UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

Reconocimiento de escenas basado en extracción de características usando imágenes de profundidad

TESIS

Que para obtener el título de

Ingeniero en Computación

P R E S E N T A

Mario Alberto Suárez Espinoza

DIRECTOR DE TESIS

Dr. Víctor Manuel Lomas Barrié

Ciudad Universitaria, Cd. Mx., 2023

Dedicado a ustedes, mamá y papá, mi apego emocional, mi refugio y lugar seguro, polvo de estrellas que me brinda cuidado, atención, apoyo y amor incondicional

Reconocimientos

Agradezco al Dr. Víctor Manuel Lomas Barrié, por involucrarme en el proyecto de investigación AYUDAME 2.0 y proporcionar guía en el desarrollo de este trabajo.

Agradezco a Luis Gerardo Hernández Chávez, quien, junto con el Dr. Víctor Lomas, ideamos y desarrollamos el método de extracción de características BOF presentado en este trabajo (sección 3.2).

Agradezco a mi familia, por la solidaridad que me brinda a mí y al resto de sus miembros.

Agradezco a mis amigos por escuchar, alentar, compartir y acompañar.

Investigación realizada gracias al Programa de Apoyo a Proyectos de Investigación e Innovación Tecnológica (PAPIIT) de la UNAM TA100721. Agradezco a la DGAPA-UNAM la beca recibida.

Resumen

El reconocimiento de escenas es un caso especial de la clasificación de imágenes en la que se pretende etiquetar a las imágenes a partir de la información semántica del lugar que representan. En el contexto de la navegación autónoma de robots, el reconocimiento de escenas provee al robot la capacidad de localizarse y de entender el contexto del lugar que lo rodea. Bolsa de palabras visuales con extracción de características locales es uno de los métodos tradicionales usado para la clasificación de imágenes. La etapa de extracción de características es de las que más tiempo de cómputo toma, por lo que su optimización es una tarea aún vigente.

BOF (siglas del inglés, *Boundary Object Function*) es un método de extracción de características que ha sido utilizado en el reconocimiento de piezas en tareas de ensamblaje robótico, y en este trabajo se extiende su uso para reconocimiento de escenas, esto gracias a que este descriptor es una alternativa de bajo coste computacional comparado con descriptores usados en el estado del arte. Las imágenes RGB-D consisten en dos canales, uno con información de color y otro con información de profundidad. En este trabajo se adopta un método que consiste en extracción de características BOF utilizando imágenes de profundidad, pues estas aportan información del modelo 3D de la escena capturada, lo que permite que los objetos sean segmentados con rapidez.

En este trabajo se describen los fundamentos teóricos relacionados al reconocimiento de escenas, se explica el método utilizado y se compara con uno basado en extracción de características locales SIFT (del inglés, *Scale Invariant Feature Transform*). Se presentan resultados de métricas de clasificación y de rendimiento en el tiempo, los cuales indican que BOF es una alternativa de bajo consumo de recursos computacionales, pero que sacrifica exactitud de clasificación.

Índice general

Índice de figuras

Índice de tablas

Glosario

BOF-BoVW Esquema para la clasificación de imágenes propuesto en esta tesis, que utiliza BOF y BoVW. 29, 30, 32, 34, 36, 38, 40–42, 44, 46, 47, 49–53, 55–57

SIFT-BoVW Esquema para la clasificación de imágenes que utiliza SIFT y BoVW. 29, 30, 34, 36, 38, 43–47, 49–53, 55–57

Siglas

BOF Boundary Object Function. 1, 2, 17–19, 29, 30, 32–35, 38, 39, 41, 51, 52, 55, 57, 58

BoVW Bag of Visual Words. 1, 19, 29, 32, 35, 36, 55

BoW Bag of Words. 19

BRIEF Binary Robust Independent Elementary Features. 13

DOG Difference Of Gaussian. 14, 15

ORB Oriented FAST and Rotated BRIEF. 13, 29, 57

RGB Red Green Blue. 6, 7, 35

RGB-D Red Green Blue Depth. 2, 7, 30, 33, 39, 55, 57

SIFT Scale Invariant Feature Transform. 3, 13, 17, 29, 30, 35, 38, 42, 51, 52, 55, 57

SLAM Simultaneous Localization and Mapping. 58

SURF Speeded Up Robust Features. 13, 29, 57

SVM Support Vector Machine. 24–29, 35, 36, 38, 39, 42, 47, 50, 52, 55

Capítulo 1

Introducción

1.1. Planteamiento del problema

Al inicio de la pandemia por COVID-19 la saturación de hospitales fue uno de los primeros problemas a los que los gobiernos de todo el mundo tuvieron que enfrentarse. Para solventar esta dificultad, es de gran utilidad la creación de robots asistentes de enfermería, los cuales pueden realizar tareas como el traslado de medicamento e instrumental y la comunicación remota con otras personas. Es así como surge el proyecto AYUDAME[1] 1.0 en el IIMAS–UNAM[2], el cual consiste en una plataforma móvil para auxiliar al personal sanitario realizando las tareas ya mencionadas. Esta plataforma es maniobrable, robusta y segura, pero no cuenta con un mecanismo de navegación autónoma, lo cual es ideal para que los operadores del robot intervengan lo menos posible.

Para proporcionar autonomía, el *Reconocimiento de Escenas* dota al sistema inteligente con la capacidad de localizarse y de entender el contexto del lugar que lo rodea. Al reconocer el lugar en el que se encuentra, el sistema inteligente puede adaptar sus acciones para conseguir sus objetivos, entre los que se incluyen ajustar sus estrategias de navegación para adecuarse al entorno y evitar obstáculos, así como ejecutar operaciones específicas por tipo de escena.

Dentro de los métodos disponibles para el reconocimiento de escenas se encuentra el método de bolsa de palabras visuales BoVW (del inglés, *Bag of Visual Words*) con extracción de características locales. Este método ha dado buenos resultados en métricas de clasificación, sin embargo, la extracción de características es una etapa con un alto coste computacional, especialmente cuando se trabaja con sistemas embebidos, los cuales frecuentemente presentan recursos computacionales limitados.

La función de frontera BOF (del inglés, *Boundary Object Function*) es un descriptor liviano que ha sido utilizado previamente para obtener la ubicación de un observador

[1] Siglas de Apoyo y Ubicación de Alimentos, Medicinas y Enseres.

[2] Siglas del Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas de la Universidad Nacional Autónoma de México.

con respecto a un punto de referencia [1], por lo que se cree, se podría extender su uso al reconocimiento de escenas, lo que resultaría en una alternativa más ligera que utilizar extracción de características locales.

Los sensores RGB-D (del inglés, *red, green, blue and depth*) son sensores de profundidad que se combinan con cámaras a color convencionales. La información de profundidad aporta una descripción del modelo 3D de la escena, lo cual permite realizar una segmentación basada en la profundidad de los objetos en la misma.

Es por ello por lo que se plantea el uso de descriptores BOF para representar la información visual de un robot y reconocer escenas en interiores como una alternativa con bajo consumo de recursos computacionales frente al uso de extracción de características locales.

1.2. Tema

Reconocimiento de escenas utilizando un descriptor liviano de imágenes de profundidad como alternativa para sistemas con recursos computacionales limitados.

1.3. Objetivos

1.3.1. General

Implementar un método de reconocimiento de escenas utilizando extracción de características BOF de imágenes de profundidad, bolsa de palabras visuales, y clasificación estadística.

1.3.2. Específicos

1. Encontrar dentro del estado del arte un conjunto de datos RGB-D adecuado que permita la implementación y evaluación del método de reconocimiento de escenas.

2. Implementar un método para agrupar una nube de puntos en planos ortogonales a un eje de coordenadas, extracción de los contornos significativos de estos planos y obtención del descriptor BOF de cada contorno. La nube de puntos es generada a partir de una imagen de profundidad.

