Una aplicación móvil para la captura y mejora de imágenes de textos*

José Luís Adelantado-Torres, Joan Pastor-Pellicer, and María José Castro-Bleda

Departamento de Sistemas Informáticos y Computación
Universitat Politècnica de València, Spain
joadtor@fiv.upv.es,{jpastor,mcastro}@dsic.upv.es

Resumen Cada vez es más frecuente la utilización de dispositivos móviles para tomar instantáneas de alta resolución en cualquier momento. Ello permite registrar imágenes de documentos de todo tipo. Este trabajo presenta una aplicación para la plataforma Android cuyo objetivo es preprocesar las imágenes de estos documentos, en concreto, corrección de perspectiva y limpieza, para facilitar su digitalización y además favorecer etapas posteriores como son el análisis del documento y reconocimiento del texto.

1. Introducción

Las plataformas móviles (como por ejemplo *IOS*, *Android*, *Windows Phone y otras*) tienen un éxito exponencial y la tendencia es disponer de terminales con mayor capacidad de cálculo con una gran cantidad de herramientas útiles que siempre están a mano cuando necesitamos usarlas, ya que las llevamos encima. Por tanto, tenemos la posibilidad de capturar instantáneas de documentos, ya sean impresos o manuscritos, nuevos o viejos, etc., sin necesidad de usar para ello dispositivos convencionales y especializados para ello, como son escáneres u otros dispositivos tales como cámaras específicas de alta resolución.

Evidentemente, tomar una instantánea desde un dispositivo móvil (o tableta) presenta una serie de inconvenientes respecto a los métodos específicos para ello:

- Resolución: las imágenes obtenidas por los dispositivos móviles son cada vez mejores pero siguen siendo inferiores en resolución si las comparamos con las obtenidas con herramientas específicas.
- Calidad: luminosidad, brillo, sombras, ruido, ... son sólo algunos ejemplos de características cuya calidad se ve afectada al tomar instantáneas desde dispositivos móviles ya que no se suelen obtener desde las condiciones ideales para ello.
- Perspectiva: al usar dispositivos móviles es común emplear la mano como soporte de apoyo del dispositivo, por tanto es más que normal orientar la cámara desde un punto si no incorrecto, al menos no ideal, además de capturar partes no deseadas que no corresponden al objeto que deseamos capturar y que habría que eliminar.

* Trabajo parcialmente subvencionado por el gobierno español (TIN2010-18958 y TIN2011-13070-E).

En particular, para una imagen de documento, manuscrito o impreso, es muy importante ser capaz de detectar y extraer la parte correspondiente al texto y/o información relevante para el usuario, así como almacenar o presentarlo en un formato digital accesible en cualquier máquina y/o dispositivo [5]. Asimismo, es posible que la imagen presente una serie de artefactos [2] como son suciedad, brillos, distorsión, perspectiva y otras características que deben ser corregidas o tratadas previamente al reconocimiento o mejorado de la imagen [1,4]. Otro elemento a tener en cuenta es la necesidad de detección o selección de zonas de interés sobre las que se va a aplicar dicho reconocimiento o mejora.

Existen algunas aplicaciones comerciales, como Google Drive o CamScanner, que permiten capturar y efectuan procesos similares, pero ninguna de ellas ofrece posibilidades en la selección de algoritmo o en el filtrado posterior como las que la nuestra ofrece. El objetivo de este trabajo es el diseño de una aplicación para aunar tanto algoritmos clásicos de binarización: Otsus, Niblack , Sauvola [6,8], como algoritmos propios de limpieza basados en redes neuronales [3,7]. Este trabajo presenta una aplicación móvil para Android, a la que hemos llamado esCam, cuyo funcionamiento explicaremos en el siguiente apartado, capaz de capturar o cargar imágenes destinadas a su posterior procesado, corrección de perspectiva y limpieza. Las tecnologías que se han empleado serán descritas en el apartado 3 y en el apartado 4 se reflejan las conclusiones.

2. Descripción de la aplicación

En esta sección se describen las principales características de la aplicación esCam. La figura 1 muestra un diagrama general del flujo de la aplicación esCam, la imagen de la figura 2 muestra el menú de la herrramienta, y en la figura 3 se muestran capturas reales de las principales fases de la aplicación.

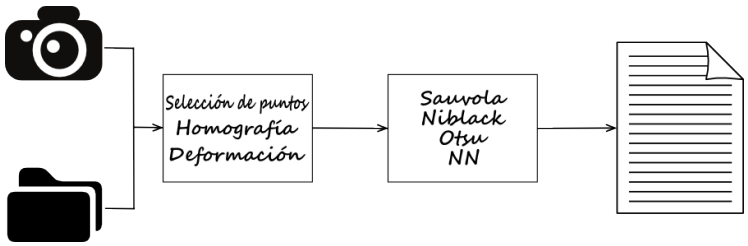


Figura 1. Flujo de la aplicación esCam.

2.1. Captura de la imagen

La entrada de imágenes a la aplicación puede provenir de la propia cámara del dispositivo o imágenes que hayamos capturado en otro momento y que se encuentren en la memoria.

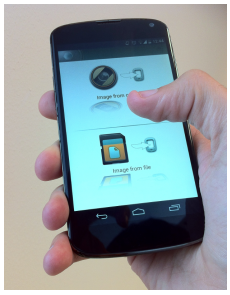


Figura 2. Menu de la aplicación esCam.

