EECS 1710
Programming for Digital Media

Lecture 21 :: Moving Images and Video

YORK UNIVERSITÉ UNIVERSITY

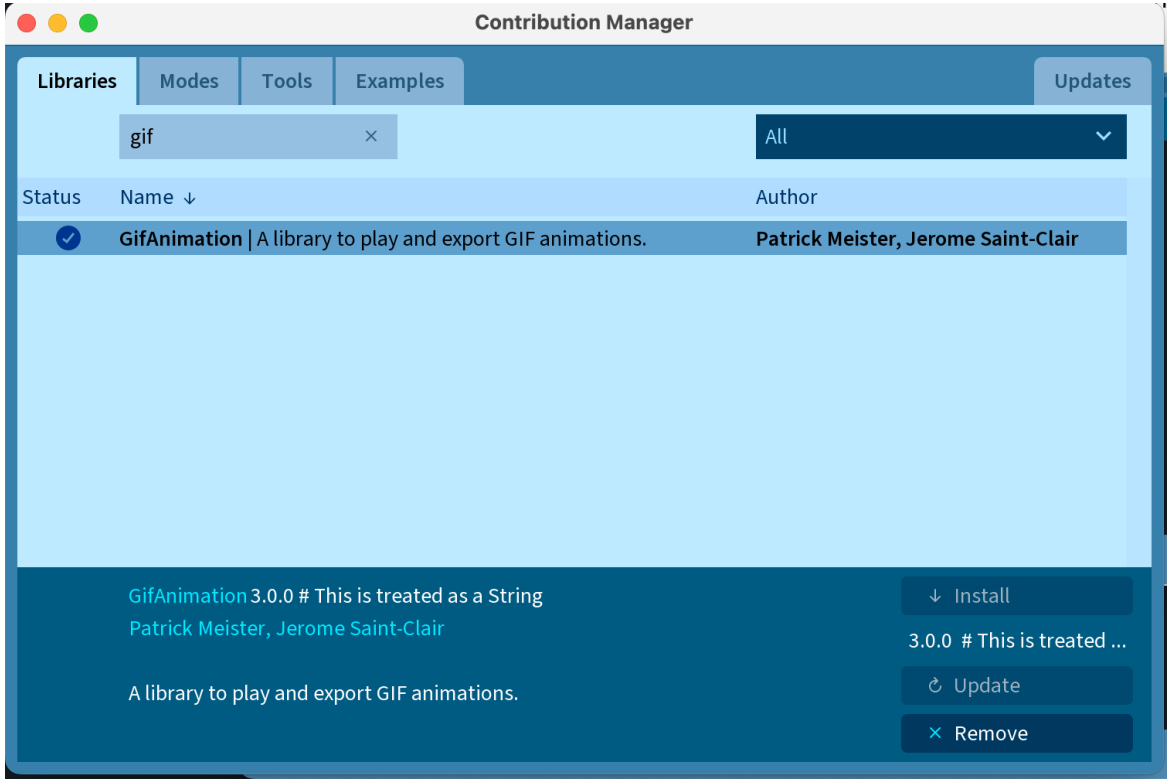
Moving/Changing Images

- *Recall → utilize draw loop*
  - *animating*
    - *modify a graphic (clear and redraw)*
    - *modify an image (all or some pixels)*

- *Capturing & saving frames (as `Pimage[]` or `ArrayList<>`)*
  - *exporting frames as Gif files*

- **Working with moving images & video**

Using the Gif Animation Library (external lib)



YORK UNIVERSITÉ UNIVERSITY

Using the Gif Animation Library (external lib)

```
import gifAnimation.*;

// TWO MAIN OBJECTS

Gif myAnimation;          // use this for loading in and running
                          // an existing gif file


GifMaker gifExport;       // use this to setup and create a gif
                          // from your app window graphics
```

- Documentation/library hosted on github (as are many projects):
  - https://github.com/extrapixel/gif-animation

