

Autor: Miguel Ángel Rodrigo Lisbona

Tutor: Daniel Limón Marruedo

Reconocimiento de Sellos en “Archivo Histórico de Osborne”

Trabajo de Fin de Grado

Grado en Ingeniería de Tecnologías Industriales

Dep. Teoría de la Señal y Comunicaciones

Escuela Técnica Superior de Ingeniería

Universidad de Sevilla

Sevilla, 2013

Proyecto Fin de Carrera

Ingeniería de Telecomunicación

**Reconocimiento de Sellos en “Archivo Histórico de Osborne”**

Autor:

Miguel Ángel Rodrigo Lisbona

Tutor:

Daniel Limón Marruedo

Profesor titular

Dep. de Ingeniería de Sistemas y Automática

Escuela Técnica Superior de Ingeniería

Universidad de Sevilla

Sevilla, 2016

Trabajo de Fin de Grado: Reconocimiento de Sellos en “Archivo Histórico de Osborne”

|  |  |
| --- | --- |
| Autor: | Miguel Ángel Rodrigo Lisbona |
| Tutor: | Daniel Limón Marruedo |

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2016

El Secretario del Tribunal

*A mi familia*

*A mis maestros*

Agradecimientos

Blabla...

*Miguel Ángel Rodrigo Lisbona*

*Sevilla, 2013*

Resumen

La Fundación Osborne posee un archivo de documentos relacionados con la empresa que datan desde principios de 1800. Previas investigaciones han permitido descubrir información sobre la vida de Tolkien. Se sabe además, que existe correspondencia con casas reales o el Vaticano. También existe correspondencia de personajes públicos como Washington Irving. Los inicios de la empresa están estrechamente relacionados con otros personajes históricos como Fernán Caballero, cuñada del fundador, o “Frasquita” Larrea, su suegra. Por ello, podría haber información importante sobre dichos personajes que podría ser de valor histórico.

Para extraer dicha información, se ha optado por desarrollar una web donde se suben los archivos escaneados junto con su transcripción. Además de ello, se quiere desarrollar un sistema que realice dichas transcripciones de manera autónoma mediante detección de los caracteres manuscritos.

En este trabajo se describe el desarrollo de un sistema que reconozca los escudos en las cartas para clasificar su autor. Además, el sistema debe calcular la posición de dichos escudos con el objetivo de eliminarlos. Esto es necesario ya que los sistemas de detección de caracteres necesitan que los elementos que no son texto sean eliminados del documento antes de empezar cualquier análisis.

Abstract

Fundación Osborne owns an archive of documents related to the company dated up to early 1800. Previous investigations on the archive have revealed information about the life of Tolkien. Moreover, it is known that the archive has correspondence with royal houses as well as The Vatican. There are also letters of historical characters such as Fernán Caballero – founder’s sister-in-law – or “Frasquita” Larrea – his mother-in-law. For those reasons, it is reasonable to think that there might be some historical value on the documents that remain not investigated.

In order to retrieve that information, there is a website in development where scanned documents will be uploaded with their respective transcription and metadata. Those transcriptions are currently being typed manually. A system to make those transcriptions automated is being developed too by recognizing handwritten characters on the documents.

The purpose of this work is to describe the development of a system which recognizes seals on letters to automatically classify them by author. Moreover, the system calculates the position of such seal to delete it from the document. This is a need because character recognition system needs every non-text element removed from the image before any analysis is performed.

Índice

Agradecimientos ix

Resumen xi

Abstract xiii

Índice xiv

Índice de Tablas xv

Índice de Figuras xvii

Notación xix

1 Introducción 1

Referencias 11

Índice de Conceptos 13

Glosario 15

# **Índice de Tablas**

**No se encuentran elementos de tabla de ilustraciones.**

# **Índice de Figuras**

**No se encuentran elementos de tabla de ilustraciones.**

Notación

|  |  |
| --- | --- |
| A\* | Conjugado |
| c.t.p. | En casi todos los puntos |
| c.q.d. | Como queríamos demostrar |
| ∎ | Como queríamos demostrar |
| e.o.c. | En cualquier otro caso |
| e | número e |
| IRe | Parte real |
| IIm | Parte imaginaria |
| sen | Función seno |
| tg | Función tangente |
| arctg | Función arco tangente |
| sen | Función seno |
| sin*xy* | Función seno de *x* elevado a *y* |
| cos*xy* | Función coseno de *x* elevado a *y* |
| Sa | Función sampling |
| sgn | Función signo |
| rect | Función rectángulo |
| Sinc | Función sinc |
| ∂y ∂x  *x*◦ | Derivada parcial de *y* respecto  Notación de grado, *x* grados. |
| Pr(*A*) | Probabilidad del suceso *A* |
| SNR | Signal-to-noise ratio |
| MSE | Minimum square error |
| : | Tal que |
| < | Menor o igual |
| > | Mayor o igual |
| \ | Backslash |
| ⇔ | Si y sólo si |

# Introducción

Individual commitment to a group effort, that is what makes a team work, a company work, a society work, a civilization work.

- Vince Lombardi -

L

a Fundación Osborne “es el vehículo canalizador de la Responsabilidad Social Corporativa del Grupo Osborne”. Posee dos fines fundamentales: “la preservación y difusión del patrimonio histórico de la compañía y la promoción de la formación y el emprendimiento en las áreas de influencia del Grupo Osborne.” (citar web fundación Osborne)

La Fundación SEPI, entre otras actividades, “promueve y gestiona un extenso Programa de Becas de Inserción Laboral que cada año facilita la formación práctica en las empresas a más de 300 jóvenes titulados.” (citar web fundación sepi)

La Fundación Telefónica tiene por misión “mejorar las oportunidades de desarrollo de las personas a través de proyectos educativos, sociales y culturales, adaptados a los retos del mundo digital.” (citar web fundación tele)

Las tres colaboran en un proyecto denominado Lab Osborne. El objetivo de dicho proyecto es la digitalización del archivo histórico que posee la empresa. Dicho archivo contiene, por un lado, facturas, notas de pago y demás documentación de carácter administrativo y, por otro, correspondencia entre miembros de la familia y con otros personajes públicos desde la época de la fundación de la empresa en 1772. Entre estos personajes se encuentran Washington Irving o J.R.R. Tolkien así como miembros de distintas casas reales o El Vaticano.

Esta correspondencia podría tener valor histórico. De hecho, existe un libro sobre la vida de Tolkien donde una gran cantidad de datos han sido extraídos de investigaciones en este archivo.

No sólo eso, las cartas de miembros de la familia también podrían contener información importante ya que algunos estos miembros en los orígenes de la empresa fueron personajes públicos. Por ejemplo, la suegra de Thomas Osborne Mann, fundador de la empresa, fue Francisca Javiera Ruiz de Larrea y Aherán, **Frasquita Larrea**. Esta señora fue una escritora gaditana impulsora de la tertulia literaria que se cree que tuvo una gran influencia en la Constitución de 1812. Su cuñada, Cecilia Böhl de Faber y Larrea, **Fernán Caballero**.

Gran parte de la documentación que se posee tiene un contenido desconocido de ahí nace la motivación por digitalizarlo, ya que se podría analizar e investigar con mayor agilidad. Para ello, se desarrolla una página web donde se muestran los documentos escaneados junto con su transcripción. Esta transcripción es necesaria por un doble motivo: por un lado, los documentos tienen en su mayoría una caligrafía de muy difícil reconociminto lo cual ralentiza enormemente la lectura e interpretación. Por otro lado, si se quiere realizar algún tipo de búsqueda en función de alguna palabra que forme parte del texto, sería imposible sin transcripción previa.

Dicha transcripción se realiza a mano, pero es una tarea lenta y tediosa. Por ese motivo se quiere desarrollar, como parte del proyecto, un sistema visión artificial que reconozca el contenido manuscrito de los documentos y genere una transcripción de manera automática. Esto puede realizarse por distintos métodos, por ejemplo, mediante reconocimiento óptico de caracteres (OCR). Esto consiste en seccionar de alguna manera el texto en caracteres para más tarde utilizar algún método para reconocer a qué carácter corresponde cada una de las secciones de imagen previamente obtenidas. (citar artículo donde se ha hecho incluyendo precisión) El proceso de seccionamiento es costoso e impreciso (citar artículo) y puede dar caracteres divididos en dos regiones o varios caracteres en una misma región, lo que al final lleva a un reconocimiento erróneo de dichos caracteres. Por ello, otra forma de hacerlo es directamente reconocer palabras completas. Esto elimina el paso de seccionamiento de los caracteres lo que supone un gran avance, ya que el seccionamiento de palabras es mucho más simple y preciso generalmente (citar). Este segundo enfoque presenta, sin embargo, una gran desventaja. El método más utilizado para reconocer los caracteres o palabras consiste en entrenar un sistema mostrándole un gran número de caracteres o palabras, según corresponda. En el caso de las palabras, es evidente que existe una variedad mucho más elevada que de caracteres, por ello para obtener un reconocimiento razonablemente preciso es necesario una cantidad de texto de entrenamiento mucho mayor. Independientemente de cuál de las dos sea la solución adoptada, ambas requieren una serie de pasos previos.

