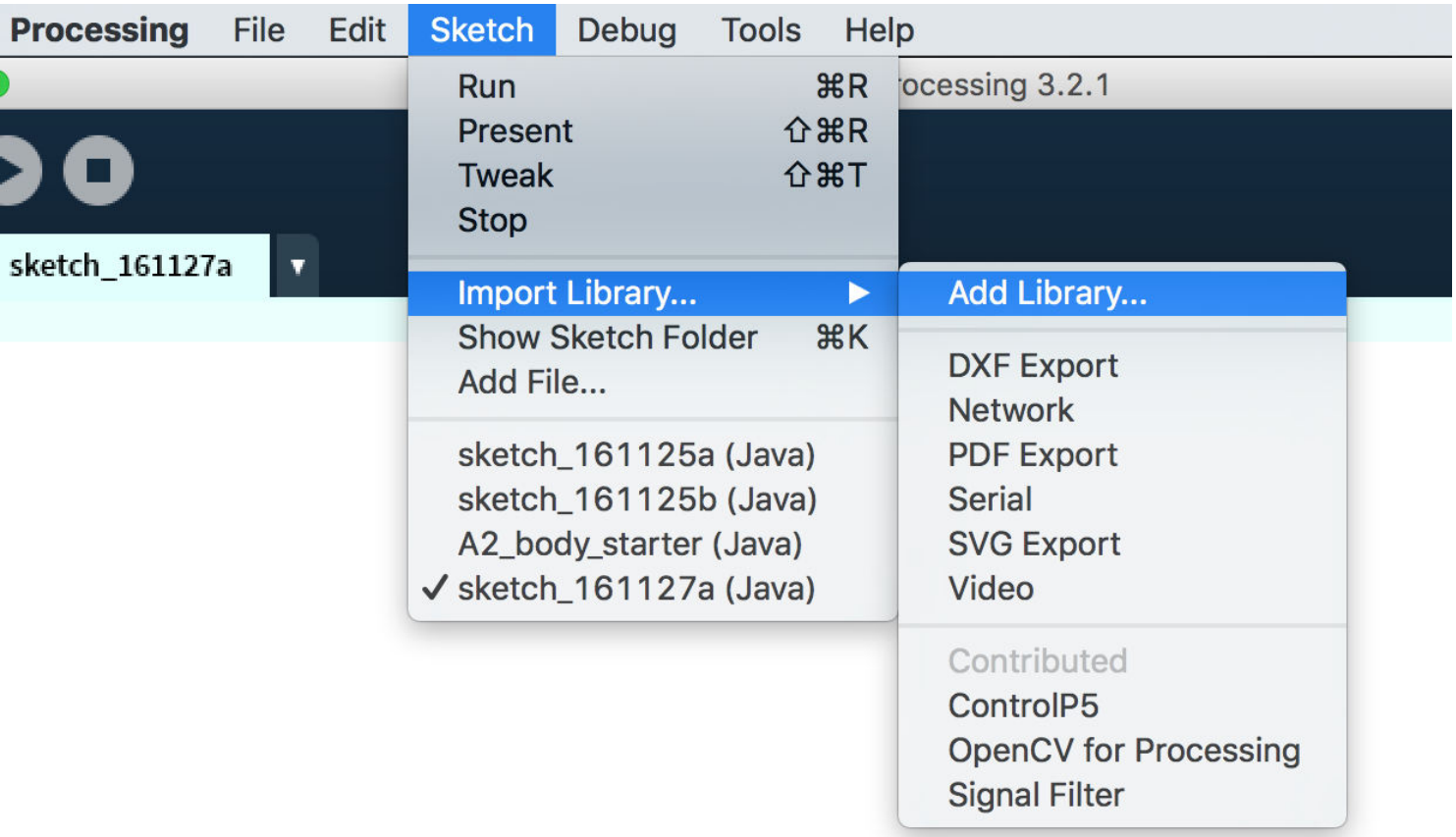
Video and Libraries

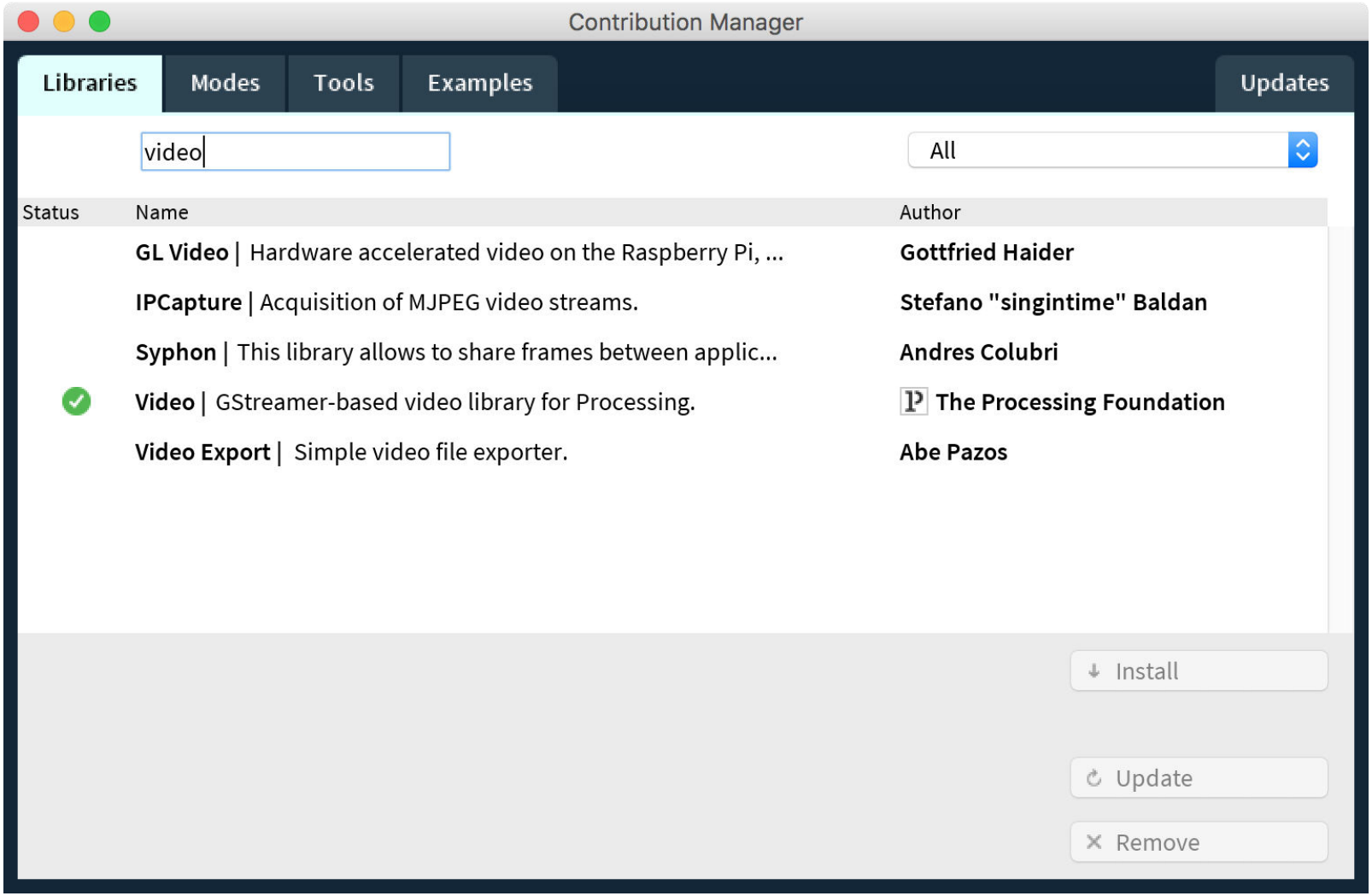
- Libraries (LP Chapter 12)
- Video Processing and Computer Vision (LP Chapter 16)

Libraries

- libraries are extensions to the "core" Processing language
- libraries must be installed (“added”) before using them



“Adding” the Video Library



“Standard” Libraries and Contributed Libraries

- https://processing.org/reference/libraries/

Libraries. Extend Processing beyond graphics and images into audio, video, and communication with other devices.

The following libraries are created by the Processing Foundation. The PDF Export, Network, Serial, and DXF Export libraries are distributed with Procesing. The Video and Sound libraries need to be downloaded through the Library Manager. Select "Add Library..." from the "Import Library..." submenu within the Sketch menu.

PDF Export
Create PDF files. These vector graphics files can be scaled to any size and printed at high resolutions.

Network
Send and receive data over the Internet through simple clients and servers.

Serial
Send data between Processing and external hardware through serial communication (RS-232).

DXF Export
Create DXF files to save geometry for loading into other programs. It works with triangle-based graphics including polygons, boxes, and spheres.

Video
Read images from a camera, play movie files, and create movies.

Sound
Playback audio files, audio input, synthesize sound, and effects.

Hardware I/O
Access peripherals on the Raspberry Pi and other Linux-based computers

Contributions

