Captura ergonómica de cheques bancarios mediante dispositivos móviles Android

Flavius Stefan Nicu

Resumen — Actualmente las empresas que no gestionan un elevado número de cheques deben acudir periódicamente a una oficina para depositarlos. En este artículo se plantea el desarrollo de una aplicación móvil Android que permita a los clientes de "La Caixa" capturar cheques de manera ergonómica, proporcionándoles un ahorro en dinero y tiempo de gestión. El sistema se basa en técnicas de visión por computador para detectar automáticamente los cheques, capturarlos y, finalmente, validar sus contenidos.

Palabras clave — Android, captura automática, detección de cheques, NDK, procesamiento a tiempo real, One-Class Support Vector Machine.

Abstract — Currently the companies that do not manage a large number of checks must periodically go to an office to make their deposit. This article describes the development of a mobile application for Android, that will allow to "La Caixa" customers to capture checks, save money and time. The system is based on computer vision techniques that allow it to automatically detect checks, capture them and finally validate their contents.

Index Terms — Android, automatic capture, check detection, NDK, real-time processing, One-Class Support Vector Machine.

1 Introducción

La entidad bancaria "La Caixa" recibe e ingresa diariamente una gran cantidad de cheques y pagarés provenientes de empresas. Para facilitar el ingreso, "La Caixa" ofrece un servicio de captura y envío de cheques. Este servicio solo se proporciona de manera gratuita a aquellos clientes que gestionan un elevado número de cheques diariamente. La solución consta de un escáner de cheques y un aplicativo on-line que envía los documentos de forma segura. A los clientes, esto les proporciona un ahorro de tiempo y costes administrativos ya que no tienen que acudir a una oficina para efectuar el ingreso. El escaneado y envío se puede realizar a cualquier hora y, además, el abono en cuenta se efectúa el mismo día en caso de los cheques.

1.1 Motivación del trabajo

La motivación de este proyecto consiste en buscar una alternativa que pueda ofrecer el mismo servicio a un mayor número de clientes de "La Caixa", evitando así el desplazamiento a una oficina para realizar el ingreso y proporcionar un ahorro en dinero y tiempo de gestión. Teniendo en cuenta que hoy en día casi todo el mundo dispone de un dispositivo móvil inteligente (Smartphone), este proyecto plantea el desarrollo de una aplicación móvil que permita a los clientes capturar los cheques de manera ergonómica, mediante la cámara del Smartphone. Una vez capturado el cheque, este se trata para encontrar automáticamente los datos de interés. De esta manera, la petición de ingreso del cheque se efectuará automática e instantáneamente mediante la conexión a Internet.

1.2 Objetivos

"La Caixa" actualmente ya dispone de una aplicación que permite hacer el proceso de captura e ingreso, pero de una manera muy pautada. La aplicación requiere una continua interacción del usuario para realizar el proceso de captura, necesitando encuadrar el cheque, buscar una iluminación óptima, enfocar la cámara y finalmente tomar la decisión de cuándo capturar el documento. Por otra parte, una gran variedad de aplicaciones como Scanbot [1], CamScanner [2] o CAPTIO [3] ofrecen la posibilidad de escanear documentos. Estas aplicaciones detectan cualquier tipo de documento, lo capturan, y ofrecen la posibilidad de extraer el texto que lo compone.

El objetivo principal del proyecto es proponer una serie de algoritmos para realizar la captura de cheques de manera mucho más ergonómica que las aplicaciones que existen a día de hoy en el mercado. Para la realización del objetivo de este proyecto, se deben cumplir los siguientes subobjetivos:

- Detectar y capturar el cheque en la escena: La aplicación final debe detectar y capturar el cheque de la escena automáticamente. Además, se deben implementar unas etapas correctivas para presentar la captura

- *E-mail de contacto: flaviusstefan.nicu@e-campus.uab.cat*
- *Mención realizada: Computación*
- *Trabajo tutorizado por: Marçal Rossinyol - Dept. Computación CVC*
- *Curso 2014/15*

como si de un documento escaneado se tratase, independientemente del punto de vista de la cámara.

- Rechazar las capturas a las que no se le pueda aplicar la extracción de datos: Se deben realizar diferentes comprobaciones para asegurar la posibilidad de extracción de datos de la captura. Por otra parte, también se deben rechazar los documentos que realmente no son cheques.

- Crear una aplicación Android para capturar cheques de forma automática en tiempo real: Una vez estudiadas y comparadas las diferentes técnicas para cumplir los objetivos anteriores, estas se combinan para la realización de una aplicación móvil Android. Se debe garantizar un alto rendimiento que permita la detección y captura en tiempo real.

Este artículo se estructura de la siguiente manera. En la sección 2 se analiza el estado del arte. La definición de detección de objetos se presenta en la sección 3 y en la 4 se detallan diferentes métodos de validación de captura. La implementación del sistema de captura en dispositivos Android se muestra en la sección 5 y en la sección 6 se evalúan los resultados obtenidos. Finalmente, en la sección 7 se repasan las conclusiones más importantes de este proyecto y en la sección 8 se proponen posibles mejoras y trabajos futuros.

2 ESTADO DEL ARTE

Describiremos en primer lugar el estado del arte en el problema de la detección de objetos, seguido de la clasificación de objetos y el procesamiento intensivo en Android.

2.1 Detección de objetos

El emparejamiento de imágenes o la detección de objetos es uno de los problemas más comunes en la Visión por Computador. Esta industria ha aumentado significativamente en los últimos años, aportando así, nuevas técnicas en el campo. En 1999, Lowe propone el detector de características SIFT (Scale Invariant Feature Transform) [4], que permite resolver el problema de la detección de objetos. SIFT representa una imagen u objeto en "puntos de interés" o características encontradas mediante DoG (Difference of Gaussians), es decir, la substracción de diferentes filtros de gaussianas convolucionados a la misma imagen. Estas características son invariantes al escalado y la rotación, y presentan robustez a cambios de iluminación y distorsión geométrica. En su estudio, Lowe propone aprovechar SIFT para encontrar las características de los objetos a detectar y guardarlas en una base de datos para el posterior emparejamiento. Las características de una nueva imagen son comparadas una a una con las características de los objetos almacenados en la base de datos. Para esto se realiza un *matching* [5] entre los descriptores locales, calculando la distancia Euclidea entre los vectores, donde una menor distancia implica una mayor semejanza.

La propuesta de Lowe presenta muy buenos resultados, pero el hecho de ser un algoritmo patentado y lento para el funcionamiento en sistemas de detección en tiempo real, motiva el estudio de nuevas técnicas. En 2011, Rublee propone el detector de características ORB (Oriented FAST and Rotated BRIEF) [6], que permite una detección robusta y en tiempo real, combinando el uso de los detectores FAST [7] y BRIEF [8].

