
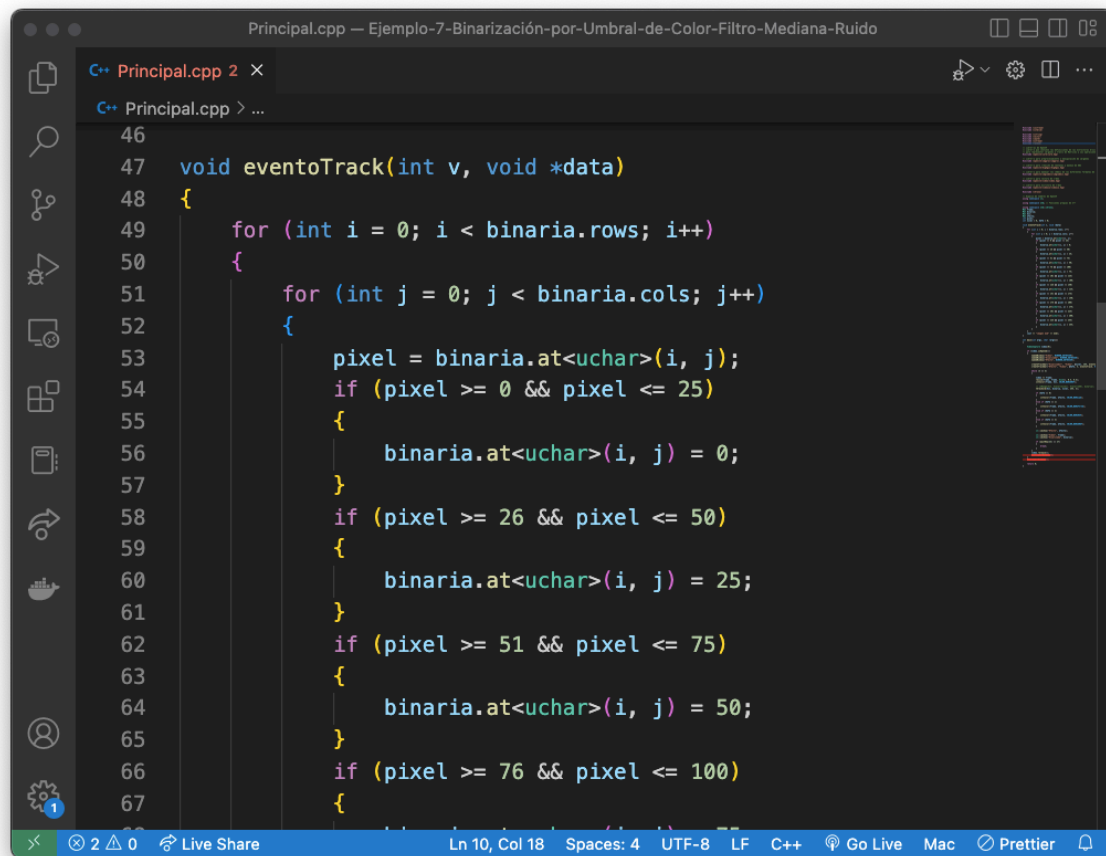

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

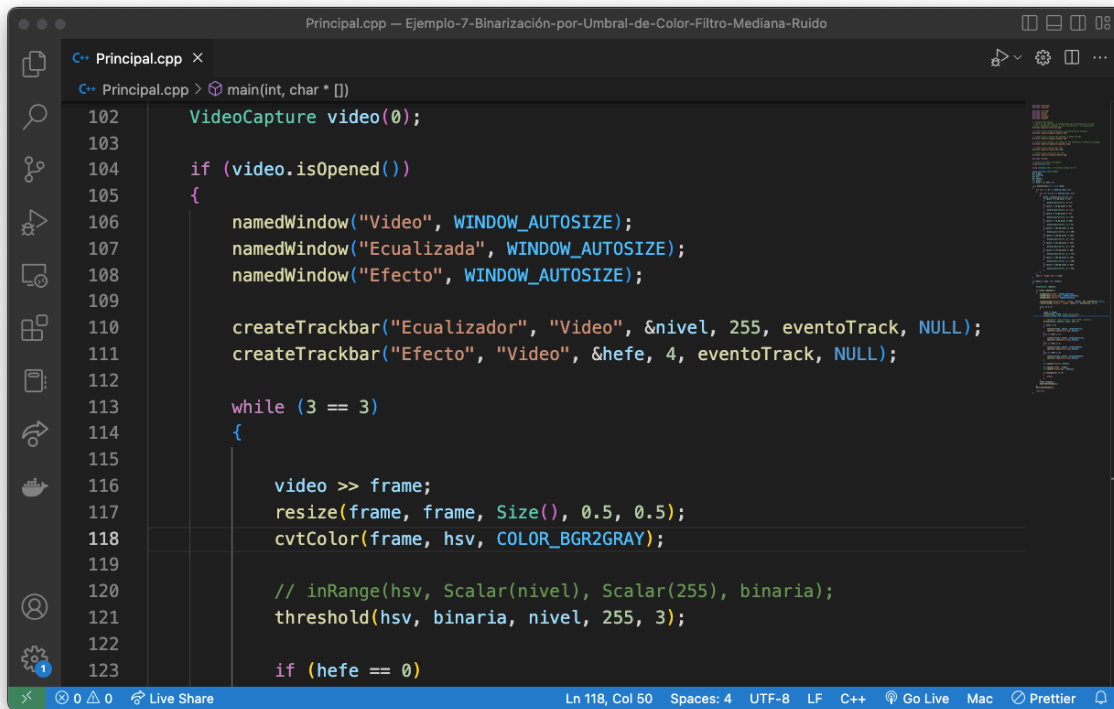
		PRÁCTICA DE LABORATORIO
CARRERA:	Computacion	ASIGNATURA: Visión por computador
NRO. PRÁCTICA:	1	TÍTULO PRÁCTICA: Fundamentos del procesamiento digital de imágenes: Espacios de Color, Histogramas y Clasificación Básica
OBJETIVO ALCANZADO: Reforzar los conocimientos adquiridos en clase sobre la conversión de espacios de color, cálculo de histogramas y el procesamiento de vídeo.		
ACTIVIDADES DESARROLLADAS		
Parte 1 Crear evento track		



```
Principal.cpp — Ejemplo-7-Binarización-por-Umbra-de-Color-Filtro-Mediana-Ruido
C++ Principal.cpp 2 x
C++ Principal.cpp > ...
46
47 void eventoTrack(int v, void *data)
48 {
49     for (int i = 0; i < binaria.rows; i++)
50     {
51         for (int j = 0; j < binaria.cols; j++)
52         {
53             pixel = binaria.at<uchar>(i, j);
54             if (pixel >= 0 && pixel <= 25)
55             {
56                 binaria.at<uchar>(i, j) = 0;
57             }
58             if (pixel >= 26 && pixel <= 50)
59             {
60                 binaria.at<uchar>(i, j) = 25;
61             }
62             if (pixel >= 51 && pixel <= 75)
63             {
64                 binaria.at<uchar>(i, j) = 50;