Contributed Libraries must be downloaded individually. Select "Add Library..." from the "Import Library..." submenu within the Sketch menu. Not all available libraries have been converted to show up in "Add Library...". If a library isn't there, it will need to be installed manually. Follow the How to Install a Contributed Library instructions on the Processing Wiki for more information.

Contributed libraries are developed, documented, and maintained by members of the Processing community. For feedback and support, please post to the Forum. For development discussions post to the Create & Announce Libraries topic. Instructions for creating your own library are on the Processing GitHub site.

3D
Animation
Compilation
Data
GUI
Geometry
Hardware
I/O
Language
Math
Other
Simulation
Sound
Typography
Utilities
Video & Vision

Using a Library

- After a library is installed ("added"), you use it in a sketch by "importing" it

e.g.

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

- Every library has a reference explaining how to use it (some libraries have better references than others)

Using the Video Library to Capture Live Camera

1) Import the video library into your sketch

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

2) Declare a global *Capture* object

```
Capture camera;
```

3) Create the *Capture* object in setup()

```
camera = new Capture(this, 320, 240);
```

4) Start the Capture object

```
camera.start();
```

5) Read a frame of video when the camera is available

```
void captureEvent(Capture cam) {
  cam.read();
}
```

video

```
import processing.video.*;

Capture camera;

void setup() {
  size(320, 240);
  camera = new Capture(this, 320, 240);
  camera.start();
}

void draw() {
  image(camera, 0, 0);
}

void captureEvent(Capture cam) {
  cam.read();
}
```

Uses the "default" camera.