No obstante, para encontrar documentos se puede utilizar una alternativa a la detección de objetos mediante características. Al tratarse de documentos de geometría conocida, se puede enfocar el proceso de captura en encontrar los contornos del documento y usar la posición de sus cuatro esquinas para poder deshacer los efectos de la perspectiva.

2.2 Clasificación de imágenes

A continuación, se detallan diferentes tipos de descriptores numéricos para representar imágenes. Dichos descriptores, usados en combinación con un clasificador estadístico (ej. kNN [9] o SVM [10]) son capaces de clasificar imágenes dependiendo de su contenido.

2.2.1 Descriptores locales

Como se ha comentado antes, SIFT y ORB extraen características locales de una imagen. La principal ventaja de estos descriptores locales es que son tanto invariantes al escalado como a la rotación, además de robustos a cambios de iluminación y distorsión geométrica. Aplicando un descriptor local a una imagen, se obtienen N vectores de características, donde N varía de imagen a imagen. Esto supone un problema ya que para los algoritmos de clasificación se necesita representar cada descriptor en un espacio M dimensional donde M sea constante para todos los descriptores. Como se explica anteriormente, el tamaño de la matriz generada aplicando SIFT u ORB, varía según la imagen, por eso, una posible solución a este problema consiste en crear un modelo *Bag-of-Visual-Words* [11].

2.2.2 Bag-of-Visual-Words

BoVW es una aproximación del modelo Bag of Words, transportado a un dominio visual. Como se puede observar en la figura 1, esta técnica consiste en crear un diccionario visual o "codebook" a partir de un conjunto de imágenes. De las imágenes se extraen descriptores y se agrupan mediante un método de agrupamiento o "clustering" como k-Means [12]. Si M es el número de centroides de cada agrupamiento, entonces el tamaño del diccionario visual es M.

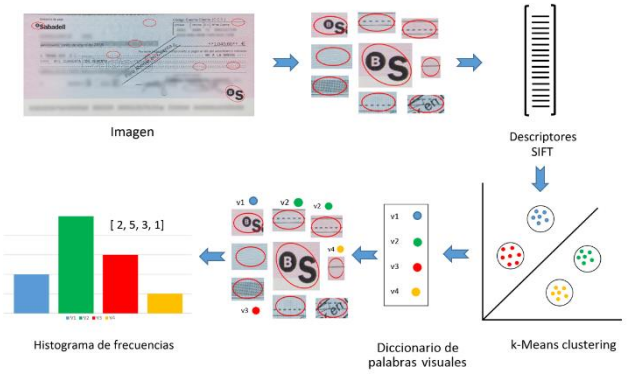


Fig. 1 - Esquema de los pasos realizados en el modelo Bag of Visual Words.

$$DIC = \{v_1, v_2, v_3, \dots, v_M\} \quad (1)$$

Dada una imagen, esta se representa por una serie de descriptores donde N es el número total de descriptores.

$$IMG = \{d_1, d_2, d_3, \dots, d_N\} \quad (2)$$

De esta manera, dado un descriptor d_k, este se asigna a una palabra del diccionario visual v_i, usando una medida de similitud como la siguiente distancia euclidiana:

$$v(d_k) = arg \min_{v \in DIC} dist(v, d_k) \qquad (3)$$

Donde d_k es el descriptor número k de la imagen y $v(d_k)$ es la palabra visual asignada al descriptor según la distancia $dist(v, d_k)$. Cada descriptor se asigna a una palabra del diccionario visual. Por lo tanto, una imagen es representada por un histograma de palabras visuales, donde el número de intervalos es igual al número de palabras en el diccionario M. Finalmente, el valor de un intervalo o subconjunto s_i del histograma, devuelve el número de apariciones que una palabra visual v_i presenta en una imagen. El histograma de frecuencias resultante de BoVW es un vector de tamaño constante donde se cuantifican los descriptores de las características extraídas de las imágenes.

2.2.3 Descriptores globales

Un descriptor global, como su propio nombre indica, representa una imagen de forma global, de manera que no busca puntos de interés en la imagen. Estos descriptores son sensibles a cambios de escalado, rotación o distorsiones geométricas, pero da buenos resultados en la representación de formas y texturas. El descriptor piramidal es un ejemplo de descriptor global.

2.2.4 Descriptor piramidal

El descriptor piramidal [13] consiste en la descomposición piramidal de una imagen para generar un vector descriptor global. Como se ve en la figura 2, la construcción del descriptor corta recursivamente la imagen en regiones, de las cuales se calcula el valor medio y genera una posición en el vector final.

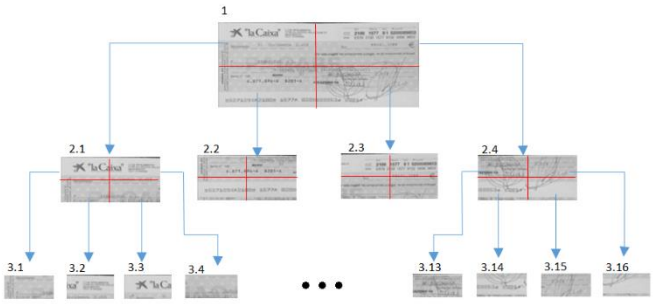


Fig. 2 - Esquema del árbol generado por el recorte recursivo de un descriptor piramidal.

Esta pirámide se divide en niveles donde el primer nivel corresponde al valor medio de la imagen global. El segundo nivel devuelve el valor medio de las 4 partes equivalentes recortadas sobre la imagen original. El tercer nivel devuelve el valor medio de las 16 partes equivalentes recortadas sobre la imagen original, y así sucesivamente en los L niveles definidos, siguiendo $\sum_{i=L-1}^{i=0} 4^i$. De esta manera, una pirámide de nivel 3 devuelve un vector de 21 posiciones (1+4+16) y una pirámide de nivel 5 devuelve un vector de 341 posiciones (1+4+16+64+256). Al ser un descriptor global siempre devuelve un vector de L posiciones, donde L es constante, independientemente de la imagen, y esto permite utilizar el descriptor directamente como entrada para los algoritmos de clasificación.

2.3 Procesamiento intensivo en Android

La visión por computador aprovecha los dispositivos móviles como los *smartphones* para generar aplicaciones que en un sentido u otro facilita la vida de las personas. La detección de peatones en los coches, visión aumentada y procesamiento de imágenes en los móviles, son ejemplos que ya han visto la luz. Sin embargo, todos estos procesos necesitan un gran rendimiento y estos dispositivos tienen recursos limitados. Por otra parte, Android se basa en el núcleo de bibliotecas Java, que funcionan en una máquina virtual Dalvik con complicación en tiempo de ejecución, lo que limita aún más el procesamiento intensivo. NDK [14] es una herramienta que permite incrustar y compilar código de bajo nivel como C o C++, aprovechando así los recursos disponibles para crear aplicaciones de procesamiento intensivo.

