APLICACIÓN BASADA EN ANDROID PARA LA TRADUCCIÓN DE TEXTO EN IMÁGENES DE ESCENAS

Marc Armengol Sayavera

Universidad Autónoma de Barcelona

RESUMEN

El propósito de este documento es el de consignar detalladamente el desarrollo de una aplicación basada en Android para la traducción de texto en imágenes de escenas. Para el reconocimiento de texto se ha integrado el módulo de reconocimiento de texto de la librería OpenCV, y para realizar la traducción se ha integrado la API de Google Translate. A partir de la aplicación, se realiza un análisis del rendimiento para evaluar el comportamiento del dispositivo frente a la aplicación, donde los resultados obtenidos se acercan bastante en precisión del reconocimiento de texto frente a otros resultados obtenidos mediante varias implementaciones de otros métodos para el reconocimiento de texto para PC.

Palabras clave: reconocimiento, texto, aplicación.

ABSTRACT

The purpose of this document is to record in detail the development of an Android based application for translating text in images of scenes. For text recognition has been integrated text recognition module of OpenCV library, and for translation has been integrated Google Translate API. From the application, performance analysis is performed to evaluate the behavior of the device to the application, where results are very close in text recognition accuracy compared with other results obtained by some implementations of other methods for text recognition to PC.

Keywords: recognition, text, application.

1. INTRODUCCIÓN

Durante la última década, la comunidad de investigación de visión por computador, ha dedicado un esfuerzo significativo en la investigación de los sistemas de extracción de texto en imágenes de escenas naturales. Como resultado, los métodos de extracción de texto en imágenes de escenas han evolucionado sustancialmente y su precisión se ha incrementado drásticamente en los últimos años [1]. Se puede ver la evolución de los algoritmos de detección en la Fig. 1, que muestra los resultados de las últimas competiciones de ICDAR. Anotar que aunque los conjuntos de imágenes sobre los cuales se han realizado las diferentes competiciones no son los mismos, los resultados son razonablemente comparables. Sin embargo, el problema aún está lejos de ser considerado como resuelto, debido a que los métodos ganadores en la última competición ICDAR obtienen solamente un 66% de precisión en la tarea de localización de texto y un 74% en la tarea de segmentación del texto.

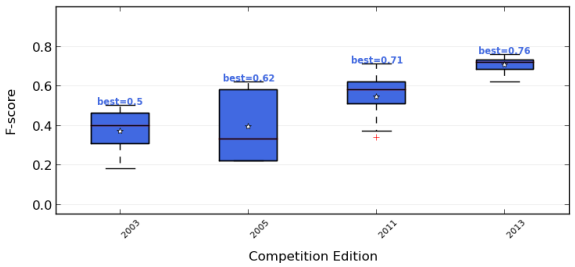

Fig. 1. Evolución de los resultados de la detección de texto de los métodos participantes en las competiciones ICDAR. Imagen obtenida de [2].

El objetivo principal de este experimento se basa en adaptar uno de estos métodos en un aplicativo móvil y ver los resultados que se obtienen tras procesar las imágenes con el dispositivo. Además, para el desarrollo de la aplicación, se pretende como objetivo aplicar las técnicas de toma de requisitos y documentación de la Ingeniería del Software, así como las de diseño, implementación y pruebas.

2. ESTADO DEL ARTE

El desarrollo de sistemas de comprensión de texto en escenas son necesarios para liberar el potencial de esta tecnología. Más allá de las piezas que los integran, estos sistemas constituyen un reto para la investigación, ya que tienen que proveer respuestas precisas y pertinentes, lo que implica una buena comprensión de la imagen y del contexto del usuario. Actualmente, existen aplicaciones de extracción de texto, como WordLens y Google Translate, que han adquirido madurez a nivel de mercado. Sin embargo, estas aplicaciones se limitan a unas condiciones bien definidas (i.e. imágenes orientadas horizontalmente o con el texto bien centrado) y, a menudo se basan en un modelo cliente-servidor y requieren de una infraestructura de datos muy grande. Los primeros intentos de reconocimiento de textos en escenas mediante métodos sin restricciones (en el sentido de no utilizar léxicos predefinidos focalizados) se propusieron hace más de diez años [3], pero los trabajos que reportaban resultados conjuntos de imágenes públicas datan de hace apenas unos años [4,5].