Capturing from other cameras

- Ask Capture what cameras are available

```
// list all available capture ’devices’
// to the console to find your camera.
String[] devices = Capture.list();
for (int i = 0; i < devices.length; i++) {
  println(i, devices[i]);
}
```

- Create a new capture object using a specific camera

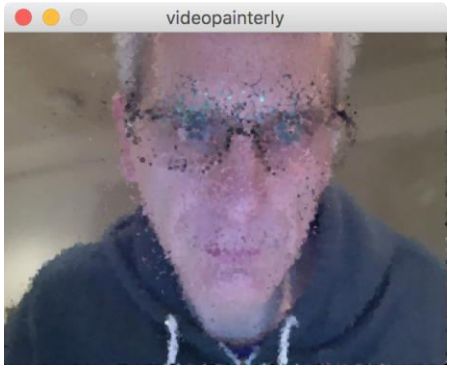
```
// e.g. open camera device ‘3’
camera = new Capture(this, 320, 240, devices[3]);
camera.start();
```

video

```
import processing.video.*;

Capture cam;

void setup() {
  size(320, 240);
  cam = new Capture(this, 320, 240);
  cam.start();
}

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

void captureEvent(Capture cam) {
  cam.read();
}
```

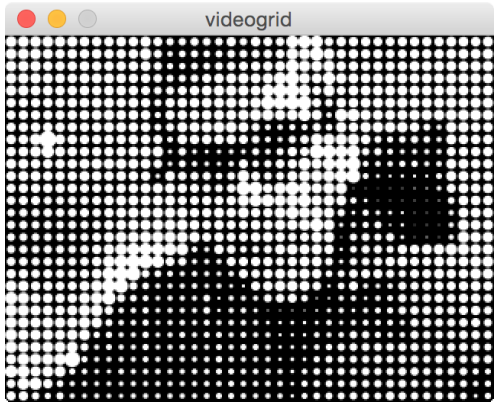
videopainterly

```
using .get(x, y) with video frame

  for (int i = 0; i < 250; i++) {
    int x = int(random(0, width - 1));
    int y = int(random(0, height - 1));

    color pixelColour = cam.get(x, y);
    fill(pixelColour, 200);

    int size = int(random(1, 5));
    ellipse(x, y, size, size);
  }
```



videogrid

```
// loop through all grid positions
  for (float x = maxSize / 2; x < width; x += maxSize) {
    for (float y = maxSize / 2; y < height; y += maxSize) {

      // get the colour at the corresponding pixel
      int vx = int(map(x, 0, width - 1, 0, cam.width - 1));
      int vy = int(map(y, 0, height - 1, 0, cam.height - 1));
      color pixelColour = cam.get(vx, vy);

      // get brightness and convert it to a size
      float b = brightness(pixelColour);
      float s = map(b, 0, 255, 0, maxSize * 1.5);

      fill(255);
      ellipse(x, y, s, s);
    }
  }
```



videopixels

```
if (cam.available()) {

  cam.read();
  image(cam, 0, 0);

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

       int i = x + y * width;

       color pixelColour = pixels[i];
       float r = red(pixelColour);
       float g = green(pixelColour);
       float b = blue(pixelColour);

       // image processing here
       pixels[i] = color(r, g, b);
    }
  }
  updatePixels();
}
```

Computer Vision

- "Computer vision" refers to a broad class of algorithms that allow computers to make intelligent assertions about digital images and video (Levin, 2006)

- Computers that can "see"
- Using the camera as a "sensor"

