**TRABAJO 6**
**MEMORIA SOBRE TRABAJO DE WEBCAM**


**NOMBRE DEL ALUMNO: LERMA MOLINA ALAN ALBERTO**

**NOMBRE DE LA MATERIA: CREANDO INTERFACES DE USUARIO**

**NOMBRE DEL PROFESOR: MODESTO F. CASTRILLÓN SANTANA**

**FECHA A ENTREGAR: 21 DE MARZO DEL 2019**

**Índice**



## 1.- Introducción

En Esta actividad se trabajará con la librería de video de processing, haciendo uso de la webcam para poder crear una escena interactiva a elección, en este caso se encargará de encontrar el píxel más rojo, más azul y más verde.
En el caso de cómo se detonará los pixeles, por decisión propia aparecerán sobre estos una tortuga(1), una ballena(2) y un fuego(3) representando al verde, azul y rojo respectivamente.

## 2.- Desarrollo

A continuación se explicara el código y las razones del por qué se utiliza cada una de las partes implementadas en este.

```
1 import processing.video.*;
2 PImage Fuego, Tuga, Ballena;
3 Capture video;
```

imagen 2.1- primeras tres líneas de código

Como se puede ver en la imagen 2.1 se puede observar las tres primeras líneas de código, donde se puede observar la representación de la importación de la librería de video de processing y a su vez la creación de tres PImage las cuales son las que representan los píxeles, como ya se había explicado con anterioridad.

```
5  void setup() {
6    size(640, 480);
7    video = new Capture(this, width, height);
8    video.start();
9    Fuego=loadImage("Fuego.png");
10   Ballena=loadImage("ballena.png");
11   Tuga = loadImage("Tuga.png");
12 }
```

imagen 2.2- Void setup dado en el código de la línea cinco a doce

En la imagen 2.2 podemos observar lo básico de lo que tiene que establecer dentro el primer frame, por ejemplo el tamaño de la pantalla, la declaración de la captura de video, su inicialización y por último las tres imágenes que serán utilizadas para poder representar los píxeles con más color.

```
14 void draw() {
15   if (video.available()) {
16     video.read();
17     image(video, 0, 0, width, height);
18     int rojoX = 0;
19     int rojoY = 0;
20     int verdeX = 0;
21     int verdeY = 0;
22     int azulX = 0;
23     int azulY = 0;
24
25     float RojoValue = 0;
26     float VerdeValue = 0;
27     float AzulValue = 0;
```

imagen 2.3.- Declaración de draw y variables

En la imagen 2.3 podemos observar que se inicia el draw, pero no solo eso sino que se da la implementación de la cámara, de esta forma ya se estará reflejando la imagen en pantalla.

dentro de las líneas desde la 18 a la 27 podemos ver la implementación de valores enteros y flotantes respectivamente para cada color, los cuales se encargan de almacenar las coordenadas de los píxeles y la cantidad de color que hay en estos, de esta forma con un for anidado que logre pasar por todos los píxeles dentro de la imagen, podremos detectar cuales son los que tienen más color.

```
29      video.loadPixels();
30      int index = 0;
31      for (int y = 0; y < video.height; y++) {
32        for (int x = 0; x < video.width; x++) {
33          int pixelValue = video.pixels[index];
34          float pixelcolorRojo = red(pixelValue);
35          float pixelcolorVerde = green(pixelValue);
36          float pixelcolorAzul = blue(pixelValue);
```

imagen 2.4.- for anidado con análisis de color.

Desde la imagen 2.4 se puede denotar el uso del for anidado anteriormente mencionado donde podemos ver que se está viendo cada pixel en la pantalla, y donde podemos ver que analizamos los valores del píxel en un array, el cual le sacamos los valores rojo verde y azul.

```
50            if (pixelcolorVerde > VerdeValue) {
51            VerdeValue = pixelcolorVerde;
52            verdeY = y; //guardamos el valor de y
53            verdeX = x; //guardamos el valor de x
54          }
55          index++;
56        }
57      }
58
59      image(Fuego,rojoX-100, rojoY-100, 200, 200);
60      image(Tuga,verdeX-100, verdeY-100, 200, 200);
61      image(Ballena,azulX-100, azulY-100, 200, 200);
```

imagen 2.5 .- comparación y representación

Tal como se muestra en al imagen 2.5, lo que resta de código es la introducción del comando que hará que si un valor no es mayor que el anterior no guarde sus coordenadas, hasta terminar con el pixel con mayores valores, tanto para rojo verde y azul, al final imprimiendo estas imágenes en su respectivo pixel.

## 3.- Conclusión

Al trabajar con este proyecto se pudo observar y aplicar de una forma correcta la forma básica en que funciona la librería de video de processing, el cual si bien no

fue un ejemplo muy complicado, sigue dando raíces de lo que se puede lograr con este.

imagen 3.1 Ejemplo 1

imagen 3.2 Ejemplo 2

## 4.- Referencias

1.- Pixabay.com. (2019). *Imagen gratis en Pixabay - Tortuga, Mar, Animales, El Agua*. [online] Available at: https://pixabay.com/es/vectors/tortuga-mar-animales-el-agua-311098/ [Accessed 21 Mar. 2019]

2.- Anon, (2019). [image] Available at: https://es.pngtree.com/freepng/free-cute-cartoon-whale-pull-material_2149808.html [Accessed 21 Mar. 2019].

3.- Anon, (2019). [image] Available at: https://es.vexels.com/png-svg/vista-previa/146886/clipart-de-fuego [Accessed 21 Mar. 2019].

Repositorio github
https://github.com/AlanLerma/Video