Lab 18

Load a video
Do something with it before you draw it
Lab 18

- **Basic Goal:**
  - Load input from web cam or video
  - Draw the input

- **Advanced A: Tracking with webcam**
  - Detect brightest spot or detect a spot closest to a certain colour
  - Have an object e.g. an ellipse follow the brightest spot

- **Advanced B: Stylizing a video**
  - Alternatively apply filters / stylization to a video
  - For top marks: can you apply something like the painterly stylized effect you did in Lab 16 to your video?
Install the video library

- Click on Sketch → Import Library → Add Library
- In the popup window (Contributions Manager) In the Libraries tab, type Video in the Search box
- Click on “Video” by the Processing Foundation. Click Install.
- Close the contributions manager
Import the video Library

- Start a new sketch
- Click on Sketch → Import Library → Video
  - (If you don’t see this option, you may have not completed the previous step correctly)

- This should add a line of text at the top of your sketch as follows:
  - import processing.video.*;
  - Basically this adds some specialized functions for dealing with video
  - For details see: [https://processing.org/reference/libraries/video/index.html](https://processing.org/reference/libraries/video/index.html)
Loading webcam feed

- To see if everything is working, let's start by running some basic code
- Cut and paste the text on the right into your sketch
- If you have a webcam, this should open up a stream and display it in a small window

```java
import processing.video.*;

Capture cam;

void setup()
{
    size(1280, 720);
    cam = new Capture(this);
    cam.start();
}

void draw()
{
    if (cam.available() == true)
    {
        cam.read();
    }
    image(cam, 0, 0, width, height);
}
```
Loading Video Feed

- Create a sketch paste in the code to the right and save it
  - You will need to make sure you installed the Video library (from previous slides)

- Open the sketch folder and create a sub folder called Data within the sketch folder

- Requires movie files in .MOV (quicktime video) format

- Download this sample: https://www.scss.tcd.ie/~dinglijl/Skylight.mov

- Place in the Data folder

- Try Running the sketch

```java
import processing.video.*;
Movie myMovie;

void setup() {
    size(640, 480);
    myMovie = new Movie(this, "Skylight.mov");
    myMovie.loop();
}

//leave this function alone
//reads next frame from video
void movieEvent(Movie m) {
    m.read();
}

//Edit this function to make changes
void draw() {
    image(myMovie, 0, 0);
}
```
Notes on Video

- **myMovie** is a Movie but can be treated almost the same way as a PImage object which we’ve dealt with before

- This function continually updates **myMovie**.
  - It is a special callback function
  - Best leave this alone

- Remember that by default draw gets called repeatedly. So it draws the updated image
  - We will change what goes on here to apply effects to the movie (JUST LIKE WE DID WITH IMAGES)

```java
import processing.video.*;
Movie myMovie;

void setup() {
    size(640, 480);
    myMovie = new Movie(this, "Skylight.mov");
    myMovie.loop();
}

//leave this function alone
//reads next frame from video
void movieEvent(Movie m) {
    m.read();
}

//Edit this function to make changes
void draw() {
    image(myMovie, 0, 0);
}
```
Some free movies (.MOV) to try out

- For convenience (careful, some are very big. You’ll need drive space):

- Free online resources for more interesting stuff
  - https://hilo.hawaii.edu/~csav/quicktime/
  - http://thecliparchive.com/

- Note that for the video library, videos need to be in Quicktime MOV format.
- There are some tools that let you convert formats e.g. http://www.videolan.org/vlc/index.html
Functions of images

- If we define new filters or effects, we may want to reuse them
- For any set of steps we might want to reuse, we should define a function

```java
// The following function takes an image as input and returns a new one
PImage stylise(PImage img_old)
{
    PImage img_new;

    // img_new = img_old; // NOTE don't do this the two images become linked
    img_new = img_old.copy();

    img_new.filter(BLUR, 1);
    img_new.filter(GRAY);

    return img_new;
}
```

Create a copy of the old image so we can do safely manipulate it
Operations applied to the new image. Add your own!
When we're done RETURN the modified image.
Functions of images

- If we define new filters or effects, we may want to reuse them.
- For any set of steps we might want to reuse, we should define a function.

//The following function takes an image as input and returns a new one
PImage stylise(PImage img_old)
{
    PImage img_new;

    //img_new = img_old; //NOTE don't do this the two images become linked
    img_new = img_old.copy();

    img_new.filter(BLUR, 1);
    img_new.filter(GRAY);

    return img_new;
}
Combine this code with the code on the previous page to load an image from a video and then stylise it using the stylise function.

Your objective is then to apply more interesting styles.