CREDIT: United States Postal Service

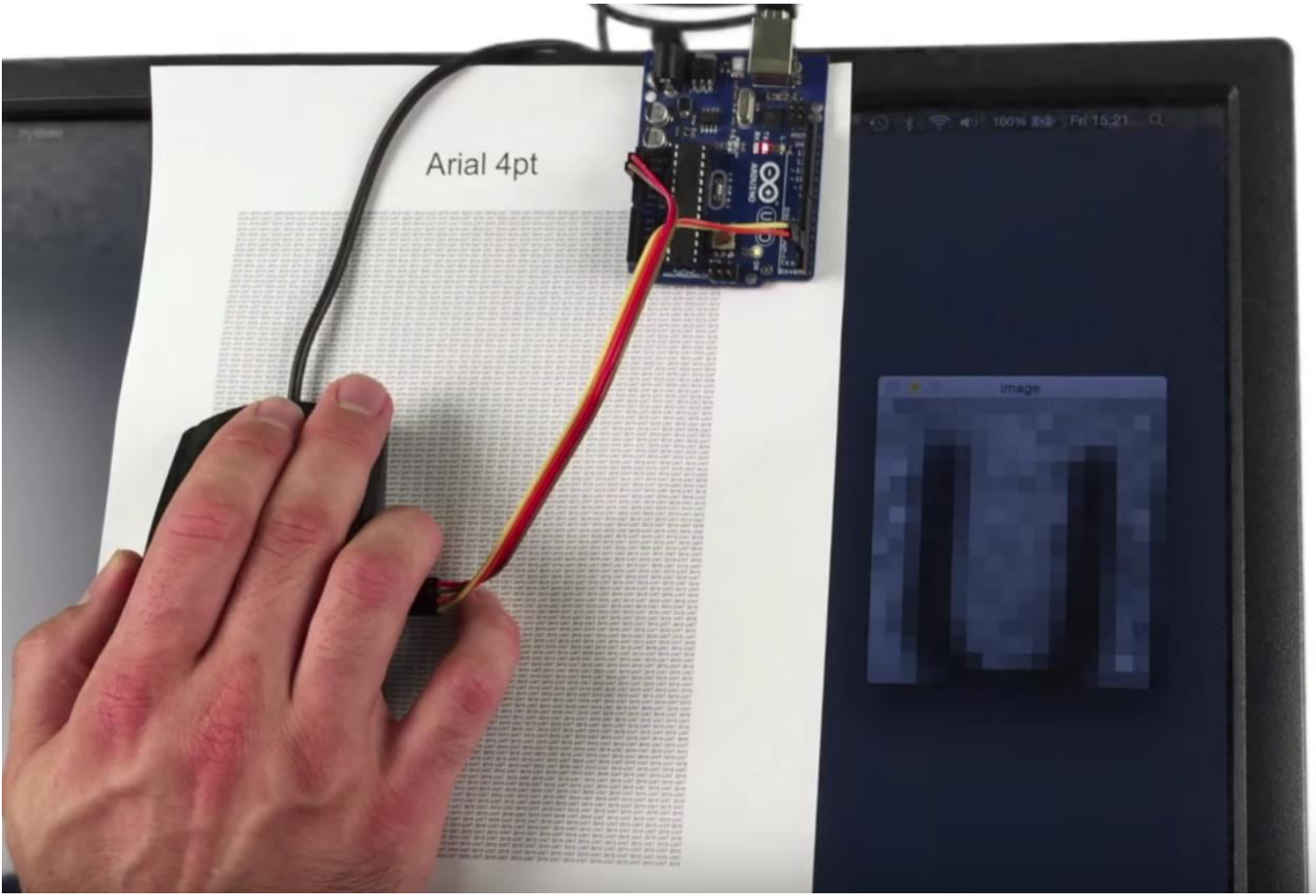
80322-4129 80206

40004 64310

33502 75346

35460 44209

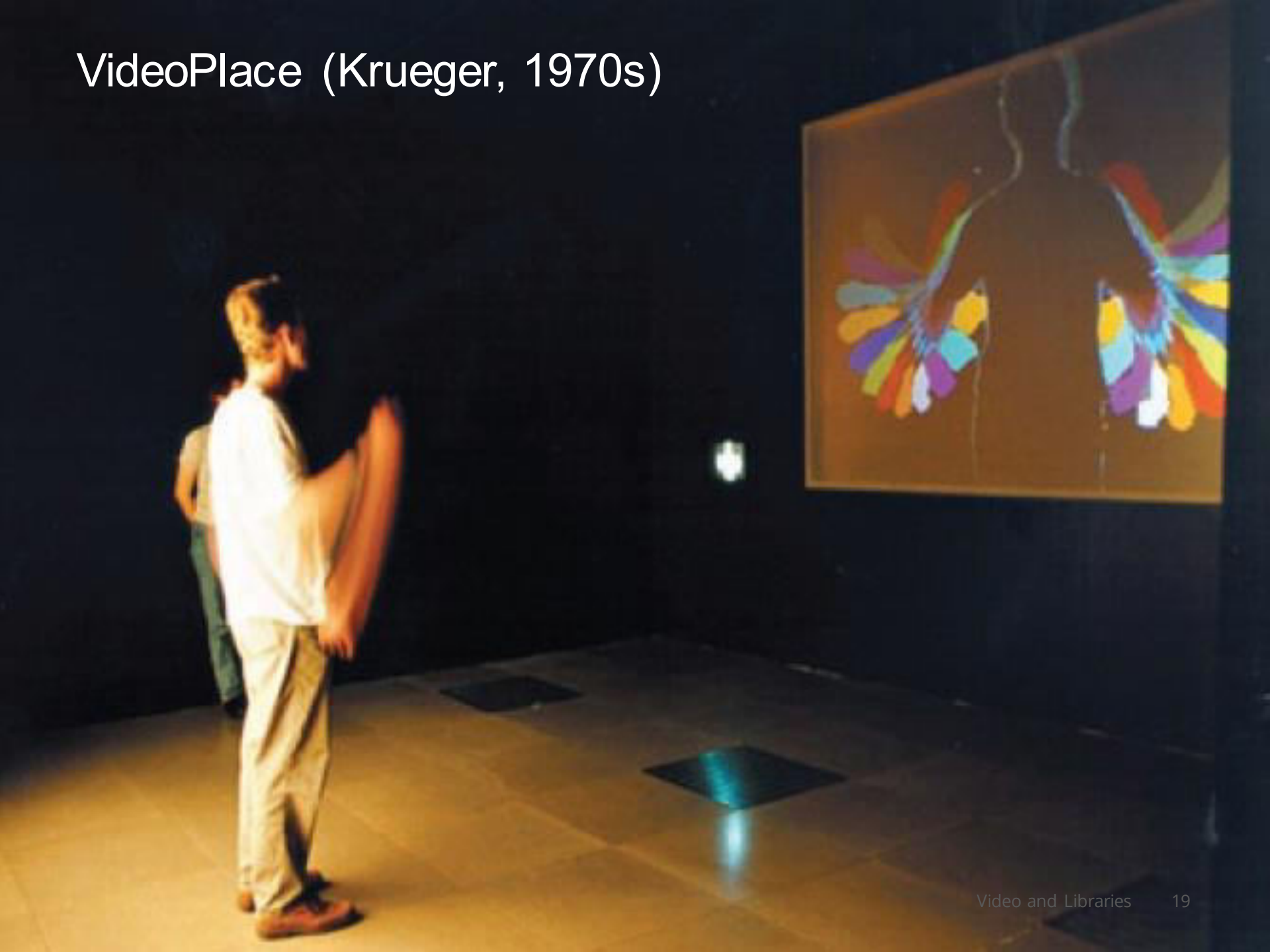
What a Mouse "Sees"



from A Simple Method for Measuring End-to-end Latency using an Optical Mouse (Casiez et al. 2015)
https://youtu.be/XB_mxGTgWvQ?t=46s

The Kinect Effect
https://youtu.be/oq98_35sQko

VideoPlace (Krueger, 1970s)



VIDEOPLACE (Krueger, 1985)

- http://youtu.be/d4DUIeXSEpk

VIDEOPLACE Mini-documentary (1988)

- https://youtu.be/dmmxVA5xhuo?t=4m5s

A Simple Computer Vision Algorithm

```
for each video frame:
    identify a "special" pixel
    based on some criteria
    use that pixel's location
    to control the computer
```

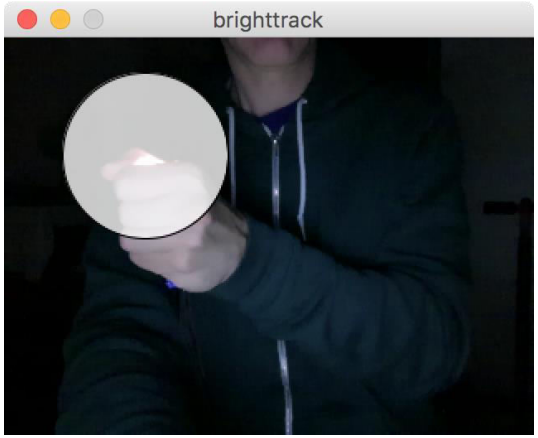
- This is just a special kind of array operation, very similar to finding the largest element in an array ...

Array Operation: Find Largest Element Index

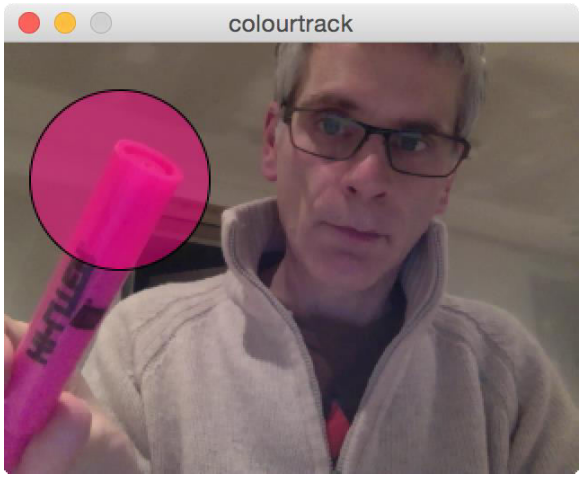
```
// find the index of the largest element
// (assuming myArray has at least 1 element)
int indexOfLargest = 0;
for (int i = 1; i < myArray.length; i++) {
  if (myArray[i] > myArray[indexOfLargest]) {
    indexOfLargest  = i;
  }
}
println("index of largest:", indexOfLargest);
println("largest:", myArray[indexOfLargest]);
```

brighttrack

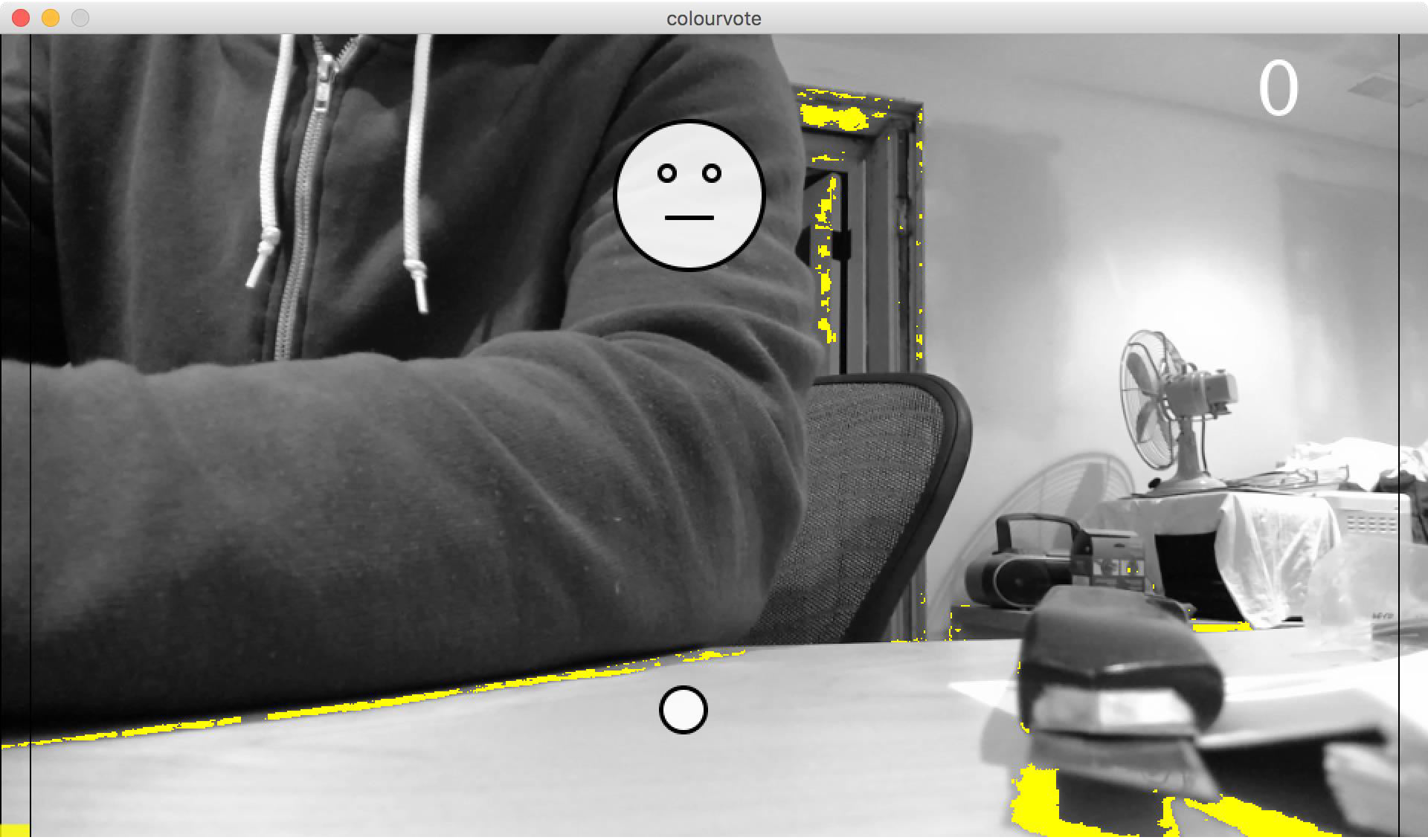
```
int brightestX = 0;
int brightestY = 0;
float brightest = 0;

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

    int i = x + y * width;
    float b = brightness(pixels[i]);

    if (b > brightest && b > 200) {
      brightest = b;
      brightestX = x;
      brightestY = y;
    }
  }
}
```