1. Primero necesitamos una **binarización** que convierta las imágenes escaneadas en una matriz binaria. Deseablemente este sistema eliminará todos los píxeles de fondo del papel, incluyendo posibles manchas o texto transparentado perteneciente a la cara opuesta del papel. Este paso incluye posibles transformaciones morfológicas o filtrados que fueran necesarios con el objetivo de eliminar ruido procedente del uso de un papel heterogéneo, de manchas, de oscurecimiento de posibles pliegues, texto transparentado, etc…Una binarización ideal no elimina nada de los elementos gráficos del documento, incluyendo firmas, fechas, sellos, además de texto, pero eliminará todos los elementos que no contengan información.
2. Posteriormente, procedemos a **segmentar** el documento, localizando qué píxeles dentro de la imagen corresponden a texto manuscrito, cuáles a texto impreso, a sellos o firmas.
3. Una vez que sabemos qué parte del documento corresponde a texto, se procede a **dividirlo por líneas**. Esta división es sencilla, precisa y computacionalmente rápida. El método más utilizado consiste en sumar cuántos píxeles de texto hay en cada una de las filas de la imagen binarizada. Ni siquiera es necesario calcular mínimos locales, aquellas filas donde el número de píxeles es menor que un cierto umbral, corresponden con los huecos entre líneas de texto.
4. El siguiente paso es obtener las coordenadas que **delimitan cada palabra**. La forma más extendida para resolver este problema se denomina *etiquetado de componentes conexas* que se explicará en detalle más adelante. Una vez que tenemos las coordenadas de cada palabra, podemos extraer las secciones de la imagen original que las contienen para entrenar al sistema que reconocerá dichas palabras.

Existen otros enfoques alternativos para realizar este procedimiento, en (cita) se propone un método alternativo que se salta la fase de binarización y trabaja directamente con imágenes en escala de grises. Este procedimiento no presenta ventajas notables para el caso de imágenes de alta resolución. Las imágenes con las que trabajamos tienen una resolución de 3525x2810 píxeles. Por ese motivo, se ha preferido atenerse al método tradicional, ya que es más sencillo y no presenta desventajas notables. También existen métodos para eliminar la fase de segmentación. En (cita) Aouadi N. y Kacem A. proponen un método basado *GHT* (*Generalized Hough Transform*). Este método no requiere segmentación y, previo entrenamiento, consigue un sistema que detecta el texto a partir de parametrizaciones aprendidas. Aparentemente, este sistema consigue muy buenos resultados, sin embargo, requiere un entrenamiento. Dado que el método pretende evitar una segmentación automática errónea, dicho entrenamiento debe realizarse a partir de muestra de texto segmentada manualmente. Eso es precisamente lo que se quiere evitar aquí, por lo tanto este algoritmo no nos es válido.

Por todo ello, se ha optado por seguir un planteamiento más ortodoxo descrito con anterioridad e ilustrado en la figura a continuación.

Este trabajo pretende describir el desarrollo de un sistema de segmentación que detecte los sellos en los textos. Esto tiene un doble propósito:

1. Por un lado y como ya se ha descrito anteriormente, es necesario conocer qué partes de la imagen escaneada del documento corresponden o no a texto.
2. Por otro lado, conocer qué píxeles en la imagen corresponden a un sello permite clasificar dicho sello con el objetivo de asociar de manera automática un autor al documento en cuestión. La mayoría de los documentos corresponden a cartas y buena parte de ellos contienen un sello en la parte superior que corresponde al remitente de dicha carta.

# Algoritmos disponibles para seccionamiento de sellos

Everything is theoretically impossible, until it is done.

- Robert A. Heilein -

E

xisten múltiples maneras de resolver este problema. En (cita) se utiliza un enfoque frecuencial para diferenciar entre regiones gráficas (sellos, imágenes…) y texto. Pruebas con ese algoritmo demuestra no ser suficiente ya que algunos de los sellos presentes en los documentos agregan componentes de alta frecuencia igual que el texto, por lo tanto, eliminarían el sello de la misma manera que eliminan el texto.

Otro procedimiento probado, consiste en localizar formas geométricas mediante HT ya que muchos sellos presentan algún tipo de circunferencia o rectas de dimensiones de los que se presentan en el texto. Las pruebas no muestran buenos resultados ya que, en lugar de figuras geométricas de gran tamaño, se detectan múltiples círculos y rectas de pequeñas dimensiones.

Se ha probado, además, extraer las características mediante SURF de un sello modelo y comparar el número de emparejamientos en los documentos; si este número es suficientemente alto, se considera que dicho sello ha sido encontrado en el documento. Este método ofrece mucho mejores resultados, sin embargo, presenta el inconveniente de necesitar una muestra de cada uno de los distintos sellos que puedan aparecer.

El último intento consiste en aislar los elementos que aparecen en el documento y almacenarlos de manera independiente para más tarde aplicar una serie de criterios heurísticos para determinar si dicha región corresponde o no a un sello. Esto se hace mediante *etiquetado de componentes conexas* (o *connected component labeling*). Este algoritmo asigna una etiqueta identificativa única para cada región conexa de píxeles. El problema de este método es la existencia de texto muy cercano al sello o incluso encima de éste. En ese caso el algoritmo detecta como una región tanto al sello como a ese texto. Cuando esto ocurre, los filtros heurísticos fallan ya que el conjunto de sello más texto no cumple las mismas condiciones que si solamente estuviera el sello.

A continuación se detalla cada procedimiento en mayor profundidad además de los resultados obtenidos en cada caso, utilizados para decidir qué solución adoptar.

## Enfoque frecuencial para seccionamiento de texto y sellos.

En (cita) se desarrolla un método para separar texto de elementos gráficos en documentos escaneados. La idea se basa en el hecho de que el texto introduce componentes armónicas de alta frecuencia a la imagen. Por lo tanto, un filtro de paso bajo eliminaría ese texto. En el algoritmo, se utiliza un filtro de Gauss para eliminar las componentes de alta frecuencia. El resultado es una imagen con los elementos de texto eliminados casi en su totalidad mientras que los elementos gráficos (en el artículo incluye también logos) permanecerían en la imagen, idealmente, en su totalidad. El algoritmo expuesto es el siguiente:

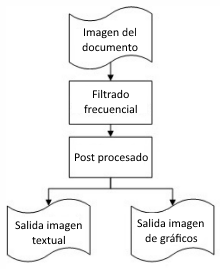


Figura X. Algoritmo espectral de seccionamiento.

El algoritmo original presenta pasos adicionales para separar distintos tipos de elementos gráficos, pero para nuestro problema, son pasos innecesarios.

### Filtrado frecuencial

El filtrado frecuencial es el paso más importante, y es donde se elimina la mayor parte del texto. Es una evidencia conocida que el texto contribuye a componentes espectrales de muy alta frecuencia en imágenes de documentos (cita). Por ello, se convoluciona con un kernel gausiano como filtrado de paso bajo y se utiliza un post procesado para eliminar restos residuales de texto. El algoritmo puede verse en la figura X.

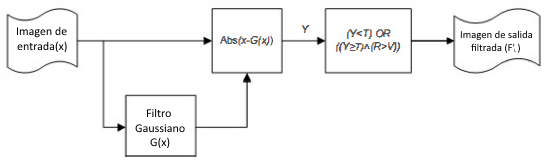


Figura X. Filtrado frecuencial.

El proceso consiste en:

1. Se toma la imagen original convertida a escala de grises, X.
2. Se le aplica un filtrado gaussiano, G(X).
3. Se calcula la imagen la imagen de **paso alto** Y, restándole G(X) a la original.
4. Un filtrado selectivo en frecuencias sencillo sería entonces

Donde I es la imagen original de dimensiones NxM, 1≤ i ≤ N y 1≤ j ≤ M. T es el umbral de frecuencias. Los valores que superan un cierto valor en la imagen filtrada Y se consideran de alta frecuencia, los que no, de baja frecuencia. Por último, Bk es el valor del color considerado de fondo. En el artículo utilizan una media de los colores de fondo del papel, obtenidos sumando los píxeles de la imagen original I, que han quedado como fondo en una binarización de la imagen en escala de grises X. En este trabajo se ha considerado innecesario ya que no se pretende reconstruir los huecos dejados por los elementos de eliminados del documento, y se ha tomado Bk como blanco directamente.

Hay que destacar que un filtrado solamente en frecuencias no es apto para elementos cromáticos cuyos tonos cambien de manera abrupta, ya que también introducen componentes de alta frecuencia y dicho filtro los eliminaría. Para solventar este problema, se conservan también aquellos elementos que tengan una cantidad de color suficientemente alta, esto se consigue cuantificar mediante la *cromaticidad*.

**Definición** cromaticidad: Se entiende por cromaticidad de un tono de color a la medida de cómo de alejado se encuentre ese color de tonos grises, independientemente de su luminancia, es decir, de cómo de claro u oscuro sea.

En un espacio de colores YCrCb, la cromaticidad R se cuantifica como

Por tanto, dada nuestra imagen original, se define la matriz de cromaticidad aquella que contiene para cada píxel el valor de su cromaticidad.

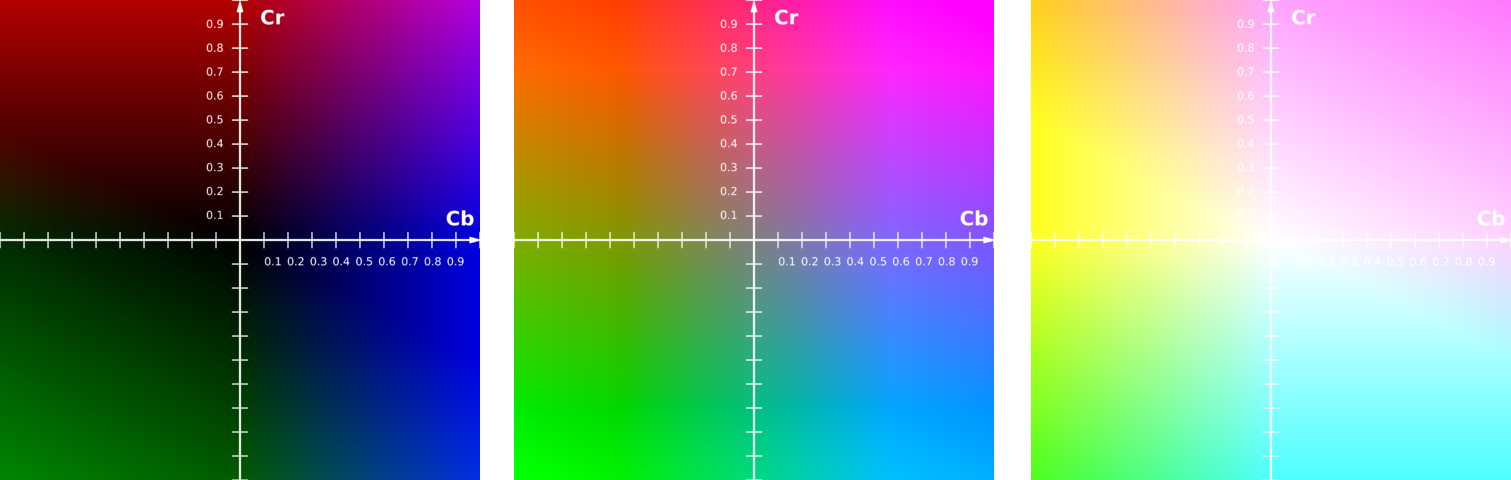


Figura x. Espacios de color YCrCb para valores de luma (Y) de 0, 0.5 y 1 respectivamente.

En la figura anterior puede verse como aquellos colores con una menor R, es decir, aquellos más cercanos al origen de coordenadas, poseen tonos más grises.

Volviendo al problema anterior, queremos evitar que al filtrar componentes de alta frecuencia eliminemos elementos gráficos que posean cambios bruscos en los colores. Para ello, imponemos un umbral V. Aquellos píxeles cuya cromaticidad sea superior a dicho umbral serán conservados en la imagen filtrada. Esto no supone ningún problema para la eliminación de texto ya que los caracteres tienen generalmente una cromaticidad muy baja.

Los resultados obtenidos mediante este procedimiento son los siguientes.



Figura X. Ejemplo de correcto funcionamiento del filtrado frecuencial.



Figura X. Ejemplo en el cual el filtrado frecuencial falla.

Como puede verse, los resultados en el primer caso son verdaderamente alentadores. Sin embargo, en el segundo caso el filtrado frecuencial elimina el sello además del texto. Esto se debe a que el sello posee claras diferencias tonales entre colores de muy baja cromaticidad. Por tanto, la condición de cromaticidad incluida para evitar la eliminación de elementos gráficos con variaciones tonales importantes, aquí no consigue salvar el sello.

Por todo eso, se procederá a probar otro método.

## Seccionamiento de texto y sellos mediante extracción de características

Definición Extracción de Características: En el contexto de tratamiento de imágenes, proceso mediante el cual, a partir de una imagen se obtiene un conjunto de datos (características), ligados a algunos píxeles específicos de la imagen (*keypoints)*. Estos datos deben ser informativos y no redundantes. En ocasiones las características pueden tener una representación física tangible, como esquinas. En otras en cambio, es un cálculo matemático sin una interpretación evidente.

El objetivo de la extracción de características es encontrar objetos que aparezcan en distintas imágenes, a pesar de presentar transformaciones homográficas o algún otro tipo de cambio menor, como cubrimiento parcial del objeto, deterioro, etc. Esto se hace partiendo de un modelo del objeto a encontrar, se realiza una extracción de características, luego se hace lo mismo en la imagen. Más tarde se comparan las características encontradas en la imagen y en el objeto modelo y aquellas que se parezcan lo suficiente suponen una correspondencia o *match*. Aquella zona de la imagen que dé suficientes matches, contiene al objeto buscado.

El proceso normal en una extracción de características es:

1. Detección de puntos clave o *keypoints*: según el algoritmo utilizado, estos puntos tienen una definición matemática diferente. En general, se busca encontrar puntos que tienen alguna propiedad matemática que los hace suficientemente diferentes al resto de puntos próximos.
2. Descripción de dichos puntos. A cada punto se le calcula un valor identificativo único denominado **descriptor**. En función del algoritmo escogido este descriptor puede ser un conjunto de vectores o un valor binario.
3. Matching. Consiste en la búsqueda de correspondencias de estos puntos clave en función de los descriptores. Para cada punto se le busca el punto de la imagen cuyo descriptor se parece más. Este “parecerse más” puede variar también según el algoritmo: mínima distancia euclídea o mínima distancia Manhattan son los criterios más comunes, aunque no los únicos.
4. Desechado de matches erróneos. Se toma algún tipo de criterio para intentar eliminar matches erróneos en el paso anterior. El criterio más común es el descrito por D. Lowe desarrollado para utilizarlo en su algoritmo SIFT, pero extendido hoy en día casi como criterio estándar de eliminación de falsos matches. Consiste en, dado un keypoint, calcular sus dos mejores matches. Si el primero y el segundo se parecen demasiado (un 70% o más es el valor utilizado por él, aunque este valor es configurable a medida para satisfacer las necesidades de cada problema), entonces es un falso match. Si el primer y el segundo candidato son suficientemente diferentes, se considera válido. Según su artículo, este método elimina un alto porcentaje de falsos positivos, eliminando un número muy bajo de positivos verdaderos. (cita).

### Elección de algoritmo para extracción de características

Existen diversos algoritmos para llevar a cabo esta tarea. Se pondrán a prueba los resultados obtenidos así como los tiempos de ejecución de los siguientes algoritmos:

* SIFT (*Scale-Invariant Feature Transform*)
* SURF (*Speeded-up Robust Features*)
* FAST (*Features from Accelerated Segment Test*)
* BRIEF (*Binary Robust Independent Elementary Features*)
* ORB (*Orientated FAST and Rotated BRIEF*)

#### SIFT

El algoritmo de extracción de características surge en 1999 por David Lowe (cita) por la necesidad de desarrollar un algoritmo de detección de características que no dependiese de la escala, es decir, que las características no desaparezcan en la detección al aparecer el objeto con distintas dimensiones en la imagen, bien por hacer zoom, bien porque el objeto se ha acercado físicamente al objetivo de la cámara.

Previamente, el algoritmo más extendido para llevar a cabo esta tarea era el detector de esquinas de Harris (cita) o el de Shi-Tomasi (cita). Éstos presenta la enorme ventaja de que una esquina detectada permanecía siendo reconocida como tal a pesar de que el objeto rotase. Es decir, dichos detectores son invariantes a la rotación. Sin embargo, al modificar la escala ocurre el siguiente fenómeno.

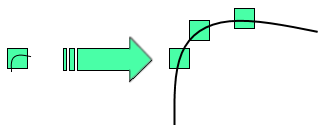


Figura X. Una esquina deja de serlo al aumentar su escala.

Como puede apreciarse en la figura, el objeto sería detectado como una esquina en la situación de la izquierda. Sin embargo, en la situación derecha el objeto ha incrementado su tamaño tanto que un que la porción que queda dentro de la ventana utilizada por el detector ha dejado de ser una esquina y, por lo tanto, ningún detector de esquinas podrá localizar dónde se encuentra dicho objeto ahora.

El detector SIFT no sufre este problema, ya que es invariante en escala. Por tanto elementos que cambien de tamaño seguirán siendo reconocidos por el algoritmo. El método funciona principalmente en cuatro pasos que se describen brevemente a continuación. (citar web de python y sift)

##### Paso 1: Detección de extremos en el espacio de escalas

De la imagen de arriba se deduce que tomar una ventana de tamaño fijo no es útil para detectar esquinas de tamaños variables. Para detectar esquinas mayores necesitaremos una ventana mayor. Para esto, se utiliza un filtrado en el espacio de escala. En él, el *Laplaciano del Gaussiano* (LdG de ahora en adelante) es calculado para la imagen con varios valores de σ.

Definición Laplaciano del Gaussiano: Dada una imagen de entrada f(x,y), esa imagen está convuelta con un kernel gaussiano:

siendo σ el denominado factor de escala del operador. Dada la imagen convuelta:

Entonces se define como Laplaciano de Gaussiano:

Este operador devuelve una fuerte respuesta positiva para regiones oscuras de dimensión y respuestas fuertemente negativas para regiones claras de igual tamaño.

(cita wikipedia <https://es.wikipedia.org/wiki/Reconocimiento_de_regiones#El_laplaciano_de_Gauss>)

El LdG actúa como un detector de blobs, que detecta blob de distintos tamaños debido al cambio en el valor de σ. En resumen, σ actúa como un parámetro de escalado. Por ejemplo, en la imagen superior, LdG con un σ pequeño devuelve valores altos para la esquina pequeña mientras que con un σ mayor, los devuelve para esquinas más grandes. Por tanto, podemos localizar los máximos locales a través tanto para cada escala como comparando escalas distintas con lo cual obtendríamos una lista de valores (x, y, σ) para cada potencial *keypoint* (x, y) en la escala σ.

El problema es que el LdG es algo costoso desde un punto de vista computacional. Para solventar este inconveniente SIFT utiliza en su lugar la *Diferencia de Gaussianos* (DdG de ahora en adelante), que es una aproximación de LdG. La DdG es obtenida como la diferencia entre dos imágenes convolucionadas con dos kernels gaussianos con distinta σ. El proceso se repite para diferentes niveles de una pirámide gaussiana.

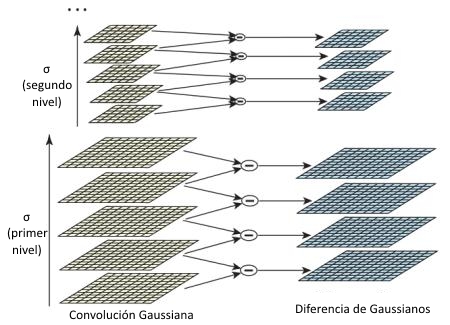


Figura X. Proceso de obtención de la diferencia de gaussianos.

Una vez que las imágenes de diferencias de gaussianos son obtenidas, se buscan extremos locales comparando en cada escala los 8 valores vecinos además de los 9 contiguos en una escala superior y los otros 9 que se encuentran en la inferior. Si es un extremo local, es un punto clave potencial. Básicamente esto significa que dicho punto se encuentra mejor representado en esa escala.

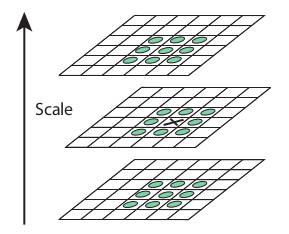


Figura X. Búsqueda de extremos locales en el espacio de escalas.

Con respecto a los parámetros a utilizar, se recomienda en (cita), basándose en datos obtenidos de manera empírica utilizar 4 niveles para la pirámide, 5 escalas partiendo de σ = 1.6 e incrementando dicho valor en .

##### Paso 2: Afinamiento en la localización de puntos clave.

Una vez que los puntos clave han sido localizados, es necesario un proceso de refinar los valores obtenidos para mejorar la precisión de los resultados. Para ello, se utilizan series de expansión de Taylor para mejorar la localización de extremos locales en el espacio de escalas. Además de esto, si este extremo no supera un cierto umbral (0.03 es el valor utilizado por Lowe), es rechazado.

Cabe destacar también, que DdG ofrece una respuesta notable para bordes, así que estos deben ser eliminados. Para esto se utiliza una idea similar a la que se usa para el detector de esquinas Harris. Se toma una matriz Hessiana de dimensión 2x2 para calcular la curvatura. En el caso de bordes en lugar de esquinas un autovalor de dicha matriz será significativamente mayor que el otro. Por lo tanto, si el ratio de un autovalor con respecto del otro es superior a un cierto umbral (10 es el valor propuesto por Lowe), es descartado.

Este paso elimina cualquier punto clave con bajo contraste, ya que es muy probable que se deban a ruido. Se eliminan además puntos situados en bordes para quedarnos sólo con las esquinas. De esta manera quedan tan sólo puntos de gran interés y utilidad.

##### Paso 3: Asignación de orientación

Ahora, asignamos una orientación a cada punto clave obtenido para conseguir la invariancia a la rotación. Se toma una vecindad alrededor del punto dependiendo de la escala, y se calculan la magnitud y dirección del gradiente en dicha región para cada píxel. Se crea un histograma de orientaciones con 36 intervalos cubriendo los 360 grados, en el cual cada orientación es ponderada en función de la magnitud de su gradiente además de un kernel gaussiano centrado en el punto y de σ igual a 1.5 veces la escala del punto. Una vez calculado el histograma, se toma el pico más elevado además de los picos que midan al menos un 80% el valor del superior. La orientación resultante es asignada al punto en cuestión.

##### Paso 4: Creación de un descriptor para el punto

Se toma una vecindad de 16x16 alrededor del punto. Ésta se subdivide a su vez en 16 bloques de dimensión 4x4. Para cada sub-bloque, se toma un histograma de orientación de 8 divisiones. En total, 128 valores son calculados. Este vector asociado a cada punto clave se denomina descriptor. Estos valores son refinados posteriormente para mejorar la robusteza contra cambios de iluminación.

##### Paso 5: Emparejamiento de puntos

El objetivo del emparejamiento de puntos consiste en localizar qué puntos en una imagen se corresponden con el mismo objeto de otra. En este paso no se describe un método para llevar esto a cabo ya que existe toda una familia de algoritmos destinados a resolver esta tarea. Se trata de proponer un método para eliminar los falsos positivos que puedan surgir en el emparejamiento de puntos con descriptores SIFT.

Se propone tomar un punto clave de una imagen y después tomar de la otra imagen otros dos puntos cuyos descriptores sean los más parecidos a los del punto original. Dada las distancias de los descriptores de estos dos puntos al punto de la imagen, si ambas difieren menos de un 80% ambos puntos son descartados y no se empareja ningún punto de la segunda imagen con el punto en cuestión de la primera.

#### SURF

Durante algún tiempo, SIFT fue el estándar a la hora de resolver esta tarea, sin embargo es computacionalmente muy costoso. No sólo los procedimientos de obtención de puntos clave y descriptores sino que, además, dichos descriptores contienen un gran número de valores. Esto hace que el posterior proceso de emparejamiento de puntos se ralentice enormemente.

En 2006, Bay, H., Tuytelaars, T. and Van Gool, L publicaron un artículo llamado “SURF: Speeded Up Robust Features”, el cual, como indica su nombre, introdujo una versión acelerada de SIFT.

##### Búsqueda de puntos clave

En SIFT, Lowe aproxima el laplaciano del gaussiano mediante una diferencia de gaussianos con el objetivo de trasladar la búsqueda de puntos clave al espacio de escalas. SURF va un paso más allá aproximando el LdG mediante lo que denominan ellos un *filtro de cajas*. La imagen inferior ilustra una demostración de dicha aproximación. En la imagen derecha se muestra el resultado de cada una de las dos derivadas segundas parciales de la imagen original y a la izquierda la matriz utilizada como kernel de convolución para aproximar dicha derivada. Una gran ventaja de esta aproximación es que la convolución con estas “cajas” puede calcularse de manera muy rápida con ayuda de imágenes integrales. También puede realizarse en paralelo de manera independiente para distintas escalas. Por otra parte, SURF utiliza el determinante de la matriz Hessiana tanto para la escala como para la localización, disminuyendo por tanto el número de cálculos necesarios.

##### Cálculo de la orientación

Para asignar una **orientación**, SURF utiliza las respuestas a *wavelets* (en concreto la *Transformada de Wavelet de Haar* o HWT por sus siglas en inglés) en las direcciones tanto en horizontal como vertical para una vecindad de dimensión 6\*s, donde *s* es la dimensión del punto calculado (no necesariamente tiene que ser un único píxel). A cada una de esas respuestas se las pondera con una adecuada función gaussiana. A continuación, se las sitúa en el espacio tal y como se aprecia en la imagen inferior. La orientación dominante se calcula sumando las respuestas que se encuentran en una abertura de 60º. Es interesante destacar que las respuestas a wavelets también se pueden calcular fácilmente mediante el uso de imágenes integrales para cualquier escala. También cabe destacar que los descriptores son independientes de esta orientación, por lo tanto, se puede prescindir de este cálculo. En nuestro caso, la invariancia a la rotación no es necesaria ya que se trata de documentos que siempre van a ser escaneados con la misma orientación. Este método es denominado Upright-SURF, o U-SURF. Mejora la velocidad de manera notable y proporciona robustez con una precisión de ±15º. Ese será el método que utilizaremos nosotros ya que los sellos pueden estar ligeramente rotados pero no nos importa calcular cuál es el valor de dicha rotación.

##### Asignación de descriptores

Para la descripción de características, SURF utiliza respuestas a wavelet de Haar en las direcciones horizontal y vertical, nuevamente con ayuda de las imágenes integrales. Se toma una vecindad de 20sx20s alrededor del punto (donde *s* es nuevamente la dimensión del punto calculado). Esta región se divide en subregiones de 4x4. Para cada una de ellas, se calcula la respuesta a wavelet nuevamente en ambas direcciones y se crea un vector de la siguiente manera:

Todos estos vectores colocados de manera consecutiva como un único vector suponen un descriptor del punto con una dimensión de 64 valores. Menor dimensión significa mayor velocidad en computación además de en el emparejamiento posterior.

Si mayor distinción para los descriptores de cada punto fuera necesaria, SURF proporciona la posibilidad de aumentar la dimensión de dichos descriptores a 128 calculando por separado las sumas de las distintas respuestas: tanto dx como |dx| se suman por separado en función de si su respectivo dy es positivo o negativo. Para dy se hace lo mismo.

##### Afinamiento de emparejamiento de puntos

Para mejorar el futuro emparejamiento SURF ofrece además otra mejora. Utilizando el signo del laplaciano (que es el mismo que la traza de la matriz hessiana) para cada punto de interés. No supone ningún aumento del coste computacional ya que dicha matriz ha sido ya calculada durante la fase de detección. El signo del laplaciano sirve para distinguir si el punto de interés supone una región clara sobre fondo oscuro o al contrario. Durante el emparejamiento sólo tenemos que comparar puntos en los cuales este valor coincida, mejorando tanto la precisión del emparejamiento como la velocidad con la que se ejecuta.

##### Comparativa con SIFT

Las pruebas demuestran que SURF supone un incremento de velocidad considerable con una precisión comparable con SIFT. SURF demuestra ser robusto ante rotación y desenfoque, sin embargo, no es muy bueno ante cambios de iluminación y de cambios en puntos de vista. Esto no supone ningún problema para nuestro caso.

#### ORB

Como alternativa a ambos expuestos anteriormente, se desarrolló un algoritmo que fuera libre. En 2011 Ethan Rublee, Vincent Rabaud, Kurt Konolige y Gary R. Bradski publicaron un artículo titulado “ORB: An efficient alternative to SIFT or SURF” (cita).

Básicamente ORB supone una fusión entre el detector de puntos clave FAST y el generador de descriptores BRIEF con modificaciones múltiples con el objetivo de mejorar el rendimiento. En un primer paso, utiliza FAST para encontrar los puntos. Después, aplica un detector de esquinas Harris con el objetivo de localizar los máximos N valores entre los puntos obtenidos previamente. Esto elimina los puntos encontrados con FAST que no posean suficiente contraste. Se utiliza además una pirámide para calcular propiedades multiescala. Un problema importante es que FAST no computa ningún valor que nos proporcione información sobre la orientación. Para conseguir invarianza ante la rotación los autores añadieron la siguiente modificación.

Se computa el centroide ponderado de la región en cuyo centro se encuentra el punto clave en cuestión. El vector que va desde el centro hasta este centroide nos proporciona la orientación. Para mejorar este valor, se computan además los momentos dentro de una región circular centrada en el punto.

Ahora, para los descriptores se hace uso de BRIEF. Nuevamente, BRIEF responde de manera pobre con respecto a la rotación, esto para el caso expuesto no supone un problema. Aún así, ORB soluciona este problema “orientando” BRIEF en la dirección calculada previamente. Para cada punto, se toma la vecindad necesaria y se rota multiplicando las coordenadas por la matriz de rotación correspondiente.

ORB discretiza en incrementos de 12º y construye una tabla de búsqueda de patrones BRIEF precomputados. Siempre y cuando la orientación utilizada inicialmente sea correcta, los valores del descriptor serán robustos con respecto a la rotación.

Una propiedad importante de BRIEF es que cada propiedad binaria obtenida tiene una alta varianza pero la media permanece cerca de 0.5. Pero una vez orientado con respecto a la dirección del punto clave esta propiedad se pierde y se vuelve más distribuido. Alta varianza significa que es el descriptor mucho más distintivo ya que responde de manera distinta ante la entrada de distintos puntos. Otra propiedad deseable sería que los test no estuvieran correlados ya que entonces cada test realizado contribuiría al resultado final. Para resolver todo esto ORB realiza una búsqueda voraz entre todos los posibles test binarios para encontrar aquellos que tienen gran varianza y media cercana a 0.5, además de no tener correlación entre sí. El resultado se denomina rBRIEF.

Para el emparejamiento de descriptores, se utiliza un LSH (del inglés *Locality-Sensitive Hashing*) el cual mejora los resultados con respecto al LSH original.

(INCLUIR AQUÍR TESTS DE ALGORITMOS DE DETECCIÓN DE ROS) y por esto usamos SURF…

### Resultados obtenidos con el método seleccionado

Para el caso en el que el sello usado para comparar encaja con el que aparece en el documento, se obtienen más de 200 correspondencias (típicamente, entre poco más de 200 hasta unas 350, según el documento) como puede apreciarse en la imagen. Muchos de ellos son falsos positivos, pero en su mayoría son buenas.

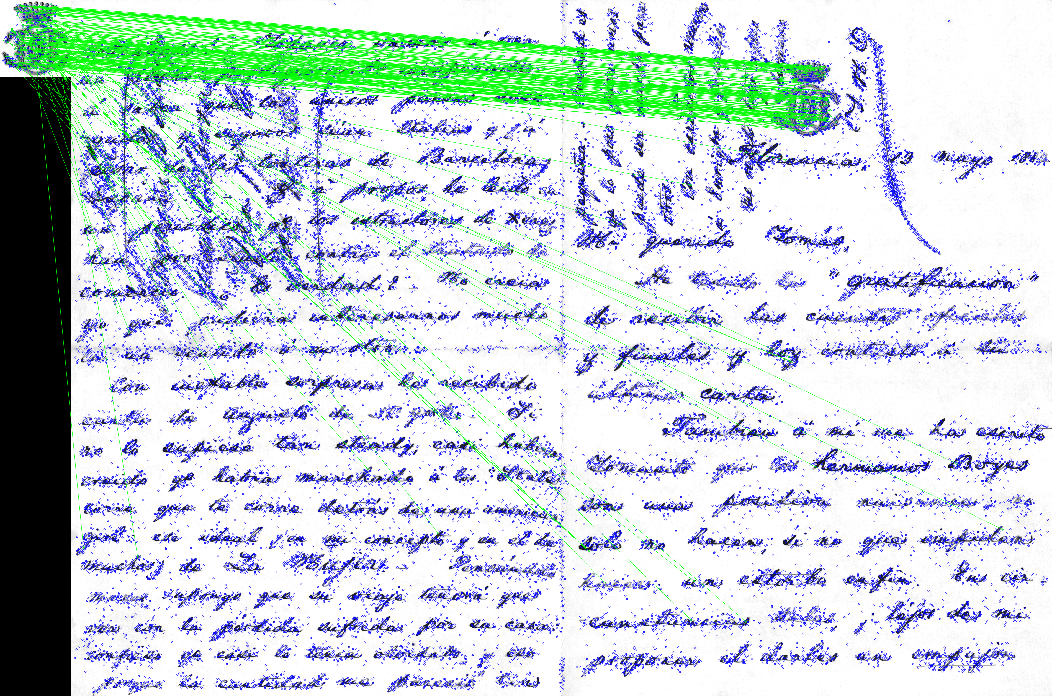


Figura X. Sello localizado en el documento mediante SURF

En cambio, cuando la muestra y el sello del documento no coinciden, el número de correspondencias encontradas, es de poco más de 100. Pueden parecer muchas pero hay que tener en cuenta que el método de Lowe para discriminar falsos positivos sólo es útil cuando el objeto buscado se encuentra en la imagen. En caso de que el número de falsos positivos cuando el sello buscado no está presente fuera demasiado grande podríamos calcular la distancia media entre descriptores emparejados en el caso de emparejamientos acertados (es decir, cuánto se parecen cuando el emparejamiento es correcto). Aquellos descriptores cuya distancia sea muy superior a un umbral prefijado (digamos por ejemplo, un 30% más que la media calculada), podrían ser descartados. Sin embargo, ya que el número de falsos positivos disminuye lo suficiente y que además los emparejamientos erróneos se encuentran distribuidos por toda la imagen, es fácil determinar cuándo el sello se encuentra en el documento y cuándo no.

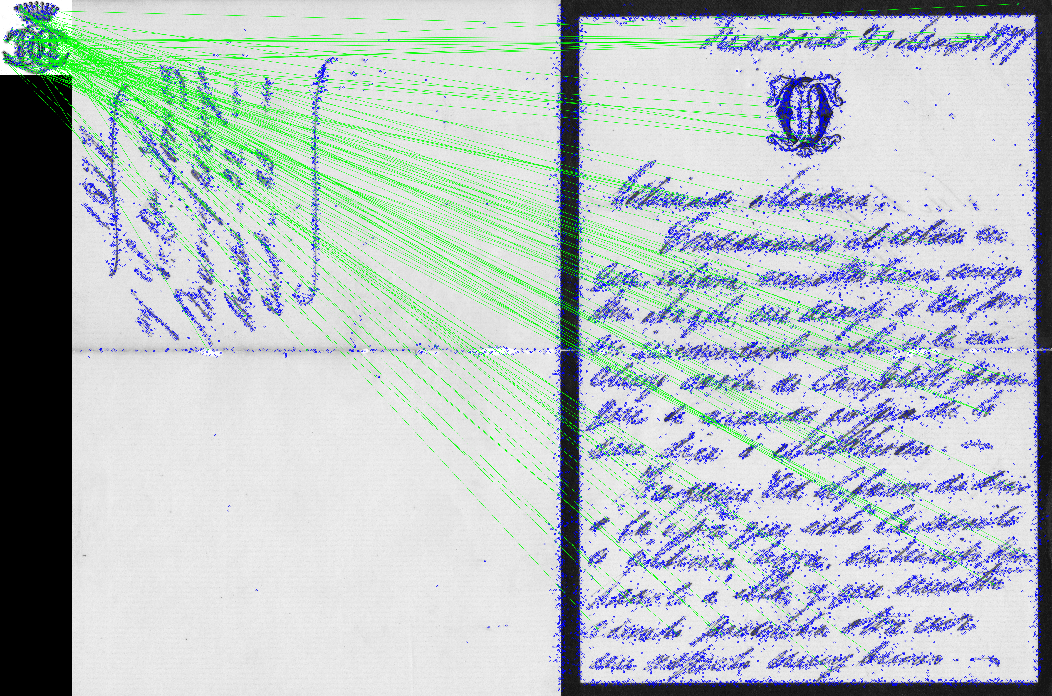


Figura X. Sello buscado mediante SURF pero no localizado en el documento.

Puede verse que con este método puede resolverse el problema de manera bastante robusta. Sin embargo, tiene una desventaja bastante importante: para que funcione es necesario partir de una base de datos que contenga todos los tipos de sello para poder comparar. Esto podría hacerse de manera manual abriendo cada documento y recortando aquellos que contengan algún sello. Pero esta solución carece de sentido ya que clasificar cada documento de manera manual supondría menos trabajo.

## Búsqueda de sellos mediante métodos heurísticos

En este caso se pretende localizar los sellos aplicando componentes conexas. Con esto se obtiene una caja conteniendo cada elemento del documento, palabras, sellos, manchas…etc. Posteriormente a cada una de esas cajas se le aplica una serie de reglas que sirven para distinguir si se trata de un sello o de cualquier otro elemento. A continuación se detalla todo el proceso en profundidad.

Partiendo de la imagen original, se realiza una binarización invertida mediante un umbral de Otsu. De esta manera obtenemos el texto en blanco y fondo en negro. A esta imagen se le aplica una dilatación con una vecindad de 11x11 con el objetivo de recuperar huecos vacíos en la binarización. Esto es importante para un test que pruebe la simetría que se verá más adelante. Estos huecos disminuirían enormemente el valor de simetría obtenido. Una vez dilatada, se procede a un etiquetado de componentes conexas. El resultado obtenido es una matriz con el valor de la etiqueta de cada píxel, de manera que aquellos píxeles considerados conexos obtienen la misma etiqueta y por tanto son considerados el mismo objeto.

Seguidamente se procede a unificar aquellas regiones lo suficientemente próximas como para considerar que en realidad se tratase de una única región. Para ello, se detectan colisiones entre cada caja ligeramente ampliada y se le asigna una etiqueta común a aquellas que se solapen. En este proceso es necesario eliminar el borde que pudiera aparecer en alguna de las imágenes ya que se detecta como una gran región y la detección de colisiones acabaría por incorporar el documento al completo a un único grupo.

Una vez separadas las regiones, se procede a aplicar múltiples filtros que vayan eliminando sucesivamente regiones que no sean sello. Estos son los siguientes:

1. El primero y más evidente de todos ellos es filtrar por tamaño. Se han hecho pruebas con distintos sellos y los que poseen una menor área son de unos 50,000px2. Por tanto aquellas cajas con menos de 40,000 son descartadas.
2. Lo siguiente es eliminar cajas demasiado alargadas. En ocasiones palabras muy largas o varias palabras que han sido unificadas tras la dilatación o la detección de colisiones poseen un área mayor que el límite establecido. Estas cajas son muy alargadas así que se eliminan imponiendo que anchura no sea más de dos veces la altura.
3. En ocasiones la una línea hacia arriba de una palabra, como una “t” se une con la parte de debajo de la línea superior, como de una “p”, por ejemplo. Esto da lugar a cajas suficientemente grandes y cuadradas como para pasar los dos filtros anteriores. Para eliminar estos casos se mide el *ratio de relleno* de las cajas como el número de píxeles de objeto que hay dentro con respecto al área total de la caja. Si es demasiado pequeño, no se trata de un sello.
4. A continuación se realiza un test de simetría vertical. Para ello, se divide la caja por la mitad de la anchura. La mitad izquierda se voltea con respecto a un eje vertical y se le resta a la mitad derecha. Si la sección de imagen fuera perfectamente simétrica, el resultado tendría cero píxeles blancos. Sería todo negro. Esto no ocurre nunca así que se mide el número de píxeles blancos resultantes con respecto al número de píxeles originales en una de las dos mitades. Las pruebas demuestran que para valores menores de 0.2, podemos considerar simetría, así que se impone un umbral de 0.25 para garantizar que no se elimina ningún nuevo sello por error.

Cabe destacar que, en caso de que un sello no supere el test de simetría en un primer paso, no se descarta inmediatamente. Es frecuente que aparezcan manchas y ruido cerca del sello lo cual harían que el centro de la caja detectada no coincida con el sello y, como cabe esperar, al dividir por la mitad y voltear el resultado sea negativo. También puede ser que el sello aparezca en un color muy claro y parte de ese sello se elimine durante la binarizarión. Por lo tanto, para garantizar que no se elimine ninguno de manera errónea, si no es suficientemente simétrica la caja se toma una un poco mayor de la imagen original. A esta nueva caja se le aplica un umbral de binarización un 90% del original, se vuelve a dilatar y se le vuelve a pasar el test. Tras un nuevo fracaso, se prueba con un umbral de un 110%. Si la prueba fracasa nuevamente, se descarta esa caja de manera definitiva.

En la imagen a continuación puede verse un caso en el cual el método funciona. En gris se recuadra dónde se ha detectado un sello. Puede apreciarse también cómo el algoritmo ignora el borde negro del papel del documento.

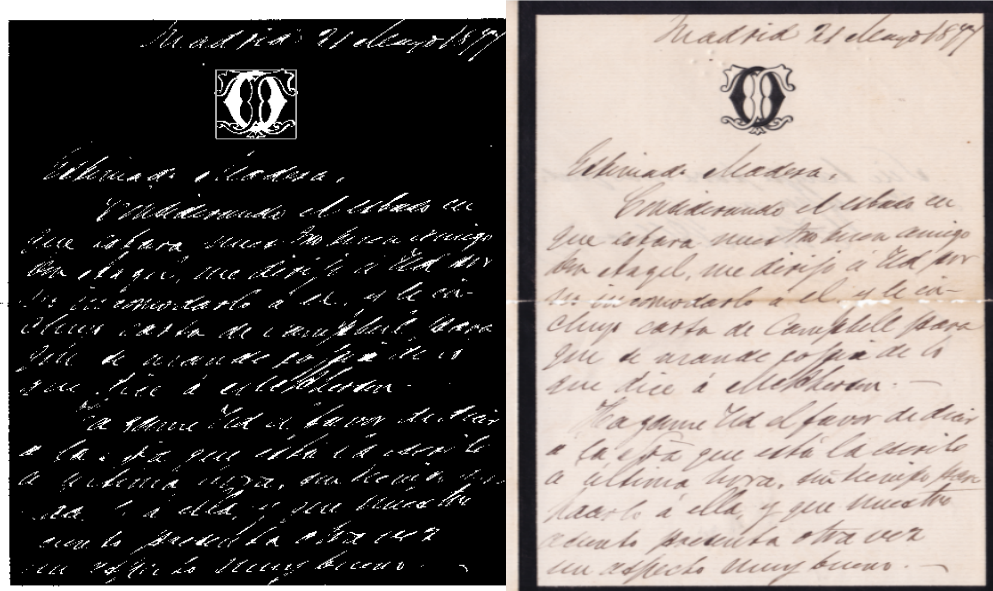


Figura X. Demostración de correcta detección del algoritmo heurístico

Este algoritmo tiene un caso claro donde falla: si existe texto sobre el sello o muy cerca de éste, el etiquetado de componentes conexas nos dará una región que, además del sello, contendrá texto. Es muy improbable que esta región supere las pruebas de simetría, ya que debería sobrar la misma longitud de texto por ambos lados del sello. También es improbable que supere el test del ratio de relleno, ya que debería además tener texto superpuesto desde la parte superior hasta la inferior del sello, todo ello sin dejar de ser simétrico.



Figura X. Ejemplo de caso en el cual la detección fallaría.

Existen numerosos casos donde esto ocurre, por ese motivo este método por sí sólo no es suficientemente robusto para ser utilizado.

## Conclusiones

Ninguno de los métodos analizados es suficientemente bueno por sí sólo. El método frecuencial elimina algunos sellos además del texto por lo cual no es una alternativa fiable. El método de extracción de características arroja resultados bastante positivos, sin embargo, no es aplicable en la práctica por requerir un registro previo de todos los sellos disponibles. El algoritmo heurístico sólo es adecuado en un número limitado de casos.

Por todo ello, se ha optado por **aplicar extracción de características** **y alimentar** ese sistema con muestras de sellos obtenidas **mediante etiquetado de componentes conexas y eliminación heurística**. Es decir, en un primer paso se procederá con el método heurístico para conseguir el mayor número posible de sellos. Luego, estos se emplearán como patrón de comparación para una extracción de características. El algoritmo al completo se explicará en profundidad en el siguiente capítulo.

# Descripción del algoritmo desarrollado

El algoritmo desarrollado tiene tres pasos. Primero se detectan mediante heurística todos los sellos posibles y se eliminan los errores producidos por este método. Después se extraen las características de los sellos y se guardan en una base de datos. Por último, cada vez que se escanea un nuevo documento se carga la base de datos que contiene los puntos clave y descriptores de cada sello. Se compara el documento con cada uno de ellos y se clasifica en función de los resultados obtenidos, además de borrar el sello localizado mediante un método detallado a continuación. Si no se ha logrado ningún emparejamiento positivo, se busca mediante componentes conexas y filtrado heurístico de nuevo en el documento. Si se encuentra algún sello nuevo, se extrae como candidato a nuevo sello a incorporar en la base de datos. En caso de que sea aprobado, se realiza tal operación y el documento se clasifica convenientemente. En caso de que no se encuentre ningún sello en el documento, éste se cataloga como documento sin sello. Estos pasos se explican en detalle a continuación.

## Obtención de sellos mediante método heurístico

El primer paso es obtener una imagen de cada uno de los sellos posibles. Para ello aplicamos el algoritmo heurístico a todos y cada uno de los documentos que se encuentran ya escaneados. Es importante destacar que los resultados obtenidos serán revisados antes de construir la base de datos de sellos. Por lo tanto, el hecho de obtener falsos positivos no supone un gran problema. Sin embargo, el hecho de que algún sello no apareciese en la base sí sería más problemático. Por ello es que los parámetros del método heurístico se han relajado lo suficiente para garantizar que ningún sello sea omitido, aunque eso suponga obtener algunos falsos positivos.

Tras correr el algoritmo se revisan manualmente los documentos para medir los resultados. Existen un total de 187 documentos escaneados y subidos a la plataforma en el momento de las pruebas entre los cuales hay 9 sellos distintos.

El algoritmo devuelve un total de 37 detecciones. Hay que tener en cuenta que una buena parte de los documentos no contienen ningún sello. De esos 37, 19 corresponden a sellos (algunos de ellos repetidos). Esto significa que un 48.65% de los resultados obtenidos corresponden falsos positivos. Esto, como ya se ha comentado, no supone un problema.

Posteriormente, se procede a eliminar los sellos que se encuentran repetidos y los falsos positivos. El algoritmo es capaz de detectar 8 de los 9 sellos existentes. Esto supone un 88.89% de efectividad. No sólo eso, sino que el sello que no es detectado sólo aparece en una única carta y con texto muy próximo. Se realiza un test en el cual se elimina ese texto de manera manual y se vuelve a correr el algoritmo y el sello es detectado. Esto significa que cuando volviese a introducirse un nuevo documento en el que apareciese dicho sello, éste sería incorporado a la base de datos.

## Creación de base de datos de sellos

Una vez que tenemos todos los sellos, realizamos una extracción de características y almacenamos los puntos clave y los descriptores de cada uno de ellos para el futuro emparejamiento. Además de ello, también se guardan las dimensiones de cada ello, que son imprescindibles para poder eliminar más adelante los sellos encontrados.

## Detección de sellos en documentos escaneados

Cada vez que se escanea un documento y se sube a la plataforma, se carga la base de datos y se realiza un emparejamiento con los descriptores obtenidos con cada uno de los sellos con respecto a los extraídos el documento. En un principio se tomaba como tipo de sello encontrado aquel que diera un máximo de puntos emparejados. Sin embargo, este método no ha resultado ser suficientemente robusto ya que algunos patrones de sellos devuelven un número de puntos clave mucho mayor que otro. En concreto, existe uno que almacena en la base de datos más de 1200 puntos, mientras que otro poco más de 400. Debido a la existencia de falsos positivos propia de este tipo de detección, ocurre en ocasiones que se encuentren más puntos del primer sello, distribuidos por todo el documento, del primer sello, a pesar de ser el segundo el que realmente aparezca. El filtrado de Lowe resulta insuficiente ya que no siempre es eficaz para comparar objetos que realmente no aparecen en la imagen.