YORK UNIVERSITÉ UNIVERSITY

Gif Animation (documentation)

- Documentation:
  - https://github.com/extrapixel/gif-animation

GifMaker class:

void setTransparent(int color)

void setTransparent(int red, int green, int blue)

void setTransparent(float red, float green, float blue)

Sets the transparent color of the GIF file. Unlike other image formats that support alpha (e.g. PNG), GIF does not support semi-transparent pixels. The way to achieve transparency is to set a color that will be transparent when rendering the GIF. So, if you set the transparent color to black, the black pixels in your gif file will be transparent.

void setQuality(int qualtiy)

Sets the quality of the color quantization process. GIF only supports 256 indexed colors per frame. So, the colors that come in your images need to be reduced to a set of 256 colors. The quality of this process can be set using this method (or by instantiating the GifMaker object with the respective constructor). Default is 10 and seems to produce good results. Higher qualities will cause the quantization process to be more expensive in terms of cpu-usage.

void setSize(int width, int height)

Sets the size of the new GIF file. If this method is not invoked, the image dimensions of the first added frame will be the size of the GIF.

void setRepeat(int count)

Sets the repeat setting in the GIF file. GIF renderers like web browsers should respect this setting and loop the animation that many times before stopping. Default is 1. 0 means endless loop.

3 ways to add frames

void addFrame()

Adds the current sketch window content as a new gif frame.

void addFrame(PImage image)

Pass a PImage to add it as a new gif frame

void addFrame(int[] pixelArray, int width, int height)

Pass a int pixel array and the width and height to add it as a new gif frame.

void setDelay(int ms)

Sets the delay (the "framerate") for the most recently added frame. This is measured in Milliseconds. This can be different for every frame. Note, this effects the playback speed of the resulting GIF-file only. So, the speed / framerate with which you wrote the frames has no effect on play- back speed.

void setDispose(int mode)

Sets the disposal mode for the current frame. Disposal modes are a special concept used in the GIF file format. It basically determines whether a frame will be overriden by the next frame, or if the next frame should be added, layed over the last frame. For convenience there are constants for the different disposal modes:

Dispose mode	
GifMaker.DISPOSE_NOTHING	Nothing special
GifMaker.DISPOSE_KEEP	retain the current image
GifMaker.DISPOSE_RESTORE_BACKGROUND	restore the background color
GifMaker.DISPOSE_REMOVE	restore the background color

boolean finish()

Finishes GIF recording and saves the GIF file to the given file name in the sketch folder. Returns true if saving the file was successful, false if not.

```
// MAKING A GIF FROM FRAMES (instead of storing/saving)


// INSIDE setup() or method where you want to create the gif

gifExport = new GifMaker(this, "export.gif");

gifExport.setRepeat(0);       // parameter = count
                              // (num times to loop before stopping;
                              // 0 = endless)


gifExport.setTransparent(0,0,0);  // sets which colour to set
                                  // to transparent.. e.g. black



// INSIDE draw() or method where you want to build the gif
gifExport.setDelay(1);         // sets time between frames
gifExport.addFrame();          // record app window as a frame
```

Hit Target example: saving stored frames (record up until hit target then make gif)

```
// inside draw()
if (frameCount%6 == 0) {           // at 60fps, save every 6th (i.e. 10fps)
    PImage thisFrame = get();

    if (animation.size()<100)
      animation.add(thisFrame);    // add if haven't gotten to 100 frames
    else {
      animation.remove(0);         // remove first frame before adding frame
      animation.add(thisFrame);    // (keeps only last 100 frames)
    }
  }
```

Saving from stored frames (hit target e.g.)

```
// inside moveParticle (when hit detected),
// call method to generate gif from saved frames
if (hitTarget(x, y)) {
    hit = true;
    effect.play();
    writeFramesToGif();
}
```

```
void writeFramesToGif() {

  gifExport = new GifMaker(this, "export.gif");

  gifExport.setRepeat(1);
  gifExport.setDelay(100);                          // 100ms apart

  for (int i=0; i<animation.size(); i++) {
    gifExport.addFrame(animation.get(i));
  }

  gifExport.finish();

}
```