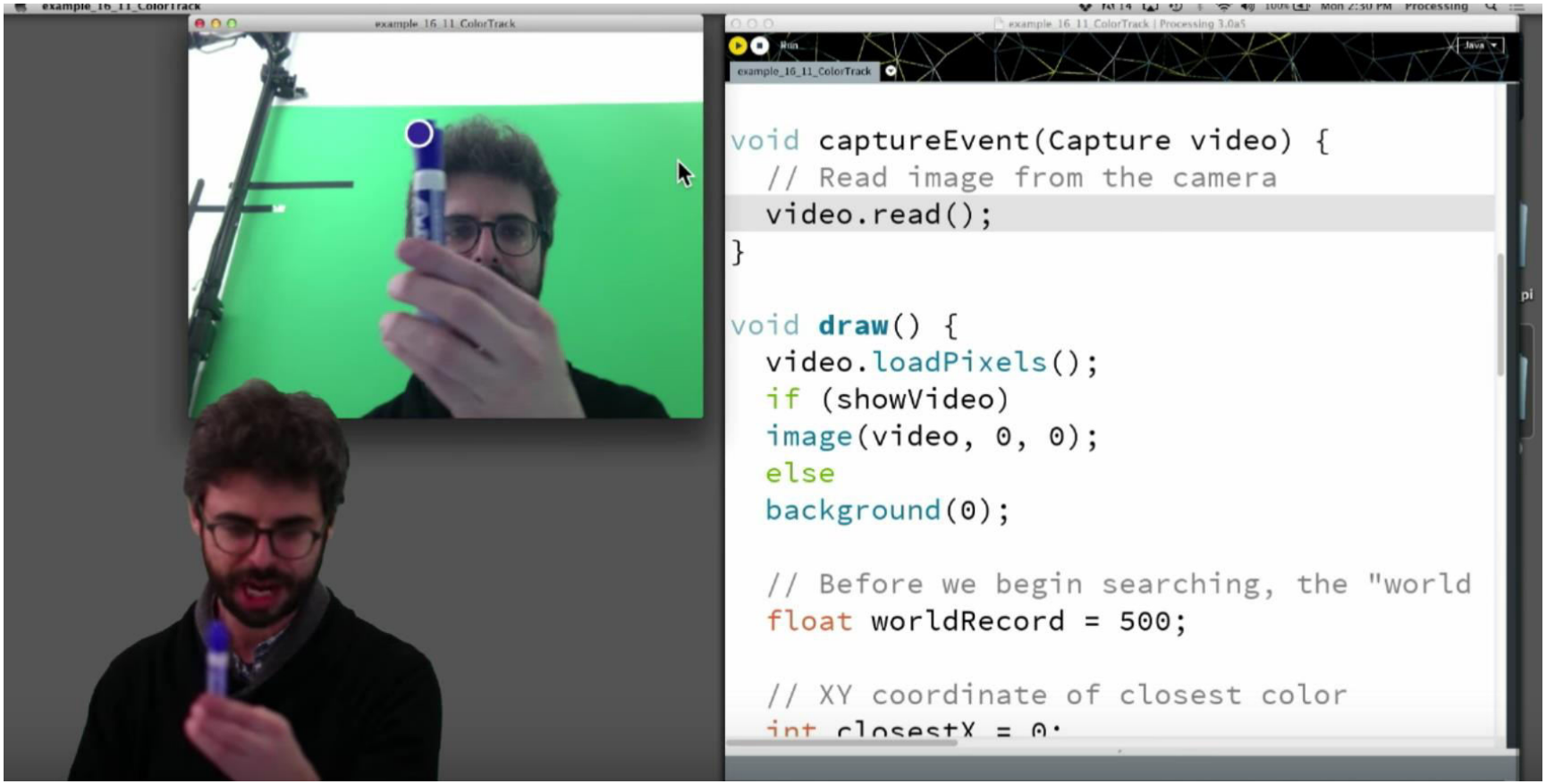


colourtrack




```
int closestX = 0;
int closestY = 0;
float closestDist = 360;

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

    int i = x + y * width;

    float b = brightness(pixels[i]);
    float s = saturation(pixels[i]);

    float h = hue(pixels[i]);
    float d = abs(trackHue - h);

    if (d < closestDist && b > 50 && s > 50) {
      closestDist = d;
      closestX = x;
      closestY = y;
    }
  }
}
```


colourvote
0



Introduction to Computer Vision

https://youtu.be/h8tk0hmWB44

Excellent Introduction

- Levin, G. "Computer Vision for Artists and Designers: Pedagogic Tools and Techniques for Novice Programmers". *Journal of Artificial Intelligence and Society*, Vol. 20.4. Springer Verlag, 2006.
  - http://www.flong.com/texts/essays/essay_cvad/

Computer Vision for Artists and Designers: Pedagogic Tools and Techniques for Novice Programmers

This article also appears in the following publications:

- Levin, G. "Computer Vision for Artists and Designers: Pedagogic Tools and Techniques for Novice Programmers". *Journal of Artificial Intelligence and Society*, Vol. 20.4. Springer Verlag, 2006.
- Reas, Casey and Fry, Ben. *Processing: A Programming Handbook for Visual Designers and Artists*. MIT Press, 9/2007. ISBN: 978-026218262.

ABSTRACT

"Computer vision" refers to a broad class of algorithms that allow computers to make intelligent assertions about digital images and video. Historically, the creation of computer vision systems has been regarded as the exclusive domain of expert researchers and engineers in the fields of signal processing and artificial intelligence. Likewise, the scope of application development for computer vision technologies, perhaps constrained by conventional structures for research funding, has generally been limited to military and law-enforcement purposes. Recently, however, improvements in software development tools for student programmers and interactive-media artists — in

Using the Video Library to Play Movies

0) Add a video ("movie") to your sketch's data folder.

1) Import the video library into your sketch

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

2) Declare a global *Movie* object

```
Movie mov;
```

3) Create the *Movie* object in setup()

```
mov = new Movie(this, "flyboard.mp4");
```

4) Start playback

```
mov.play();
```

5) Read a frame of video when the camera is available

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

movie

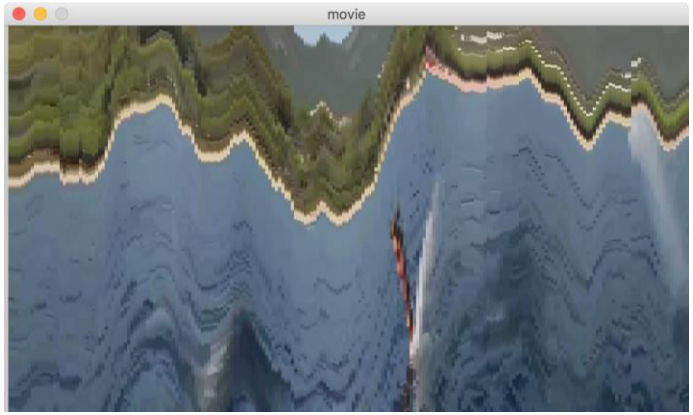

```
import processing.video.*;
Movie mov;

void setup() {
  size(640, 360);
  mov = new Movie(this, "flyboard.mp4");
  mov.play();
}

void draw() {
  image(mov, 0, 0);
}

// Called every time a new frame is available to read
void movieEvent(Movie m) {
  m.read();
}
```

Detect when movie is done playing

```
void draw() {
  image(mov, 0, 0);

  if (mov.time() >= mov.duration()) {
     background(255, 0, 0);
  }
}
```

movie (slit demo)

```
int x = 0;

void draw() {
  image(mov, x, 0);
}

void movieEvent(Movie m) {
  m.read();
  x += 3;
}
```



moviescratch


```
void draw() {
  if (mov.available()) {
    mov.read();
    // A new time position is calculated
    // using the current mouse location:
    float f = map(mouseX, 0, width, 0, 1);
    float t = mov.duration() * f;
    mov.play();
    mov.jump(t);
    mov.pause();
  }
  image(mov, 0, 0);
}
```

Movie Methods

- see reference

Methods	frameRate()	Sets the target frame rate
	speed()	Sets the relative playback speed
	duration()	Returns length of movie in seconds
	time()	Returns location of playback head in units of seconds
	jump()	Jumps to a specific location
	available()	Returns "true" when a new movie frame is available to read.
	play()	Plays movie one time and stops at the last frame
	loop()	Plays a movie continuously, restarting it when it's over.
	noLoop()	Stops the movie from looping
	pause()	Pauses the movie
	stop()	Stops the movie
	read()	Reads the current frame

Using the Sound Library to Play Sounds

0) Add a sound file to your sketch's data folder.