3. Generar un conjunto de datos extrayendo los descriptores BOF de los fotogramas de profundidad de una escena.

4. Construir un vocabulario de palabras visuales utilizando los descriptores BOF de acuerdo con el esquema de bolsa de palabras visuales y generar una representación de cada fotograma en la escena.

5. Utilizar el conjunto de datos de la representación de los fotogramas para entrenar un clasificador estadístico y que este identifique la escena a la que pertenecen los fotogramas.

6. Comparar el método de reconocimiento de escenas planteado con uno basado en extracción de características locales SIFT (del inglés, *Scale Invariant Feature Transform*). La comparación incluye métricas de clasificación y de rendimiento en el tiempo y almacenamiento.

1.4. Hipótesis

El uso de un descriptor liviano en un esquema para el reconocimiento de escenas resulta en una alternativa de bajo coste computacional a comparación de un esquema que utiliza extracción de características locales.

1.5. Estructura de la tesis

El resto del documento está organizado de la siguiente manera. El capítulo 2 introduce los fundamentos teóricos necesarios para la aplicación de un esquema tradicional para el reconocimiento de escenas. El capítulo 3 presenta los detalles de implementación de los métodos de reconocimiento de escenas propuestos. El capítulo 4 muestra los resultados obtenidos y realiza una discusión de los mismos. Por último, el capítulo 5 presenta las conclusiones, así como propuestas para continuar el trabajo a futuro.

Capítulo 2

Marco teórico

2.1. Fundamentos de imágenes digitales

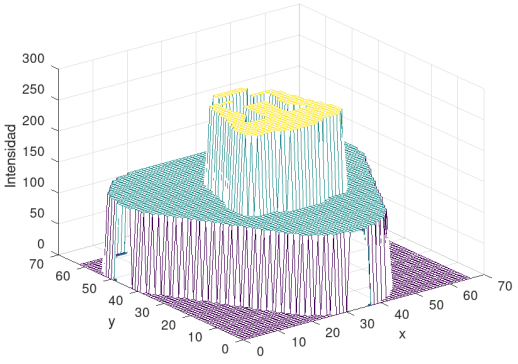
2.1.1. Representación de imágenes digitales

Una imagen es la representación visual de un objeto. La forma más común de representar imágenes es la de mapa de bits, la cual consiste en un arreglo bidimensional $f(x, y)$ o matriz, que contiene M filas y N columnas. Ver capítulo 2 de [2].

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{bmatrix} \tag{2.1}$$

Cada elemento de la matriz es conocido como pixel (del inglés, *picture element*).

La función f depende de dos variables, las cuales están relacionadas con la posición del pixel en el espacio bidimensional. Esta función puede ser visualizada como una superficie 3D (figura 2.1a), pero su forma de visualización más común es la de la figura 2.1b, en donde la intensidad de cada punto es proporcional al valor de f en dicho punto.



(a) Imagen graficada como una superficie 3D

(b) Imagen desplegada en escala de grises. Los tonos negro, gris y blanco representan los valores de 0, 127 y 255 respectivamente.

Figura 2.1: Representación gráfica de una imagen

La imagen mostrada en la figura 2.1 tiene un tamaño de 67×63 pixeles, lo que quiere decir que contiene 67 filas y 63 columnas, además, es de 8 bits, lo que significa que cada pixel almacena un valor que utiliza dicha cantidad de memoria. A las imágenes que solo aportan información de intensidad y no de color, se les conoce como *imágenes a escala de grises*. Con 8 bits, pueden representarse valores enteros desde 0 a 255, en donde el valor de 0 representa un tono negro, el valor de 255 un tono blanco y valores intermedios tonos grises que varían su luminosidad, siendo grises oscuros para valores cercanos a 0 y grises claros para valores cercanos a 255.

2.1.2. Imágenes a color

El uso del color en el procesamiento digital de imágenes aporta información que facilita tareas como la detección y extracción de objetos de una escena, además, ayuda en el análisis manual de imágenes, ya que para los humanos es más cómodo discernir entre diferentes matices de color a comparación de diferentes tonos sobre una escala de grises. Ver capítulo 6 de [2].

Para facilitar la especificación de colores de forma estandarizada se han creado diferentes modelos de color. Según Gonzalez y Woods [2], un modelo de color es la especificación de un sistema de coordenadas y un subespacio dentro del cual cada color es representado por un punto. El modelo más utilizado en imágenes digitales es el RGB (sigla del inglés, *red, green, blue*), el cual se describe a continuación.

2.1.2.1. Modelo de color RGB

En este modelo cada color es una combinación de rojo, verde y azul. Está basado en el sistema de coordenadas cartesiano de tres dimensiones, en donde cada eje corresponde a un canal de color, tal como se muestra en la figura 2.2.

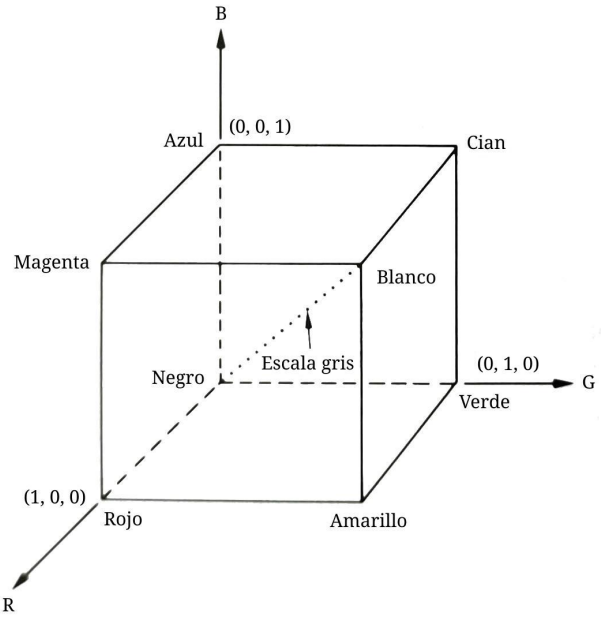


Figura 2.2: Representación del modelo RGB en un sistema de coordenadas cartesiano. Figura obtenida del capítulo 6 de [2].

Las imágenes representadas en un modelo de color RGB consisten en tres imágenes componentes, una por cada canal de color. Considerando una imagen RGB en la cual cada canal rojo, verde y azul son imágenes de 8 bits resulta en una profundidad total de 24 bits. El número total de colores que pueden representarse en una imagen RGB de 24 bits es de $(2^8)^3 = 16\,777\,216$.

2.1.3. Imágenes a color y profundidad

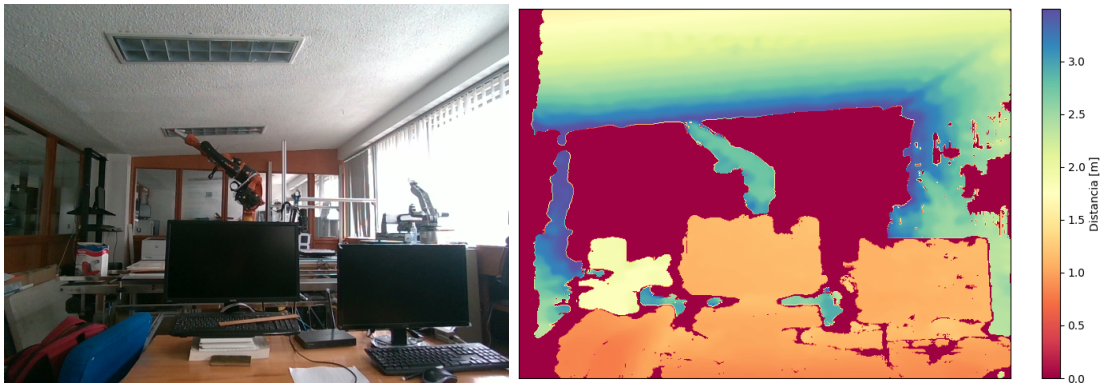
Las imágenes a color y profundidad son imágenes con información de color a las que se les agrega un canal de profundidad, el cual consiste en una imagen en la que cada pixel almacena la distancia que existe entre el sensor que capta la imagen y el objeto que ocupa dicha posición dentro de la imagen. También se les conoce como imágenes RGB-D (sigla del inglés, *red, green, blue and depth*).

De acuerdo con Kadambi et al. [3], existen dos principales tecnologías para la adquisición de una imagen de profundidad: luz estructurada (en inglés, *structured light*) y tiempo de vuelo (en inglés, *time of flight*). Una cámara de luz estructurada proyecta un patrón activo de luz y obtiene la profundidad analizando la deformación del patrón. Una cámara de tiempo de vuelo mide el tiempo que la luz ha estado en vuelo para determinar la distancia. Independiente del método utilizado para obtener la distancia, es común que la imagen de profundidad contenga huecos, es decir, pixeles con datos

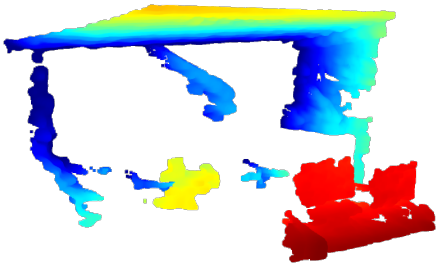
perdidos.

En la figura 2.3b se muestra el canal de profundidad con falso color para una mejor visualización. El color rojo más oscuro corresponde a un valor de cero, los cuales son huecos. Colores cercanos al rojo indican que el objeto captado por la imagen está cerca del sensor, mientras que los colores cercanos al azul indican objetos alejados del sensor.

A partir de la imagen de profundidad es posible obtener la representación de los puntos en el espacio tridimensional de los objetos captados en la propia imagen. A esta representación se le conoce como nube de puntos, la cual se presenta en la figura 2.3c.



(a) Imagen RGB **(b)** Imagen o canal de profundidad



(c) Nube de puntos

Figura 2.3: Imagen RGB-D y su representación en nube de puntos

De acuerdo con Sturm et al. [4], dado el valor de profundidad p en la coordenada (u, v) de la imagen, su punto $(x, y, z) \in \mathbb{R}^3$ correspondiente es:

$$z = \frac{p}{p_s} \tag{2.2a}$$

$$x = \frac{(u - c_x)z}{f_x} \tag{2.2b}$$

$$y = \frac{(v - c_y)z}{f_y} \tag{2.2c}$$

donde p — profundidad en la coordenada (u, v) de la imagen
p_s — escala de profundidad
(f_x, f_y) — longitud focal
(c_x, c_y) — centro óptico

La escala de profundidad en (2.2a) se refiere al factor sobre el cual se debe dividir el valor del pixel para obtener un metro de distancia. Por ejemplo, con una escala de profundidad de 5 000, un valor de pixel de 10 000 corresponde a 2 metros de distancia.

La longitud focal y el centro óptico en la ecuación (2.2) son parámetros intrínsecos de la cámara y dependen del sensor utilizado.

2.2. Introducción al reconocimiento de escenas

2.2.1. Clasificación de imágenes

Pérez-Careta et al. [5] señalan que la clasificación de imágenes consiste en asignar una etiqueta a una imagen basándose en las propiedades de su contenido. Para ello se necesita de un modelo que pueda realizar la asignación de la etiqueta de forma automática a partir de aprender el criterio de asignación de acuerdo con un conjunto de imágenes previamente etiquetadas. El criterio de asignación de etiquetas es subjetivo y depende del problema que se quiera resolver, por ejemplo, un problema sería separar las imágenes que contengan gatos de las que contengan perros, o imágenes donde haya que clasificar el tipo de fruta (sandías, peras, manzanas y fresas).

2.2.2. Reconocimiento de escenas

El reconocimiento de escenas es un caso especial de la clasificación de imágenes en la que se pretende etiquetar a las imágenes a partir de la información semántica del lugar que representan. Algunos ejemplos son la clasificación de escenas en exteriores (costa, bosque, carretera, montaña, edificios, etc.) y la clasificación de escenas en interiores (dormitorio, comedor, cocina, sala, corredor, baño, etc.).

De acuerdo con Xie et al. [6], en el contexto de la navegación autónoma de robots, el reconocimiento de escenas provee al robot la capacidad de localizarse y de entender el contexto del lugar que lo rodea, permitiendo así generar estrategias para completar su tarea, o bien, adaptarse a los cambios que se presentan en el entorno.

Según Wang et al. [7], el reconocimiento de escenas presenta dos principales desafíos, mismos que son ejemplificados en la figura 2.4:

1. **Inconsistencia visual**: Se refiere a que existe mucha variación entre imágenes de una misma categoría de escena, lo que dificulta extraer patrones que se asocien a una categoría en específico.

2. **Ambigüedad de etiquetado**: Se presenta cuando algunas categorías de escenas comparten características visuales similares, lo que provoca que sea fácil confundir las unas con las otras.

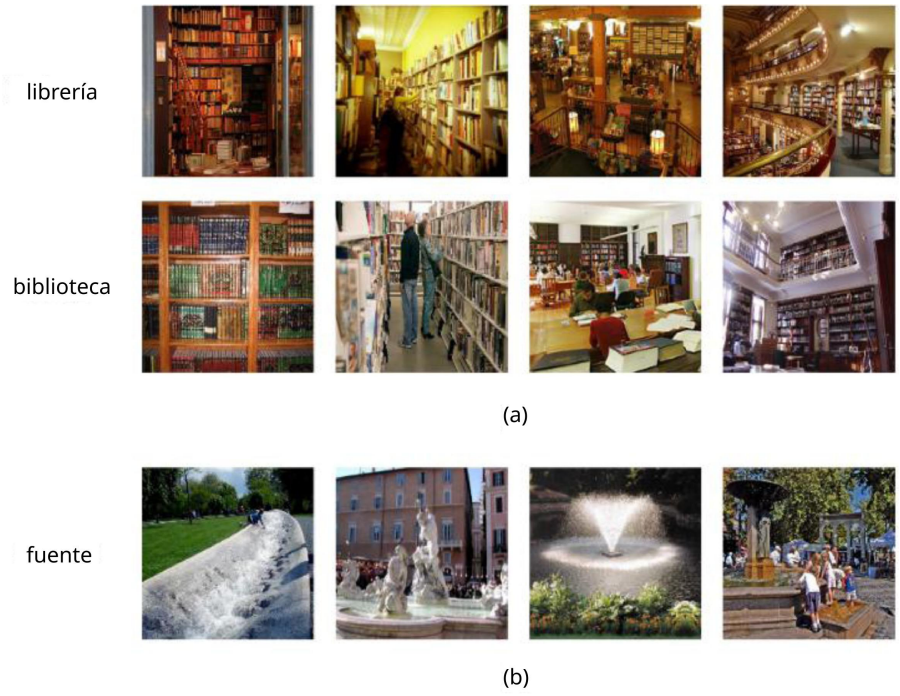


Figura 2.4: Desafíos en el reconocimiento de escenas. (a) Ejemplo de *ambigüedad de etiquetado*; imágenes de "librería" son demasiado parecidas a las imágenes de "biblioteca". (b) Ejemplo de *inconsistencia visual*; imágenes de "fuente" son muy distintas entre sí. Figura obtenida de [6].

2.2.3. Esquema general para la clasificación de imágenes

De acuerdo con Xie et al. [6], el esquema general para la clasificación de imágenes consiste en tres etapas principales: **extracción de características**, **transformación de características** y **clasificación**.

En la primera etapa se detectan puntos de interés, que son puntos dentro de la imagen que son relevantes. La relevancia de los puntos está dada a partir de aquellos pixeles que presentan alta variabilidad en su intensidad con respecto a sus vecinos. Además, se obtienen descripciones de estos puntos de interés, los cuales son vectores numéricos que representan la información visual alrededor de los puntos.

La etapa de **transformación de características** se encarga de capturar las descripciones o características visuales obtenidas en la etapa anterior para formar una representación de la imagen.

Por último, la etapa de **clasificación** utiliza la representación de la imagen para generar un modelo que permita asignarle una etiqueta, siguiendo los criterios del conjunto de imágenes de entrenamiento.

El esquema general para la clasificación de imágenes está representado en la figura 2.5, en donde los rectángulos representan las etapas, mientras que las elipses entradas y salidas de dichas etapas.

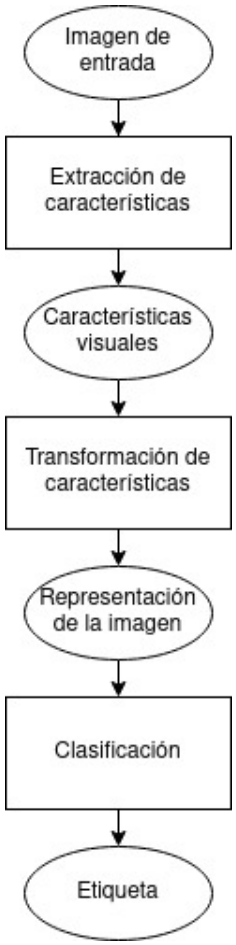


Figura 2.5: Esquema general para la clasificación de imágenes

Los métodos que han cobrado relevancia para la clasificación de imágenes durante los últimos años son las *redes neuronales convolucionales profundas*, esto gracias a los buenos resultados de clasificación que otorgan y los grandes avances en el hardware necesario para generar los modelos. Las redes neuronales convolucionales son clasificadores que trabajan de forma natural con imágenes y unifican las etapas presentadas anteriormente dentro de un mismo esquema. La desventaja de las redes neuronales convolucionales profundas es que necesitan de datos masivos para su entrenamiento, además, requieren de una alta capacidad de cómputo, lo que va en contra del objetivo de este trabajo, que es encontrar un método de reconocimiento de escenas de bajo coste computacional. Es por ello por lo que el tratado de las redes neuronales convolucionales profundas queda fuera de los alcances de este trabajo, sin embargo, puede encontrarse una introducción a estos métodos en el capítulo 9 de [8].

2.3. Extracción de características

Como ya se introdujo en la subsección *Esquema general para la clasificación de imágenes* 2.2.3, en la etapa de extracción de características se buscan puntos en la imagen que sean relevantes y que permitan identificar a una zona de la imagen, de tal forma que esta zona pueda ser rastreada o comparada con otras zonas. A estos puntos relevantes se les conoce como *puntos característicos* (en inglés, *feature points*) y por lo regular se encuentran a partir de aquellos pixeles cuya intensidad presenta una alta variabilidad con respecto a sus vecinos.

Según Gao y Zhang [9], los puntos característicos deben tener las siguientes propiedades:

1. **Repetibilidad**: El mismo punto característico debe poder ser encontrado en diferentes imágenes.
2. **Distintividad**: Diferentes puntos característicos deben tener una representación diferente.
3. **Eficiencia**: En la misma imagen, el número de puntos característicos debe ser mucho menor que el número de pixeles.
4. **Localidad**: El punto característico debe estar relacionado solo a una pequeña área de la imagen.

Los puntos característicos se construyen a partir de dos fases: la *detección* y la *descripción*. En la primera fase se detectan *puntos clave*, que se refieren a la posición 2D de los puntos característicos en el plano de la imagen, además, algunos métodos también mantienen información de orientación y tamaño. En la segunda fase, la de descripción, se genera un vector multidimensional que contiene información de los pixeles alrededor de los puntos clave; a este vector se le conoce como *descriptor*. Los descriptores están diseñados de tal forma que puedan ser comparados en el espacio multidimensional, y que

aquellos puntos característicos con apariencia similar tengan descriptores similares. La comparación de descriptores se realiza usando métricas de distancia, la más común es la distancia euclidiana, sin embargo, existen descriptores basados en cadenas binarias, donde la métrica de distancia que debe utilizarse es la distancia de Hamming.

Dentro de los algoritmos de extracción de características más comunes se encuentran Scale Invariant Feature Transform (SIFT), Speeded Up Robust Features (SURF) y Oriented FAST and Rotated BRIEF (ORB), todos ellos cuentan con etapa de detección y descripción, pero hay otros algoritmos que solo se centran una etapa, como Binary Robust Independent Elementary Features (BRIEF), que solo se enfoca en la etapa de descripción.

Para el propósito de este trabajo, se da una breve introducción a SIFT.

2.3.1. Scale Invariant Feature Transform (SIFT)

Scale Invariant Feature Transform (SIFT) es un método de extracción de características desarrollado por Lowe [10]. Los puntos característicos que proporciona son invariantes a la escala y a la rotación, y permite robustez a través un rango substancial de distorsión afín, cambios en el punto de vista tridimensional, adición de ruido y cambios en la iluminación. Es uno de los métodos más usados en visión computacional. Fue un método patentado [11], cuya expiración ocurrió en 2020, por lo que actualmente es fácil encontrar su implementación en distintas bibliotecas de programación sobre visión computacional.

Sus cuatro principales etapas son:

1. **Detección de extremos en el espacio-escala**: Busca alrededor de todas las escalas y localizaciones de la imagen para identificar posibles puntos de interés.

2. **Localización de puntos clave**: Consiste en hacer un refinamiento de la localización y escala de los puntos de interés candidatos utilizando un modelo detallado.

3. **Asignación de orientación**: Una o más orientaciones son asignadas a cada ubicación de punto clave basándose en las direcciones del gradiente en dicha localidad. Posteriormente todas las operaciones son realizadas en datos de la imagen que han sido transformados relativos a su orientación, escala y ubicación de cada punto característico, por lo tanto, proporcionando invariancia a dichas transformaciones.

4. **Descripción del punto clave**: Los gradientes locales de la imagen son calculados en la escala seleccionada alrededor de cada punto clave. Estos gradientes son transformados en una representación que tenga en cuenta niveles significativos de distorsión de forma local y cambios en iluminación.

2.3.1.1. Detección de extremos en el espacio-escala

Como primer paso, se define al espacio-escala de la imagen como la función $L(x, y, \sigma)$, que es producida por la convolución de una Gaussiana, $G(x, y, \sigma)$, con la imagen de entrada, $I(x, y)$:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \tag{2.3}$$

donde (x, y) — coordenadas de los pixeles en la imagen
σ — desviación estándar de la Gaussiana (escala)

La Gaussiana está definida por:

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.4}$$

La función diferencia de Gaussianas DOG (del inglés, *Difference Of Gaussian*) se define de la siguiente manera:

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma) \tag{2.5}$$

La imagen de entrada es incrementalmente convolucionada con Gaussianas para producir imágenes separadas por un factor k en el espacio escala, mismas que se muestran en la figura 2.6 en la columna de la izquierda. Posteriormente, imágenes adyacentes en escala son restadas para producir diferencias de Gaussianas, tal como se muestra en la figura 2.6 en la columna de la derecha. Después de un número determinado de imágenes Gaussianas procesadas, aquella que queda en lo alto de la pila es sub muestreada por un factor de 2 y el proceso es repetido.

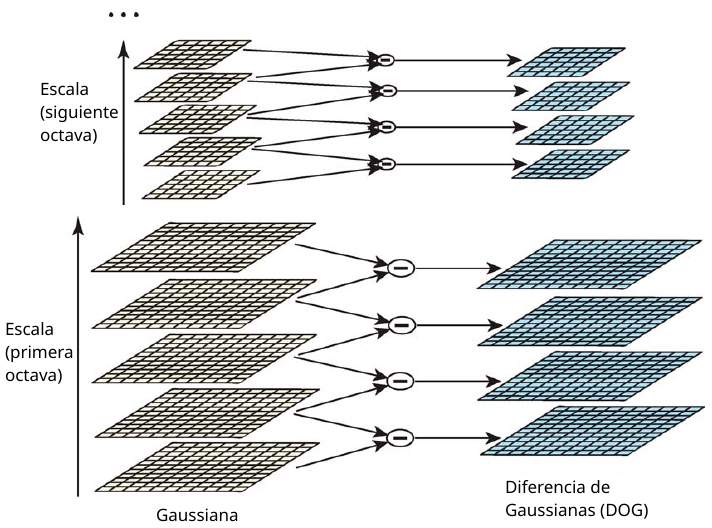


Figura 2.6: Construcción de imágenes Gaussianas a diferentes escalas. Figura obtenida de [10].

Una vez construida la pila de diferencias de Gaussianas, se obtienen los puntos clave candidatos como aquellos máximos o mínimos en la DOG alrededor de una escala. Para ello, cada punto en la DOG es comparado con sus ocho vecinos en la propia imagen y con sus nueve vecinos en la escala de arriba y abajo, tal como se muestra en la figura 2.7.

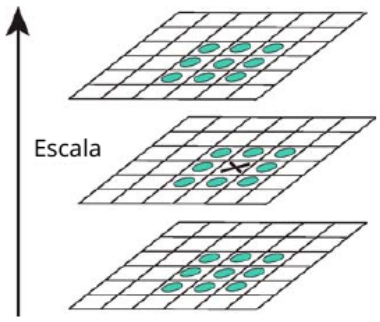


Figura 2.7: Máximos y mínimos de diferencias de Gaussianas. Figura obtenida de [10].

2.3.1.2. Localización de puntos clave

Algunos de los puntos clave obtenidos en el paso anterior son inestables, por lo que se hace un refinamiento:

1. Se hace una interpolación de datos cercanos usando la expansión de Taylor cuadrática, y con ello se refina la localización subpixel del punto clave.
2. Se descartan puntos clave de bajo contraste.
3. Se eliminan puntos clave ubicados en contornos, pues estos poseen una localización pobremente definida.

2.3.1.3. Asignación de orientación

Una vez obtenidos los puntos clave, una o más orientaciones les son asignadas.

Se selecciona la imagen Gaussiana suavizada $L(x, y)$ con la escala σ más cercana a la del punto clave, con el objetivo de que los cálculos sean realizados con invariancia a la escala.

Por cada imagen muestra $L(x, y)$, en la escala σ, la magnitud del gradiente $m(x, y)$, y su orientación $\theta(x, y)$, es calculada usando diferencias de pixeles:

$$\Delta x = L(x+1, y) - L(x-1, y) \tag{2.6a}$$
$$\Delta y = L(x, y+1) - L(x, y-1) \tag{2.6b}$$
$$m(x, y) = \sqrt{(\Delta x)^2 + (\Delta y)^2} \tag{2.6c}$$
$$\theta(x, y) = tan^{-1}\left(\frac{\Delta y}{\Delta x}\right) \tag{2.6d}$$

Posteriormente se forma un histograma de orientación a partir de las orientaciones del gradiente de los puntos muestra alrededor del punto clave. El histograma de orientación tiene 36 intervalos, cubriendo el rango de orientaciones de 360 grados. Cada muestra agregada al histograma es multiplicada por su magnitud de gradiente y por una ventana Gaussiana con σ igual a 1.5 veces la escala del punto clave.

Los picos en el histograma de orientación corresponden a direcciones dominantes de los gradientes locales. Se seleccionan las orientaciones del pico máximo y de aquellos picos con más del 80 % del pico máximo y son asignadas al punto clave. Para puntos clave con más de una orientación, se crea un punto clave adicional con la misma localización y escala, pero con orientaciones distintas, por cada orientación adicional.

2.3.1.4. Descripción del punto clave

La descripción consiste en construir un vector que represente la información de los pixeles alrededor de cada uno de los puntos clave obtenidos en los pasos anteriores. Para ello, se utiliza la imagen Gaussiana $L(x, y)$ correspondiente a la escala del punto clave, y se toma una región de interés de $m \times m$ pixeles alrededor de dicho punto. Para lograr invarianza a la orientación, las coordenadas de la región de interés son rotadas relativamente a la orientación del punto clave obtenida en pasos anteriores. Posteriormente la región de interés es dividida en $n \times n$ subregiones, y por cada subregión se genera un histograma de orientación de 8 intervalos. Para construir cada histograma, las magnitudes de los gradientes son multiplicadas por una función Gaussiana con σ igual a la mitad del ancho de la ventana de la región de interés, esto para dar mayor importancia a las muestras cercanas al centro de la región de interés. El descriptor del punto clave se conforma como un vector que concatena cada uno de los histogramas obtenidos.

Para ilustrar el proceso de la construcción del descriptor se tiene la figura 2.8, la cual a la izquierda contiene una región de interés de 8×8 ($m = 8$), misma que es dividida en 2×2 subregiones ($n = 2$); el círculo azul ilustra la función Gaussiana; y cada recuadro representa los gradientes. En la parte derecha de la figura 2.8 se ilustra la construcción de los histogramas en cada una de las subregiones, siendo cada histograma de 8 intervalos (8 flechas en la figura).

De forma experimental, el autor del método (Lowe [10]) encontró que se obtienen buenos resultados con regiones de interés de 16×16 ($m = 16$), y 4×4 subregiones ($n = 4$). Como cada histograma contiene 8 intervalos, el tamaño del descriptor es

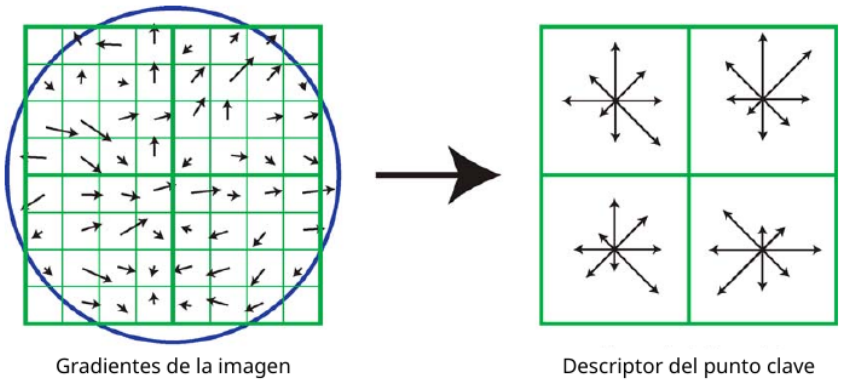


Figura 2.8: Creación del descriptor SIFT. Figura obtenida de [10].

$4 \times 4 \times 8 = 128$ (tamaño de la división en subregiones multiplicado por el tamaño de cada histograma).

Por último, el descriptor es modificado para reducir los efectos debidos a cambios de iluminación. Para ello el vector es normalizado a una longitud unitaria. Así mismo, para reducir la influencia de grandes gradientes, en el vector unitario se toman valores no mayores a 0.2, y nuevamente se normaliza a longitud unitaria.

2.3.2. Función de frontera (BOF)

La función de frontera BOF (del inglés, *Boundary Object Function*) es un vector numérico utilizado para describir la forma de un objeto. Se ha usado principalmente para el reconocimiento de piezas en tareas de ensamblaje robótico [12]. Difiere con los descriptores de características locales presentados previamente (como lo es SIFT), en el hecho de que describe la forma de un objeto, pero no describe la vecindad de un punto característico.

De acuerdo con sus autores Peña-Cabrera et al. [13], su metodología está basada en la teoría psicológica de los Principios de Gestalt[1].

[1]Los principios de Gestalt son reglas de la organización de las escenas perceptivas. Estos principios tienen el objetivo de formular las regularidades según las cuales la entrada perceptiva es organizada en formas unitarias. Los principios son aplicados principalmente a la visión, pero también hay aspectos análogos en la percepción auditiva y somatosensorial. En la percepción visual, tales formas son las regiones del campo visual cuyas porciones son percibidas como agrupadas o unidas, y estas son segregadas del resto del campo visual [14].

Si un sistema puede aprender formas canónicas y estas están embebidas en la cognición inicial, es posible proporcionar información del objeto en el mundo real (mundo tridimensional) con representaciones bidimensionales, tal como se muestra en la figura 2.9.

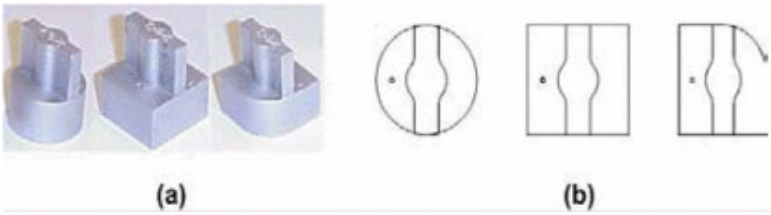


Figura 2.9: Representación en formas canónicas de piezas de ensamblaje. a) Piezas de trabajo, b) Formas canónicas. Figura obtenida de [13].

2.3.2.1. Obtención de la función frontera de un objeto

El primer paso es la obtención del contorno y del centroide del objeto, los cuales son problemas habituales en el procesamiento digital de imágenes. El contorno es un conjunto de puntos cuya cantidad depende de la resolución de la imagen, pero se puede considerar que están en el orden de las centenas. Para la construcción de BOF no se necesitan todos los puntos, sino solo un subconjunto, así que el contorno es cuantizado a n puntos, siendo n el tamaño del vector BOF.

La función de frontera del objeto BOF se obtiene calculando las distancias de los puntos en el contorno del objeto al centroide de este:

$$d_i = \sqrt{(x_i - c_x)^2 + (y_i - c_y)^2} \tag{2.7}$$

donde (x_i, y_i) — Coordenadas del i-ésimo punto del contorno
(c_x, c_y) — Coordenadas del centroide

Posteriormente estas distancias son concatenadas.

$$BOF' = [d_1, d_2, d_3, \ldots, d_n] \tag{2.8}$$

donde n — Tamaño del vector BOF

Por último, el vector es normalizado.

$$BOF = \frac{BOF'}{\text{máx}(BOF')} \tag{2.9}$$

donde $\text{máx}(BOF')$ — Máxima componente del vector BOF'

A partir de la representación BOF, se pueden obtener las coordenadas de los puntos en el contorno.

$$x_i = BOF_i \cos\left(\frac{2\pi i}{n}\right) \tag{2.10a}$$

$$y_i = BOF_i \sin\left(\frac{2\pi i}{n}\right) \tag{2.10b}$$

donde BOF_i — Elemento i-ésimo del vector BOF
i — Índice del elemento i-ésimo

De acuerdo con Peña-Cabrera et al. [13], BOF es un descriptor invariante a la rotación, escalamiento y translación.

En algunos artículos, como Peña-Cabrera et al. [13], Peña-Cabrera et al. [12], al vector BOF presentado en este trabajo se le concatenan otras propiedades como el centroide del objeto, la orientación y la altura, obteniendo así un vector al que le llaman CFD&POSE. Para el propósito de este trabajo la información extra de CFD&POSE no se utiliza, y se mantiene la representación BOF descrita en la ecuación (2.9), tal como lo hacen en Lomas-Barrie et al. [1] y Peña-Cabrera et al. [15].

2.4. Representación de la imagen

2.4.1. Bolsa de palabras visuales

Bolsa de palabras visuales BoVW (del inglés, *Bag of Visual Words*) es un método que integra los descriptores extraídos de una imagen en una representación de esta misma. Corresponde con la etapa de Transformación de características que se introdujo en la subsección *Esquema general para la clasificación de imágenes* 2.2.3. El método toma como entrada un conjunto de descriptores $\{d_1, \ldots, d_n\}$ (dónde $d_i \in \mathbb{R}^m$, m es el tamaño del descriptor) y los agrupa en una representación de un solo vector h (dónde $h \in \mathbb{R}^k$, k el tamaño del vocabulario de palabras visuales). Al vector h se le conoce como *bolsa de palabras visuales*.

Uno de los primeros trabajos que utilizó BoVW es el de Csurka et al. [16], quienes implementaron todo el esquema de clasificación de imágenes que se aplica también en este trabajo. De acuerdo con estos mismos autores, el método está inspirado como analogía al método de clasificación de textos, bolsa de palabras BoW (del inglés, *Bag of Words*). En clasificación de textos, la *bolsa de palabras* es un vector que cuenta la ocurrencia de palabras en el documento, mientras que, en clasificación de imágenes, la *bolsa de palabras* es un vector que cuenta la ocurrencia de puntos característicos dentro de la imagen.

La implementación de BoVW se resume en dos pasos:

1. Generación del vocabulario de palabras visuales (también conocido como cuantización del espacio de características).

2. Generación del histograma (también conocido como generación de la bolsa de palabras visuales).

El primer paso consiste en cuantizar el espacio de descriptores utilizando unos grupos predeterminados. Al conjunto de centroides de los grupos predeterminados se le conoce como *vocabulario de palabras visuales*. Para generar el *vocabulario de palabras visuales* se utilizan métodos de cuantización vectorial, entre ellos, uno de los más simples y utilizados es k-medias (ver subsección 2.4.2).

En el segundo paso, cada uno de los descriptores que se extrajeron de la imagen es asignado al grupo más cercano (en otras palabras, a cada descriptor de la imagen se le asigna la palabra del vocabulario visual más cercana), y, por último, se genera la *bolsa de palabras visuales*, la cual es el conteo de cada una de las palabras visuales en la imagen. La *bolsa de palabras visuales* es un histograma sobre el espacio de palabras visuales, y representa la distribución de este espacio en la imagen.

2.4.2. Algoritmo de agrupamiento k-medias

K-medias es un algoritmo de agrupamiento popular dentro de los métodos de aprendizaje automático no supervisado. Su objetivo es la partición de un conjunto de puntos en k grupos, donde k es un número predefinido. Funciona de forma iterativa asignando puntos a su centroide más cercano y después actualizando a los centroides en función de las nuevas asignaciones de grupo. Ver capítulo 9 de [17].

Los pasos del algoritmo k-medias son:

1. Inicialización. De forma aleatoria se inicializan los k centroides de los grupos.
2. Asignación de grupo. Se asigna cada punto a su centroide de grupo más cercano.
3. Actualización de los centroides. Se actualiza cada centroide de grupo calculando la media de todos los puntos asignados a dicho grupo.
4. Repetición. Se repiten los pasos 2 y 3 hasta que el algoritmo converge, que suele definirse como un cambio muy pequeño en los centroides de grupo, o en un máximo número de iteraciones.

La salida del algoritmo k-medias es un conjunto de k centroides de grupo, los cuales tienen como característica que la distancia de ellos a los puntos que les fueron asignados es mínima. Por lo tanto, el algoritmo k-medias puede entenderse como un problema de optimización.

2.5. Clasificación estadística

Según el capítulo 1 de Zhou [17], en el contexto del aprendizaje automático, una predicción es un problema que mapea $f : \mathcal{X} \mapsto \mathcal{Y}$ de un espacio de entrada $\mathcal{X}$ a un espacio de

salida $\mathcal{Y}$ aprendiendo de un conjunto de datos de entrenamiento $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_m, y_m)\}$. A la función de mapeo se le conoce como *modelo*. Si el espacio de salida $\mathcal{Y}$ es un conjunto finito, al problema de predicción se le conoce como *clasificación*. Si $|\mathcal{Y}| = 2$, se trata de un problema de *clasificación binaria*, mientras que para $|\mathcal{Y}| > 2$, se trata de un problema de *clasificación multiclase*. A cada elemento del espacio de entrada $\mathcal{X}$ se le conoce como *muestra de entrenamiento*, mientras que a los elementos en el espacio de salida $\mathcal{Y}$ se les conoce como *etiquetas*. La idea es que el modelo aprendido sea capaz de generalizar, es decir, que cuando se le presenten muestras fuera del conjunto de entrenamiento, el modelo sea capaz de predecir su etiqueta.

2.5.1. Evaluación de un modelo

Se necesita evaluar la capacidad de generalización del modelo. Para ello se utiliza un *conjunto de prueba* que permite estimar la habilidad de clasificar nuevas muestras. Es necesario que los conjuntos de entrenamiento y de prueba sean mutuamente excluyentes, pues si una muestra del conjunto de prueba se utiliza en el entrenamiento del modelo, la evaluación de este resulta engañosa al no demostrar capacidad de generalización con nuevas muestras. Ver capítulo 2 de [17].

Para evaluar la capacidad de generalización se utilizan métricas cuantitativas que se discuten en la subsección *Métricas de evaluación* 2.5.1.3.

Dado un conjunto de m muestras $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_m, y_m)\}$, se puede producir un conjunto de entrenamiento y de prueba a partir de los métodos de retención y de validación cruzada[1].

2.5.1.1. Método de retención

Del inglés *hold-out method*, consiste en dividir el conjunto de datos D en dos conjuntos disjuntos, el conjunto de entrenamiento S y el conjunto de entrenamiento T, donde $D = S \cup T$ y $S \cap T = \phi$. Se entrena al modelo en el conjunto S y se predicen las muestras del conjunto T. En la práctica, el tamaño del conjunto S es mayor que el del conjunto T porque mientras mayor cantidad de muestras se tengan para generar el modelo, mejor será su rendimiento, sin embargo, no se debe descuidar el tamaño del conjunto T o se caerá en una desconfianza de las métricas de evaluación.

2.5.1.2. Método de validación cruzada

Del inglés *cross-validation method*, consiste en dividir al conjunto de datos D en k conjuntos disjuntos con tamaños similares, $D = D_1 \cup D_2 \cup \ldots \cup D_k$, $D_i \cap D_j = \phi (i \neq j)$. Se realizan k iteraciones, en la primera se utilizan k–1 conjuntos para entrenar al modelo y el restante para evaluar su rendimiento. Se repite el proceso de tal forma que en cada

[1] Existen más métodos para generar los conjuntos de entrenamiento y prueba, pero para propósitos de este trabajo solo se mencionan estos dos. Ver capítulo 2 de [17]

iteración se ocupa a un conjunto D_i como conjunto de prueba. Finalmente se promedian las métricas de cada iteración para obtener una métrica global. La figura 2.10 ilustra el proceso de validación cruzada con $k = 10$.

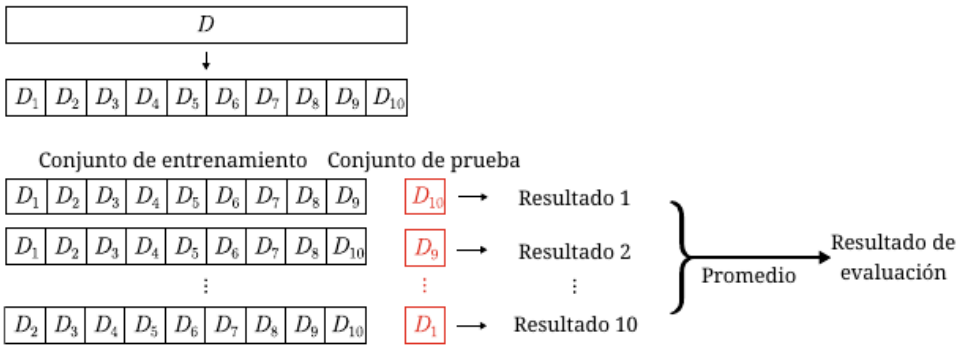


Figura 2.10: Proceso de validación cruzada con $k = 10$. Figura obtenida del capítulo 2 de [17].

El proceso de validación cruzada implica entrenar y evaluar k veces, por lo que para modelos complejos (como lo son las redes neuronales), resulta en una complejidad computacional alta en tiempo.

2.5.1.3. Métricas de evaluación

La principal métrica de evaluación en clasificación es la *exactitud* (del inglés, *accuracy*), la cual se calcula como el cociente del número de aciertos a entre el total de muestras m.

$$Acc = \frac{a}{m} \tag{2.11}$$

La tasa de error es el complemento de la exactitud.

$$E = 1 - Acc \tag{2.12}$$

Para clasificación binaria existen cuatro tipos de formas de combinar los resultados de clases verdaderas y clases predichas: verdadero positivo TP, falso positivo FP, verdadero negativo TN y falso negativo FN. Verdadero/falso se refieren a si la predicción fue correcta y positivo/negativo al tipo de clase. Estas cuatro combinaciones se expresan en una *matriz de confusión* (tabla 2.1).

Clase verdadera	Clase predicha	
	Positivo	Negativo
Positivo	*TP*	*FN*
Negativo	*FP*	*TN*

Tabla 2.1: Matriz de confusión para clasificación binaria

En la tabla 2.1, *TP* se refiere al número de muestras verdaderas positivas (del inglés, *true positive*), *FN* al número de muestras falsas negativas (del inglés, *false negative*), *FP* al número de muestras falsas positivas (del inglés, *false positive*) y *TN* al número de muestras verdaderas negativas (del inglés, *true negative*).

La *precisión* P (del inglés, *precision*) y la *exhaustividad* R (del inglés, *recall*) se definen como:

$$P = \frac{TP}{TP + FP} \tag{2.13}$$

$$R = \frac{TP}{TP + FN} \tag{2.14}$$

Por último, el *valor* $F1$ (del inglés, *F1-score* o *F1-measure*), se define como la media armónica de la precisión y exhaustividad.

$$F1 = \frac{2 \times P \times R}{P + R} = \frac{2 \times TP}{2 \times TP + FP + FN} \tag{2.15}$$

Para clasificadores multiclase, estas métricas son calculadas para cada clase. Si se quieren considerar las métricas del clasificador completo (para todas las clases) se tienen las *métricas macro promedio* (del inglés, *macro average*). Para un clasificador con m clases estas métricas son:

$$\textit{macro-}P = \frac{1}{m} \sum_{i=1}^{m} P_i \tag{2.16}$$

$$\textit{macro-}R = \frac{1}{m} \sum_{i=1}^{m} R_i \tag{2.17}$$

$$\textit{macro-}F1 = \frac{2 \times \textit{macro-}P \times \textit{macro-}R}{\textit{macro-}P + \textit{macro-}R} \tag{2.18}$$

donde M_i —métrica M de la clase i

Existen casos en donde el conjunto de datos posee diferente número de muestras por cada clase. A esta condición se le conoce como desequilibrio de clases (del inglés, *class imbalance*). Para aliviar el desequilibrio de clases en las métricas de evaluación se tiene el concepto de *métricas macro promedio ponderado* (del inglés, *weighted macro average*), en donde la métrica de cada clase es ponderada usando el peso de su soporte. El *soporte* se refiere al número de muestras que contiene cada clase. Las métricas macro promedio ponderado son:

$$macro\text{-}P\ ponderada = \frac{1}{n}\sum_{i=1}^{m} n_i P_i \tag{2.19}$$

$$macro\text{-}R\ ponderada = \frac{1}{n}\sum_{i=1}^{m} n_i R_i \tag{2.20}$$

$$macro\text{-}F1\ ponderada = \frac{1}{n}\sum_{i=1}^{m} n_i F1_i \tag{2.21}$$

donde n_i — número de muestras de la clase i (también conocido como soporte)
n — total de muestras en el conjunto de datos
m — número de clases

2.5.2. Máquina de soporte vectorial (SVM)

Del inglés Support Vector Machine (SVM), es un modelo de clasificación estadística. Ver capítulo 6 de [17]. La idea básica es: dado un conjunto de entrenamiento $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_m, y_m)\}$, en dónde $y_i \in \{-1, +1\}$, utilizar al conjunto D para encontrar un hiperplano que separe a las muestras de diferentes clases. Pueden existir diferentes hiperplanos que separen a las muestras, pero se debe elegir aquel que tenga mayor tolerancia perturbaciones locales, tal como se ilustra en la figura 2.11 en donde el hiperplano ideal está en color rojo.