3 DETECCIÓN Y CAPTURA DEL CHEQUE

La AEB (Asociación Española de Banca) [15] ha especificado que el formato de un cheque debe ser de 80x175mm y una tolerancia de ±2mm, independientemente de la entidad bancaria. Por otra parte, también está fijada la posición de cada elemento que constituye el cheque, tal como se indica en el apéndice A1. Así pues, para detectar un cheque en una imagen se han estudiado diferentes técnicas, partiendo de la base de que un cheque es un documento rectangular de dimensiones y forma estándar o normalizada.

3.1 Detección basada en características

Para el propósito de esta aplicación, los cheques serán deslizados uno por uno sobre una superficie regular y la cámara del dispositivo móvil se encargará de capturar imágenes de la escena.

Como se explica en el apartado 2.1, una de las técnicas utilizadas para detectar objetos se basa en la búsqueda de "puntos de interés" o características que forman la imagen. Primeramente, se ha creado un programa que extrae las características de una imagen de entrada, en este caso un cheque, y las guarda en memoria. Seguidamente, recoge el flujo de imágenes producido por una webcam, extrae las características de cada *frame* y lo compara con las características de la imagen de entrada para comprobar si existe semejanza. En este experimento se han probado tanto SIFT como ORB. Para encontrar semejanzas entre características y por consiguiente encontrar el cheque en los *frames* de la *webcam*, los descriptores de las características de SIFT se comparan mediante una distancia euclidiana y en caso de ORB, mediante distancias de Hamming.

Realizando diferentes pruebas se ha observado que tanto SIFT como ORB detectan sin problemas el cheque en la escena, sin importar la escala, la rotación, pequeños cambios de luminosidad o incluso oclusiones, como muestra la figura 3.

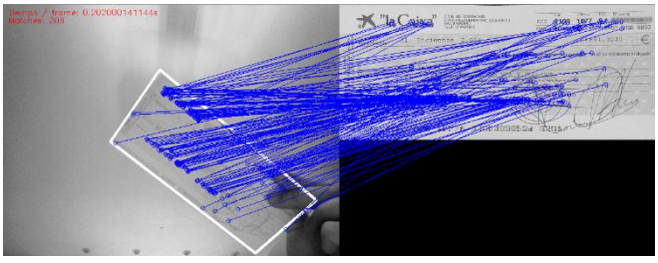


Fig. 3 – Detección de un cheque basada en características ORB mediante una webcam.

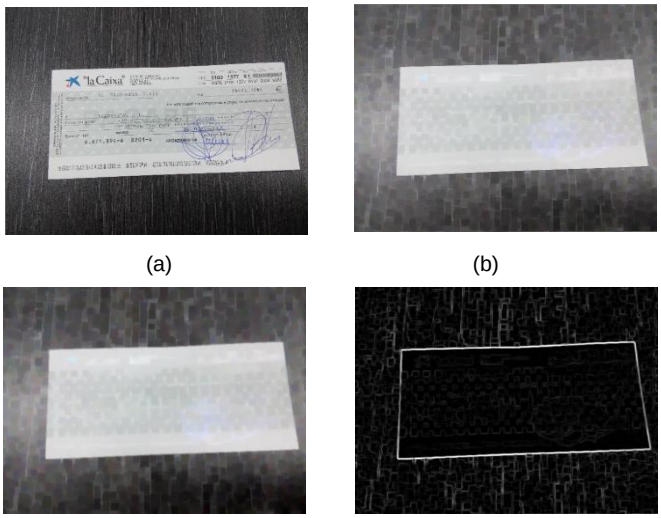
3.2 Detección basada en contornos

Dadas las circunstancias de captura mencionadas en el apartado 3.1, otra posible solución para detectar los cheques se basa en encontrar los contornos de la imagen.

3.2.1 Representación de bordes en la imagen

Para mostrar el proceso de esta detección, se toma como ejemplo una imagen que contenga un cheque (Fig. 4 a). En este proceso, solo interesan los contornos del cheque ya que aportan información sobre su localización. La búsqueda de contornos se basa en encontrar cambios de intensidad en la imagen, es por eso que el texto del cheque o la textura de la mesa pueden afectar en el proceso y se considera como ruido. El primer paso consiste en aplicar matemática morfológica, que elimina parte del ruido (Fig. 4 b). Se ha utilizado en este caso un elemento estructurante cuadrado de dimensiones 3x3 y el operador *closing* o cierre, y se ha iterado 5 veces sobre la imagen.

Para suavizar o eliminar pequeños cambios de intensidad, se utiliza una técnica de *smoothing (Fig. 4 c)*, concretamente *Gaussian Smoothing*, con un *kernel* de tamaño 5x5. Llegado a este punto, se utiliza el operador de Sobel para encontrar los posibles bordes de la imagen. El operador de Sobel calcula el gradiente de la intensidad de una imagen. Este operador utiliza un "kernel" de tamaño 3x3, uno vertical y otro horizontal para aplicar una convolución a la imagen y calcular aproximaciones a las derivadas. Formalmente, si IMG es la imagen original, G_x y G_y los gradientes en X y en Y respectivamente, se obtiene:



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

Fig. 4 – Proceso de la detección por contornos. En los apartados de esta figura se muestran: (a) imagen original; (b) resultado de aplicar operador *closing;* (c) *smoothing* aplicado; (d) canal de luminosidad del espacio de colores HSL.

$$G_x = \begin{matrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{matrix} \times IMG \qquad (4)$$

$$G_y = \begin{matrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{matrix} \times IMG \qquad (5)$$

$$S = \sqrt{{G_x}^2 + {G_y}^2} \qquad (6)$$

siendo S la imagen resultante después de aplicar el operador de Sobel a la imagen original. Este resultado cuantifica el cambio de intensidad de la imagen.

Otros detectores de bordes como el operador Canny [16] ofrecen buenos resultados y pueden ser utilizados en este paso. En este proyecto se ha optado por el operador de Sobel ya que no necesita tantos recursos de cómputo como los demás y los resultados que ofrece son suficientes para el propósito de este proceso. Así pues, se utiliza el operador de Sobel para cada uno de los 3 canales RGB y posteriormente se unifica el resultado de cada canal. La imagen resultante se convierte a una imagen HSL (*Hue, Saturation, Lightness*) y el posterior procesado se centra en el canal de luminosidad normalizado (Fig. 4 d).

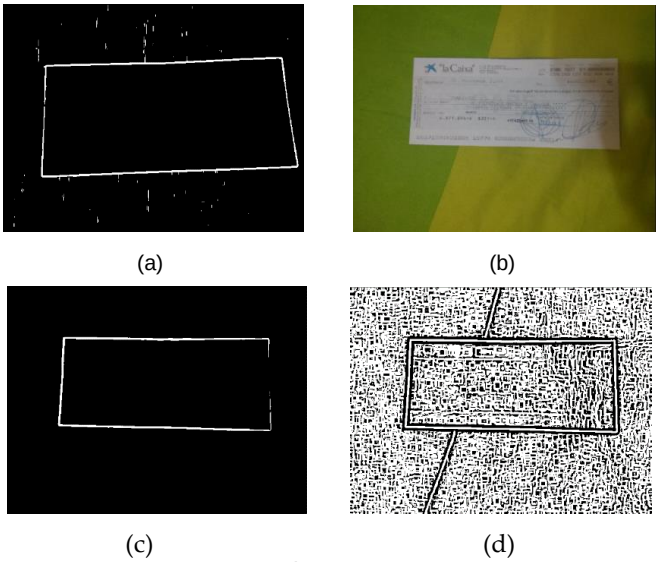
3.2.2 Obtención de contornos

Para aplicar una búsqueda de contornos sobre los bordes encontrados previamente, se debe fijar un criterio para definir lo que es contorno y lo que no. En este paso se utiliza una binarización del canal de luminosidad obtenido anteriormente, donde el resultado de la binarización devuelve una imagen con valores 0 y 1. Se toman los píxeles con valor 1 como parte del contorno y los de valor 0 como "background". El principio de binariazación se basa en la siguiente función:

$$g(x,y) = \begin{cases} 1, & f(x,y) > T \\ 0, & f(x,y) \leq T \end{cases} \qquad (7)$$

Donde $f(x,y)$ es la intensidad del píxel en las coordenadas x e y, T es el umbral y $g(x,y)$ es el valor resultante de aplicar la binarización. Para encontrar el umbral de la binarización se ha utilizado el método de Otsu [17]. Este se basa en cálculos estadísticos para encontrar el umbral óptimo de la región.

En las primeras pruebas se calculó el umbral de la imagen de forma global, de manera que, en las imágenes que presentan una luminosidad uniforme y sin sombras, se conseguía identificar el contorno del cheque (Fig. 5 a). Por lo contrario, si la imagen tenía sombras en alguna parte del contorno del cheque (Fig. 5 b), este contorno no se identificaba (Fig. 5 c). Para arreglar este problema, se utiliza el cálculo de umbral local o *Adaptative Thresholding* [18]. Esta técnica separa la imagen en bloques, y calcula el umbral para cada uno de ellos, evitando que una sombra afecte de manera global a la imagen (Fig. 5 d).



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

Fig. 5 – Proceso de la detección por contornos. En los apartados de esta figura se muestran: (a) binarización global mediante Otsu aplicada a (Fig. 4 a); (b) imagen que presenta sombras en los bordes del cheque; (c) binarización global mediante Otsu aplicada a (Fig. 5 b); (d) binarización local mediante Otsu aplicada a (Fig. 5 b).

3.2.3 Identificación del contorno del cheque

Tomando como ejemplo la imagen de contornos obtenidos mediante "Adaptative Thresholding", se puede identificar cada contorno y estimar el área de cada uno. Sin la presencia de ruido, el cheque constituye el contorno con el área más grande de la escena. A consecuencia de eso, se ordenan los contornos obtenidos según el área, y se eligen solamente los cinco mayores (Fig. 6 a), asegurándose así, que el área del cheque esté presente en esas posiciones (Fig. 6 b).

(a) (b)

Fig. 6 – Proceso de la detección por contornos. En los apartados de esta figura se muestran: (a) los cinco contornos de mayor área mostrados sobre el canal de luminosidad; (b) el contorno rojo mostrado sobre la imagen original

El siguiente seudocódigo resume las comprobaciones realizadas para determinar cuál de los contornos pertenece al cheque.

```
Procedimiento obtenerConornoCheque (contornosEnImagen)
  Para cada contorno ∈ contornosEnImagen hacer
    perímetro ← calcularPerimetro(contorno)
    aproximación ← encontrarPoligonoAproximado(perimetro)
    Si poligonoDeCuatroLados(aproximación) entonces
      altura, anchura ← obtenerAlturaAnchura(aproximación)
      Si anchura ≠ 0 entonces
        ratioTemporal ← altura / anchura
        Si RATIO_MIN ≤ ratioTemporal y RATIO_MAX ≥ ratioTemporal entonces
          contornoCheque ← aproximación
          Fin procedimiento
        Fin si
      Fin si
    Fin si
  Fin para cada
Fin procedimiento
```

3.3 Cambio de perspectiva

El dispositivo móvil estará en continuo movimiento, así pues, no se puede asegurar que el punto de vista de la cámara y la posición del cheque sean fijas. Aun así, la aplicación final debe simular la función de un escáner de cheques, de manera que la perspectiva del cheque siempre sea la misma. Para resolver este problema, utilizamos la localización de los contornos proporcionada por la detección de cheques. A partir de los contornos, se calcula la altura y anchura del cheque de la imagen, creando una plantilla temporal. En base a esto, se calcula la matriz de transformación entre las esquinas de la plantilla y las esquinas de los contornos del cheque. Así, mediante

la matriz de transformación, se aplica una transformación homográfica (Fig. 7) aplicando RANSAC [19].

Fig. 7 – Cambio de perspectiva aplicado para mostrar el cheque como si de un documento escaneado se tratase.

3.4 Detector implementado

El proyecto está enfocado a la creación de una aplicación móvil y este puede presentar diferentes prestaciones. El tiempo que conlleva cada proceso es un punto a tener en cuenta a la hora de elegir qué método de detección de cheque utilizar.

Para determinar qué método es el más adecuado para esta aplicación, se ha contabilizado el tiempo medio que tarda cada técnica en obtener la posición del cheque en la escena. Se debe tener en cuenta que para estas pruebas, el tiempo se ha empezado a contabilizar una vez cargada la imagen en memoria. A fin de comparación, este procedimiento se ha aplicado a 3 imágenes que contienen cheques, y se ha iterado 10000 veces cada método de detección.

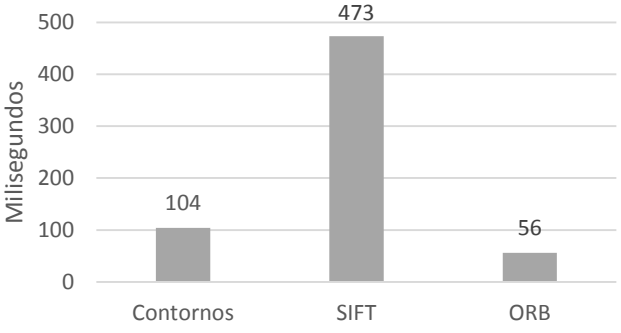


Fig. 8 – Tiempo medio en detectar un cheque en la escena. Se compara la detección basada en contornos y en características SIFT y ORB.

En la gráfica (Fig. 8) observamos que SIFT presenta un tiempo muy elevado. En comparación, la detección por contornos es casi cinco veces más rápida que la detección mediante SIFT, pero la mitad de rápido que ORB. En una primera instancia, podemos pensar que utilizar ORB para detectar los cheques es la mejor elección, pero debemos tener en cuenta que esta técnica busca parecidos entre par de imágenes. Esto significa que para encontrar la posición de un cheque en una imagen de entrada, esta se debe comparar con todos los modelos de cheques de las entidades bancarias que la aplicación puede aceptar. Es por eso que el tiempo de detección del cheque en la escena, se multiplica por N, donde N es el n-ésimo modelo con el que se compara la escena. La aplicación debe aceptar cheques de cualquier entidad bancaria, por tanto, la detección mediante contornos es significantemente más rápida que la técnica de detección por contenidos.

4 Validación de la captura

Una vez se haya detectado y capturado el cheque, en primer lugar se debe comprobar que se trata de una imagen enfocada de la cual posteriormente se pueda extraer información. En segundo lugar, si la imagen está enfocada, se debe verificar que realmente se trata de un cheque y no otro tipo de documento. En los siguientes subapartados se explican las diferentes técnicas utilizadas para conseguir estos objetivos.

4.1 Comprobación de enfoque

El operador de Laplace, es uno de los métodos utilizados en procesamiento de imágenes, para comprobar si la imagen está enfocada. En una imagen, este operador calcula la segunda derivada de la intensidad, tanto en el eje X como en el eje Y, de modo que, si $f''(x,y) = 0$, se produce un cambio de intensidad en $f(x,y)$. De esta manera, el operador de Laplace viene definido por:

$$Laplace(f) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \qquad (8)$$

Para verificar si la captura está enfocada, el operador de Laplace se aplica a la imagen de un cheque en escala de grises (Fig. 9).

Fig. 9 – El resultado de aplicar el operador de Laplace a una imagen enfocada se muestra en la izquierda y el resultado de una imagen desenfocada en la derecha.

De esta imagen se calcula la desviación estándar, de manera que se puede observar que en una imagen enfocada, σ es mayor que en una imagen desenfocada (Fig. 10). En este paso, se han realizado diferentes pruebas para encontrar el umbral a partir del cual una imagen está enfocada. Así pues, para este proyecto se ha definido:

$$Imagen = \begin{cases} enfocada, & \sigma \geq 10 \\ desenfocada, & \sigma < 10 \end{cases} \qquad (9)$$

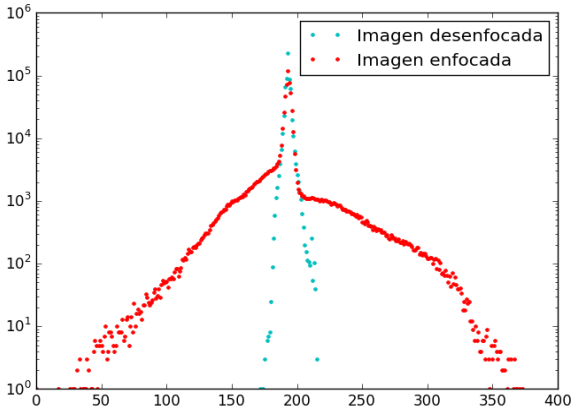


Fig. 10 - Comparación de la distribución de datos en una imagen enfocada y desenfocada después de aplicar el operador Laplace. El grafico muestra los datos en escala logarítmica en el eje Y. Imagen desenfocada $\sigma = 2.319$, imagen enfocada $\sigma = 20.710$.

4.2 Clasificación de la captura

En el proceso de detección y captura, existe la posibilidad de que un objeto de forma parecida a un cheque pase las diferentes comprobaciones, por eso, se debe identificar si realmente es un cheque lo que se ha capturado. En este proyecto se han comparado diferentes descriptores y métodos de clasificación supervisada, para evaluar su rendimiento y la posibilidad de implementación en la aplicación final.

4.2.1 Descriptores

En este artículo se analizan los descriptores locales SIFT y ORB con la respectiva cuantificación mediante BoVW, y también el descriptor piramidal que funciona a nivel global, tal como se explica a lo largo del apartado 2.2. No obstante, los descriptores ORB no han podido ser analizados debido a un error conceptual detectado en la cuantificación del descriptor. El modelo de BoVW no se puede aplicar tal cual con descriptores binarios como ORB ya que el algoritmo de K-means solo funciona con descriptores con números reales. En el espacio de Hamming, el concepto de media no está definido. Con lo cual se debería redefinir el modelo de BoVW usando un método de K-medoids [20] con distancia de Hamming para la construcción del diccionario.

Para determinar el tiempo medio de generación de los diferentes descriptores, este se ha calculado desde el almacenamiento de la imagen en memoria, hasta la obtención de un descriptor de dimensión fija, independiente de la imagen. De este modo, el tiempo incluye BoVW en el caso del descriptor SIFT. Con el fin de comparar, este procedimiento se ha aplicado a 3 imágenes aleatorias de diferentes tamaños, y se ha iterado 10000 veces para cada imagen. Los resultados son mostrados en la figura 11.

La diferencia de tiempos presentada entre SIFT y el descriptor piramidal se debe en primer lugar al tipo de descriptor. SIFT es un descriptor local, de manera que el tiempo de ejecución está correlacionado con el tamaño y el contenido de la imagen de entrada. Por otra parte el

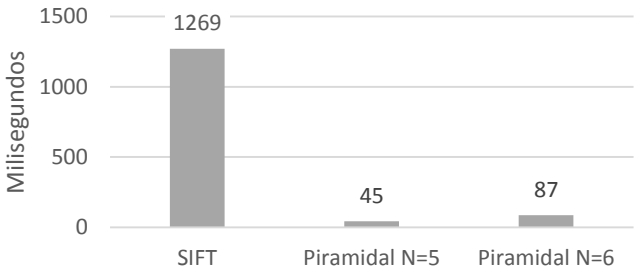


Fig. 11 – Comparación del tiempo medio en generar 10.000 descriptores de 3 imágenes aleatorias de diferentes tamaños mediante SIFT y el descriptor piramidal de nivel 5 y 6.

descriptor piramidal es un descriptor global, donde su complejidad se ve menos afectada por el tamaño de la imagen, y no es afectada por su contenido. La diferencia de tiempo entre los niveles 5 y 6 de este descriptor se debe a la complejidad del algoritmo. El nivel 6 genera un descriptor de cuatro veces más posiciones que el nivel 5.

A partir de los datos anteriores, se puede decir que la mejor opción sería utilizar los descriptores piramidales de nivel 5, siempre que en la clasificación de nuevas muestras presenten unos resultados aproximadamente iguales a los demás descriptores.

4.2.2 Métodos de clasificación supervisada

k-Nearest Neighbors (k-NN)

k-NN es un método de clasificación supervisada que consiste en situar vectores en un espacio N dimensional, asignando una etiqueta o clase a cada uno de estos vectores. En este caso los vectores serán los descriptores de las imágenes y se asignarán como clase positiva aquellas imágenes que sean cheques, y negativas, aquellas que no lo sean. Para predecir la clase de una nueva muestra, esta se sitúa en el espacio definido en la fase de entrenamiento y se busca los k vectores más cercanos a esta. Para encontrar la distancia entre una muestra y sus vecinos, en estas pruebas se ha utilizado la distancia euclidiana, pero se puede utilizar la distancia de Mahalanobis, entre otras. Finalmente, la etiqueta de la nueva muestra vendrá definida por la clase predominante en los k vecinos más cercanos, de manera que si $k = 1$, entonces la clase del vecino más cercano determina la clase de la nueva muestra.

Support Vector Machines (SVM)

Los SVM son algoritmos de clasificación supervisada que, al igual que k-NN, consiste en situar vectores en un espacio N dimensional, dotando con una etiqueta o clase a cada uno de estos vectores. SVM utiliza diferentes funciones *kernel*, lineales y no lineales (*polynomial*, *radial basis function*, *sigmoid*), que tratan de encontrar un hiperplano N dimensional, que separa de forma óptima los puntos de diferentes clases. De esta manera, dada una nueva muestra (descriptor de una imagen), ésta se sitúa en el espacio definido en la fase de entrenamiento y se predice la clase a la que pertenece según el modelo creado.

One-Class Support Vectors Machines (OC-SVM)

En este proyecto, se desea conocer si una muestra es cheque o no, pero, ¿qué significa que una muestra no sea cheque? Existen infinitas muestras que no sean cheques y a SVM se le debe definir en una fase de entrenamiento la etiqueta de estas. One Class SVM [21] permite resolver este problema, proporcionando solamente muestras de la clase positiva, en este caso, cheques. Para determinar la clase de una nueva muestra, el algoritmo trata de encontrar el radio óptimo de una hiperesfera, que envuelva y separe todo el conjunto de muestras positivas. De esta manera, si la proyección de un punto (descriptor de una imagen) en el espacio *N* dimensional está dentro del radio de la hiperesfera, este se etiqueta positivamente, y en el caso contrario, negativamente.

4.2.3 Evaluación de los métodos de clasificación

Para evaluar los resultados que proporciona cada clasificador, "La Caixa" ha facilitado un conjunto de 6000 muestras de clase positiva formado por cheques escaneados de diferentes entidades bancarias. En cuanto a la clase negativa (no cheque), no se dispone de un conjunto de imágenes que englobe este significado, ya que este está formado por infinitas muestras. En el caso de SVM i k-NN se debe proporcionar conjuntos tanto de clase positiva como negativa. Por eso, en las pruebas realizadas se ha creado un conjunto de 900 imágenes formadas por 101 clases diferentes [22] y que forman parte del conjunto de muestras negativas.

Para evaluar los diferentes modelos (Tabla 1), se ha realizado una validación cruzada de la combinación de los descriptores SIFT y piramidal de nivel 5 y 6, con los métodos de clasificación k-NN, One Class SVM y SVM (para cada "*kernel*").

	SIFT	Piramidal de nivel 5	Piramidal de nivel 6
3-NN	98,58%	97,00%	97,19%
LINEAR SVM	94,44%	97,49%	99,01%
POLY SVM	99,72%	97,49%	98,66%
RBF SVM	99,28%	99,49%	99,97%
SIGMOID SVM	3,86%	48,10%	51,12%
OCSVM	94,91%	90,41%	90,47%

Tabla 1 – *Accuracy* presentado por los diferentes métodos de clasificación utilizando los descriptores SIFT y el descriptor piramidal de nivel 5 y 6.

A falta de un modelo de clasificación que destaque sobre los demás, no se puede decidir mediante los resultados de clasificación el modelo que se debe utilizar, por lo consiguiente, se debe elegir el modelo que mejor se adapte a las necesidades de este proyecto. En este caso la rapidez a la hora de clasificar es un buen ejemplo teniendo en cuenta que el proyecto está enfocado a un sistema en tiempo real. Como se ha explicado en el apartado 4.2.2, el algoritmo k-NN calcula la distancia de una nueva muestra a todas las muestras entrenadas. En estas pruebas se eligen los 3 vecinos más cercanos a esta muestra. Por este motivo, el coste computacional a la hora de clasificar una muestra depende de la cantidad de muestras proporcionadas en la fase de entrenamiento y la *k* selecionada. Por lo contrario, el tiempo de clasificación de SVM y OCSVM es totalmente independiente al número de muestras entrenadas, así que esta penalización de tiempo descarta la utilización de un modelo k-NN.

En cuanto a SVM se necesita proporcionar tanto la clase "cheque" como "no cheque", pero como se explica previamente, la clase negativa que nosotros creamos es un subconjunto del infinito conjunto de muestras negativas. Por otra parte, OCSVM solamente necesita la definición de lo que es cheque para crear un modelo. Por estos motivos, se puede concluir que el método de clasificación OCSVM es el que más se adapta a las necesidades de este proyecto. Este modelo con descriptores piramidales de nivel 5 obtiene un 90,41% *accuracy*. No obstante, debido a que la aplicación final deberá detectar cheques reales y que el modelo se entrena a partir de un conjunto de cheques escaneados, esto puede influir en el resultado final.

5 IMPLEMENTACIÓN DEL SISTEMA EN ANDROID

La aplicación final implementa una serie de algoritmos que detectan y capturan de manera automática un cheque, por eso, el sistema necesita realizar un procesado intenso y en tiempo real. Para cumplir estos dos requisitos, se crea una librería que implementa el procesado de la imagen en C++. De esta manera, como se explica en el apartado 2.3, se importa esta librería en la aplicación Android mediante NDK y se crea una interfaz o *wrapper* entre C++ y Java mediante SWIG [23]. Para la implementación de las diferentes técnicas utilizadas en este proyecto, se utiliza también la librería OpenCV [24]. Esta librería ofrece una gran variedad de funciones optimizadas, lo que facilita el desarrollo de tecnologías de visión por computador.

Debido a que este proyecto se centra en implementar una serie de algoritmos en dispositivos móviles, esta aplicación no implementa el envío del cheque para la extracción de datos. Para mostrar el correcto funcionamiento del sistema de captura, se ha creado una sencilla aplicación (Fig. 12) que consta de una *Android activity* principal, con dos botones que dan acceso a:

1. Empezar a detectar cheques: Esta *activity* permite al usuario detectar, capturar y almacenar el documento en el dispositivo sin la necesidad de encuadrar, enfocar y pulsar un botón para capturar. Este proceso se realizara de manera automática, con la simple acción de pasar los cheques por delante del objetivo de la cámara. El dispositivo reproduce un sonido "*beep*" que informa al usuario de que se ha capturado y almacenado un nuevo cheque.
2. Galería de cheques: Esta segunda *activity* permite al usuario, con un simple deslizamiento sobre la pantalla, visualizar los cheques capturados por el dispositivo.