2.2. Corrección de la perspectiva

Para corregir la perspectiva obtenemos cuatro puntos, que introduce el usuario, y que representarán las esquinas de nuestro futuro documento, convirtiéndose en ellas mediante la corrección usando homografías. Es posible que la imagen original sea grande o esté en un ángulo distinto al deseado, por lo cual se establecerán formas de controlar dichas situaciones como el rotado o escalado de esta previo cálculo de la homografía.

2.3. Limpieza y binarizado

La aplicación permite elegir entre varios métodos de binarizado:

Niblack/Sauvola [8]: algoritmo que calcula el umbral utilizando el píxel central de una ventana que se va desplazando a lo largo de toda una imagen escalada a grises. El cálculo del umbral se basa en la media local y la desviación típica de la ventana.

Otsu [6]: método de cálculo automático del umbral, basado en la idea de separar entre fondo y objeto. Para ello emplea métodos estadísticos como la varianza de los distintos tonos de gris presentes en la imagen.

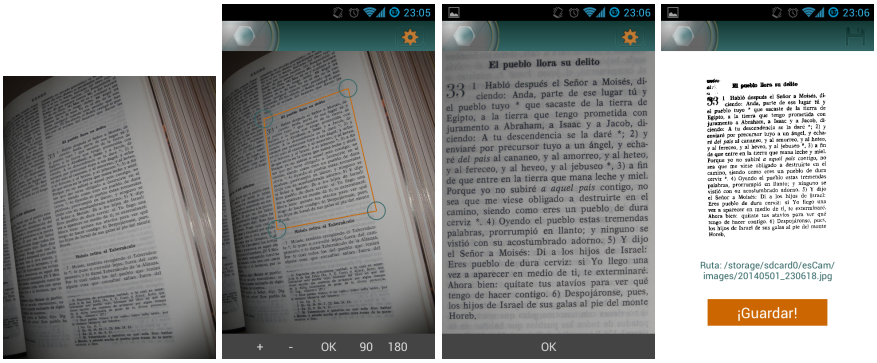
Redes neuronales [1]: una técnica basada en un perceptrón multicapa que recorre la imagen y, a partir de una ventana de píxeles, genera una salida con el valor del píxel limpio. La red neuronal ha sido entrenada previamente con muestras similares a los documentos que se van a escanear o fotografiar.

3. Tecnologías utilizadas

Para el desarrollo de la aplicación, además del propio SDK destinado al desarrollo de aplicaciones para la plataforma de Android, se ha utilizado la librería OpenCV ya que ofrece mayor eficiencia en el cálculo relacionado con imágenes.

4. Conclusiones

En este trabajo se presenta una aplicación, que hemos llamado esCam, capaz de tratar las primeras etapas del preproceso, especialmente aquellas propias



(a) (b) (c) (d)

Figura 3. Capturas de la aplicación esCam: (a) Fotografía original, (b) Selección de la zona de la hoja para la corrección de perspectiva, (c) Imagen con la perspectiva normalizada, (d) Imagen final tras aplicar filtros de limpieza y binarizado.

de las imágenes tomadas con dispositivos móviles: corrección de perspectiva y limpieza y binarizado. Con esto se consigue una mejor representación de la información para las siguientes etapas como el análisis del documento (extracción de lineas, layout) y/o reconocimiento y búsqueda de palabras clave. Además se reduce el tamaño de la imagen tras aplicar binarizados y eliminar partes de la fotografía que no se corresponden con el documento. La aplicación se encuentra aun en fase de desarrollo pero debe hacerse notar que los algoritmos de corrección de la perspectiva y los algoritmos adaptativos funcionan lo suficientemente bien para continuar en la misma línea de la aplicación que se ha venido desarrollando.

Referencias

1. España Boquera, S., Castro-Bleda, M.J., Gorbe-Moya, J., Zamora-Martinez, F.: Improving offline handwritten text recognition with hybrid hmm/ann models. Pattern Analysis and Machine Intelligence, IEEE Transactions on 33(4), 767–779 (2011)
2. Gatos, B., Pratikakis, I., Perantonis, S.J.: Adaptive degraded document image binarization. Pattern recognition 39(3), 317–327 (2006)
3. Hidalgo, J.L., España, S., Castro, M.J., Pérez, J.A.: Enhancement and cleaning of handwritten data by using neural networks. In: Pattern Recognition and Image Analysis, pp. 376–383. Springer (2005)
4. Mori, S., Suen, C.Y., Yamamoto, K.: Historical review of ocr research and development. Proceedings of the IEEE 80(7), 1029–1058 (1992)
5. Nagy, G.: Twenty years of document image analysis in pami. IEEE TPAMI 22 (2000)
6. Otsu, N.: A threshold selection method from gray-level histograms. Automatica 11(285-296), 23–27 (1975)
7. Pastor-Pellicer, J., Zamora-Martínez, F., España-Boquera, S., Castro-Bleda, M.J.: F-measure as the error function to train neural networks. In: Advances in Computational Intelligence, pp. 376–384. Springer (2013)
8. Sauvola, J., Pietikäinen, M.: Adaptive document image binarization. Pattern Recognition 33(2), 225–236 (2000)