More examples: image manipulation vs time

- Simple: Fading images (in/out)
- Simple: Adding noise to images (making grainy/speckly)
- Harder: Warping images

```
float fadeFactor;
float fadeValue = 255;

void imageFade(PImage img) {

  background(BLACK);

  rWidth = frameCount%FPS/FPS * orig.width;
  rHeight = frameCount%FPS/FPS * orig.height;

  if (fade)
    fadeFactor = 0.95;
  else
    fadeFactor = 1.05;

  fadeValue *= fadeFactor;

  image(img, width/2, height/2);
  tint(fadeValue,fadeValue,fadeValue);

}
```

Fading images in/out

Noisy Image (uniform “white” noise)

```
PImage imageNoise(PImage img, float amp) {

  background(BLACK);
  PImage result = img.copy();

  img.loadPixels();
  result.loadPixels();

  for (int i=0; i<result.pixels.length; i++) {
    result.pixels[i] = img.pixels[i]+
                          color(random(amp),random(amp),random(amp));
  }
  result.updatePixels();

  image(result, width/2, height/2);

  return result;


}
```

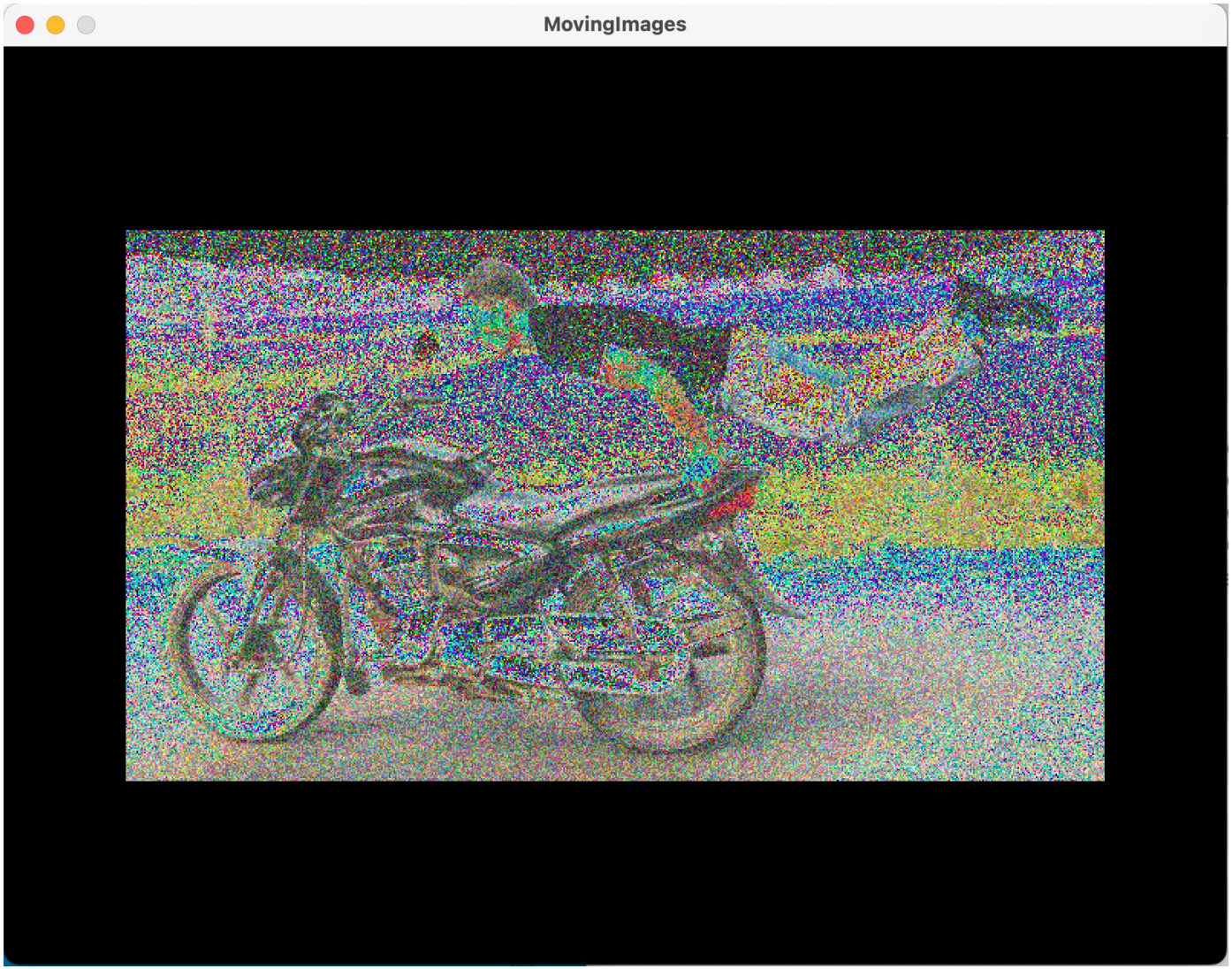


Image Warp

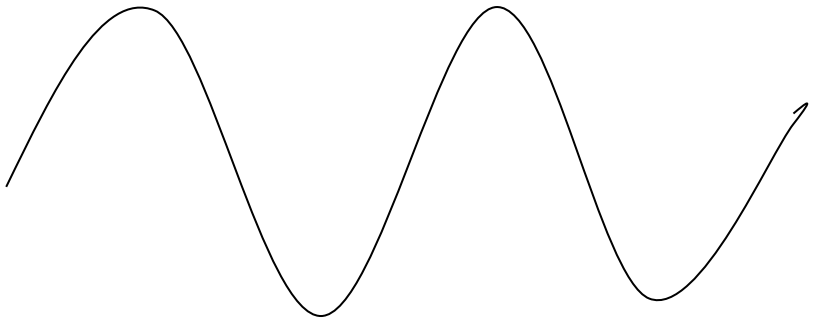
- Use some function as a basis to sample where we are extracting pixels from (and setting them to current location)

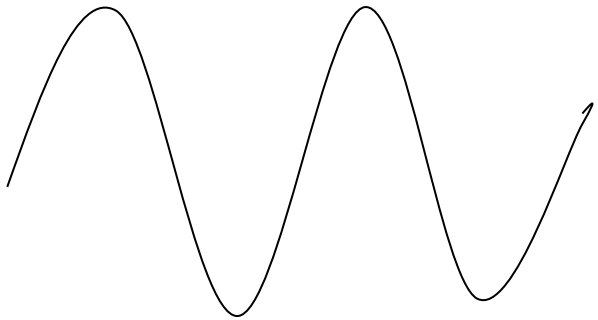
- E.g. if I grab pixels from i, j in an original image, and put them into a result image (but offset i or j by some fixed value)…. then I make the image shift

- If the offset varies due to some function, then I can warp the image (i.e. spread pixels in some areas, compress in others)

Image warp with a sine wave (stretches/compresses pixels in X)