65             }
66             if (pixel >= 76 && pixel <= 100)
67             {
```

Llamamos al evento cuando modificamos los niveles

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		



```

Principal.cpp — Ejemplo-7-Binarización-por-Umbra-de-Color-Filtro-Mediana-Ruido
Principal.cpp x
C++ Principal.cpp > main(int, char * [])
102     VideoCapture video(0);
103
104     if (video.isOpened())
105     {
106         namedWindow("Video", WINDOW_AUTOSIZE);
107         namedWindow("Ecuilizada", WINDOW_AUTOSIZE);
108         namedWindow("Efecto", WINDOW_AUTOSIZE);
109
110         createTrackbar("Ecuilizador", "Video", &nivel, 255, eventoTrack, NULL);
111         createTrackbar("Efecto", "Video", &hefe, 4, eventoTrack, NULL);
112
113         while (3 == 3)
114         {
115             video >> frame;
116             resize(frame, frame, Size(), 0.5, 0.5);
117             cvtColor(frame, hsv, COLOR_BGR2GRAY);
118
119             // inRange(hsv, Scalar(nivel), Scalar(255), binaria);
120             threshold(hsv, binaria, nivel, 255, 3);
121
122             if (hefe == 0)
123

```

Parte 2

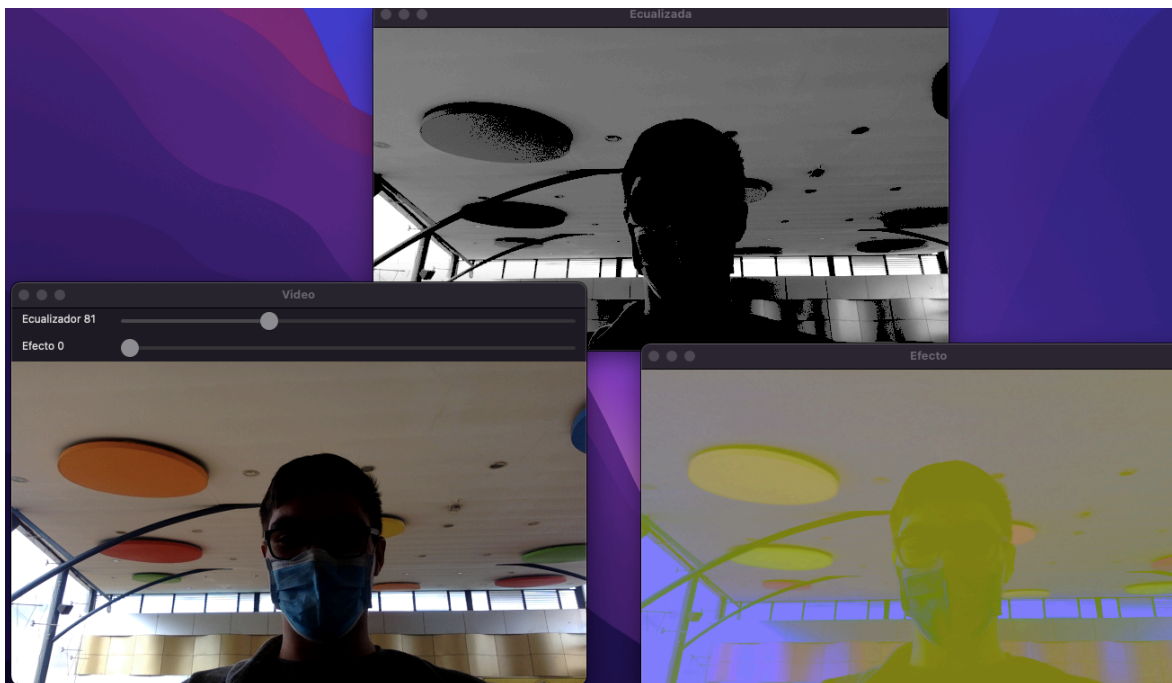
Utilizamos otro tracebar para aplicar un efecto al video cada vez que movemos los valores y además guardamos la imagen.