La gran mayoría de las arquitecturas se construyen alrededor de un proceso incremental, donde el texto de localización/segmentación es seguido por algún paso de análisis (como agrupación de palabras o de líneas de texto), y finalmente por un módulo de reconocimiento independiente. Está demostrado que estas arquitecturas simplistas pueden actualmente producir buenos resultados si se combina OCR con una segmentación del texto decente.

Otros enfoques han presentado arquitecturas más compactas para el reconocimiento del texto, como la integración de análisis de distribución y reconocimiento [4,6], o la detección y reconocimiento de caracteres [7,8]. El reconocimiento en estos casos se basa generalmente en simples combinaciones de clasificadores de caracteres y modelos de lenguaje ad-hoc que están mucho menos desarrollados que sus contrapartes para el procesamiento de imágenes. Como resultado, estos métodos generalmente no son capaces de producir mejores resultados que las

soluciones mencionadas anteriormente utilizando módulos de OCR.

3. METODOLOGÍA

Este apartado se centra en detallar la parte más técnica del desarrollo de la aplicación, desde los rasgos más generales de su funcionamiento hasta los rasgos más específicos del núcleo. Además, se centra también en la parte de implementación de la aplicación dónde se detalla el cómo y con qué herramientas se ha realizado el desarrollo de la aplicación.

3.1 REQUERIMIENTOS Y DISEÑO DE LA APLICACIÓN

A partir de un análisis de requisitos previo al desarrollo de la aplicación, la funcionalidad principal de la misma se centra en el análisis de imágenes mediante el módulo integrado de reconocimiento de texto de OpenCV[1]. La obtención de las imágenes para su análisis se puede hacer mediante la cámara del dispositivo, limitando el número de imágenes a una por análisis, o bien mediante la selección de múltiples imágenes almacenadas en la memoria del dispositivo, que permite, de este modo, un análisis de múltiples imágenes a la vez. Tras la realización de un análisis, ya sea de una sola imagen o de múltiples, la aplicación dibuja en cada una de las imágenes el texto reconocido y después, permite el almacenamiento del mismo, o bien la traducción del texto reconocido en cada una de las imágenes mediante la API de GoogleTranslate. Si se decide almacenar el análisis, la aplicación almacena todas las imágenes analizadas en la memoria del dispositivo.

Por otra parte, si se decide realizar la traducción, la aplicación substituye el texto original detectado por el traducido en el idioma indicado y además, permite también almacenar el resultado de la traducción. Para establecer tanto el idioma de la aplicación como el idioma de traducción del texto reconocido en las imágenes analizadas, entre otras opciones, la aplicación dispone de un menú de opciones para indicarlos. Finalmente, la aplicación permite visualizar los resultados de todos los análisis realizados, ordenados de manera cronológica.

3.2 DETALLES TÉCNICOS

Entrando un poco más en el núcleo de la aplicación, ésta permite realizar el análisis de las imágenes mediante dos funciones nativas que ejecutan dos versiones del algoritmo de detección de texto: DetectTextMSER y DetectTextER. Estas dos funciones implementan los algoritmos para la detección de texto descritos detalladamente en [4,6,9], y lo que hacen es realizar una serie de hipótesis sobre las localizaciones de texto que detectan, y posteriormente son verificadas.

La función DetectTextER requiere de los siguientes parámetros: matAddrGr (*long*) y matAddrRgba (*long*), que son punteros a la imagen que se va a procesar (*escala de grises y color respectivamente*); filters (*long*), que es un puntero a una serie de filtros inicializados previamente para el detector de texto; boxes (*long*), que es un puntero a una matriz donde se guardan los resultados del algoritmo de detección.

Los filtros nombrados anteriormente utilizan una serie de valores para filtrar las hipótesis sobre las localizaciones de texto detectadas para poder determinar si las hipótesis son correctas o se descartan. La generación de los filtros para el algoritmo de detección de texto, se realiza mediante dos funciones distintas: createERFilterNM1(*const Ptr<ERFilter::Callback> &cb, int thresholdDelta=1, float minArea=0.00025, float maxArea=0.13, float minProbability=0.4, bool nonMaxSuppression=true, float minProbabilityDiff=0.1*) y createERFilterNM2(*const Ptr<ERFilter::Callback> &cb, float minProbability=0.3*). Los valores de estas dos funciones pueden ser variables, pero en este caso se han utilizado los recomendados por la documentación del módulo de texto de OpenCV. Una variación en estos parámetros puede afectar en diversos factores a los resultados de los análisis, como por ejemplo hacer variar el umbral de aceptación sobre el texto reconocido.

Por otra parte, la función DetectTextMSER, igual que la función anterior, ejecuta otra versión del algoritmo de detección de texto, y requiere los mismos parámetros que la función anterior pero sin los filtros, debido a que esta versión del algoritmo no los utiliza.

Finalmente, para la traducción del texto reconocido por cualquier de los dos métodos nombrados anteriormente, se lanza una petición GET mediante HTTP hacia la API de Google Translate, donde se envían como parámetros: el texto que se desea traducir, el idioma de dicho texto y el idioma al que se desea traducir el texto.

3.3 IMPLEMENTACIÓN

En una primera instancia se pretende realizar la integración del módulo de texto de la última versión compilada de OpenCV para Android, la 2.4.10, pero finalmente se realiza a partir de la versión 3.0.0 que, a pesar de ser una versión beta, proporciona ciertas mejoras respecto a la versión 2.4.10. El único y principal problema que presenta dicha versión, es que no existe aún una versión para Android, y se procede a realizar una compilación cruzada de la API en un sistema operativo Windows 8.1.
Para el desarrollo de la aplicación se ha utilizado la última versión de la interfaz de desarrollo Eclipse (*Juno)*, configurada con el ADT (*Android Development Tools)* junto con las librerías obtenidas tras la compilación cruzada de la versión 3.0.0 de OpenCV. Además, se ha creado una cuenta para poder utilizar la API de Google Translate y se ha integrado en la aplicación.
Una vez el entorno está configurado, se dispone de un dispositivo móvil con la versión 4.3 de Android para realizar las depuraciones y pruebas para poder desarrollar la aplicación.

4. APLICACIÓN

La aplicación consta de un menú principal, donde se podrá acceder diversas pantallas. Siguiendo los botones de la Fig. 2, desde arriba en orden descendente, el primero lleva a otra pantalla donde se podrá escoger si se desea realizar un análisis de imágenes de la memoria del dispositivo o bien realizar un análisis de una imagen tomada desde la cámara del dispositivo. El segundo botón permite ir a una nueva pantalla para visualizar los resultados de los análisis anteriores. Dichos resultados, están ordenados por fechas y, en cada fecha, ordenados de manera numérica. El último de los botones es el menos relevante, pues simplemente lleva a una pantalla para escoger las opciones de la aplicación (i.e. idioma de detección o idioma de traducción).

[1] http://docs.opencv.org/trunk/modules/text/doc/erfilter.html

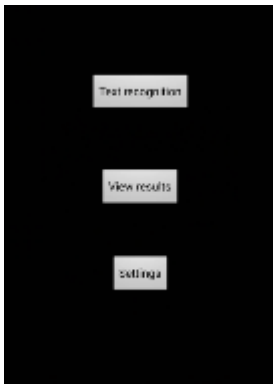

Fig. 2. Menú principal

La Fig. 3, nos muestra la opción de realizar un análisis mediante la toma de una fotografía de la cámara del dispositivo. La toma de la fotografía se realiza pulsando simplemente en la cámara.

Fig. 3. Cámara.

La Fig. 4, nos muestra la opción de guardar los resultados o de traducirlos, tras haber realizado la fotografía con la cámara. Si guardamos el resultado, se vuelve al menú principal, y si se decide traducir el resultado, mostrará la imagen de la Fig. 5, que posteriormente a la traducción, se ofrece también la opción de guardar el resultado.

Fig. 4. Se muestra el resultado de la detección.

Fig. 5. Se muestra el resultado de la traducción.

La Fig. 6, muestra la visualización de los resultados para una fecha determinada ordenados numéricamente. Desplegando el análisis deseado se muestran todas las imágenes analizadas en él. Si se pulsa sobre cualquier imagen, muestra el resultado guardado.

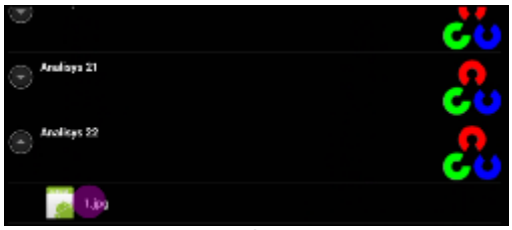

Fig. 6. Visualización de los resultados.

5. PROTOCOLO DE EVALUACIÓN

El propósito de este apartado es el de detallar el protocolo seguido para la evaluación del rendimiento de la aplicación dónde se pretende comparar la versión Android con la versión PC del método de extracción de texto. Por otra parte, se detalla también el protocolo seguido para evaluar el rendimiento respecto a las imágenes tomadas mediante la cámara del dispositivo.

Previamente se ha desarrollado una simple aplicación para PC, para poder realizar las mismas pruebas que en la aplicación móvil, y así poder realizar una comparativa respecto a los resultados del dispositivo, pero en este caso, solamente con una de las versiones del método de reconocimiento de texto, la versión DetectTextER. Los experimentos se realizan sobre el conjunto de datos de ICDAR2013 que se compone de 233 imágenes de escenas reales con diferentes tamaños. Para comparar los resultados obtenidos con los correctos, existe el mismo conjunto de datos con el texto de todas las imágenes reconocido, que será el ground truth sobre el que los evaluaremos. El ground truth se define a nivel de palabras mediante recuadros delimitadores orientados sobre el eje de abscisas, y sus correspondientes anotaciones de texto sobre la imagen.

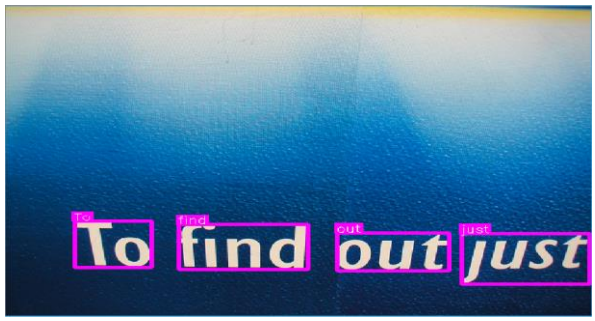

Fig. 7. Imagen del conjunto de datos del ground truth

A partir del ground truth y de los resultados obtenidos, se realiza un proceso para la evaluación de los resultados que consta de dos partes: la Localización del texto y su Transcripción. En la Localización, para que una detección sea considerada como correcta, el área de superposición a_0 entre el recuadro delimitador predicho B_p y el recuadro delimitador del ground truth B_{gt} debe superar el 50% mediante la siguiente fórmula [10]:

$$a_o = \frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})},$$

dónde la parte superior de la fórumla denota la intersección entre el recuadro predicho y el recuadro del ground truth, y la parte inferior su unión. Las detecciones resultantes son asignadas a las anotaciones de una imagen del ground truth satisfaciendo el criterio de superposición con el fin de ser clasificadas por la (decreciente) confianza. En la parte de Transcripción, para que un texto detectado sea considerado correcto, todos y cada uno de los caracteres deben coincidir con los del ground truth, ya sean mayúsculas o minúsculas.

Para que un texto detectado y transcrito sea considerado como correcto, debe haber superado con éxito las dos partes del proceso de evaluación. Una vez se han obtenido el total de palabras correctas, se calcula la Precisión y el Recall del análisis realizado. La Precisión es un valor entre 0 y 1 que se calcula mediante la división entre el total de palabras correctas y el total de palabras detectadas en el análisis. Cuanto más se acerque el valor de la división a 1, mayor Precisión habrá tenido el método utilizado para realizar el análisis. Por otra parte, el Recall es otro valor entre 0 y 1 que se calcula mediante la división entre el total de palabras correctas y el total de palabras del ground truth. Cuanto más se acerque el valor de la división a 1, mayor Recall habrá tenido el método utilizado para realizar el análisis.

Para la evaluación del rendimiento de la cámara del dispositivo, se realizan una serie de pruebas para determinar en qué rango de resoluciones ésta obtiene mejores resultados. Para ello, se toma una fotografía del mismo texto con distintas resoluciones de la cámara y, para cada resolución diferente, se toma una fotografía del mismo texto desde distintos ángulos y desde distintas distancias.

Las pruebas de PC se realizan mediante un dispositivo con un procesador i5-2410M 2.3GHz y con una memoria RAM de 4GB, mientras que las pruebas de dispositivo móvil se realizan mediante un dispositivo con un procesador Exynos 4 Quad de cuatro núcleos a 1.4GHz, 1GB de memoria RAM y equipado con la versión 4.3 de Android.

6. DISCUSIÓN DE LOS RESULTADOS

Tras la realización de las pruebas de evaluación de rendimiento, en este apartado se detallan los resultados obtenidos.

La siguiente tabla muestra los resultados obtenidos tras la evaluación de los análisis realizados sobre el conjunto de datos de ICDAR2013 compuesto por 233 imágenes:

Método	Precisión	Recall	F-Score	Tiempo(s)
PC (ER)	0.46	0.36	0.40	0.85
Móvil(ER)	0.43	0.28	0.34	5.67
Móvil(MSER)	0.38	0.16	0.23	2.06

Table 1. Resultados de la evaluación de los análisis.

Como se puede presenciar en la tabla, el análisis realizado mediante PC, es el que mejores resultados ha obtenido a un tiempo de 0.85 segundos por imagen analizada de media. Bastante cerca le sigue el análisis realizado mediante dispositivo móvil para la misma versión del método de detección de texto en cuanto a la Precisión aunque, no ocurre lo mismo respecto al Recall, que cae casi un punto, ni con el tiempo promedio que sube hasta los 5.67 segundos de media por imagen analizada. Teniendo en cuenta el abismo que existe entre las características de los dos dispositivos y que la versión de la librería Tesseract, utilizada por ambos métodos, es menos flexible en su versión para Android, ya que no permite obtener un valor de confidencia para cada reconocimiento, cosa que en su versión para PC sí, y lo aprovecha para filtrar reconocimientos con baja confidencia. Más lejos se queda la otra versión del método, MSER, que la no utilización de filtros no le permite obtener una Precisión y Recalls más altos pero, sin embargo, baja el tiempo promedio a 2.06 segundos por imagen analizada. A continuación podemos ver los resultados de los análisis para la imagen anterior del ground truth por los distintos métodos y dispositivos que hemos utilizado para realizar las pruebas:

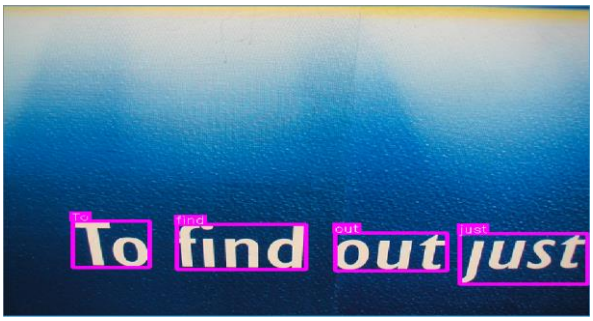

Fig. 8. Resultados del análisis de la versión del método ER para PC.

Fig. 9. Resultados del análisis de la versión del método ER para Android.

Fig. 10. Resultados del análisis de la versión del método MSER para Android.

En las imágenes anteriores se ven reflejados los resultados de la evaluación del rendimiento de la aplicación, donde la imagen analizada mediante la aplicación de PC se ajusta perfectamente a la del ground truth, la imagen analizada mediante la versión del método ER se acerca bastante a la del ground truth, aunque en vez de realizar un recuadro por cada palabra ha realizado uno para las cuatro, y la imagen analizada mediante la versión del método MSER presenta algunas imperfecciones tanto a la hora de detectar el texto como a la hora de transcribirlo.

Finalmente, comparamos los resultados obtenidos con los resultados obtenidos por diferentes métodos similares del estado del arte, que utilizaron el conjunto de imágenes de IDCAR2011. Dichos resultados pueden ser similares si se utiliza el conjunto de imágenes de IDCAR2013, por lo tanto, la siguiente comparativa es válida para el experimento [2].