Por lo tanto se ha procedido a filtrar los puntos de la siguiente manera. Como puede apreciarse en la figura siguiente, los puntos emparejados se encuentran repartidos por todo el documento. Sin embargo, existe una mayor concentración de puntos donde se encuentra el sello. Esta propiedad nos va a permitir filtrar los falsos positivos y quedarnos tan solo con aquellos puntos que pertenezcan al sello.

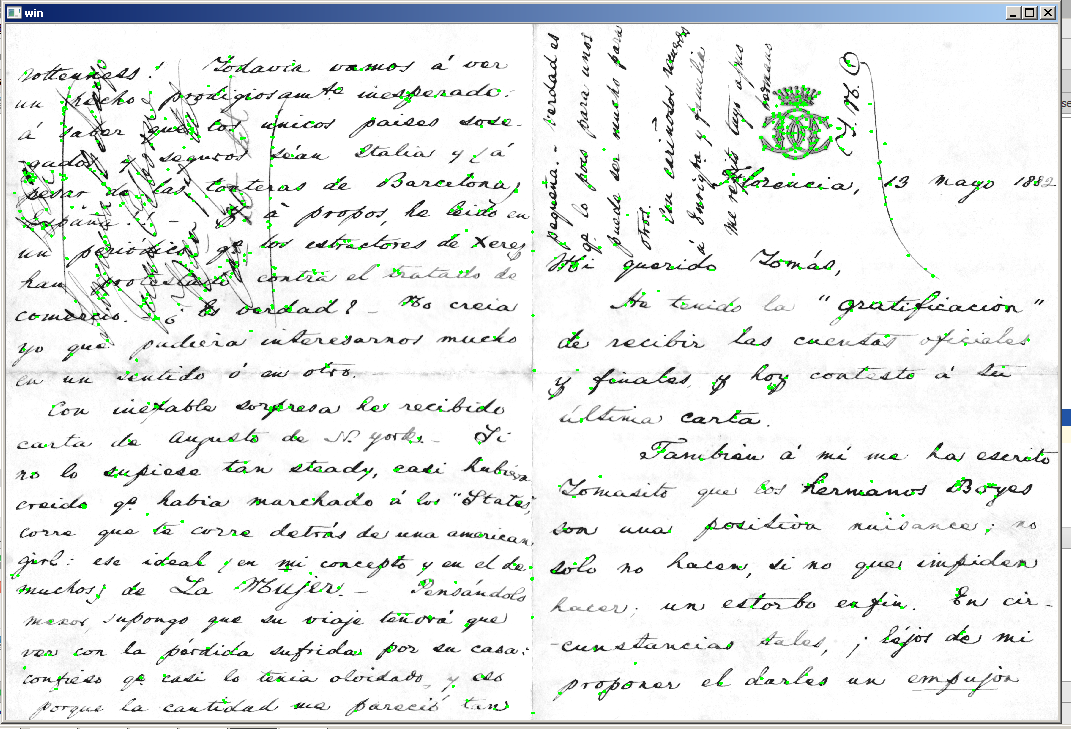


Figura X. Distribución de puntos emparejados con el sello modelo en la imagen del documento.

Para realizar el filtrado, dividimos primero la imagen en secciones de un tamaño prefijado. En la figura siguiente se muestra un mallado de 100x100px sin embargo, este valor se ha tomado tan sólo por legibilidad de la ilustración. En la práctica, se ha utilizado un mallado de 10x10px. Una vez dividido, contamos cuántos puntos se encuentran dentro de cada división. Al resultado se le ha denominado *matriz de acumulación de evidencias* y tiene, en nuestro caso, las dimensiones de la imagen divididas por un factor 10. En esta matriz se realiza una convolución con un kernel unidad y con dimensiones del sello a comparar, también divididas por el factor pertinente (recuérdese que las dimensiones de cada sello también se encuentran almacenadas en la base de datos creada previamente). Esto nos da cuántos puntos existen en una vecindad del mismo tamaño del sello. Aquel elemento de la matriz convolución que presente un máximo, será el centro del sello. En la práctica es frecuente que varios elementos presenten el mismo máximo, todos ellos dentro del sello, mientras que los elementos que no pertenecen al sello obtienen valores varias veces inferiores. Por ello, se hace una media de las coordenadas de todos los máximos y se toma el resultado como centro. Aquellos puntos cuya coordenada vertical u horizontal se aleje del centro más de la mitad de la dimensión correspondiente del sello, se considera que no pertenece a éste y es descartado.

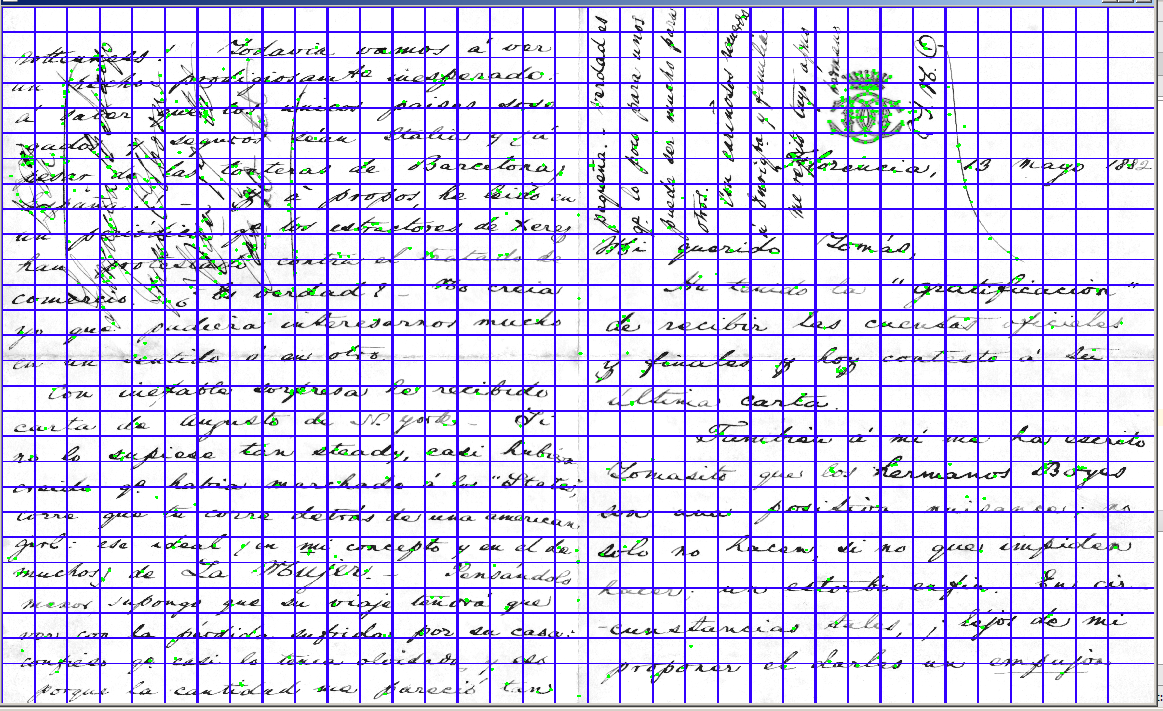


Figura X. Mallado de la imagen a 100x100px

En la figura a continuación puede verse como, de todo el documento, el punto detectado como centro es bastante bueno.

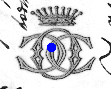


Figura X. Punto detectado como centro del sello

En caso de que para ningún sello se cumpla que el número de puntos no descartados que hayan sido emparejados no supere el 20% de los puntos totales hallados en el modelo tomado de dicho sello, se considerará que no se ha detectado ninguno. Dada esa situación, el algoritmo vuelve a utilizar el método heurístico para ver si existe algún sello en el documento que no haya sido incorporado a la base de datos aún. Si encontrase algún candidato, lo colocaría pendiente de aprobación para ser incorporado a la base de datos.

# Referencias

|  |  |
| --- | --- |
| [1] | Autor, «Este es el ejemplo de una cita,» *Tesis Doctoral,* vol. 2, nº 13, 2012. |
| [2] | O. Autor, «Otra cita distinta,» *revista,* p. 12, 2001. |

# **Índice de Conceptos**

conceptos 9

# **Glosario**

ISO: International Organization for Standardization 4

UNE: Una Norma Española 4