i offset

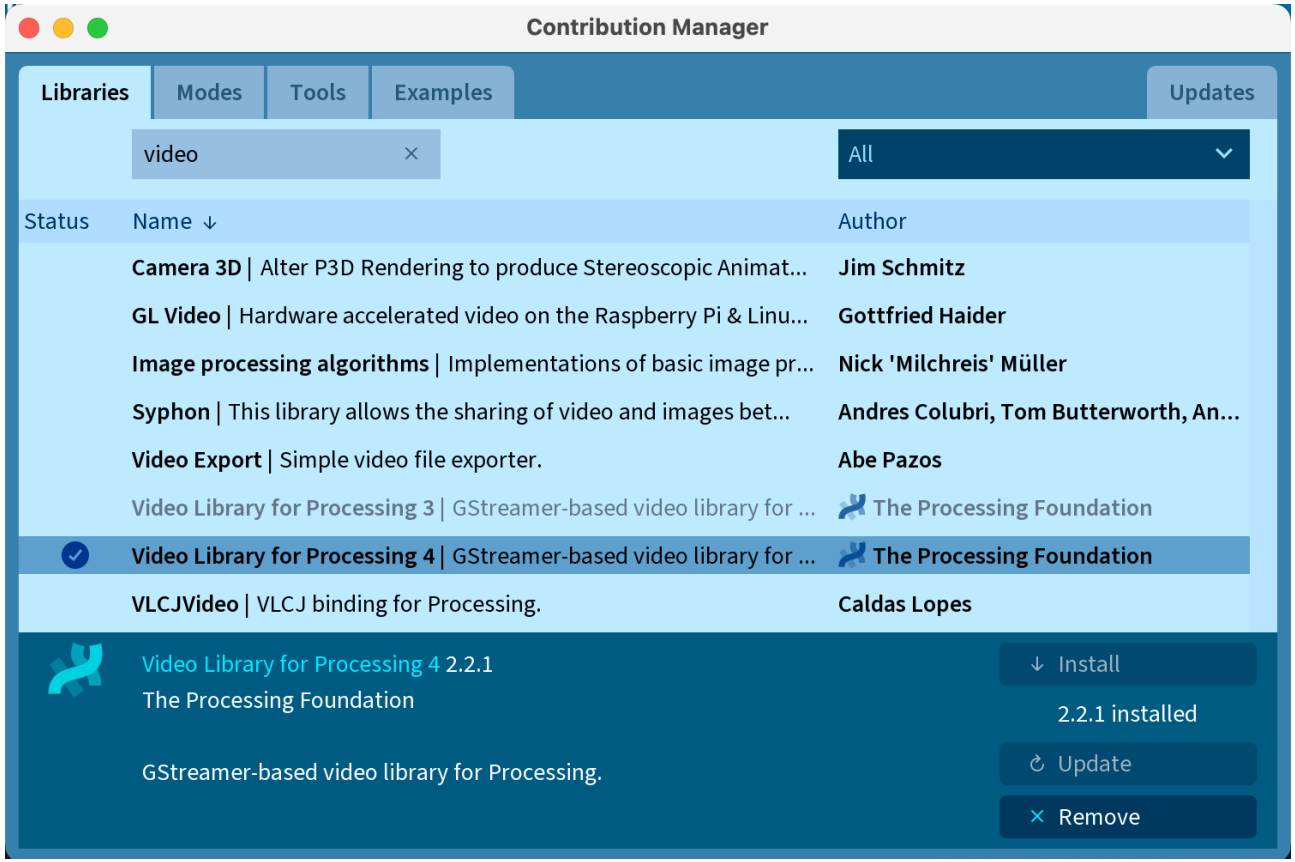




```
PImage imageWarp(PImage img, float ampX, float ampY) {

  background(BLACK);
  PImage result = img.copy();
  img.loadPixels();
  result.loadPixels();

  for (int i=0; i<result.width; i++) {
    for (int j=0; j<result.height; j++) {

      int loc = i + j*result.width;
      int offsetX = (int)(ampX * sin(PI*i/150.0));

      if ( ((i+offsetX)>=0) && ((i+offsetX)<result.width) ) {
        int loc2 = i+offsetX + j*result.width;

        result.pixels[loc] = img.pixels[loc2];
      }
      else {
        result.pixels[loc] = color(0, 0, 0);
      }
    }
  }
 result.updatePixels();
  image(result, width/2, height/2);
  return result;
}
```

Playing/Working with Video

Sketch (menu) → Import Library → Manage Libraries
- *Video Library for Processing 4*



Playing Video Files/ Video from Camera

```
import processing.video.*;



// OBJECTS

Movie myMovie;          // use this for loading in and running
                        // an existing video file

Capture cam;            // use this to capture live stream from
                        // a camera device (e.g. webcam)
```

Reference:
https://processing.org/reference/libraries/video/index.html

YORK UNIVERSITÉ UNIVERSITY

Movie object

Constructors

```
Movie(parent, filename)
```

Parameters

parent	PApplet
filename	String

Methods

`frameRate()`	Sets how often frames are read from the movie.
`speed()`	Sets the relative playback speed of the movie.
`duration()`	Returns the length of the movie in seconds.
`time()`	Returns the location of the playback head in seconds.
`jump()`	Jumps to a specific location within a movie.
`available()`	Returns ”true” when a new movie frame is available to read.
`play()`	Plays a movie one time and stops at the last frame.
`loop()`	Plays a movie continuously, restarting it when it’s over.
`noLoop()`	If a movie is looping, this will cause it to play until the end and then stop on the last frame.
`pause()`	Pauses a movie during playback.
`stop()`	Stops a movie from continuing.
`read()`	Reads the current frame of the movie.

Example

from examples→ Libraries → Video Library for Processing 4

```
/**
 * Scratch
 * by Andres Colubri.
 *
 * Move the cursor horizontally across the screen to set
 * the position in the movie file.
 */

import processing.video.*;

Movie mov;

void setup() {
  size(560, 406);
  background(0);

  mov = new Movie(this, "launch2.mp4");

  // Pausing the video at the first frame.
  mov.play();
  mov.jump(0);
  mov.pause();
}
```

```
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;          // set t to a % of duration

    mov.play();
    mov.jump(t);
    mov.pause();
  }

  image(mov, 0, 0);
}
```

Syntax	`map(value, start1, stop1, start2, stop2)`		
Parameters	**`value`**	`(float)`	the incoming value to be converted
	`start1`	`(float)`	lower bound of the value's current range
	`stop1`	`(float)`	upper bound of the value's current range
	`start2`	`(float)`	lower bound of the value's target range
	`stop2`	`(float)`	upper bound of the value's target range
Return	`float`		

Capturing Live Video (e.g. webcam)

```
// CAMERA CAPTURE SETUP
 String[] cameras = Capture.list();
 if (cameras.length == 0) {
   println("there are no cameras available for capture");
   exit();
 }
 else {
   println("Available cameras:");
   for (int i = 0; i < cameras.length; i++) {
     println(cameras[i]);
   }
 }

 // to capture, normally you just need this line :
 // cam = new Capture(this, cameras[0]);


 // on apple silicon, there is a bug that is still not updated
 cam = new Capture(this, 640, 480,
        "pipeline:avfvideosrc device-index=0 ! video/x-raw,
         width=640, height=480, framerate=30/1");

 cam.start();
```

Capturing Live Video (e.g. webcam)

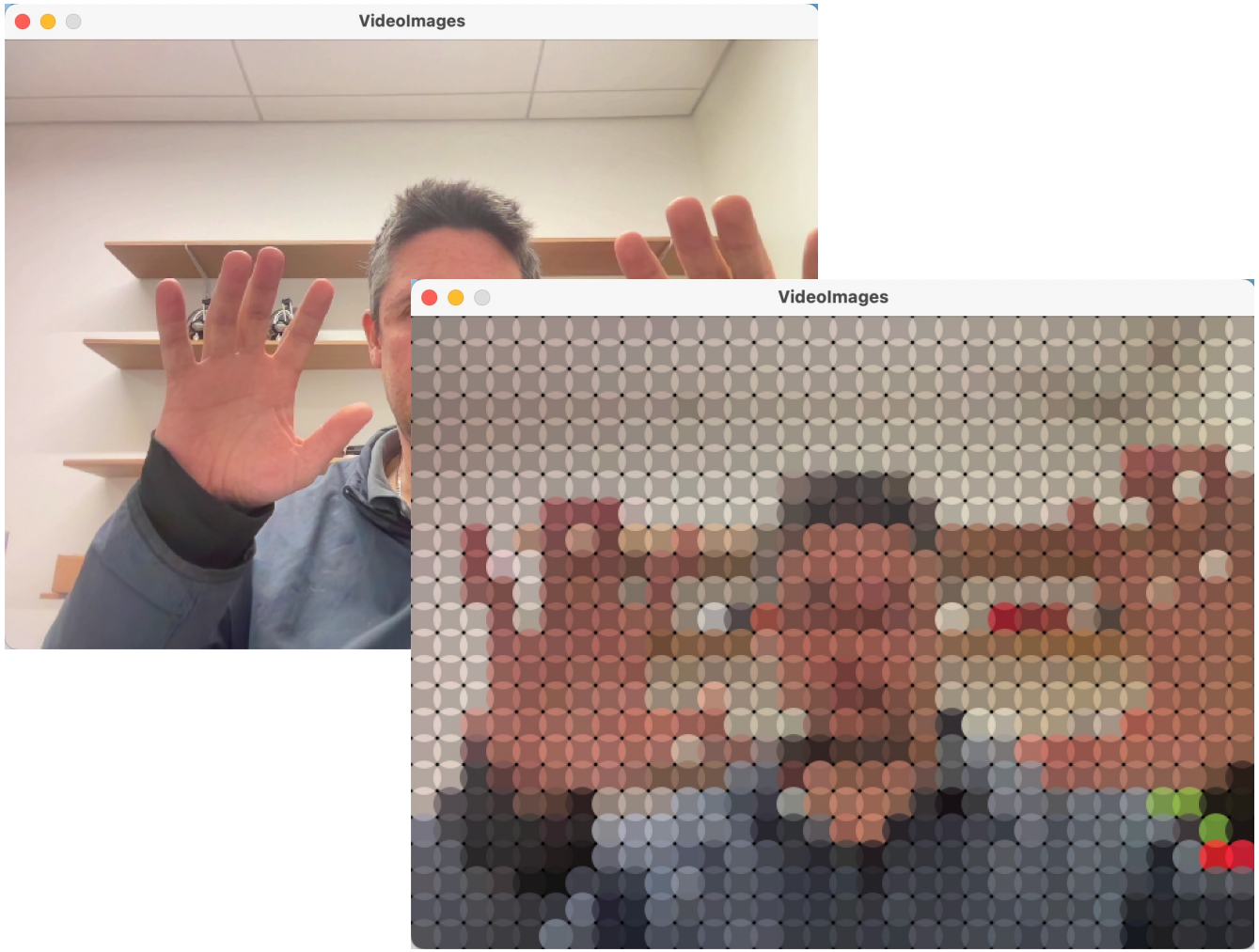
```
// CAMERA DRAW LOOP


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

// display cam image frame to app window
image(cam, width/2, height/2);

// OR load pixels and process them here... every draw loop
// example, draw some objects around the mouse

loadPixels();

// Do stuff with pixels



updatePixels();
```

Example
(use video frame pixels to draw shapes)

```
  if (cam.available() == true) {
    cam.read();
  }
  cam.loadPixels();
  colorMode(RGB, 255, 255, 255, 100);

  int cellSize = 20;
  for (int i = 0; i < cam.width/cellSize; i++) {
    for (int j = 0; j < cam.height/cellSize; j++) {
      int x = i*cellSize;
      int y = j*cellSize;
      int loc = x + y*cam.width;

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

      // Make a new color with an alpha component
      color c = color(r, g, b, 75);
      ellipseMode(CENTER);
      fill(c);
      noStroke();
      ellipse(x+cellSize/2, y+cellSize/2, cellSize+6, cellSize+6);
    }
  }
  cam.updatePixels();
}
```

YORK UNIVERSITÉ UNIVERSITY

Drawing ellipses (driven by video pixels)..



YORK UNIVERSITÉ UNIVERSITY