Método	Precisión	Recall	F-Score
Milayev et al.	66.0	46.0	54.0
CSER + ABBYY	59.2	39.3	47.2
Yao et al.	49.2	44.0	45.4
Neumann and Matas	44.8	45.4	45.2
CSER + Tesseract	52.9	32.4	40.2
Neumann and Matas	37.8	39.4	38.6
Neumann and Matas	37.1	37.2	36.5
ER + Tesseract (Móvil)	43.0	28.0	34.0
MSER + Tesseract (Móvil)	36.0	16.0	23.0

Tabla 2. Comparación de los resultados con los obtenidos por otros métodos mediante el conjunto de imágenes de ICDAR 2011.

Tal y como se puede apreciar, en general los resultados obtenidos tras realizar el análisis del conjunto de imágenes de IDCAR2013, están bastante lejos de la mayoría de métodos del estado del arte. De todos modos, es importante anotar que aunque haya un abismo entre los resultados obtenidos mediante la versión del método MSER y los métodos del estado del arte, los resultados obtenidos mediante la versión del método ER, en cuanto a precisión superan a dos métodos del estado del arte y, sobretodo, anotar que es la versión Android del mismo mientras que los demás no lo son. Este punto es muy importante para nuestro experimento, pues pese a las restricciones tanto a nivel de la versión del método como a nivel de cómputo del dispositivo, los resultados son buenos.

Para evaluar el rendimiento de la cámara, se han realizado para tres tipos de resoluciones diferentes (una alta, una media y una baja), los dos tipos de pruebas nombrados en el apartado 4, pero en este caso, solamente para la versión ER del método de reconocimiento de texto. Primeramente, se analiza el efecto rotacional de una imagen con texto respecto a la cámara del dispositivo. Las Fig. 11, 12 y 13 muestran la fotografía tomada a 0º de inclinación respecto al eje de las abscisas, para cada una de las tres resoluciones y, como se puede apreciar, se obtiene el mismo resultado.

Fig. 11. Resolución de 960x720 píxeles.

Fig. 12. Resolución de 600x480 píxeles

Fig. 13. Resolución de 352x288 píxeles

Las Fig. 14, 15 y 16 muestran la misma fotografía tomada a una cierta inclinación respecto al eje de las abscisas de entre unos 10º y 15º aproximadamente. Tal y como se refleja en las imágenes, para una pequeña inclinación del texto de la imagen sobre el eje de abscisas solamente se detecta el texto, pero no llega a transcribirse.

Fig. 14. Resolución de 960x720 píxeles.

Fig. 15. Resolución de 600x480 píxeles.

Fig. 16. Resolución de 352x288 píxeles.

Las Fig. 17, 18 y 19 muestran la misma fotografía tomada con cierto grado de inclinación negativo respecto al eje de las abscisas, de unos -10º a -15º aproximadamente. Como se puede apreciar, para una resolución baja ya no se detecta la totalidad del texto y además, para ninguna de las resoluciones se transcribe el texto. De este modo, a partir de los resultados de las inclinaciones, tanto negativa como positiva, se aconseja tomar las fotografías lo más recto posible para obtener unos mejores resultados.

Fig. 17. Resolución de 960x720 píxeles.

Fig. 18. Resolución de 600x480 píxeles.

Fig. 19. Resolución de 352x288 píxeles.

Si se aumentan los grados de inclinación nombrados anteriormente, la aplicación ya no es capaz de reconocer el texto de la imagen. Las Fig. 20, 21 y 22 muestran la misma fotografía tomada a una inclinación de 45º respecto al eje de las abscisas. Por lo que se observa en los resultados de este caso, para una inclinación negativa de -45º también ocurriría lo mismo.

Fig. 20. Resolución de 960x720 píxeles.

Fig. 21. Resolución de 600x480 píxeles.

Fig. 22. Resolución de 352x288 píxeles.

Finalmente, se analiza el efecto de la distancia para cada una de las tres resoluciones diferentes. Teniendo en cuenta que la distancia también depende del tamaño del texto a reconocer, según el tamaño de la letra de la fotografía se consideran dos tipos de distancias para esta prueba, la distancia media y la lejana. La distancia corta no se evalúa en esta prueba ya que según los resultados de la prueba de rotación se ha podido ver que a distancias cortas la aplicación obtiene buenos resultados. Las Fig. 23, 24 y 25 muestran la misma fotografía para las tres resoluciones diferentes a una distancia media. Tal y como se puede apreciar, para una distancia media, a una baja resolución la aplicación no obtiene el resultado esperado, mientras que para una resolución media y alta sí.

