Procesado de imagen y visión por computador
Curso 2018/2019
Programación de aplicaciones con Python y OpenCV (II)

1. Trabajar con vídeos

La librería OpenCV incluye funciones para trabajar con vídeos de distintos formatos, como por ejemplo el formato MPEG4, MJPG o XVID. Dichas funciones permiten, entre otras cosas, extraer fotogramas del vídeo y trabajar con ellos como si fueran imágenes independientes. Para poder realizar esto, se puede emplear la siguiente clase, y sus correspondientes métodos:

- `VideoCapture.` Clase principal para trabajar con un vídeo.
  - `cv2.VideoCapture(filename) → Objeto para capturar.` Abre el vídeo especificado y devuelve un objeto que permite acceder a él.
  - `cv2.read()→ retval, image`: Extrae el fotograma actual del vídeo. Una posterior llamada a la misma función extraería la siguiente imagen, y así sucesivamente hasta el final del vídeo. Devuelve en *retval* si la operación ha sido correcta.

El archivo *Apartado1.py* incluye el código que permite abrir y mostrar un vídeo por la pantalla. La variable *time* está ajustada a la velocidad correcta de visualización mediante la lectura el número de Frames por segundo (fps) del vídeo (Si la velocidad del vídeo es de 25 imágenes por segundo, *time* vale 1000 ms/25 fps=40 ms). La velocidad se lee con la orden:

```
fps = cap.get(cv2.CAP_PROP_FPS)
```

Ejecutar el programa, abriendo un vídeo cualquiera, y comprobar que dicho vídeo se muestra por pantalla a la velocidad correcta.

Posteriormente, modificar el código anterior para que el vídeo muestre, superpuesto a la secuencia original, el nombre del alumno. Para poder imprimir texto sobre una imagen, se puede emplear la función *cv2.PutText.* Para definir el color del texto, puede emplearse una tupla (B,G,R)[1]. Para probarlo ponga el texto en rojo.

Guardar el código fuente del programa anterior, incluida la modificación para imprimir sobre la imagen el nombre del alumno.

1a.- Guardar un vídeo:

Para guardar un vídeo con OpenCV, podemos emplear la clase *VideoWriter*. Dicha clase incluye, entre otros, los siguientes métodos:

- `cv2.VideoWriter`: Crea y abre el vídeo especificado. Si el archivo ya existe, se sobreescribirá. Hay que pasarle los parámetros básicos del vídeo.

- `cv2.write(img)`: Añade una nueva imagen al vídeo.

Al abrir el archivo tenemos que tener en cuenta varios detalles. En primer lugar debemos especificar el codec y el formato del vídeo. Esto se especifica con el código *FOURCC*, que es un conjunto de cuatro caracteres que identifican el codec y formato a utilizar. En la web www.fourcc.org pueden

1 Como ayuda: http://docs.opencv.org/master/dc/da5/tutorial_py_drawing_functions.html#gsc.tab=0

consultarse todos los códigos disponibles. Para construir el código a partir de los 4 caracteres, podemos utilizar:

- `fourcc = cv2.VideoWriter_fourcc(*'MJPG')`

En el ejemplo se especifica el formato MJPEG. Bastará con incluir, como segundo argumento del constructor la variable `fourcc` creada, por ejemplo:

- `out = cv2.VideoWriter('output.avi',fourcc, 20.0, (640,480))`

El tamaño de los fotogramas debe ser el mismo que el que pasemos como argumento al crear el *VideoWriter*. Si damos otro tamaño, el vídeo no se creará. Así mismo, todas las imágenes que añadamos deben ser en color (formato BGR).

Modificar el código del apartado anterior para que se guarde una copia del vídeo original[2], pero con la modificación indicada en dicho apartado, es decir, superponiendo el nombre del alumno. Como formato, emplearemos el formato MJPEG. Si el vídeo no se genera, habrá que revisar todos los detalles mencionados anteriormente, ya que si no se ajustan todos los parámetros a su valor correcto, el codec no funciona correctamente. Guardar el vídeo con el nombre *videoNombre.avi*.

2. Trabajar con cámaras de vídeo

En OpenCV, trabajar con los fotogramas en tiempo real capturados por una cámara de vídeo no ofrece ninguna diferencia significativa con respecto a trabajar con vídeos salvo que muchas de las propiedades del vídeo no se pueden leer, por ejemplo, la velocidad del vídeo no es accesible y dará un error si se intenta leer. Para abrir una cámara, emplearemos, igual que se hizo con los vídeos en el apartado anterior, la clase *VideoCapture*. La única diferencia estriba en que, a la hora de abrir el vídeo, en lugar de especificar el nombre del archivo que contiene el vídeo, habrá que especificar el dispositivo (cámara de vídeo) a emplear, utilizando la función:

- `cap=cv2.VideoCapture(device)`: Abre el dispositivo especificado.

donde *device* es el identificador de la cámara. Por ejemplo, cuando trabajamos con cámaras *USB*, y sólo tenemos una única cámara, el dispositivo sería el 0. Si tuviéramos dos cámaras, una de ellas sería la 0, y la siguiente la 1, y así sucesivamente.

Utilice como base el código de la página web de la documentación de OpenCV[2] para abrir, capturar y mostrar por pantalla la escena visualizada por una cámara USB. Conectar la cámara USB al ordenador, y comprobar que funciona correctamente (quizá sea necesario enfocar correctamente la cámara). Posteriormente, modificar el código para hacer que guarde la última imagen visualizada al salir de la aplicación tras presionar “q”.

2 Ayuda: http://docs.opencv.org/3.4.3/dd/d43/tutorial_py_video_display.html