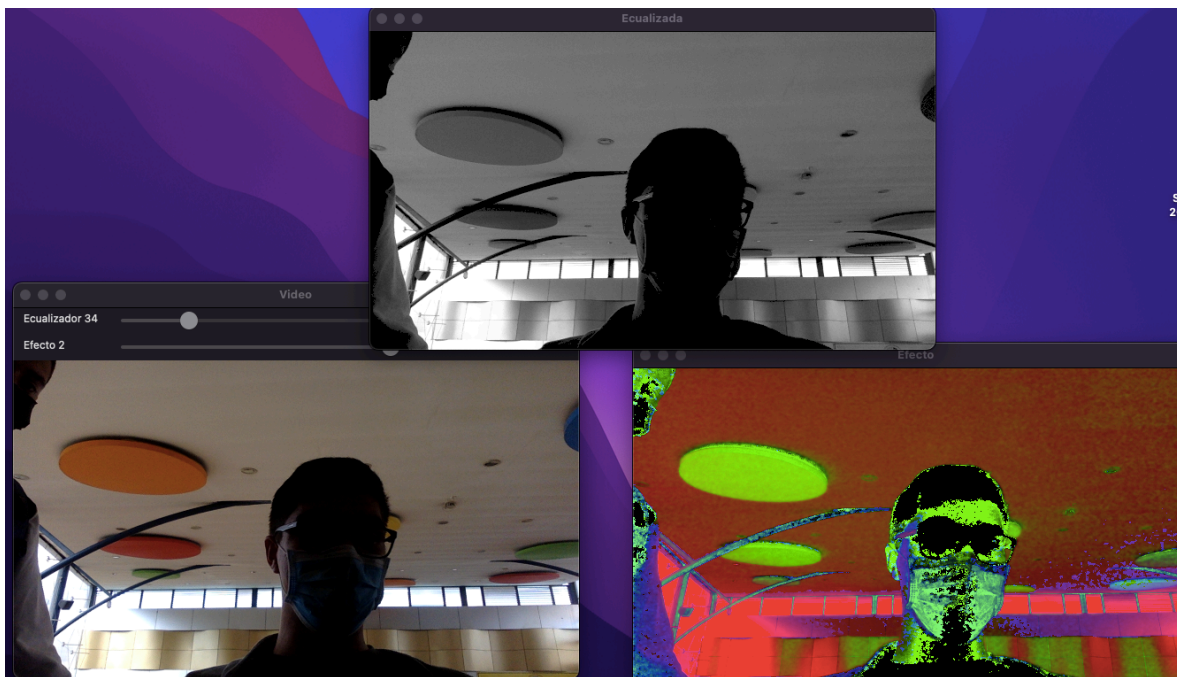
```
Principal.cpp — Ejemplo-7-Binarización-por-Umbra-de-Color-Filtro-Mediana-Ruido

Principal.cpp x
C++ Principal.cpp > main(int, char * [])

122
123     if (hefe == 0)
124     {
125         cvtColor(frame, efecto, COLOR_BGR2Lab);
126         imwrite("imagenEfecto.jpg",efecto);
127     }
128     else if (hefe == 1)
129     {
130         cvtColor(frame, efecto, COLOR_BGR2YCrCb);
131         imwrite("imagenEfecto.jpg",efecto);
132     }
133     else if (hefe == 2)
134     {
135         cvtColor(frame, efecto, COLOR_BGR2HSV);
136         imwrite("imagenEfecto.jpg",efecto);
137     }
138     else if (hefe == 3)
139     {
140         cvtColor(frame, efecto, COLOR_BGR2GRAY);
141         imwrite("imagenEfecto.jpg",efecto);
142     }
143
```

RESULTADO(S) OBTENIDO(S):





CONCLUSIONES:

Se debe trabajar diferente dependiendo de si se quiere trabajar a color o en grises.

Nombre de estudiante: Eduardo Antonio Ayora Ochoa

Firma de estudiante:

