INSTITUTO TECNOLÓGICO SUPERIOR DE JEREZ

Jerez de García Salinas

28/05/2020

Ivan Gamboa Ultreras

16070125

gamboita9@gmail.com

Ingeniería en Sistemas Computacionales

Programación Móvil

8vo Semestre

Reporte:

Uso de OpenCV en Android

Dr. Jorge R. Manjarrez Sánchez

Introducción

OpenCV es una biblioteca libre de visión artificial que puede ser usada en distintos lenguajes y ambientes de implementación como lo puede ser Android Studio.

Utilizando como base una aplicación capaz de incluir imágenes de dos modos, por medio de la galería y a través de la cámara, se implementará OpenCV para la modificación de las imágenes, estas imágenes serán persistentes por lo que se almacenarán en el dispositivo, las modificaciones de muestra serán sobre el modelo de color de la imagen, tomando el modelo RGB como el original y haciendo el cambio a los modelos gris, hsv, lab, hsl.

Configuración Previa

Para utilizar OpenCV en Android Studio es necesario incluir el módulo y sus librearías al proyecto. A continuación, los pasos a seguir de manera resumida.

Primero tenemos que descargar alguna versión de OpenCV para Android desde la página https://opencv.org/releases/, se descomprime el archivo descargado y lo siguiente es agregar el módulo a Android, para esto “File->New->Import module…” nos mostrara una ventana en donde seleccionamos la dirección “OpenCV-android-sdk\sdk\java” dentro de la carpeta en donde descomprimimos el archivo, con esto el modulo queda agregado al Android Studio, lo siguiente es crear la dependencia necesaria para poder usar el módulo, ahora vamos a “File->Project Structure…” después damos en donde dice “Dependencias” seleccionamos el modulo de nuestra aplicación, por defecto esta se llama “app”, aquí vamos a agregar una nueva dependencia con la opción de “Module Dependency”, si se agregó bien el módulo ahí deberá de aparecer, para que funcione debemos asegurarnos de que los apartados de “compileSdkVersion”, “minSdkVersion” y “targetSdkVersion” sean las mismas en ambos gradles. Por ultimo se agregan las librerías nativas, para esto vamos a “File->New->Folder->JNI Folder”, en donde le cambiamos el nombre de “jni” a “jniLibs”, ya con la carpeta creada se agregan los archivos de nuestro archivo descargado en la ruta “OpenCV-android-sdk\sdk\native\libs”, en caso de no tener

un Ndk asignado vamos a “File->Project Structure” en “SDK Location” seleccionamos nuestro NDK o descargamos uno de ser necesario.

Desarrollo

El aspecto visual de la aplicación consta de una pantalla en donde se muestran todas las imágenes agregadas, cuenta con un botón en la parte inferior izquierda para agregar más imágenes de ser necesario, para editar las imágenes la aplicación despliega un menú al presionar cualquiera de las imágenes, este menú tiene 5 opciones:

- Eliminar Imagen
- RGB a Gris
- RGB a HSV
- RGB a LAB
- RGB a HSL

El menú desplegado se muestra en la Ilustración 1

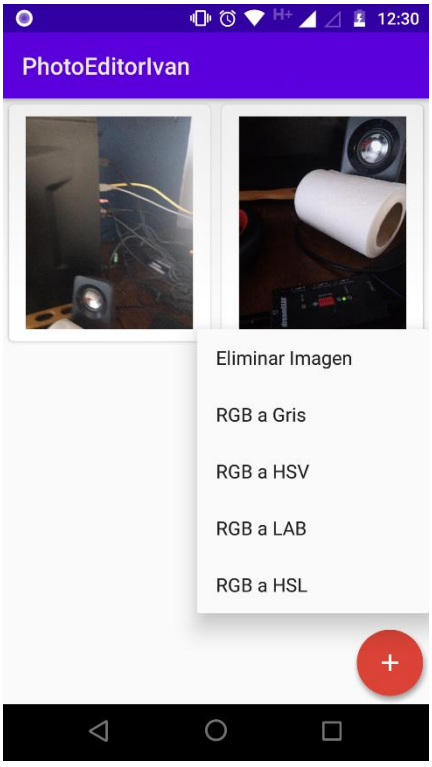

Ilustración 1. Despliegue Menú

En los “ImageView” se muestra una versión escalada de las imágenes, de esta forma no se ve afectado el rendimiento de la aplicación, pero la edición se hace sobre la imagen original.

Para poder utilizar OpenCV es necesario llamar el método “initDebug” de la clase “OpenCVLoader“, esto puede ser en el método “onCreate” de la activity principal, o lo mejor seria agregando un objeto nuevo a la clase como este:

```
private BaseLoaderCallback mLoaderCallback = new BaseLoaderCallback(this) {
    @Override
    public void onManagerConnected(int status) {
        switch (status) {
            case LoaderCallbackInterface.SUCCESS:
            {
                Log.i(TAG, "OpenCV loaded successfully");
            } break;
            default:
            {
                super.onManagerConnected(status);
            } break;
        }
    }
};
```

y en el método “onResume” agregar los siguiente:

```
super.onResume();
if (!OpenCVLoader.initDebug()) {
    Log.d(TAG, "Internal OpenCV library not found. Using OpenCV Manager for
initialization");
    OpenCVLoader.initAsync(OpenCVLoader.OPENCV_VERSION_3_0_0, this,
mLoaderCallback);
} else {
    Log.d(TAG, "OpenCV library found inside package. Using it!");
    mLoaderCallback.onManagerConnected(LoaderCallbackInterface.SUCCESS);
}
```

Para realizar la edición de la imagen se utiliza el código:

```
Bitmap b = MediaStore.Images.Media.getBitmap(cr,u);
Mat tmp = new Mat(b.getWidth(),b.getHeight(), CvType.CV_8UC1);
Utils.bitmapToMat(b,tmp);
Imgproc.cvtColor(tmp,tmp,filtro);
Utils.matToBitmap(tmp, b);
FileOutputStream out = new FileOutputStream(u.getPath());
b.compress(Bitmap.CompressFormat.JPEG, 100, out);
adapter.notifyItemChanged(listImages.indexOf(u));
```

En este código se muestra como se obtiene la imagen a partir de el “Uri” de la imagen, después se crea el objeto “Mat” para manipular la imagen y utilizando el método “cvtColor” de la clase “Imgproc” se le aplica el filtro, este filtro depende de

cual de las 4 opciones fue seleccionada, una vez se aplicar el filtro se actualiza la imagen almacenada usando el mismo “Uri”, el código de cada filtro es el siguiente:

```
case R.id.popup_menu_gray_item:
    filtro = Imgproc.COLOR_RGB2GRAY;
    break;
case R.id.popup_menu_hsv_item:
    filtro = Imgproc.COLOR_RGB2HSV;
    break;
case R.id.popup_menu_lab_item:
    filtro = Imgproc.COLOR_RGB2Lab;
    break;
case R.id.popup_menu_hls_item:
    filtro = Imgproc.COLOR_RGB2HLS;
    break;
```

A continuación, se muestra una de las imágenes pasada a escala de grises en la Ilustración 2.

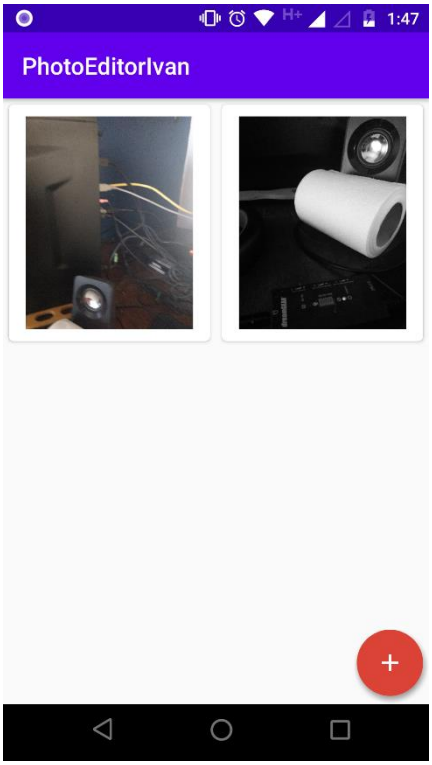

Ilustración 2. Ejemplo

Conclusión

OpenCV es una librería muy robusta, y aunque solo se mostro un ejemplo muy simple, las capacidades de la librería son muy amplias, y su punto mas fuerte es en procesamiento de imagen sobre video, sabiendo aplicarlo bien se puede hacer detección de objetos, segmentación de imágenes, traking, entre otras.