Fig. 23. Resolución de 960x720 píxeles.

Fig. 24. Resolución de 600x480 píxeles.

Fig. 25. Resolución de 352x288 píxeles.

Las Fig. 26, 27 y 28 muestran la misma fotografía para cada una de las tres resoluciones desde una distancia lejana. Como se observa en los resultados, a una distancia lejana solamente a alta resolución se obtienen los resultados esperados, por lo que se recomienda utilizar la resolución más alta posible si se desea obtener buenos resultados.

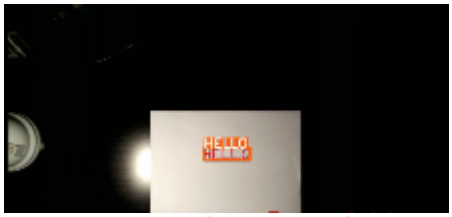

Fig. 26. Resolución de 960x720 píxeles.

Fig. 27. Resolución de 600x480 píxeles.

Fig. 28. Resolución de 352x288 píxeles.

7. CONCLUSIONES

Para concluir el experimento, anotar que se han logrado los objetivos respecto a realizar la integración de dos método para el reconocimiento de texto en un dispositivo móvil, utilizando las técnicas descritas anteriormente de la Ingeniería del Software para el desarrollo de la aplicación y, que los resultados de la evaluación del rendimiento del método en el dispositivo han sido positivos ya que se han acercado bastante a dos de los métodos cuyos hemos comparado. No obstante, anotar también, que si nos fijamos en los métodos cuyos hemos comparado que han obtenido mejores resultados, los resultados de la evaluación del rendimiento se quedan un poco lejos, y eso garantiza que aún queda mucho por investigar.

Agradecimientos: El autor quiere dar las gracias a D. Karatzas y L. Gomez por el soporte, consejo y ayuda en la implementación y evaluación de la aplicación desarrollada en este experimento.

8. BIBLIOGRAFÍA

[1] D. Karatzas et al. "ICDAR 2013 robust reading competition." *Document Analysis and Recognition (ICDAR),*

2013 12th International Conference on 25 Aug. 2013: 1484-1493.

[2] D. Karatzas and L. Gómez. “Scene text recognition: no country for old men?”.
[3] X. Chen, Xiangrong, and A.L. Yuille. "Detecting and reading text in natural scenes." *Computer Vision and Pattern Recognition, 2004. CVPR 2004*. II-366-II-373 Vol. 2.

[4] L. Neumann, and J. Matas. "A method for text localization and recognition in real-world images." *Computer Vision–ACCV 2010* (2011): 770-783.

[5] K. Wang, B. Babenko, and S. Belongie. "End-to-end scene text recognition." *Computer Vision (ICCV), 2011 IEEE International Conference on 6 Nov. 2011*: 1457-1464.

[3] Y.F. Pan, H. Xinwen, et al. "Text localization in natural scene images based on conditional random field." *International Conference on Document Analysis and Recognition (ICDAR) 2009.*

[4] L. Gomez and D. Karatzas. “Multi-script text extraction from natural scenes.” *International Conference Document Analysis and Recognition (ICDAR), 2013*.

[5] T. Novikova et al. "Large-lexicon attribute-consistent text recognition in natural images." *Computer Vision–ECCV 2012 (2012):* 752-765.

[6] L. Neumann and J. Matas. "Real-time scene text localization and recognition." *Computer Vision and Pattern Recognition (CVPR)*, 2012.

[7] C. Yao, B. Xiang, and L. Wenyu. "A Unified Framework for Multi-Oriented Text Detection and Recognition.", 2014.

[8] L. Neumann, and J. Matas. "Scene text localization and recognition with oriented stroke detection." *International Conference on Computer Vision (ICCV)*, 2013.

[9] L.Gomez and D. Karatzas. “A Fast Hierarchical Method for Multi-script and Arbitrary Oriented Scene Text Extraction”, 2014.

[10] A. Zisserman, C. K. I. Williams, J. Winn, L. Van Gool, M. Everingham and S. M. Ali Eslami. “The Pascal Visual Object Classes Challenge – a Retrospective.