1) Import the sound library into your sketch

```
import processing.sound.*
```

2) Declare a global *Sound* object

```
Sound honk;
```

3) Create the *Sound* object in setup()

```
honk = new Sound(this, "honk.wav");
```

4) Play sound when you want

```
honk.play();
```

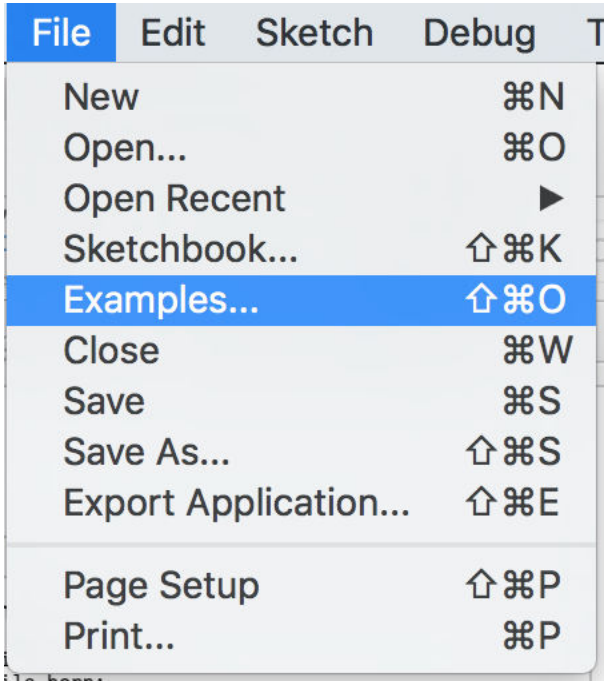
sound

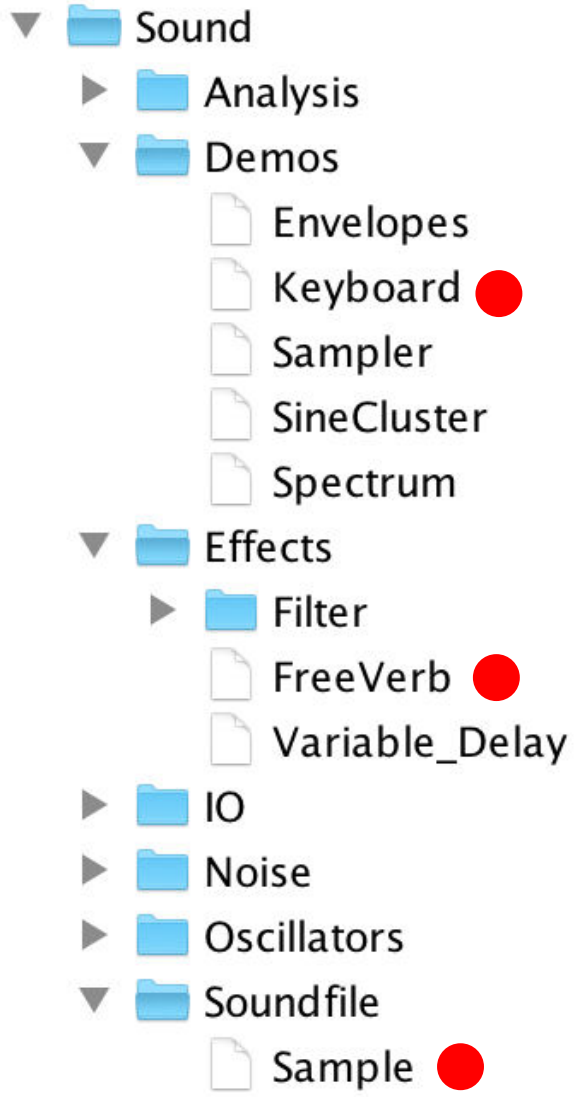
```
import processing.sound.*;

SoundFile sound;

void setup() {
    // make sure file is in data sketch directory
    sound = new SoundFile(this, "honk.wav");
}

void mousePressed() {
   sound.play();
}
```

sounds

```
SoundFile honk;
SoundFile horn;

void setup() {
  // make sure file is in data sketch directory
  honk = new SoundFile(this, "honk.wav");
  horn = new SoundFile(this, "horn.wav");
}

void mousePressed() {
  if (mouseX < 50) {
    honk.play();
  } else {
    horn.play();
  }
}
```

Sound Examples





More Libraries

- There are 100s of Processing libraries
  - https://processing.org/reference/libraries/

3D
Animation
Compilation
Data
GUI
Geometry
Hardware
I/O
Language
Math
Other
Simulation
Sound
Typography
Utilities
Video & Vision

- Some of my favourites
  - **Ani** for animating variables
  - **Signal Filter** to filter noisy values

evaluate

- https://evaluate.uwaterloo.ca/
- Prof. Kevin Harrigan
  - **questions about the instructor are about me**
    not your lab TAs, ISAs or other course staff
  - **try to evaluate me and CS 105 independently**