Ejercicio 1a

Introducción de texto

Para introducir texto al vídeo, he empleado el siguiente fragmento de código:

```
# Imprime el texto
cv2.putText(frame,"Aitor Ingelmo Martin",
            Abajo_izquierda,
            font,
            fontScale,
            fontColor,
            thickness)
```

Donde las variables las he definido previamente de la siguiente forma:

```
#Defino los parámetros para el texto que aparece en el vídeo:

font = cv2.FONT_HERSHEY_SIMPLEX    # Tipo de fuente
Abajo_izquierda = (10,500)         # Posición del texto
fontScale              = 1         # Escala de la fuente
fontColor              = (0,255,0) # Color del texto
thickness              = 2         # Grosor del texto
```

Introducción del logo

Para introducir el logo, primero he leído y rescaldo el logo de la siguiente manera:

```
# Introducción del logo:
logo_original = cv2.imread("Videos/logo.png") # Leo el logo
logo = cv2.resize(logo_original,(150,150))    # Re-escalo el logo
```

Como ahora tengo la certeza de que el logo mida 150x150, podemos introducir el logo en cualquier zona deseada de cada frame. Como el vídeo tiene abajo a la derecha el logotipo de La 2, decidí poner ahí el logo.

Para ello, empleo la siguiente línea:

```
  # Inserta el logo
  frame[(576-150):576,(720-150):720] = logo
```

Donde 576 y 720 son las dimensiones totales del frame. Es decir, voy a escribir los últimos 150x150 pixeles de la matriz total que es el frame.

Guardado del video

Para escribir en memoria una copia del vídeo, requerimos las siguientes dos variables:

```
fourcc = cv2.VideoWriter_fourcc(*'MJPG')                                #Establezco el formato de grabación
out    = cv2.VideoWriter('videoAitorIngelmo.avi',fourcc, 25.0, (720,576)) # Introduzco los parámetros del video
```

Fourcc me va a guardar el formato de la grabación. **Out** me va a guardar el constructor con los parámetros del video para poder guardarlo.

Donde el constructor "VideoWriter" requiere 3 parámetros fundamentales definidos en orden:

- **Fourcc**: Definido siguiendo el guion de la práctica. La variable con el mismo nombre.
- **FPS**: Gracias a la función get() de openCV podemos obtener los FPS del vídeo. Este valor lo he introducido a mano a partir de lo devuelto por la siguiente línea:

  cap.get(cv2.CAP_PROP_FPS)

- **Dimensión del vídeo**: Al igual que lo sucedido con FPS, la función get() puede devolvernos el valor del ancho y largo del vídeo. Estos valores los he introducido a mano a partir de lo devuelto por las siguientes líneas:

  width = cap.get(cv2.CAP_PROP_FRAME_WIDTH)

  height = cap.get(cv2.CAP_PROP_FRAME_HEIGHT)

Pol último, debo invocar a Out para hacer el guardado de cada frame:

```
# Guardo el "frame"
out.write(frame)
```

Ejercicio 2

Para poder extraer en tiempo real un frame al pulsar la letra “q”, primero he definido una función llamada extraccion_de_canales.

```python
copias_rgb = []
def extraccion_canales(imagen):
    blue = imagen.copy()
    # Extraigo el canal azúl
    blue[:, :, 1] = 0
    blue[:, :, 2] = 0

    green = imagen.copy()
    # Extraigo el canal verde
    green[:, :, 0] = 0
    green[:, :, 2] = 0

    red = imagen.copy()
    # Extraigo el canal rojo
    red[:, :, 0] = 0
    red[:, :, 1] = 0
    return blue, green, red
```

Esta función me devuelve los valores “Blue, green, red”, los cuales extraen el color correspondiente de la imagen dada. Adicionalmente se puede ver el array vacío copias_rgb en el cual voy a guardar las 3 copias de diferentes colores.

Llegados a este punto, solo debemos ir visualizando los frames grabados (aunque yo uso un video), y en el caso de pulsar la “q” guardar el frame con diferentes canales.

Para esto, en el “if” que se activa al pulsar la letra “q”, realizo las siguientes acciones

1. Copio el frame para poder manipularlo.
2. Lo paso por mi función.
3. Concateno en horizontal y vertical para obtener una única imagen final.
4. Escribo esta imagen en memoria.

```python
while(True):
    # Capture frame-by-frame
    ret, frame = cap.read()
    # Our operations on the frame come here
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    # Display the resulting frame
    cv2.imshow('frame',gray)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        copia_frame = frame.copy()
        copias_rgb = extraccion_canales(copia_frame)

        concatenacion_horizontal_1 = cv2.hconcat([frame,copias_rgb[0]])
        concatenacion_horizontal_2 = cv2.hconcat([copias_rgb[1],copias_rgb[2]])
        concatenacion_final        = cv2.vconcat([concatenacion_horizontal_1,concatenacion_horizontal_2])

        cv2.imwrite("Aitor.png",concatenacion_final)
        break
# When everything done, release the capture
```