Universidad de Oviedo

ESCUELA POLITÉCNICA DE INGENIERÍA DE GIJÓN.

GRADO EN INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

ÁREA DE INGENIERÍA DE SISTEMAS Y AUTOMÁTICA

TRABAJO FIN DE GRADO Nº 17010434

RECONOCIMIENTO DE OBJETOS MEDIANTE VISIÓN POR COMPUTADOR PARA AYUDAS A INVIDENTES

D.ª ROOS HOEFGEEST TORIBIO, Sara
TUTOR: D. GONZÁLEZ DE LOS REYES, Rafael Corsino

FECHA: junio de 2017

ÍNDICE

ÍNDICE DE LA MEMORIA

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1.

MEMORIA

1. Introducción

1.1 Objetivo

El proyecto tiene como objetivo el estudio de la viabilidad de un sistema de reconocimiento de objetos mediante visión por computador y la evaluación de su funcionamiento, en especial del nivel de fiabilidad en la detección. Para establecer esta viabilidad se reconocerán objetos de una serie, cuyas imágenes han sido adquiridas y almacenadas previamente en una base de datos.

El método de reconocimiento de objetos aplicado se basa en la extracción de puntos característicos y sus descriptores basado en la transformada de Hough. Estos se emparejan con las características de las imágenes de entrenamiento para llevar a cabo un agrupamiento mediante un algoritmo de *clustering*, generando hipótesis sobre las posiciones de los objetos. Dichas hipótesis se verifican mediante una homografía entre las posiciones de los objetos, especificadas en la base de datos, y las correspondientes determinadas en la imagen.

Este método potencialmente será válido para cualquier tipo de objeto, sin embargo, es más apropiado para objetos que presenten diferentes texturas. En cuanto al número de objetos a reconocer en una imagen, podrá detectar tantos como haya en la base de datos.

1.2 Estado del arte

La visión por computador o visión artificial es un campo de la ingeniería cuya finalidad es, mediante el uso de las técnicas adecuadas, la extracción de información del mundo físico a partir de imágenes por medio de un computador.

Desde su creación, la visión artificial ha sido un área de investigación en auge que no ha frenado su evolución, permitiendo el desarrollo de aplicaciones muy diversas en ámbitos dispares como la industria, la medicina, en aplicaciones de seguridad o en el desarrollo de sistemas para la mejora de la calidad de vida de personas con algún tipo de deficiencia visual.

Según datos de la Organización Mundial de la Salud publicados en el año 2014 [1], en el mundo hay aproximadamente 285 millones de personas que padecen algún tipo de discapacidad visual, de las cuales alrededor de 39 millones presentan ceguera total. Los sistemas de visión por computador como sustitutos de la visión humana, pueden constituir una herramienta muy importante para el desarrollo de aplicaciones de apoyo a personas invidentes, permitiendo mejorar la percepción de su entorno. En esta línea podrían implementarse sistemas de reconocimiento de objetos, guiado en el interior de edificios, acceso a información impresa, orientación o interacción social, entre otros.

El reconocimiento de objetos presenta potenciales ventajas en cuanto a implementación de aplicaciones de ayuda a personas con discapacidad visual, otorgándoles una mayor libertad e independencia tanto en espacios cerrados como abiertos. En interiores les permitiría desenvolverse con mayor soltura ayudando a identificar objetos cotidianos como escaleras, puertas o ventanas o encontrar objetos de uso habitual, como mandos a distancia, libros o teléfonos. En exteriores, mediante la identificación de diferentes señales, interpretación de semáforos, localización de pasos de peatones o paradas de autobús. Estas aplicaciones permitirían cubrir algunas de las carencias que tienen los bastones, perros guía y otros dispositivos básicos que utilizan las personas invidentes.

Hasta la fecha se ha intentado desarrollar diferentes sistemas de ayuda, destinados a personas con algún tipo de discapacidad visual, basados en el reconocimiento de objetos.

Una *startup* de Israel desarrolló una cámara, bajo el nombre de *OrCam MyEye* [2], para ayudar a mejorar la calidad de vida de las personas con algún tipo de deficiencia visual. Consiste en una cámara que se acopla a unas gafas corrientes. Este dispositivo se activa con un simple gesto y permite la lectura de textos y el reconocimiento de caras conocidas y de diferentes objetos en entornos como el de hacer la compra. Además, incluye un sistema de transmisión de los resultados mediante comandos de voz. Este producto ya está disponible en el mercado.

Otro sistema con fines similares al anterior es el de Toyota mediante el proyecto BLAID [3]. Es un *wearable* en forma de herradura que se coloca sobre el cuello y hombros del usuario. Integra una serie de cámaras que permiten detectar objetos circundantes, incluyendo letreros, e identificar baños, escaleras mecánicas, escaleras, ascensores o puertas. Además, estará equipado con sistemas de reconocimiento de voz, botones, altavoces y vibración para

permitir una mejor interactuación con el usuario. Además, la tecnología *Bluetooth* permitirá al usuario emparejarse con su teléfono inteligente para una funcionalidad adicional. Este proyecto aún es un prototipo, por lo que aún no está disponible en el mercado.

También existen otros sistemas de reconocimiento de objetos implementados en aplicaciones para *smartphones*, como por ejemplo *Aipoly Vision* [4]. Aplicación que reconoce en tiempo real diferentes objetos a través de la cámara del móvil.

Una vez vistas las aplicaciones existentes en el mercado, es útil remarcar cuál es la estructura general de cualquier proceso de reconocimiento de objetos. Se puede dividir en cuatro etapas básicas, que se ejecutan consecutivamente para obtener el objetivo deseado.

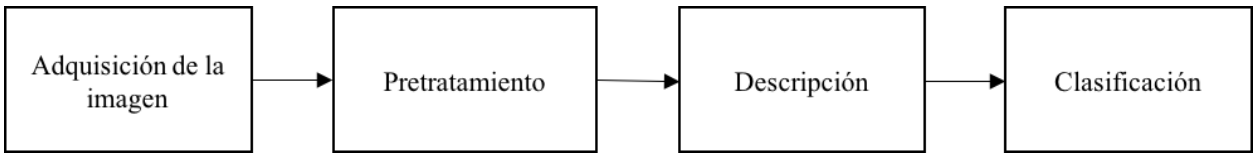

Figura 1. Etapas generales del reconocimiento de objetos

Según el problema a tratar, puede resultar necesaria una etapa de pretratamiento para contrarrestar una posible degradación producida al capturar la imagen, manifestándose en forma de ruido y pérdida de definición. En esta, se engloban las distintas operaciones y transformaciones que se aplican sobre las imágenes con el fin de contrarrestar dicha degradación, de manera que se faciliten las operaciones de las etapas posteriores. En algunos casos, y a continuación de esta etapa, puede ser necesario realizar una segmentación, cuyo objetivo sea la separación de los objetos para su identificación y descripción posterior. Se basa en la división de la imagen en diferentes regiones, estableciéndolas según una o más características a definir, y que tienen una fuerte relación con objetos del mundo real. Existen diferentes enfoques para realizar la segmentación, mediante técnicas basadas en umbralización (*Thresholding*) [5, Sec. 6.1], en detección de contornos [5, Sec. 6.2], en el crecimiento de regiones [6, Sec. 6.3], MSER (*Maximally Stable External Regions*) [7], según el color o la textura o la técnica del *template matching [5, Sec. 6.4.1]* son algunas de las más utilizadas.

Para identificar un objeto es necesario tener una descripción del mismo. En pos de resolver esta problemática, se implementará, en la tercera etapa, un algoritmo descriptor, cuya finalidad sea obtener características, fácilmente tratables, de una serie de imágenes previamente presentadas al sistema que contengan los objetos susceptibles a ser localizados. Existen diferentes tipos de descriptores, basados en regiones, como los descriptores de Fourier [8], u otros basados en la extracción de puntos característicos. Dentro de esta última categoría algunas alternativas son HOG (*Histogram of Oriented Gradientes*) [9], LBP (*Local Binary Pattern*) [10], SIFT (*Scale-invariant feature transform*) [11], SURF (*Speeded Up Robust Features*) [12] u ORB (*Oriented FAST and Rotated BRIEF*) [13].

Finalmente, en la última etapa, se desarrollará un algoritmo clasificador que, gracias a las descripciones de los diferentes objetos obtenidas previamente, asigne un elemento entrante no etiquetado en una categoría concreta conocida. En la actualidad, existen diversas técnicas de clasificación, pudiendo dividirse, principalmente, en dos grupos según el tipo de aprendizaje. Aprendizaje supervisado y no supervisado. En aprendizaje supervisado, para llevar a cabo la clasificación es necesario disponer de un conjunto de entrenamiento, el cual debe proveer la información necesaria para discernir si un futuro dato de entrada forma parte de la colección de objetos predefinidos. Algunos de los clasificadores más utilizados son SVM (*Support Vector Machine*) [14], *Adaboost* [15], K-NN (*K-Nearest Neighbor*) [16] o Redes neuronales artificiales (ANN) [17]. El aprendizaje no supervisado, no dispone de un conjunto de entrenamiento que permita etiquetar los datos, por lo que se hace necesario recurrir a técnicas de agrupamiento que construyan esas etiquetas. Este sistema de agrupamiento, o *clustering*, tiene como finalidad catalogar los objetos en conjuntos tales que los que estén en el mismo sean muy semejantes entre sí, pero diferentes a los objetos de otros grupos. Dentro de este tipo de clasificadores, se puede destacar el algoritmo K-MEANS [18].

Otra técnica para implementar un clasificador, consistiría en aplicar un algoritmo de *clustering* basado en la trasformada de Hough [19]. Para ello sería necesario desarrollar una etapa previa de emparejamiento de puntos característicos de la imagen de entrada y los de la base de datos que contiene los objetos a identificar.

Otra manera de implementar las etapas de descripción y clasificación, es recurrir a la implementación de Redes Neuronales Convolucionales (CNN) [20], algoritmos que trabajan emulando el comportamiento de las neuronas de un cerebro biológico, al igual que las ANN

pero de mayor complejidad. Consisten en múltiples capas con distintos propósitos, al inicio se encuentra la fase de extracción de características y, al final, se encuentra una etapa de clasificación a partir de las características extraídas.

1.3 Método utilizado

Tras analizar las diversas técnicas de implementación de los diferentes algoritmos de cada bloque, se desarrollará una etapa de descripción basada en el descriptor SURF y una etapa de clasificación utilizando un método de *clustering* basado en la transformada de Hough.

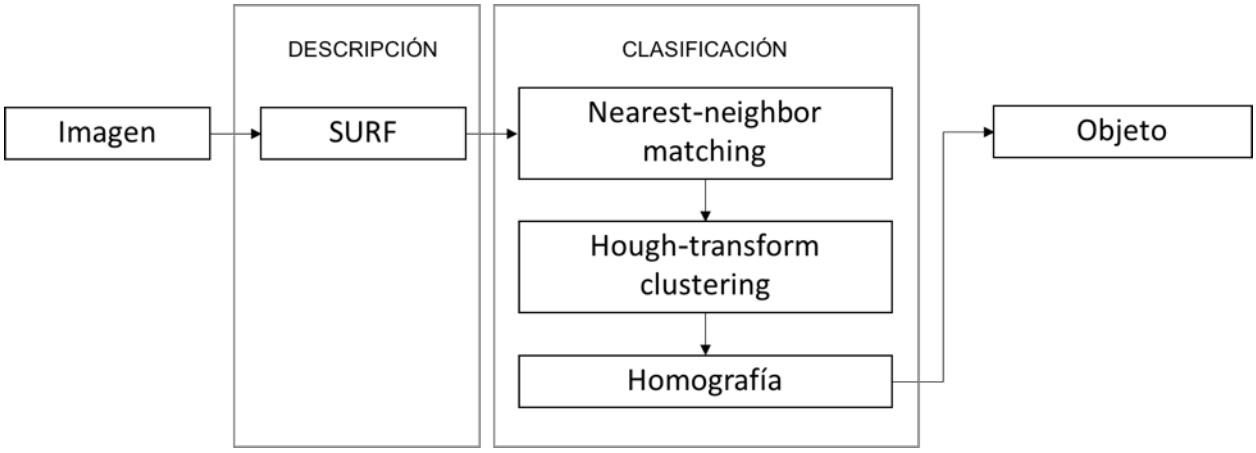

Figura 2. Etapas del método aplicado para el reconocimiento de objetos

El método que se seguirá para resolver el problema del reconocimiento de objetos es el presentado en la Figura 2, tomando como base el artículo *Object Recognition Using Hough-transform Clustering of SURF Features,* de Viktor Seib, Michael Kusenbach, Susanne Thierfelder y Dietrich Paulus [21].

A partir de una imagen de entrada, se extraen sus puntos característicos y descriptores mediante el algoritmo SURF [22]. En la siguiente etapa, se buscan correspondencias entre los puntos obtenidos y los puntos de las imágenes de la base de datos aplicando FLANN [23], una librería que busca el algoritmo óptimo basado en *nearest-neighbor*. En este paso se generan numerosas falsas correspondencias que serán refinadas en la siguiente fase, un

algoritmo de *clustering*, basado en características extraídas por la transformada de Hough. Permitirá generar hipótesis sobre localizaciones del objeto a reconocer mediante la agrupación de las correspondencias obtenidas, siendo estas verificadas tras la aplicación de una homografía entre las posiciones de los objetos, especificadas en la base de datos, y las correspondientes determinadas en la imagen.

Para llevar a cabo el reconocimiento se dispondrá de una base de datos formada por distintas imágenes con vistas diferentes de los objetos a reconocer. Las imágenes contendrán una vista de un único objeto centrado sobre un fondo de color uniforme para facilitar su segmentación. Para ello se empleará un algoritmo de umbralizado para permitir separar el objeto del fondo.

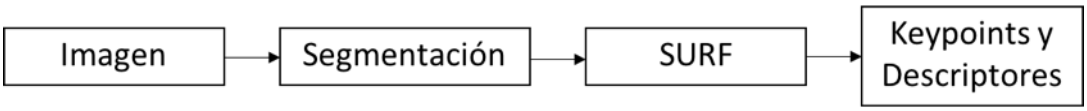

Figura 3. Fase de entrenamiento

Una vez segmentadas las imágenes, se aplicará el algoritmo SURF para extraer los puntos característicos y sus descriptores. Estos serán almacenados en archivos con el fin de agilizar la etapa de reconocimiento, no necesitando repetir la fase de entrenamiento cada vez que se ejecute el programa.

2. Marco teórico

2.1 Puntos característicos y descriptores

Para obtener la descripción de los objetos se utilizará un descriptor basado en la extracción de puntos característicos, o *keypoints*. Estos puntos deben cumplir una serie de requisitos. Repetitividad, es decir, deben poder localizarse a pesar de transformaciones geométricas o fotométricas. Saliencia, los puntos deben de ser fácilmente distinguibles. Representación compacta y eficiente, habiendo menos puntos característicos que píxeles en la imagen. Localización, un punto característico debe ocupar un área relativamente pequeña y ser robusto frente a oclusiones y desorden en la imagen.

La detección de puntos característicos implica la individualización de puntos que, por sus características, ayudan a definir las imágenes. Un detector ideal localiza dichos puntos repetidamente a pesar del cambio del punto de vista, siendo fiable ante transformaciones de la imagen.

La descripción de dichos puntos consiste en la extracción de información relevante y distintiva de la región que los rodea, de manera que la misma estructura pueda ser reconocida si es encontrada en otra imagen.

Existen algoritmos que permiten llevar a cabo un eficiente procesamiento en la detección y descripción de puntos característicos, como SIFT(*Scale-invariant feature transform*) [11], SURF (*Speeded Up Robust Features*) [12] u ORB (*Oriented FAST and Rotated BRIEF*) [13].

2.2 SURF (Speeded Up Robust Features)

El algoritmo SURF, desarrollado por Herbert Bay [22], toma como base el SIFT, desarrollado por David G. Lowe [24]. Es un detector y descriptor de puntos característicos. Ambos descriptores son invariantes ante cambios de escala, orientación e iluminación, presentando resultados similares. La característica diferenciadora es que el SURF mejora la velocidad de cálculo de su antecesor. Esto se debe a que, en la generación de los descriptores, se reduce la dimensión de los vectores y la complejidad en el cálculo. Los puntos detectados continúan siendo suficientemente característicos e igualmente repetitivos.

2.2.1 Detector Fast-Hessian

El primer paso de este algoritmo es la búsqueda de puntos característicos. El detector, llamado Fast-Hessian, se basa en la matriz Hessiana debido a su buen rendimiento en tiempo de cálculo, precisión y robustez. Se utiliza una aproximación del determinante de dicha matriz para seleccionar la ubicación y la escala de los puntos.

$$H(p,\sigma) = \begin{bmatrix} L_{xx}(p,\sigma) & L_{xy}(p,\sigma) \\ L_{xy}(p,\sigma) & L_{yy}(p,\sigma) \end{bmatrix} \quad (1)$$

Dado un punto p = (x,y) en una imagen I, la matriz Hessiana H(p, σ) en p y escala σ se define como vemos en (1). Donde Lxx (p, σ) es la convolución de la segunda derivada de la gaussiana $\frac{\partial}{\partial p^2} g(\sigma)$ con la imagen I en el punto p. De la misma manera se definen L_{xy}(p, σ) y L_{yy}(p, σ). Siendo la distribución gaussiana: $g(x,y,\sigma) = e^{-\frac{(x^2+y^2)}{2\sigma^2}}$.

Los filtros gaussianos son óptimos para el análisis de ubicación y escala, sin embargo, en la práctica deben ser discretizados y recortados, ver Figura 4. Para acelerar los cálculos, los filtros gaussianos de segundo orden se aproximan por filtros de caja, pudiendo ser eficientemente calculados mediante imágenes integrales. El concepto de imagen integral, desarrollado en el artículo de Viola y Jones en 2001 [25], se expone en el apartado 2.2.2.

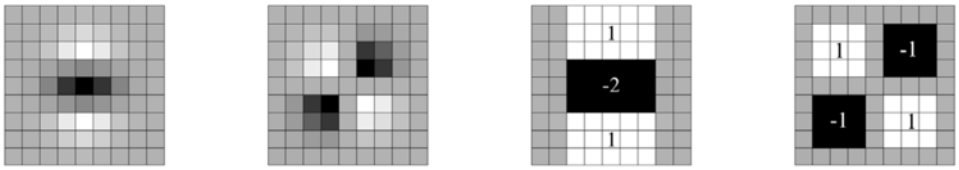

Figura 4. Aproximaciones de las derivadas parciales de segundo orden del gaussiano (izquierda), por filtros de caja (derecha). Imagen extraída del artículo original de SURF

Los filtros de caja 9x9 de la Figura 4 son aproximaciones de la segunda derivada de la gaussiana con $\sigma = 1.2$.

Los puntos característicos deben ser localizados a distintas escalas, entre otros motivos porque la búsqueda de correspondencias a menudo requiere su comparación entre imágenes de escalas diferentes. Los espacios de escala se aplican normalmente como pirámides de imagen, considerando cada planta de la pirámide una octava. Las imágenes se suavizan repetidamente con un filtro gaussiano y, posteriormente, se submuestrean con el fin de conseguir un nivel superior de la pirámide. Debido al uso de filtros de caja e imágenes integrales, no tenemos que aplicar iterativamente el mismo filtro a la salida de una capa previamente filtrada, sino aplicar estos filtros de cualquier tamaño exactamente a la misma velocidad directamente en la imagen original. Por lo tanto, el espacio de escala se analiza escalando el tamaño del filtro en vez de reducir iterativamente el tamaño de la imagen. La salida del filtro 9×9 anterior se considera como la capa de escala inicial o escala $s = 1.2$, correspondiente a derivados gaussianos con $\sigma = 1.2$. Las capas siguientes se obtienen filtrando la imagen con máscaras gradualmente más grandes, teniendo en cuenta la naturaleza discreta de las imágenes integrales y la estructura específica de nuestros filtros. Esto resulta en filtros 9x9, 15x15, 21x21, 27x27, etc. Al ascender una octava se duplica el tamaño del filtro.

Para encontrar los puntos de interés, se calcula el determinante de la aproximación de la matriz Hessiana, obteniendo sus localizaciones y escalas.

1.1.1 Descriptor SURF

El descriptor tiene como objetivo proporcionar una descripción única y robusta de un conjunto. Describe la distribución de intensidad del contenido dentro del punto de interés de los puntos vecinos. Es generado basándose en el área circundante de un punto de interés, por lo que se obtiene un vector descriptor para cada punto de interés.

El siguiente paso, una vez detectados los puntos característicos, consiste en el cálculo de la orientación de dichos puntos. Este proceso se llevará a cabo mediante el cálculo de la respuesta de *Haar [26]* en las direcciones x e y, en una región circular alrededor del punto de interés de radio 6*s*, siendo *s* la escala del punto de interés. Este paso otorga al descriptor invariancia ante rotación.

Para extraer el descriptor se construye una región cuadrada alrededor del punto de interés con la orientación calculada en el paso anterior. La región se divide regularmente en 4x4 sub-regiones cuadradas. Para cada una de las regiones se van calculando las respuestas de Haar en las direcciones x e y, se les llamará dx y dy. Para proporcionar una mayor robustez a deformaciones, ruido y traslaciones, se someten los resultados a un filtro Gaussiano. Para cada región se suman los resultados y se calculan los valores absolutos. Así, cada sub-región proporciona un vector v. Las respuestas de Haar se calculan aprovechando las imágenes integrales por su eficiencia y rapidez de cálculo.

$$v = (\sum dx, \sum dy, \sum |dx|, \sum |dy|) \quad (2)$$

El descriptor del SURF se obtiene mediante la unión de los vectores de las diferentes regiones.

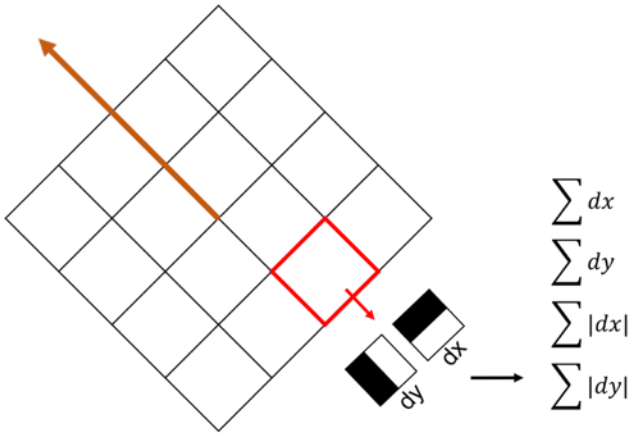

Figura 5. Cálculo de las características de Haar en cada subregión.

2.2.2 Imagen integral

El concepto de imagen integral fue desarrollado en el artículo de Viola y Jones en 2001 [25].

La imagen integral es una representación matricial que contiene información de imagen global. Está construida de manera que su valor ii(x,y) en la posición (x,y) representa la suma de todos los valores de los píxeles situados arriba y a la izquierda del punto en cuestión (x,y),

$$ii(x,y) = \sum_{x' \leq x,\, y' \leq y} i(x',y') \quad (3)$$

donde ii(x,y) es la imagen integral e i(x,y) es la imagen original. La imagen integral se puede calcular en un paso sobre la imagen original mediante las siguientes ecuaciones:

$$s(x,y) = s(x,y-1) + i(x,y) \quad (4)$$

$$ii(x,y) = ii(x-1,y) + s(x,y) \quad (5)$$

Donde s (x, y) es la suma de la fila acumulativa.

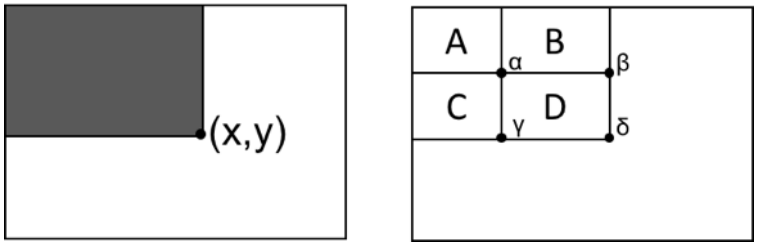

Figura 6. Imagen integral

La suma de los píxeles dentro del rectángulo D se puede calcular con cuatro referencias de matrices. El valor de la imagen integral en la posición δ es la suma de los píxeles en el rectángulo A. El valor en la posición β es A + B, en la posición γ es A + C, y en la posición δ es A + B + C + D. La suma de los píxeles del rectángulo D se puede calcular como:

$$D_{sum} = ii(\delta) + ii(\alpha) - (ii(\beta) + ii(\gamma)) \quad (6)$$

donde $ii(\alpha)$ es la imagen integral en el punto α, igual para $ii(\beta), ii(\delta), ii(\gamma)$.

Utilizando la imagen integral, cualquier suma rectangular se puede calcular en cuatro valores de imagen integral (ver Figura 6). Claramente la diferencia entre dos sumas rectangulares puede ser calculada en ocho referencias. Dado que las características de dos rectángulos definidas anteriormente implican sumas rectangulares adyacentes, pueden calcularse en seis referencias, ocho en el caso de las características de tres rectángulos y nueve para características de cuatro rectángulos.

2.3 Emparejamiento de puntos característicos

Los puntos característicos extraídos de una imagen son emparejados con los puntos de los objetos de la base de datos mediante un algoritmo basado en correspondencias del vecino más

cercano. Consiste en comparar los descriptores de los puntos buscando correspondencias entre ellos.

Para muchos problemas de visión por computador, la parte más costosa desde el punto de vista computacional de muchos algoritmos consiste en encontrar correspondencias entre puntos, en ocasiones con vectores de muchas dimensiones que representan las muestras de entrenamiento. Con el fin de evitar estos problemas, Marius Muja y David G. Lowe desarrollaron FLANN (*Fast Library for Approximate Nearest Neighbors*) [27]. FLANN es una librería que busca el mejor algoritmo para estimar las mejores correspondencias entre puntos sin realizar todas las comparaciones posibles, como hacen los algoritmos de fuerza bruta.

2.4 *Clustering* mediante Transformada de Hough

La transformada de Hough es una técnica ampliamente utilizada en visión por computador para la detección de figuras en una imagen. Este algoritmo fue patentado por Paul Hough en 1962 [28], y se aplicaba solamente a la detección de rectas. Posteriormente, se extendió a la identificación de cualquier figura que se pudiese describir a partir de unos cuantos parámetros de manera matemática, como circunferencias y elipses. Esta ampliación fue desarrollada por Richard Duda y Peter Hart en 1972 [29]. Finalmente, Dana H. Ballard, en 1981 [30], desarrolló este método hasta la detección de figuras arbitrarias, lo que se conoce como Transformada de Hough generalizada.

La transformada de Hough generalizada fue una modificación de la original basada en el principio del *Template Matching [5, Sec. 6.4.1],* permitiendo la detección de un objeto arbitrario descrito por su modelo. En la trasformada de Hough generalizada, este problema se traduce en encontrar el parámetro de transformación que hace corresponder el modelo en la imagen.

El algoritmo de la transformada de Hough utiliza una matriz llamada acumulador, de dimensión igual al número de parámetros desconocidos del problema. Para la construcción

del acumulador es necesario discretizar los parámetros que describen la figura a detectar. Cada celda del acumulador representa una figura cuyos parámetros se pueden obtener a partir de la posición de dicha celda.

En este proyecto, se aplicará un método de *clustering* basado en la transformada de Hough para encontrar agrupaciones de las correspondencias entre puntos característicos de la imagen de entrada y las imágenes de la base de datos, proporcionando así posibles posiciones de los objetos a detectar.

Esta etapa de agrupación se implementará como se describe en según el artículo *Object recognition using hough-transform clustering of surf features* de V. Seib, M. Kusenbach, S. Thierfelder, y D. Paulus [21].

Se crea un acumulador de cuatro dimensiones (x,y,θ,σ) para proporcionar hipótesis de posibles localizaciones del objeto a detectar. Cada dimensión se corresponderá con un parámetro diferente, siendo estos la posición del centroide (x,y), la rotación (θ) y la escala (σ) de los objetos hipotéticos. El objetivo es encontrar una posición del objeto consistente para eliminar falsas correspondencias de la etapa anterior.

Cada correspondencia de puntos característicos es una hipótesis para una localización del objeto y, se añade a la celda correspondiente del acumulador. Para minimizar errores de discretización, cada hipótesis vota además en la casilla de índice inmediatamente superior, lo que da un total de 16 entradas por correspondencia. Los grupos de máximos en el espacio de Hough corresponden a las posiciones de objetos más probables, mientras que las casillas correspondientes a objetos erróneos obtienen sólo pequeños votos, eliminando así los valores atípicos y obteniendo las posiciones correctas del objeto.

Para el cálculo de la celda de la posición correspondiente al centroide del objeto encontrado en la escena, se parte del centroide del objeto de la imagen de entrenamiento. Primero, se obtiene el vector de traslación (v_0) del centroide (c_0) a la posición de los puntos característicos obtenidos en las correspondencias (p_0), en la imagen de entrenamiento.

$$v_0 = p_0 - c_0 \quad (7)$$

Este vector es normalizado mediante la relación de escalas de los puntos característicos de la imagen de la escena (σ_s) y de la de entrenamiento (σ_0).

$$v_0 = (p_0 - c_0) \cdot \frac{\sigma_s}{\sigma_0} \quad (8)$$

El vector resultante es ahora rotado para tener en cuenta la posible rotación del objeto en la escena. Para ello, se aplicará una matriz de rotación con ángulo α, siendo este el ángulo de rotación entre las orientaciones de los puntos característicos del objeto en la imagen de entrenamiento (θ_0) y en la escena (θ_s), $\alpha = |\theta_0 - \theta_s|$.

$$v_s = \begin{pmatrix} \cos\alpha & -sen\ \alpha \\ sen\ \alpha & \cos\alpha \end{pmatrix} \cdot v_0 \quad (9)$$

Finalmente, se puede obtener el centroide del objeto en la escena (c_s) a partir del vector de traslación del centroide al punto característico de la escena (p_s) que tiene una correspondencia con el punto característico de la imagen de entrenamiento p_0.

$$c_s = (c_{s,x}, c_{s,y}) = p_s - v_s \quad (10)$$

En cuanto al cálculo del índice de las celdas correspondientes del acumulador de las dimensiones x e y, i_x e i_y respectivamente, se establece una relación directamente proporcional entre el tamaño de la imagen (w, h) y del acumulador.

$$i_x = c_{s,x} \cdot \frac{b_x}{w} \quad (11)$$

$$i_y = c_{s,y} \cdot \frac{b_y}{h} \quad (12)$$

La tercera dimensión del acumulador se corresponde con la rotación (θ). El índice de la celda correspondiente (i_r), se calcula a partir de α, siendo éste un ángulo entre [0, 2π] radianes.

$$i_r = \alpha \cdot \frac{b_r}{2\pi} \quad (13)$$

Siendo b_r el número total de casillas reservadas para esta dimensión.

En la última dimensión del acumulador se almacena la información correspondiente a la escala (σ). El cálculo del índice de la casilla correspondiente se calcula a través de la siguiente ecuación.

$$i_s = \left(\frac{\log_2\left(\frac{\sigma_s}{\sigma_0}\right)}{2 \cdot (n-1)} + 0.5 \right) \cdot b_s \quad (14)$$

Siendo b_s el número total de casillas reservadas para esta dimensión, σ_s y σ_0 la escala de los puntos característicos de las correspondencias entre la imagen de la escena y la de entrenamiento respectivamente y *n* el número de octavas utilizado por el algoritmo SURF en la extracción de puntos característicos.

Además de votar en las casillas calculadas, también se hará en las casillas de índice inmediatamente superior, minimizando los errores de discretización.

Una vez aplicado el método, todas las correspondencias encontradas en etapas anteriores se traducen en votos en el acumulador de Hough. La casilla más votada será la seleccionada para ser procesada en la siguiente etapa, eliminando así muchas de las falsas correspondencias obtenidas. Se aplica una homografía sobre dicha casilla, calculando una transformación en perspectiva entre las características de la celda y los puntos correspondientes en la base de datos bajo la suposición de que todas las características se encuentran en un plano 2D.

2.5 Homografía

Una homografía es una transformación proyectiva que determina una correspondencia entre dos figuras geométricas planas.

La homografía entre dos vistas se puede estimar con correspondencias entre puntos de las imágenes, de manera que se obtenga una matriz H que mapea puntos de una vista a otra.

$$p' = H \cdot p = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (15)$$

Siendo *p'* un punto de la primera vista y *p* un punto de la segunda. Hay un factor de escala arbitrario, $h_{33} = 1$. La matriz de homografía sólo tiene 8 grados de libertad, por ende, puede ser estimada si conocemos al menos cuatro correspondencias entre puntos de las diferentes vistas.

La aplicación de la homografía permite verificar si las agrupaciones realizadas en la etapa anterior se corresponden con el objeto buscado, ya que determina la correspondencia entre dos figuras planas.

Para identificar la mejor homografía para el conjunto de correspondencias se utiliza RANSAC (*Random Sample Consensus*) [31].

2.6 RANSAC (*Random Sample Consensus*)

RANSAC [31], se trata de un método iterativo utilizado para calcular los parámetros de un modelo matemático de un conjunto de datos que contiene valores atípicos o *outliers,* sin que estos influyan en el resultado.

El algoritmo de RANSAC se basa en la repetición de los siguientes pasos:

1. Seleccionar un subconjunto aleatorio de los datos originales.
2. Un modelo se ajusta en el conjunto de esos datos.
3. Todos los demás datos se prueban contra el modelo ajustado. Los puntos que se ajusten al modelo se consideran como parte del conjunto de consenso.
4. El modelo estimado es bueno si se han clasificado suficientes puntos como parte del conjunto de consenso.
5. Después, el modelo puede ser mejorado volviendo a estimar usando todos los miembros del conjunto de consenso.

Este procedimiento se repite un número fijo de veces, produciendo o bien un modelo que es rechazado porque no hay suficientes puntos en el conjunto de consenso, o un modelo aceptado con suficientes puntos en el conjunto de consenso. En este último caso, mantenemos el modelo si su conjunto de consenso es más grande que el modelo guardado previamente.

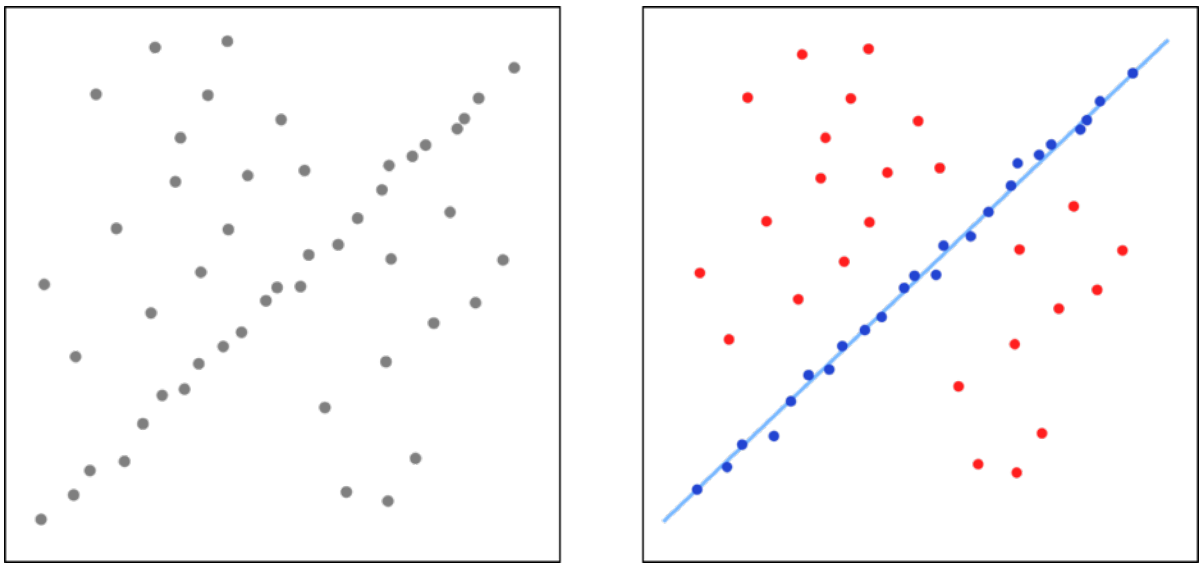

Figura 7. Izquierda: Conjunto de datos con outliers. Derecha: Línea calculada mediante RANSAC, los outliers no influyen en el resultado.

2.7 Modelos de color

La escala de grises es una escala empleada en la imagen digital en la que el valor de cada píxel posee un valor equivalente a una graduación de gris. En imágenes de 8 bits, puede haber hasta 256 tonos de gris. Cada píxel de una imagen en escala de grises tiene un valor de brillo comprendido entre 0, negro, y 255, blanco.

Los dispositivos de captura y visualización de imágenes captan y generan el color mezclando los colores primarios de la luz, rojo, verde y azul, con diferentes intensidades, modelo RGB. Sin embargo, la relación entre la cantidad de cada color no resulta intuitiva para el ojo humano. Por ello, surgieron otros modelos de color como el HSV, basado en la manera de organización de los colores al ser percibidos por las personas en términos de atributos que definen el color, tono, saturación y brillo.

2.7.1 Modelo RGB

El RGB (*Red, Green, Blue*) es un modelo de color basado en la síntesis aditiva, pudiendo representar un color mediante la mezcla por adición de los tres colores de luz primarios, rojo, verde y azul.

Este modelo maneja 3 canales diferentes, uno para cada color primario. La representación tridimensional del modelo es cúbica. Para indicar con qué proporción es mezclado cada color, se asigna un valor a cada uno de los colores primarios, de manera que el valor 0 representa que no interviene en la mezcla y, a medida, que ese valor aumenta, se entiende que aporta más intensidad a la mezcla. Aunque el intervalo de valores podría ser cualquiera, cada canal se suele codificar en un byte, por lo tanto, el valor máximo es 255.

Así, la intensidad de cada una de las componentes se mide según una escala que va del 0 al 255 y cada color es definido por un conjunto de valores escritos entre paréntesis, correspondiendo a valores *R, G, B*.

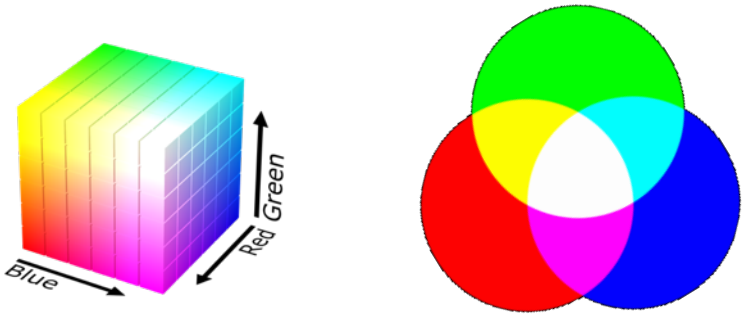

Figura 8. Modelo RGB.

La ausencia de color, representada con un valor de 0 en los tres canales, genera el negro puro, mientras que la suma de los tres colores primarios, es decir, un valor de 255 en todos los planos de color, genera el blanco puro. La escala de grises en este nuevo espacio tridimensional es la diagonal del cubo, que une el vértice del color blanco puro (0,0,0) con el vértice del negro puro (255,255,255). A lo largo de esta recta, los tres planos de color coinciden en su valor y cada punto de la recta es equidistante hacia los ejes de cada dimensión.

2.7.2 Modelo HSV

El modelo HSV (*Hue, Saturation, Value*) define un modelo de color en términos de sus componentes, tono, saturación y brillo. Se trata de una transformación no lineal del espacio de color RGB.

Este modelo posee una representación tridimensional cónica. La circunferencia base del cono se corresponde al tono, que se representa como un grado de ángulo cuyos valores posibles van de 0 a 360°. Los colores primarios se encuentran separados unos de otros 120º.

El brillo se representa en la altura del cono, coincidiendo con el eje que une el blanco puro y el negro. El blanco puro se encuentra en el centro de la circunferencia base y el negro en el vértice del cono. Los valores posibles van del 0 al 100%, siendo 0 el negro y,

dependiendo de la saturación, 100 podría corresponder a blanco o un color más o menos saturado.

La saturación es la distancia al eje de brillo. Los valores posibles van del 0 al 100%. Cuanto menor sea la saturación, el color poseerá una mayor tonalidad grisácea.

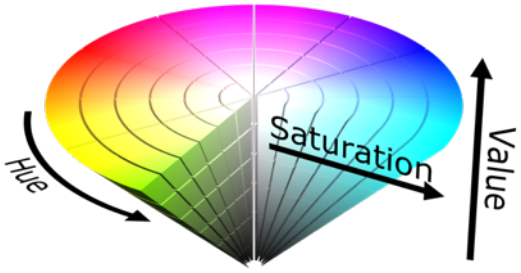

Figura 9. Modelo HSV

A partir del modelo RGB se puede obtener geométricamente el modelo HSV. Al proyectar el cubo RGB sobre un plano perpendicular a la diagonal del cubo que va desde el negro puro al blanco puro, se obtiene un hexágono cuyos vértices corresponden a los colores rojo, amarillo, verde, cian, azul y magenta. El plano de proyección se denomina plano cromático.

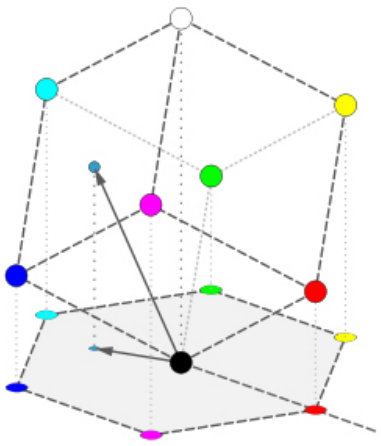

Figura 10. Proyección del cubo RGB en el plano cromático

El tono se representa por el ángulo del vector que apunta a un color del cubo una vez proyectado sobre el plano cromático, mientras que la saturación sería la distancia desde el punto proyectado al origen del plano. A partir de ese plano, se añade el brillo en el eje perpendicular, llegando a la representación tridimensional cónica.

2.8 Umbralizado

El umbralizado es una técnica de segmentación empleada cuando la diferencia entre los objetos a extraer y el fondo es notable. Los principios que rigen este método son la similitud entre los píxeles pertenecientes al objeto y sus diferencias respecto al resto. Por lo tanto, se suele aplicar para segmentar un objeto u objetos de un fondo uniforme.

Se busca un umbral óptimo que permita distinguir en una imagen el fondo del primer plano. En el caso que se expone en este proyecto, se pretende separar un objeto de un fondo uniforme de color verde.

Para mayor facilidad, se transforma la imagen RGB al modelo HSV. En este modelo resulta más sencillo definir un rango de colores, ya que, el representar cada uno como un vector de tres: rojo, verde y azul, acarrea un problema a la hora de reconocer un color, pues, dependiendo de las condiciones de iluminación, puede tener varios tonos. Mientras que en el modelo HSV será más cómodo localizar un color en concreto, puesto que para uno en particular se podrá definir un rango más o menos pequeño de tono y mediante la variación de los parámetros de la saturación y el brillo se podrá adaptar al rango de colores buscado con mayor facilidad.

3. Experimentos y resultados

Los algoritmos expuestos anteriormente se desarrollan sobre el sistema operativo Ubuntu 14.04. Se han implementado en lenguaje C++ a partir de las librerías de código abierto OpenCV (*Open Source Computer Vision*) [32], en la versión 3.1, especializadas en visión artificial.

3.1 Fase de entrenamiento

Se dispone de una base de datos que alberga una colección de imágenes, de tamaño 1024x768 píxeles, con diferentes vistas de los objetos que queremos localizar. Se somete a estas imágenes a una fase de entrenamiento con el fin de obtener sus puntos característicos y descriptores.

Las imágenes se capturan con un fondo homogéneo para facilitar su segmentación. Al utilizar un fondo de color uniforme, se empleará la técnica de umbralizado. La imagen se transformará al espacio de color HSV y, se establecerán unos límites en el canal H (tono), para eliminar el color verde del fondo. También se establecen límites para los canales de la saturación y el brillo para conseguir que el color del fondo sea seleccionado.

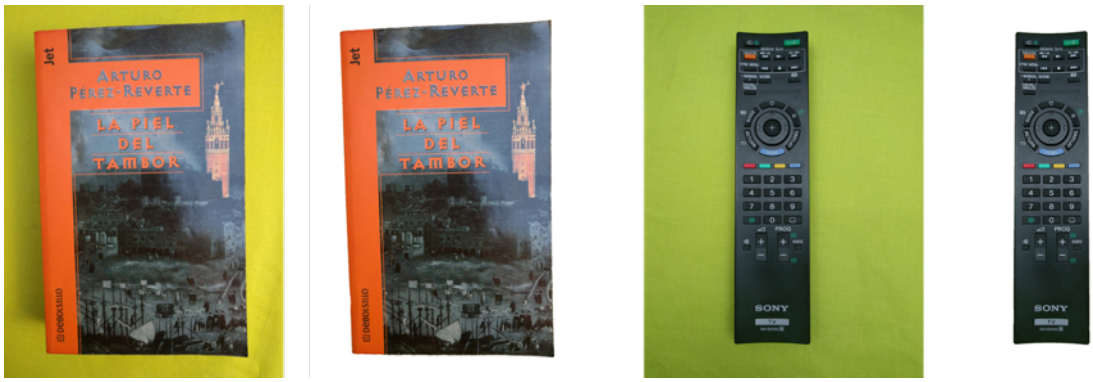

Figura 11. Imágenes de la base de datos. Originales y segmentadas.

Una vez segmentadas, se les aplicará el algoritmo SURF, tras ser convertidas a escala de grises, para extraer los puntos característicos y los descriptores, siendo almacenados en archivos tipo YAML, archivos de texto plano pensados para el almacenamiento de datos. De

esta manera se podrá acceder a estos parámetros rápidamente. En estos archivos, además de los puntos característicos y descriptores, también se almacenarán los centroides de los objetos contenidos en las imágenes, ya que es un parámetro necesario para aplicar el *clustering*.

3.2 Comparación algoritmos SURF y SIFT

Se han llevado a cabo una serie de pruebas comparando los algoritmos SIFT y SURF. El estudio se lleva a cabo sobre dos objetos diferentes (O1 y O2), 12 imágenes cada uno, que forman parte de la base de datos. El tamaño de las imágenes para esta prueba es de 1024x768 píxeles. La comparación se realiza mediante el tiempo de procesamiento de cada algoritmo y el número de puntos característicos extraídos. Se fija un número de *keypoints* a extraer por SIFT, necesario en OpenCV, de 2000 y se repite el experimento cinco veces para los mismos objetos. Además, se comprueba el número de *keypoints* extraídos para las 12 imágenes del objeto 2.

TABLA 1 . COMPARACIÓN TIEMPO DE ejecución ALGORITMOS SURF Y SIFT, EN SEGUNDOS

	1		2		3		4		5	
OBJETO	O1	O2	O1	O2	O1	O2	O1	O2	O1	O2
SURF	2.92	3.07	2.91	3.36	2.89	2.75	2.93	2.78	2.90	2.81
SIFT	3.28	4.49	3.27	4.43	3.27	4.25	3.28	4.16	3.29	4.22

TABLA 2. COMPARACIÓN NÚMERO DE PUNTOS CARACTERÍSTICOS EXTRAÍDOS, ALGORITMOS SURF Y SIFT

Imagen (O2)	0	1	2	3	4	5	6	7	8	9	10	11
SURF	2359	1861	710	2570	1435	1241	1042	889	1242	1446	1064	1080
SIFT	2000	1436	674	2000	1668	1380	719	695	1473	1429	1237	1288

Según los datos recogidos en la Tabla 1, se puede comprobar que SURF trabaja a mayor velocidad, incluso para un mayor número de puntos característicos, como se refleja en la Tabla 2, extrayendo, de media, 343 puntos más por imagen.

3.3 Color SURF

El algoritmo SURF está diseñado para trabajar en escala de grises. Sin embargo, el color provee información valiosa en problemas de detección, por ello, partiendo de una imagen RGB, se aplica el algoritmo SURF en sus tres canales individualmente, consiguiendo así un mayor número de puntos característicos, aportando más información de las imágenes. Esta idea se desarrolla con mayor profundidad en el artículo *Increase Efficiency of SURF using RGB Color Space* [33].

Se puede comprobar que al aplicar SURF en imágenes RGB se obtienen un mayor número de puntos característicos. En concreto, en la imagen de la Figura 12, al aplicar el algoritmo en color, se extraen 8170 puntos, frente a los 2573 de la imagen en escala de grises. Sin embargo, cuando se aplica este método a todas las imágenes de la base de datos, el número de puntos característicos a manejar por el programa aumenta drásticamente. Lo que deriva en un coste computacional demasiado elevado, ralentizando el proceso en exceso. Por ello, se descarta este método y se aplica el algoritmo SURF a imágenes en escala de grises, obteniendo resultados igualmente satisfactorios.

Figura 12. Keypoints aplicado SURF a imagen RGB (izquierda).
Keypoints aplicando SURF a imagen en escala de grises (derecha).

3.4 Comparación FLANN y Brute Force

Se ha realizado una comparación entre FLANN y un algoritmo de fuerza bruta, *Brute Force* [34], que consiste en comparar cada punto con todos los demás. El experimento se ha desarrollado sobre imágenes de la base de datos y una imagen que contiene dicho objeto, en escala de grises.

Tabla 3. Comparación tiempo de ejecución FLANN y algoritmo de fuerza bruta, en segundos.

Tiempo FLANN (s)	7.59	7.61
Tiempo *Brute Force* (s)	25.89	25.04

Los resultados arrojados por esta prueba han sido que el FLANN es un método indudablemente más veloz que el algoritmo de fuerza bruta, consiguiendo el mismo número de correspondencias, 14498 en este caso concreto.

3.5 Etapas del reconocimiento

El primer paso consiste en aplicar el algoritmo SURF para extraer los puntos característicos y descriptores de la imagen de entrada, ver Figura 13.

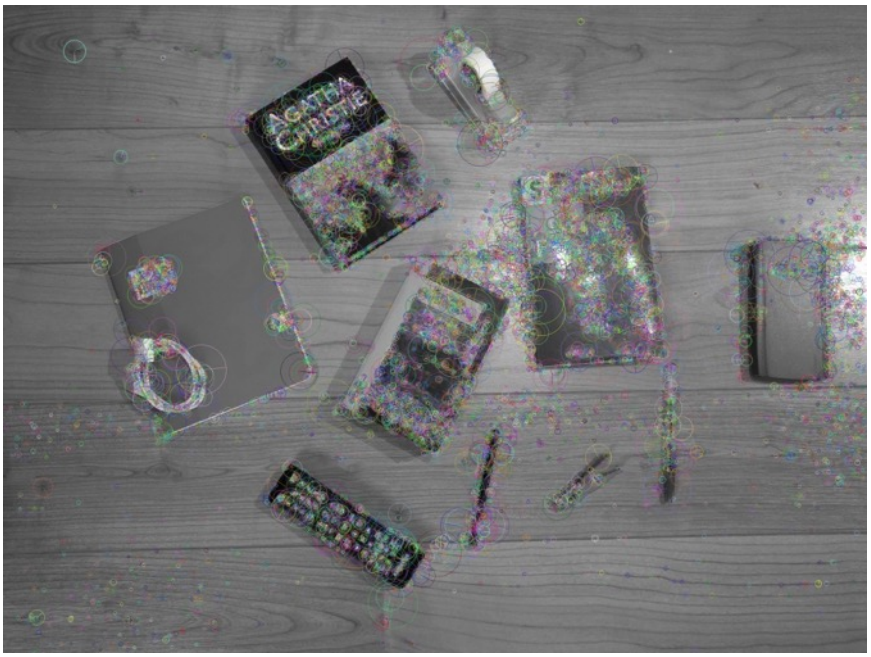

Figura 13. Aplicación del algoritmo SURF a una escena. Extracción de puntos característicos y descriptores.

En la siguiente etapa, se buscan correspondencias entre esos puntos extraídos y los de las imágenes de la base de datos mediante FLANN. Como se puede apreciar en la Figura 14, se obtienen *outliers* o falsas correspondencias.

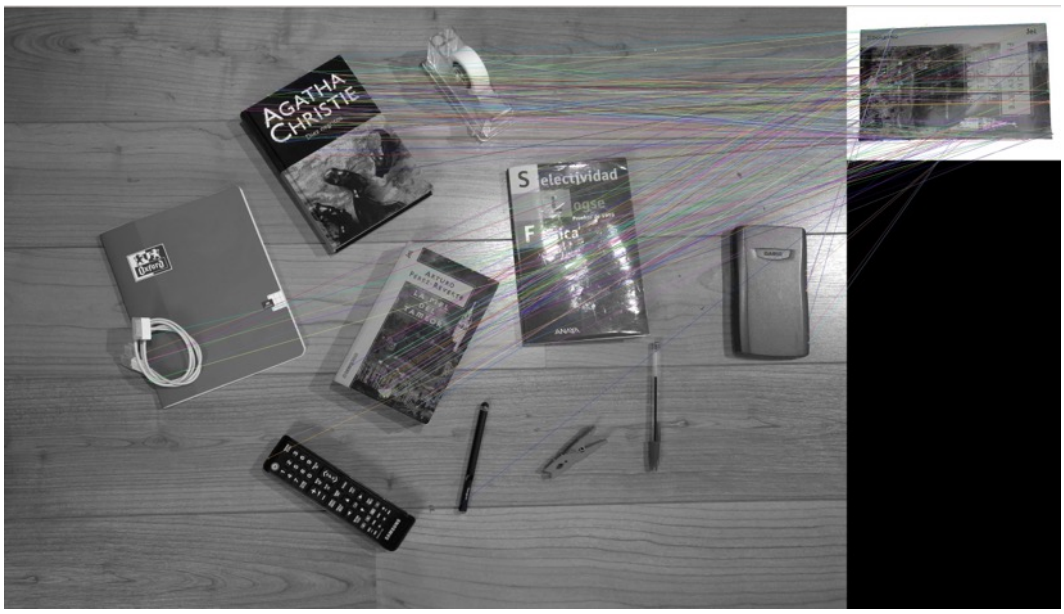

Figura 14. Obtención de correspondencias entre la imagen de entrenamiento (derecha) y la imagen de la escena (izquierda) mediante FLANN. Se pueden apreciar *outliers*.

Posteriormente, se realiza una agrupación de las correspondencias mediante el algoritmo de *clustering* basado en la transformada de Hough expuesto anteriormente, eliminando los *outliers*. Una vez agrupadas las correspondencias, se busca la imagen de la base de datos que más votos tenga en el acumulador y, se aplica una homografía entre los puntos de las correspondencias de la imagen de la base de datos y la escena, comprobando así si se trata del objeto correspondiente. Para aplicar la homografía, es necesario conocer las esquinas del objeto conocido de la base de datos. Para ello, se crea una *bounding box*, cuadrilátero más pequeño capaz de almacenar al objeto, y se toman sus esquinas.

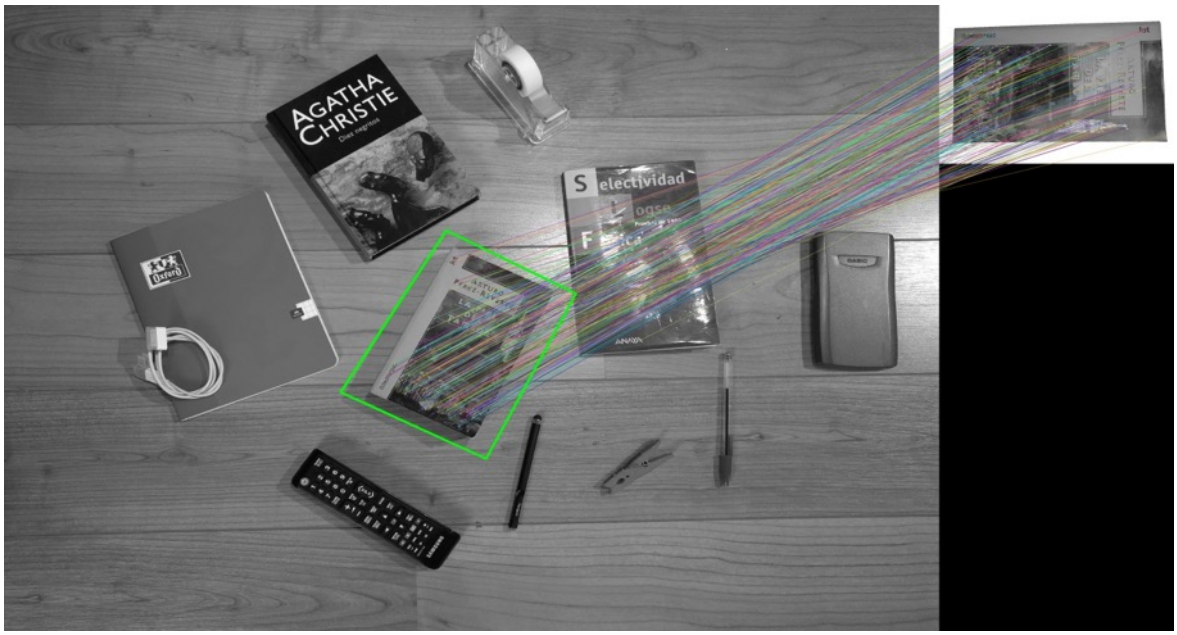

Figura 16. Resultado tras la aplicación del algoritmo de clustering y homografía en una escena sencilla

En las figuras siguientes, se muestran diferentes resultados de la última etapa ante cambios de orientación u oclusiones.

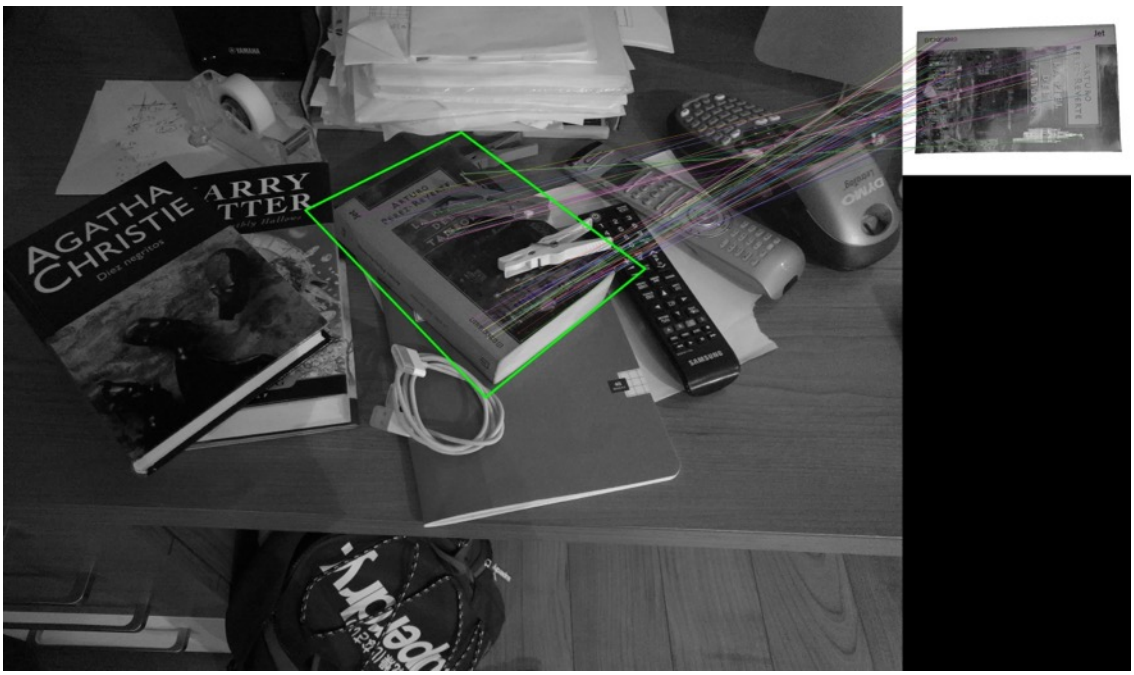

Figura 15. Resultado tras la aplicación del algoritmo de clustering y homografía en una escena más complicada

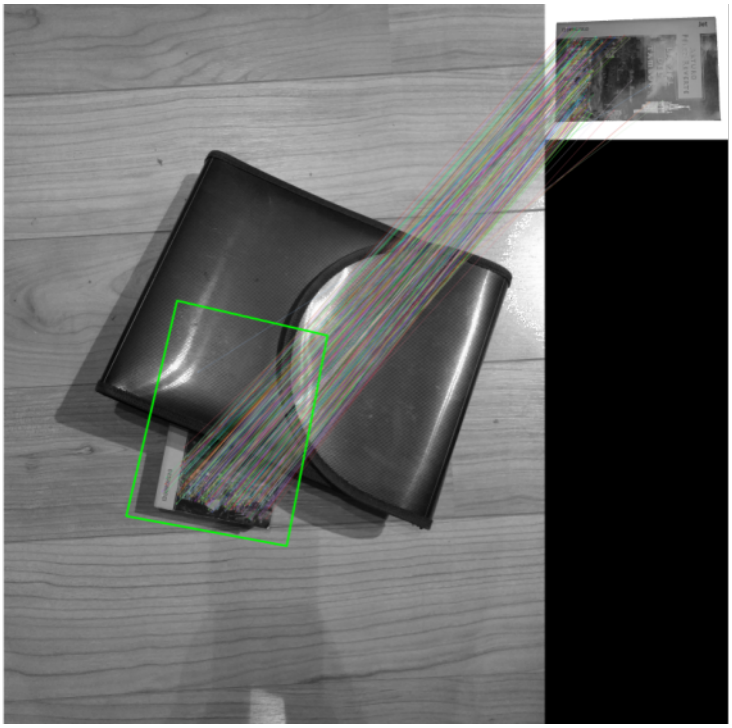

Figura 17. Resultado tras la aplicación del algoritmo de clustering y homografía con oclusión parcial del objeto.

3.6 Resultado final

En este apartado se exponen los resultados finales del proyecto ante diferentes escenas. En la Figura 18 y Figura 19 se muestra el reconocimiento de los 5 objetos que componen la base de datos.

Figura 18. Reconocimiento de los 5 objetos de la base de datos.

Figura 19. Reconocimiento de los 5 objetos de la base de datos con oclusión parcial

Sin embargo, no es necesario que se encuentren presentes todos los objetos, como se muestra en la Figura 20.

Figura 20. Reconocimiento en escena con 3 de los 5 objetos

Además, se han hecho pruebas sobre escenas más complejas, con orientaciones diferentes de los objetos a detectar.

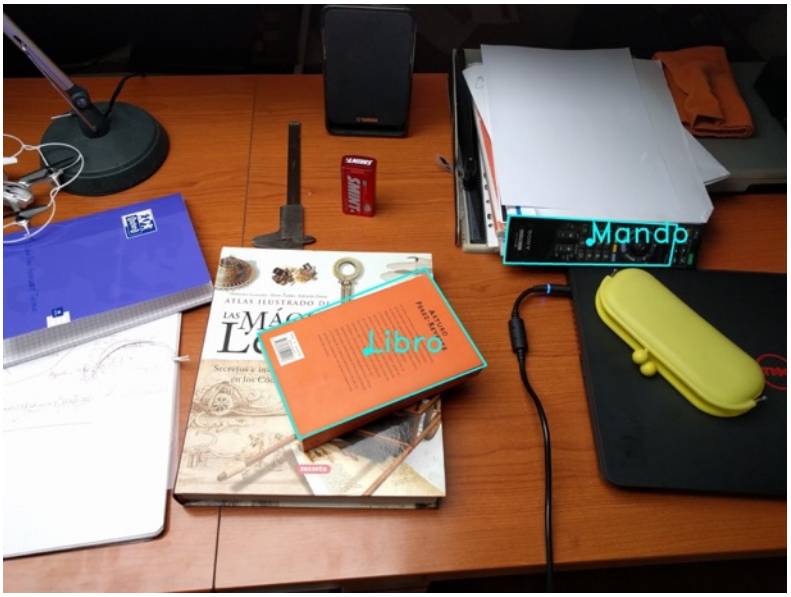

Figura 21. Reconocimiento en escena variada. 2 objetos.

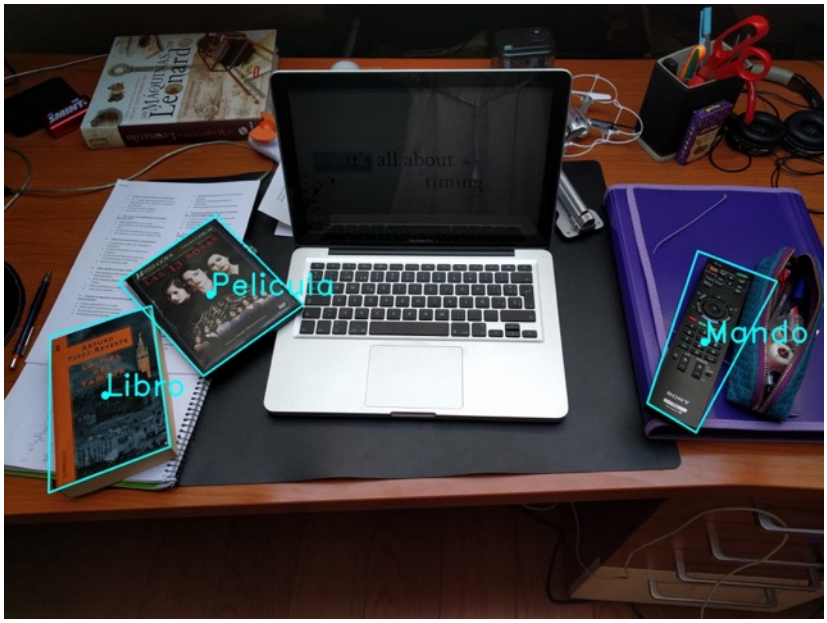

Figura 22. Reconocimiento en escena variada. 3 objetos.

El método aplicado permite reconocer un determinado objeto en una escena, por lo tanto, permite distinguirlo de objetos similares. Por ejemplo, en la Figura 23, se expone el reconocimiento del mando de la base de datos junto a otros mandos similares.

Figura 23. Reconocimiento del mando de la base de datos entre otros mandos diferentes.

Sin embargo, también tiene ciertas limitaciones. Al ser un algoritmo basado en puntos característicos, si el objeto no presenta cambios en la superficie podría no ser detectado, como puede ocurrir con la parte trasera del mando. Esto no significa que no detecte el objeto en ninguna ocasión si se encuentra colocado de esta manera, sino que, en ciertas situaciones, dependiendo de diversos factores como la iluminación, la orientación o la distancia a la cámara podría no ser detectado. En escenas controladas, sin brillos y un buen enfoque sí se podría detectar. Esto se ilustra en la Figura 24.

Figura 24. Detección del mando cuando se encuentra boca abajo.
En la escena de la izquierda falla.

Además, depende directamente de las imágenes que conforman la base de datos, por lo tanto, a mayor número de imágenes con diferentes vistas del objeto, mayor será el número de detecciones, pues agrupa más posibilidades de orientaciones del objeto. Sin embargo, al aumentar el número de imágenes también aumenta el número de puntos característicos a tratar, por consiguiente, el programa se vuelve más pesado y lento. Lo mismo ocurriría si las imágenes de la base de datos fuesen tomadas con una resolución mayor.

El programa funciona mejor cuando los objetos de la escena están en un primer plano y cuando la iluminación es adecuada, sin aparecer demasiados brillos o sombras.

Para intentar cuantificar el nivel de fiabilidad en la detección, se ha llevado a cabo un experimento sobre 20 imágenes de escenas similares. En todas las escenas aparecen los cinco objetos que conforman la base de datos. Se trata de 10 escenas controladas con el fondo homogéneo y otras 10 escenas sobre distintos fondos heterogéneos, similares a las presentadas en la Figura 25.

Figura 25. Escenas sobre las que se desarrolló el experimento. Izquierda: Fondo homogéneo. Derecha: Fondo heterogéneo

TABLA 4. RECOPILACIÓN DE RESULTADOS DE RECONOCIMIENTO SOBRE DIFERENTES ESCENAS. ERRORES EN LA DETECCIÓN ENTRE PARÉNTESIS.

Objeto	Fondo homogéneo	Fondo heterogéneo
Objeto 0: Película	100%	100%
Objeto 1: Mando a distancia	70%	70%
Objeto 2: Libro	100%	100%
Objeto 3: Monedero	100%	100%
Objeto 4: Teléfono	40%	30% (1)

Tal y como se recoge en la Tabla 4, la tasa de reconocimiento para los objetos 0, 2 y 3 ha sido del 100% en los dos tipos de escenas, mientras que los objetos 1 y 4 no alcanzan tales porcentajes. Además, el teléfono ha tenido un error en la detección en una de las imágenes, siendo detectado fuera de lugar.

A la vista de los resultados se podría afirmar que la escena no influye demasiado, ya que los resultados son similares. En mayor medida que el fondo influye la lejanía a la cámara, puesto que el método depende principalmente de los puntos característicos. Por lo tanto, si se encuentra cerca, los puntos del objeto serán detectados con mayor facilidad, sin importar los encontrados en el fondo. Por el contrario, si el objeto se encuentra lejos, no detectará correctamente los puntos, independientemente de la escena. Los *keypoints* detectados en el fondo podrían promover la aparición de falsos positivos, pero no sucede a menudo ya que deberían tener patrones similares a los de los objetos de la base de datos.

Los malos resultados proporcionados en la detección del teléfono se deben a que es un objeto con pocos cambios en su superficie, siendo el número de puntos característicos detectados bastante bajo. En la Tabla 5, se recoge el número de *keypoints* detectados en las imágenes de la base de datos en las que el objeto se encuentra de frente, Figura 26. Correspondiendo el mando y el teléfono con el menor número de puntos detectados y más baja tasa de reconocimiento, en especial este último.

Figura 26. Objetos de la base de datos. De izquierda a derecha y de arriba a abajo: Objeto 0: Película, Objeto 1: Mando a distancia, Objeto 2: Libro, Objeto 3: Monedero, Objeto 4: Teléfono

TABLA 5. PUNTOS CARACTERÍSTICOS DETECTADOS EN UNA IMAGEN EN LA QUE EL OBJETO ES VISTO DE FRENTE.

Objetos	***Keypoints***
Película	2175
Mando	1529
Libro	2570
Monedero	2706
Teléfono	594

4. Discusión

En los experimentos llevados a cabo, cada dimensión del acumulador de Hough se compone de 10 celdas, resultando un espacio de 10^4 celdas que describen las diferentes localizaciones hipotéticas del objeto.

Un mayor número de celdas por dimensión permitiría una cuantificación más fina de las posiciones características. Sin embargo, esto podría desembocar en una dispersión de los máximos entre las celdas vecinas, impidiendo un correcto reconocimiento de objetos con suficientes correspondencias. Por otra parte, un menor número de celdas podría conducir a las correspondencias de las características que votan para las posiciones incorrectas del objeto.

En el apartado de experimentación se ha demostrado que el método aplicado funciona más que aceptablemente en escenas controladas, si bien presenta algunas limitaciones. Si el objeto presenta una superficie homogénea no se detectarán muchos puntos característicos, lo que podría impedir un correcto reconocimiento. El algoritmo funciona mejor cuando se trata de un objeto con cambios en su superficie, ya sean dibujos, letras o texturas.

Otra de las limitaciones del método es que depende de las imágenes que completan la base de datos. Si el número de vistas es limitado puede ocurrir que no se cubran todas las posibles orientaciones del objeto a reconocer. Además, si las imágenes son tomadas con una resolución demasiado pequeña se podrían perder detalles de gran ayuda en el reconocimiento.

El programa presenta un funcionamiento más correcto cuando los objetos de la escena están en un primer plano y cuando la iluminación es adecuada, sin aparecer demasiados brillos o sombras.

En el algoritmo desarrollado, al finalizar el *clustering* se aplica la homografía directamente sobre la celda más votada del acumulador de Hough, calculando una transformación en perspectiva entre las características de la celda y los puntos correspondientes en la base de datos bajo la suposición de que todas las características se encuentran en un plano. Una posible mejora sería, en vez de trabajar con la casilla más votada, considerar más celdas como máximos, aplicar la homografía a todas ellas y considerar

la homografía con la mayoría de las correspondencias de puntos el objeto correcto. Sin embargo, el coste computacional al realizar una homografía es elevado, lo que ralentizaría notablemente la ejecución del programa.

En una escena se podrán reconocer tantos objetos como haya en la base de datos. En caso de haber dos objetos idénticos, se reconocerá el de mayor número de correspondencias.

Se podría mejorar el programa añadiendo nuevos objetos al reconocimiento. El código está diseñado de tal manera que no reviste dificultad la incorporación de nuevos objetos al sistema.

5. Conclusiones

El reconocimiento de objetos mediante visión artificial no es una tarea fácil, ya que en la actualidad no existe un sistema capaz de reconocer cualquier tipo de objeto en condiciones variadas de iluminación con una fiabilidad del 100 %.

Por lo tanto, se puede concluir que los objetivos marcados al principio del proyecto se han cumplido con éxito, consiguiendo reconocer una serie de objetos predeterminados, almacenados en la base de datos, dentro de una escena.

A modo de resumen, se recupera la figura que muestra las etapas que sigue el método aplicado.

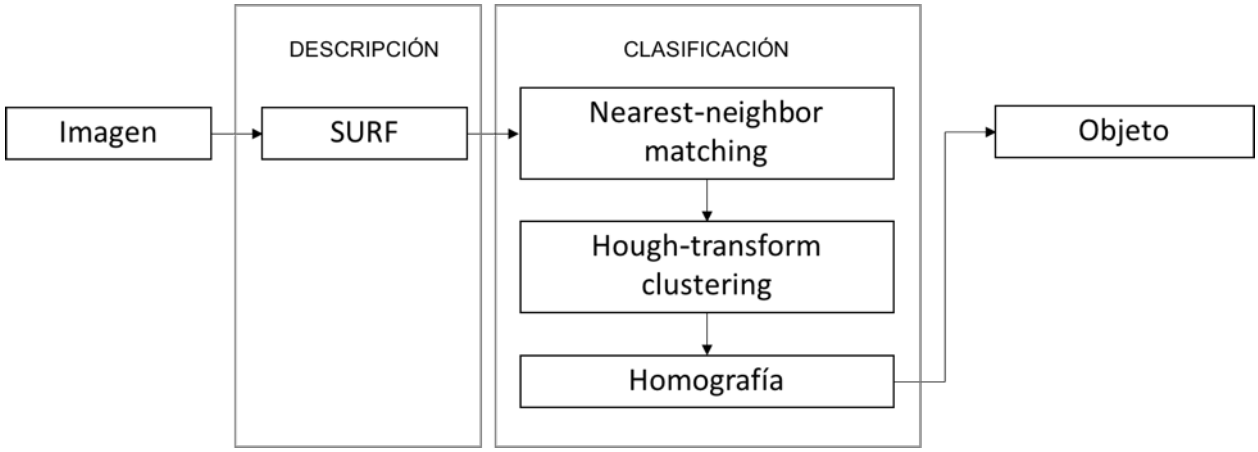

Figura 27. Etapas del método aplicado para el reconocimiento de objetos

Por último, cabe destacar el aprendizaje adquirido durante el desarrollo de este proyecto y los retos a los que se ha hecho frente. El uso del sistema operativo Linux, nunca antes manejado. Puesta en marcha del algoritmo en el lenguaje de programación C++ y el manejo de la librería de visión artificial OpenCV, tampoco utilizados hasta el momento. La investigación y estudio, de manera autónoma, de un estado del arte, ahondando en una de las múltiples vías de solucionar el problema, y de nuevos conceptos, como el desarrollo de un método de *clustering* basado en la transformada de Hough, tanto a nivel teórico como de código.

6. Referencias

[1] "OMS | Ceguera y discapacidad visual," *WHO*. [Online]. Available: http://www.who.int/mediacentre/factsheets/fs282/es/. [Accessed: 07-Dec-2016].

[2] "OrCam – See For Yourself." [Online]. Available: http://www.orcam.com/. [Accessed: 30-Apr-2017].

[3] "Toyota USA | Blind & Visually Impaired Aids | Project BLAID." [Online]. Available: https://www.toyota.com/usa/story/effect/projectblaid.html. [Accessed: 30-Apr-2017].

[4] "Aipoly Vision: Sight for Blind & Visually Impaired on the App Store." [Online]. Available: https://itunes.apple.com/us/app/aipoly-vision/id1069166437?ls=1&mt=8. [Accessed: 30-Apr-2017].

[5] M. Sonka, V. Hlavac, and R. Boyle, *Image Proccessing, Analysis, and Machine Vision*. CENGAGE Learning.

[6] L. G. Roberts, "Machine perception of three-dimensional solids," Thesis, Massachusetts Institute of Technology, 1963.

[7] M. Donoser and H. Bischof, "Efficient maximally stable extremal region (MSER) tracking," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, 2006, vol. 1, pp. 553–560.

[8] R. Diaz De Leon and L. E. Sucar, "Human silhouette recognition with Fourier descriptors," 2000, vol. 3, pp. 709–712.

[9] S. Wang, L. Duan, C. Zhang, L. Chen, G. Cheng, and J. Yu, "Fast pedestrian detection based on object proposals and HOG," in *2016 International Joint Conference on Neural Networks (IJCNN)*, 2016, pp. 3972–3977.

[10] Y. X. Zhang, Y. Q. Zhao, Y. Liu, L. Q. Jiang, and Z. W. Chen, "Identification of wood defects based on LBP features," in *2016 35th Chinese Control Conference (CCC)*, 2016, pp. 4202–4205.

[11] D. G. Lowe, "Distinctive image features from scale-invariant *keypoints*," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[12] V. Gurunathan, T. Sathiyapriya, and R. Sudhakar, "Multimodal biometric recognition

system using SURF algorithm," in *2016 10th International Conference on Intelligent Systems and Control (ISCO)*, 2016, pp. 1–5.

[13] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *2011 International conference on computer vision*, 2011, pp. 2564–2571.

[14] Z. Yin, J. Liu, M. Krueger, and H. Gao, "Introduction of SVM algorithms and recent applications about fault diagnosis and other aspects," in *2015 IEEE 13th International Conference on Industrial Informatics (INDIN)*, 2015, pp. 550–555.

[15] Y. Freund, R. Schapire, and N. Abe, "A short introduction to boosting," *J.-Jpn. Soc. Artif. Intell.*, vol. 14, no. 771–780, p. 1612, 1999.

[16] G. Xiufeng and X. Changzheng, "K-means Multiple *Clustering* Research Based on Pseudo Parallel Genetic Algorithm," in *2010 International Forum on Information Technology and Applications (IFITA)*, 2010, vol. 1, pp. 30–33.

[17] J. P. N. Cruz, M. L. Dimaala, L. G. L. Francisco, E. J. S. Franco, A. A. Bandala, and E. P. Dadios, "Object recognition and detection by shape and color pattern recognition utilizing Artificial Neural Networks," in *2013 International Conference of Information and Communication Technology (ICoICT)*, 2013, pp. 140–144.

[18] J. Qi, Y. Yu, L. Wang, and J. Liu, "K*-Means: An Effective and Efficient K-Means *Clustering* Algorithm," 2016, pp. 242–249.

[19] C. F. Olson, "Efficient pose *clustering* using a randomized algorithm," *Int. J. Comput. Vis.*, vol. 23, no. 2, pp. 131–147, 1997.

[20] Z. Zhang, Y. Wang, Q. Liu, L. Li, and P. Wang, "A CNN based functional zone classification method for aerial images," in *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 2016, pp. 5449–5452.

[21] V. Seib, M. Kusenbach, S. Thierfelder, and D. Paulus, "Object recognition using hough-transform *clustering* of surf features," *RoboCup Home Tech. Chall.*, 2012.

[22] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *European conference on computer vision*, 2006, pp. 404–417.

[23] M. Muja and D. G. Lowe, "Fast Approximate Nearest Neighbors with Automatic

Algorithm Configuration.," *VISAPP 1*, vol. 2, no. 331–340, p. 2, 2009.

[24] D. G. Lowe, "Object recognition from local scale-invariant features," in *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, 1999, vol. 2, pp. 1150–1157.

[25] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, 2001, vol. 1, p. I–511.

[26] "Haar wavelet," *Wikipedia*. 03-Jun-2016.

[27] M. Muja and D. G. Lowe, "Scalable Nearest Neighbor Algorithms for High Dimensional Data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 11, pp. 2227–2240, Nov. 2014.

[28] HOUGH, P. V. C., "Method and means for recognizing complex patterns.," Dec. 1962.

[29] R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," *Commun. ACM*, vol. 15, no. 1, pp. 11–15, 1972.

[30] D. H. Ballard, "Generalizing the Hough transform to detect arbitrary shapes," *Pattern Recognit.*, vol. 13, no. 2, pp. 183–194, 1991.

[31] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[32] "OpenCV | OpenCV." [Online]. Available: http://opencv.org/. [Accessed: 09-Dec-2016].

[33] M. Wafy and A. M. M. Madbouly, "Increase Efficiency of SURF using RGB Color Space," *transformation*, vol. 6, no. 8, 2015.

[34] "Feature Matching — OpenCV 3.0.0-dev documentation." [Online]. Available: http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_feature2d/py_matcher/py_matcher.html. [Accessed: 11-Dec-2016].

CAPÍTULO 2.

PRESUPUESTO

ÍNDICE DEL PRESUPUESTO

1. Costes de ejecución material

El coste de ejecución material incluye tres categorías, coste de equipos, coste de software y coste de mano de obra por el tiempo empleado en el proyecto.

1.1 Costes de equipos

Se considera que el tiempo de duración de los equipos es de 5 años y la duración del proyecto es de, aproximadamente, 3 meses.

CONCEPTO	PRECIO UNITARIO	CANTIDAD	SUBTOTAL
Ordenador portátil MacBook Pro	1300 €	5 %	65 €
Cámara digital CANON	500 €	5 %	25 €

Subtotal: ……. 90 €

1.2 Costes de software

CONCEPTO	PRECIO UNITARIO	CANTIDAD	SUBTOTAL
Sistema operativo Linux: Ubuntu 14.04	0 €	1	0 €
Librería OpenCV 3.1	0 €	1	0 €
Microsoft Office 365	4,20 €/mes	3	12,60 €

Subtotal: ……. 12,60 €

1.3 Costes de mano de obra

El coste unitario de la mano de obra será de 35 €/h. La dedicación será de 6 horas al día, 5 días a la semana.

CONCEPTO	CANTIDAD	SUBTOTAL
Estudio del problema	15 días	3150 €
Estudio OpenCV	5 días	1050 €
Creación base de datos	5 días	1050 €
Desarrollo algoritmo de entrenamiento	5 días	1050 €
Desarrollo algoritmo de reconocimiento	15 días	3150 €
Pruebas de funcionamiento	5 días	1050 €
Realización de la documentación	5 días	1050 €

Subtotal: ……. 55 días ……. 11550€

La duración del proyecto es de 55 días laborables, aproximadamente 3 meses.

1.4 Coste total del presupuesto de ejecución material

CONCEPTO	SUBTOTAL
Coste de equipos	90 €
Coste de software	12,60 €
Coste de mano de obra	11550 €

Subtotal: ……. 11652,60 €

2. Gastos generales y beneficio industrial

Los gastos generales y beneficio industrial son los gastos obligados que se derivan de la utilización de las instalaciones de trabajo más el beneficio industrial. Se estima un porcentaje del 16 % sobre el coste de ejecución material

CONCEPTO	SUBTOTAL
Gastos generales y beneficio industrial	1864,42 €

3. Importe total

CONCEPTO	SUBTOTAL
Coste total del presupuesto de ejecución material	11652,60 €
Gastos generales y beneficio industrial	1864,42 €

TOTAL: ……. 13517,02 €

IVA 21%: ……. 2838,58 €

TOTAL, IVA INCLUÍDO: ……. 16355,60 €

El Importe Total del proyecto suma la cantidad de:

Dieciséis Mil Trescientos Cincuenta y Cinco, con Sesenta Céntimos

CAPÍTULO 3.

MATERIALES ADJUNTADOS

1. Código

Se adjuntan los diferentes algoritmos desarrollados, incluyendo los diferentes archivos .cpp y .h. Los archivos principales de cada código se encuentran bajo el nombre de *main.cpp*. Además, se incluyen los diferentes directorios *build* con los archivos de compilación.

1.1 Segmentación

Código de segmentación de las imágenes de la base de datos.

1.2 Fase de entrenamiento

Código que extrae los puntos característicos y descriptores de las imágenes de la base de datos y los almacena en archivos tipo YALM. Se adjunta bajo el nombre de *TrainData*.

1.3 Reconocimiento

Código que se encarga de la fase de reconocimiento.

2. Base de datos

En el directorio *data* se adjunta la base de datos que contiene las imágenes de los 5 objetos que la forman. Además, incluye los archivos tipo YALM que almacenan los puntos característicos y descriptores de cada imagen.

Para que los códigos descritos anteriormente se ejecuten correctamente, la base de datos debe estar en el mismo directorio que las carpetas que contienen el ejecutable de los mismos.

3. Escenas y resultados

Se adjuntan diferentes escenas sobre las que se han realizado pruebas con el código del reconocimiento. Además, se incluyen los resultados obtenidos.

4. Vídeo

Se realizó un video, TGF.mov, que ilustra las diferentes etapas del proceso del método aplicado.