CS 106 Winter 2018

Lab 07: Image Processing and Video

Credit: Parts of this Lab is derived from a CS105 lab by Professor Dan Vogel.

Due: Wednesday, March 6th, 11:59pm

This lab will allow you to practice using Image Processing and Video Processing. There are three questions. Each question starts on a separate page.

SAVE each sketch as “L07Q01”, “L07Q02”, and “L07Q03”.

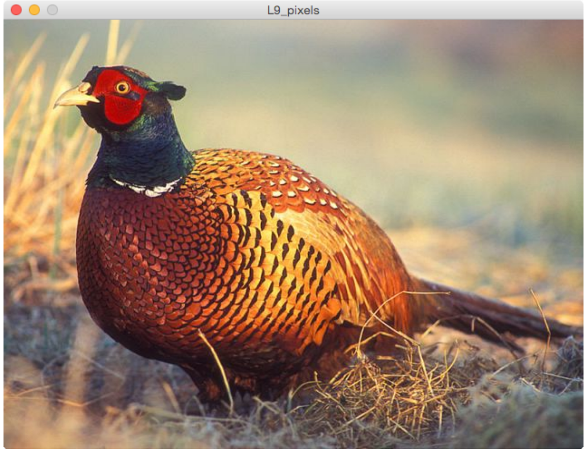
QUESTION ONE: Working with Pixels

a. Create a sketch that loads and displays the image “pheasant.jpg”. You may want to start with the demo code imagepixels.pde and remove or comment out the following lines:

```
//filter(GRAY);

//// get the colour of the pixel at mouse position
//color pixelColour = img.get(mouseX, mouseY);

//// draw rectangle using image pixel colour at that location
//fill(pixelColour);
//rect(mouseX, mouseY, 40, 40);
```



b. Convert your code from part a, to draw an abstract version of the pheasant using ellipses.

In lecture, you learned how to "get" the color of any pixel in an image using "dot get" and the color data type. For example, to get the colour of the pixel at position (100, 100) in the image stored in `img`, you could do the following:

```
color pixelColour = img.get(100, 100);
```

Once you have the colour in the `pixelColour` variable, you can use it to set the fill colour like this:

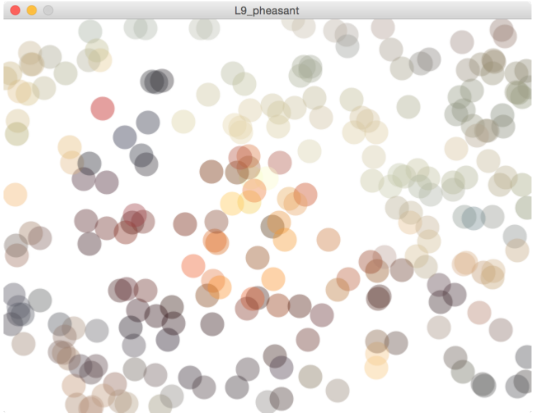
```
fill(pixelColour);
```

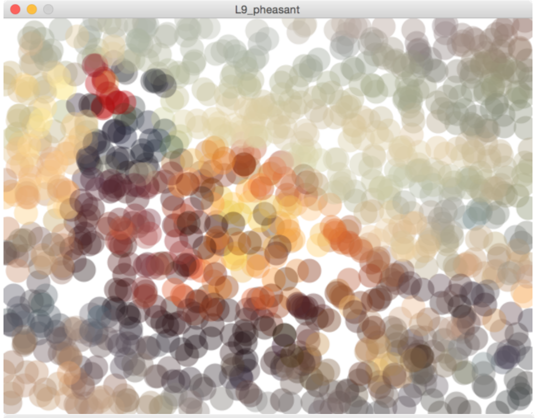
We used this in imagepixels.pde. You'll use this basic technique to make an animated abstract pattern.

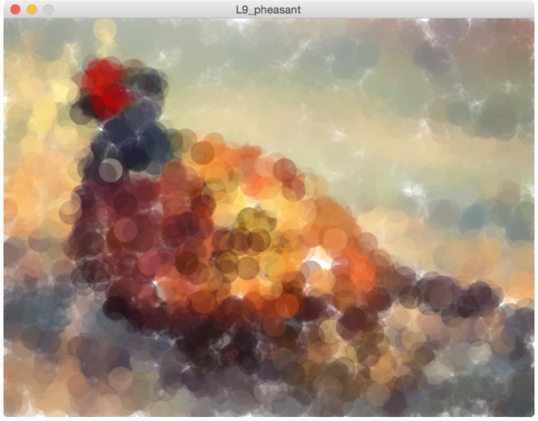
Comment out the function to draw the image in the canvas (we want the canvas to start out blank). Then in each frame, "get" the pixel colour value at a random x and y position in the image and set the fill to that same colour. Make the fill semi-transparent. Draw an ellipse in the canvas at the same random x and y position. I made my ellipse size 30, but you can experiment with different sizes.

HINT: *Do not* clear the background each time draw is called. All the code for this part 'b' is similar to imagepixels.pde.

When you run the sketch for a while, you'll see the pheasant slowly appear.



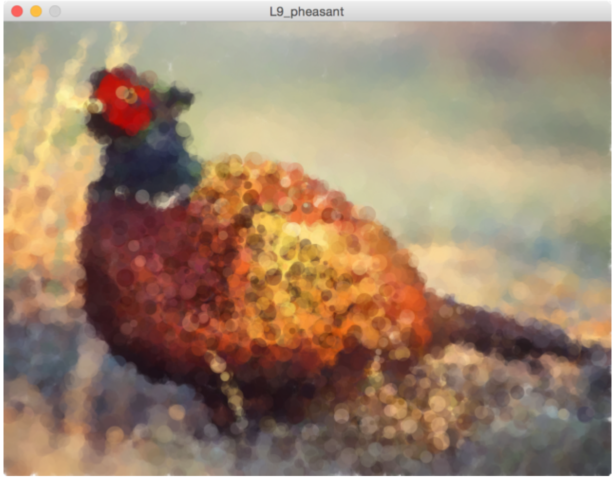




c. The pattern in part b looks great, but it’s very slow to appear and a bit too regular looking.

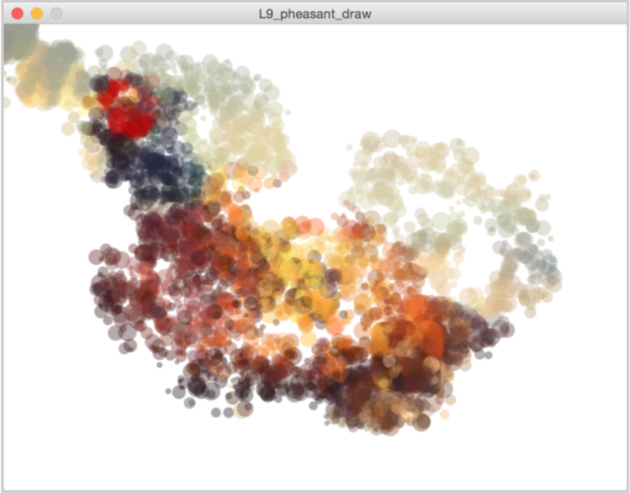
Add a loop to your code to draw multiple ellipses at random positions (each filled with the corresponding image pixel). Make the ellipse size and transparency random as well.

I draw 50 ellipses each frame, each filled with random transparency between 10 and 50 and a random size between 5 and 20. You can try these values, but experiment with other ones too.



Now, make this abstract drawing interactive. Instead of picking random ellipse positions anywhere in the canvas (anywhere in the image), pick random positions near the mouse position.

I reduced my loop iterations to 10 and I pick random positions up to 30 pixels away from the mouse in x and y directions.



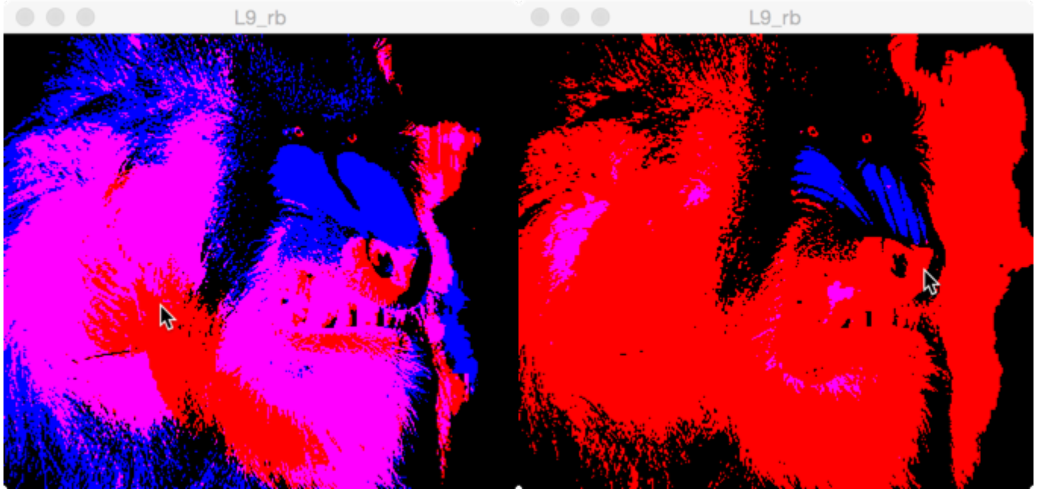
QUESTION TWO: Image Processing

Start from the `processcanvaspixels` lecture demo code. Change the code to load the image file “mandrill.jpg”.

Change the code to interactively threshold the red and blue components.

In the lecture, you saw how to interactively threshold and image to be black and white based on the brightness. Here, you want to do something similar, but threshold the red and blue components.

Use the mouseY position and map to set a threshold for red and mouseX and map to set a threshold for blue. If the red component of the pixel is above the red threshold, set red to 255. If the blue component of the pixel is above the blue threshold, set blue to 255. Always set green to 0.



QUESTION THREE: Video

Start from the videopixels.pde lecture demo code.

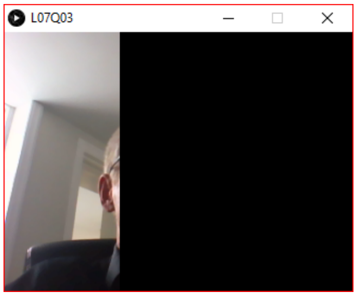
Your code should play the live video from the camera as in videopixels.pde but instead of showing the content of the video in a circle around the mouse it should perform a hittest.

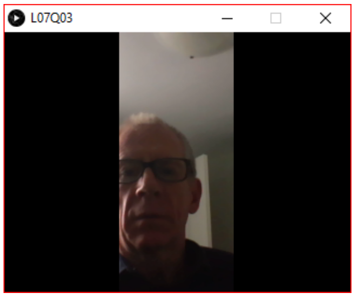
If the mouse is in the left third of the screen we should see the left third of the video only.

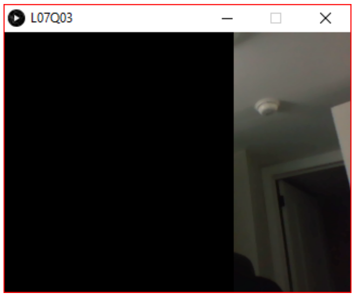
If the mouse is in the middle third of the screen we should see the middle third of the video only.

If the mouse is in the right third of the screen we should see the right third of the video only.

The three possibilities are as follows:







Submission

Submit all sketch directories from this lab as one ZIP file called L07.zip to the lab dropbox on Learn.

It is your responsibility to submit to the correct dropbox with the correct files before the deadline. Otherwise you will receive a mark of 0.