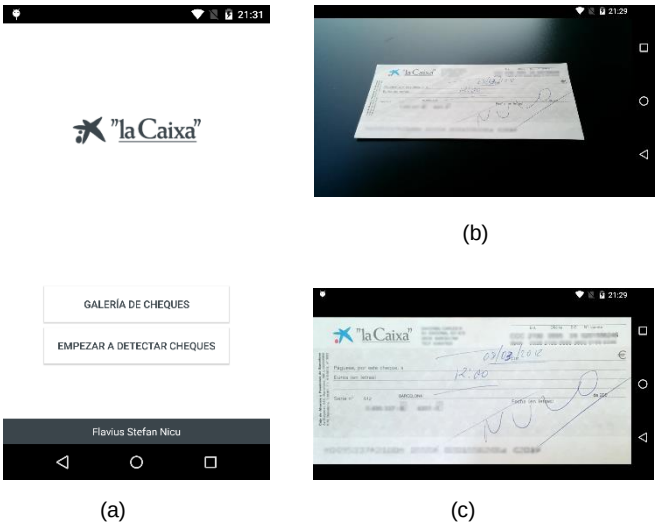


Figura 12 – (a) *Activity* principal de la aplicación; (b) detección de cheques; (c) galería de cheques

Para asegurar una continua usabilidad en la interfaz de usuario, desde el proceso principal de la aplicación se abre un nuevo hilo de ejecución para la detección y captura del cheque. De esta manera, durante el procesamiento de la imagen, los *frames* capturados por la cámara se van mostrando en la interfaz de usuario pero no todos se procesan. Cuando el hilo de ejecución acabe el procesamiento de la imagen, el último *frame* capturado por la cámara entra a procesarse. Para complementar esta explicación, en el apéndice A2 se puede ver el diagrama de flujo que refleja el funcionamiento de la aplicación.

6 Resultados finales

Una vez implementada la librería de captura en la aplicación móvil, se han realizado diferentes pruebas para evaluar cualitativa y cuantitativamente los resultados obtenidos con el estudio de cada técnica o método implementado.

Detección del cheque en la imagen basada en contornos

La detección de contornos busca bordes en imágenes y estos son generados por los cambios de intensidad. Por eso, para la correcta detección de los contornos, debe haber suficiente contraste entre el cheque y el *background.* Se han realizado pruebas de detección de contornos en imágenes con diferentes sombras y tipos de luminosidad. A pesar de utilizar *Adaptative Thresholding* como se explica en el apartado 3.2.3, si la imagen presenta sombras o destellos intensos en los bordes, estos no son detectados. No obstante, el usuario puede evitar este problema moviendo el punto de vista de la cámara del dispositivo.

Rechazo de capturas que no son cheque

Una vez implementado el sistema en la aplicación Android, se ha capturado 20 cheques reales, diferentes entre ellos, y 20 documentos de iguales proporciones al cheque. Con estas muestras se ha comprobado si el modelo responde de la misma manera que en el test de validación (90,41% *accuracy*). En la tabla 2 se puede ver los resultados de esta prueba.

		Clase predicha	
		Cheque	No cheque
Clase real	Cheque	20	0
	No cheque	14	6

Tabla 2 – Matriz de confusión obtenida con la captura de 20 cheques y 20 documentos de igual dimensión a un cheque, mediante la aplicación creada.

Se puede ver que en este caso todos los cheques se han clasificado correctamente y todos los errores de predicción se hallan en la clase "no cheque". El *accuracy* obtenido en este caso es del 65%. Este descenso en los resultados se debe a varias razones:

- El modelo de clasificación se crea en base a las imágenes proporcionadas de cheques escaneados y la aplicación trata de clasificar capturas de cheques reales.
- Los descriptores piramidales son descriptores globales, sensibles a cambios de iluminación, escalados, rotación y distorsión geométrica, lo que puede provocar que una muestra negativa se sitúe en el espacio de muestras positivas.
- Se utiliza OCSVM como método de clasificación. Este algoritmo busca una hiperesfera que engloba las muestras positivas y las separa de las negativas. Al proporcionarse solamente muestras positivas en el entrenamiento del modelo, este método no puede definir una separación exacta entre ambas clases.

Por otra parte, uno de los objetivos de este proyecto es capturar los cheques en tiempo real. Se ha realizado una prueba donde se mide el tiempo de captura de 20 cheques diferentes mediante la aplicación creada, y se han obtenido los siguientes resultados:

- Tiempo total en capturar 20 cheques: 77s.
- Tiempo medio en capturar 1 cheque (este tiempo incluye el movimiento del dispositivo móvil que se realiza al buscar otro cheque): 3.85s.
- Tiempo medio en procesar un *frame*: 2.21s.

7 Conclusiones

A lo largo de este documento se han estudiado y comparado diferentes técnicas que permiten llegar a implementar, un sistema final que captura de manera ergonómica, cheques bancarios mediante dispositivos móviles Android. Las decisiones que se han llevado a cabo durante este proyecto se han tomado en base a los recursos y tiempo de cálculo necesario para cada subproceso.

La aplicación implementa una detección de cheques mediante contornos. Esta detección es sensible a destellos de luz y sombras, pero es rápida y aplicable a cheques de cualquier entidad bancaria, ventajas de las que no disponen los detectores SIFT y ORB.

Las capturas se pueden realizar desde diferentes puntos de vista debido a que el dispositivo móvil está en

movimiento. Este problema se puede solucionar aplicando un cambio de perspectiva mediante RANSAC. Finalmente se puede definir un umbral que decida si la captura está enfocada o no mediante el operador de Laplace.

Para decidir si una captura es cheque o no, se han estudiado diferentes métodos de descripción de imágenes como SIFT, ORB y el descriptor piramidal. Se han comparado la utilización de estos descriptores para la clasificación mediante kNN, SVM y OCSVM. Debido a que el concepto de "no cheque" puede ser definido por infinitas muestras, el tiempo de extracción de descriptores y el tiempo clasificación, se ha decidido que lo más adecuado para proyecto es utilizar OCSVM con descriptores piramidales de nivel 5.

Para acabar, se ha creado una cadena de procesado o librería de captura que implementa los métodos elegidos. La implementación se ha realizado mediante OpenCV en C++, un lenguaje compilado que permite programar a bajo nivel y ofrece un gran rendimiento. Finalmente, esta librería se ha integrado en una aplicación Android para que cualquier usuario pueda capturar cheques bancarios de manera ergonómica.

8 Posibles mejoras y trabajo futuro

La separación de muestras de clase positiva y negativa mediante OCSVM y los descriptores piramidales presentan errores de predicción. Una posible solución a este problema sería, por una parte, acabar de investigar la cuantización de descriptores ORB mediante BoVW. Por otra parte, se podría reentrenar un nuevo modelo SVM, donde el conjunto de muestras negativas esté formado por capturas "no cheque" aceptadas por la aplicación actual.

Este proyecto ofrece la base para la creación de un sistema automatizado de captura documental bancaria. Como ampliación de la aplicación actual se podrían plantear los siguientes objetivos:

- Enviar las capturas en un servidor para extraer los datos del cheque y posteriormente enviarlo a la entidad bancaria del usuario.
- Crear una interfaz de usuario completa e integrar el proyecto para los sistemas operativos Android, iOS y Windows Phone.

Agradecimientos

Quiero mostrar mi agradecimiento a Marçal Rossinyol, tutor de este trabajo final de grado, por orientarme, aconsejarme y por mostrar constante interés hacia el proyecto, ofreciéndome parte de su tiempo para resolver mis dudas.

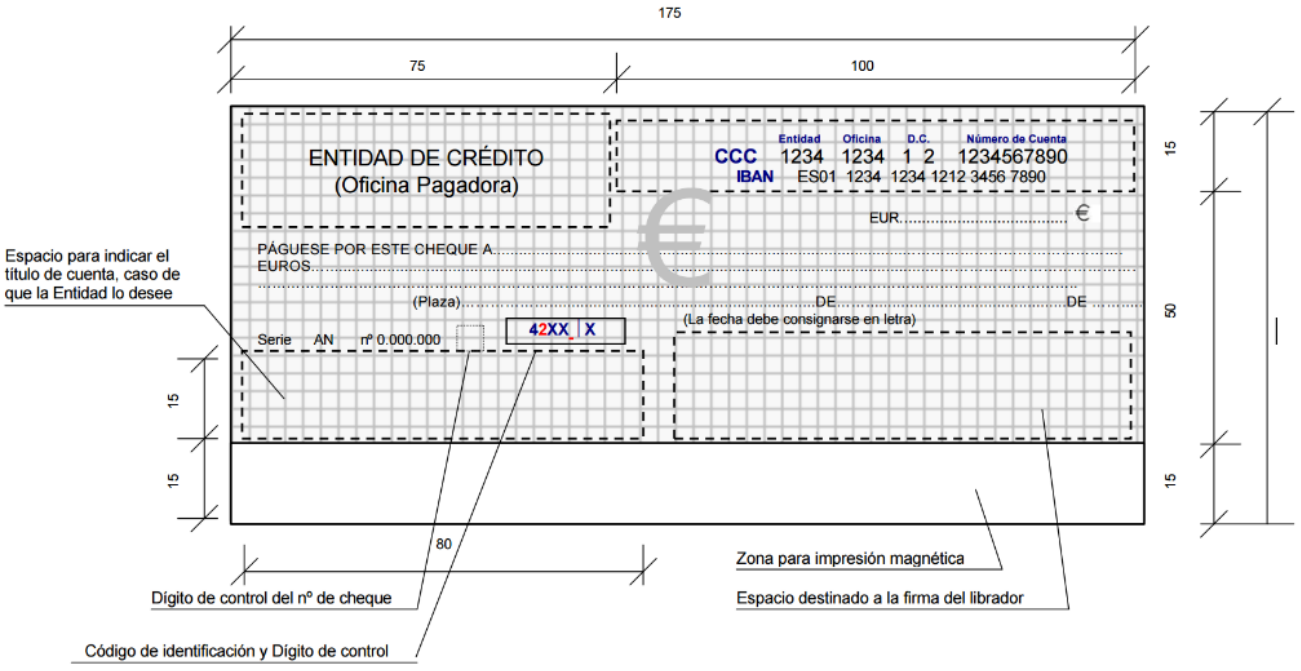
Por otra parte, me gustaría agradecer a Jordi Cortada por ofrecerme la posibilidad de participar en este proyecto junto a Marçal.

Referencias

[1] Scanbot, https://scanbot.io/en/ (25/06/2015)
[2] CamScanner, https://www.camscanner.com/ (25/06/2015)
[3] CAPTIO, http://www.captio.net/ (25/06/2015)
[4] D. G. Lowe. Distinctive image features from scale-invariant keypoints. IJCV, 60(2):91-110, 2004.
[5] Z. Zhang and R. Deriche *et al.*, "A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry", *INRIA*, nº 2273, France, (1994).
[6] Rublee, E.; Rabaud, V.; Konolige, K.; Bradski, G., "ORB: An efficient alternative to SIFT or SURF," Computer Vision (ICCV), 2011 IEEE International Conference on, vol., no., pp.2564,2571, 6-13 Nov. 2011.
[7] E. Rosten and T. Drummond. Machine learning for highspeed corner detection. In European Conference on Computer Vision, volume 1, 2006.
[8] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. Brief: Binary robust independent elementary features. In In European Conference on Computer Vision, 2010.
[9] Cover, T.; Hart, P., "Nearest neighbor pattern classification," *Information Theory, IEEE Transactions on* , vol.13, no.1, pp.21,27, January 1967.
[10] R. Duda, P. Hart, D. Stork. "Pattern Classification", second edition, September 3, 1997.
[11] Fei-Fei, L.; Perona, P., "A Bayesian hierarchical model for learning natural scene categories," *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on* , vol.2, no., pp.524,531 vol. 2, 20-25 June 2005.
[12] *Proc. Fifth Berkeley Symp. on Math. Statist. and Prob.*, Vol. 1 , 281-297 (Univ. of Calif. Press), 1967.
[13] Héroux, P., Diana, S., Ribert, A., Trupin, E.: Classification method study for automatic form class identification. In: Proceedings of the Fourteenth International Conference on Pattern Recognition, pp. 926–928, 1998.
[14] NDK, https://developer.android.com/intl/es/ndk/index.html (25/06/2015)
[15] AEB, https://www.aebanca.es/es/index.htm (25/06/2015)
[16] Canny, John, "A Computational Approach to Edge Detection," Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol.PAMI-8, no.6, pp.679, 698, Nov. (1986).
[17] N. Otsu. A Threshold Selection Method from Gray-Level Histograms," Systems, Man and Cybernetics, IEEE Transactions on, vol.9, no.1, pp.62, 66, Jan. (1979).
[18] J. Bernsen, Dynamic thresholding of gray-level images. In Int. Conf. Pattern Recognition, vol. 2, 1251– 1255, 1986.
[19] M. Fischler and R. Bolles, "Random sample consensus: A paradigm for model fitting with application to image analysis and automated cartography", *Communications of the ACM*, 1981.
[20] Costantino Grana, Daniele Borghesani, Marco Manfredi, and Rita Cucchiara. A fast approach for integrating orb descriptors in the bag of words model. , 2013.
[21] Yunqiang Chen; Xiang Sean Zhou; Huang, T.S., "One-class SVM for learning in image retrieval," *Image Processing, 2001. Proceedings. 2001 International Conference on* , vol.1, no., pp.34,37 vol.1, 2001.
[22] Caltech 101, http://www.vision.caltech.edu/ (25/06/2015)
[23] SWIG, http://www.swig.org/ (25/06/2015)
[24] OpenCV, http://opencv.org/ (25/06/2015)

APÉNDICE

A1. MODELO DE CHEQUE DE CUENTA CORRIENTE NORMALIZADO



A2. FUNCIONAMIENTO DE LA APLICACIÓN