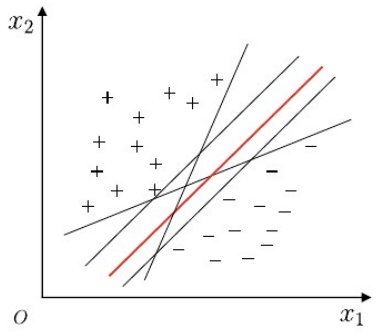


Figura 2.11: Hiperplanos que separan muestras de entrenamiento. Figura obtenida del capítulo 6 de [17].

Un hiperplano se expresa de la siguiente manera.

$$\mathbf{w}^\top \mathbf{x} + b = 0 \tag{2.22}$$

donde $\mathbf{w} = (w_1; w_2; \ldots; w_d)$ — Vector (columna) normal al hiperplano
$\mathbf{x} = (x_1; x_2; \ldots; x_d) \in \mathcal{X}$ — Vector (columna) variable d-dimensional del espacio de entrada $\mathcal{X}$

b — Sesgo (en inglés, *bias*), (denota la distancia del hiperplano al origen)

Suponiendo que el hiperplano $(\mathbf{w}, b)$ puede clasificar correctamente las muestras de entrenamiento, significa que para $(\mathbf{x}_i, y_i) \in D$ existe $\mathbf{w}^\top \mathbf{x}_i + b > 0$ cuando $y_i = +1$, así como, $\mathbf{w}^\top \mathbf{x}_i + b < 0$ cuando $y_i = -1$. Sea entonces:

$$\begin{cases} \mathbf{w}^\top \mathbf{x}_i + b \geq +1, & y_i = +1 \\ \mathbf{w}^\top \mathbf{x}_i + b \leq -1, & y_i = -1 \end{cases} \tag{2.23}$$

La expresión (2.23) se ilustra en la figura 2.12, en donde el hiperplano ideal está marcado con color rojo. Los puntos más cercanos al hiperplano (aquellos que caen sobre las líneas punteadas en la figura) son llamados *vectores de soporte* (del inglés, *support vectors*). A la distancia de dos vectores de soporte de diferentes clases al hiperplano se le conoce como margen γ, y está definido por:

$$\gamma = \frac{2}{\|\mathbf{w}\|} \tag{2.24}$$

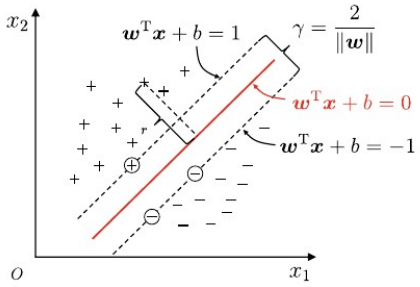


Figura 2.12: Vectores de soporte y margen. Figura obtenida del capítulo 6 de [17].

Para encontrar al hiperplano ideal, es necesario maximizar el margen, lo cual equivale a encontrar los parámetros que maximizan γ sujeto a las restricciones (2.23), es decir:

$$\begin{aligned} \underset{\mathbf{w},b}{\text{máx}} \quad & \frac{2}{\|\mathbf{w}\|} \\ \text{sujeto a} \quad & y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1, \qquad i = 1, 2, \ldots, m \end{aligned} \tag{2.25}$$

La optimización se realiza maximizando $\|\mathbf{w}\|^{-1}$, lo cual es equivalente a minimizar $\|\mathbf{w}\|^2$. Por lo tanto, la expresión (2.25) se reescribe como:

$$\begin{aligned} \underset{\mathbf{w},b}{\text{mín}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 \\ \text{sujeto a} \quad & y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1, \qquad i = 1, 2, \ldots, m \end{aligned} \tag{2.26}$$

La expresión (2.26) es la forma primordial (del inglés, *primal form*) de las máquinas de soporte vectorial SVM.

2.5.2.1. SVM con margen suave y funciones kernel

La expresión (2.26) asume que las muestras de entrenamiento son linealmente separables. Esto es un problema porque existen tareas de clasificación que no cumplen con esta condición. Para solucionar esto, se introduce el concepto de *funciones kernel*, que son funciones que permiten mapear las muestras de entrenamiento a un espacio de características de mayor dimensión, de tal forma que, en ese nuevo espacio, las muestras sean linealmente separables.

Existen ocasiones en que es necesario permitir al modelo cometer algunos errores, con el objetivo de evitar el sobreajuste[1]. Para ello, se agregan variables de holgura (del inglés, *slack variables*) al problema de optimización primordial (2.26), las cuales permiten que algunas muestras violen la restricción planteada $y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1$. Se dice que cuando el modelo SVM permite algunos errores, este posee *margen suave*.

Agregando los conceptos de función kernel y margen suave al problema (2.26) se tiene la formulación primordial completa de SVM. A esta formulación Chang y Lin [18] la llaman C-SVM.

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}\xi_i \\ \text{sujeto a} \quad & y_i(\mathbf{w}^\top \phi(\mathbf{x}_i) + b) \geq 1 - \xi_i, \\ & \xi_i \geq 0, \quad i = 1, 2, \ldots, m \end{aligned} \tag{2.27}$$

En dónde C es un parámetro de regularización para las variables de holgura ξ_i. $\phi(\mathbf{x}_i)$ mapea $\mathbf{x}_i$ en un espacio de mayor dimensión.

La función de mapeo ϕ está relacionada directamente con la función kernel $\mathcal{K}$ de la siguiente manera:

$$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j) \tag{2.28}$$

Tanto C como la función kernel $\mathcal{K}$ en la expresión (2.27) son parámetros que se establecen al momento de entrenar el modelo, e introducen cierta incertidumbre porque su desempeño depende del problema a resolver y de las muestras de entrenamiento.

Algunas de las funciones kernel más utilizadas se presentan en la tabla 2.2.

[1] El sobreajuste se presenta cuando el modelo corresponde de forma cercana a los datos de entrenamiento, pero no generaliza.

Nombre	Expresión	Parámetros
Kernel lineal	$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^\top \mathbf{x}_j$	
Kernel polinomial	$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \left(\mathbf{x}_i^\top \mathbf{x}_j\right)^d$	$d \geq 1$ es el grado del polinomio
Kernel gaussiano o de base radial (RBF)	$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$	$\sigma > 0$ el ancho del kernel gaussiano
Kernel laplaciano	$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|}{\sigma}\right)$	$\sigma > 0$
Kernel sigmoide	$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \tanh\left(\beta \mathbf{x}_i^\top \mathbf{x}_j + \theta\right)$	tanh es la función tangente hiperbólica, $\beta > 0$, $\theta < 0$

Tabla 2.2: Funciones kernel más comunes

2.5.2.2. Extensión de SVM a clasificación multiclase

La formulación de SVM 2.27 supone un problema de clasificación binaria, por lo que para extender su uso a clasificadores multiclase se tienen dos principales estrategias: *uno contra todos* (del inglés, *one vs all*) y *uno contra uno* (del inglés, *one vs one*). Ver [19].

En la estrategia **uno contra todos** se entrenan k modelos SVM, donde k es el número de clases. El i-ésimo SVM es entrenado con todos los ejemplos de la clase i-ésima como clase positiva y el resto de los ejemplos como clase negativa. Esto resulta en k funciones de decisión

$$\begin{gathered} (\mathbf{w}^1)^\top \phi(\mathbf{x}) + b^1 \\ \vdots \\ (\mathbf{w}^k)^\top \phi(\mathbf{x}) + b^k \end{gathered}$$

La predicción en la estrategia de *uno contra todos* sigue la siguiente función de decisión

$$\text{clase de } \mathbf{x} \equiv \arg \max_{i=1,\ldots,k} ((\mathbf{w}^i)^\top \phi(\mathbf{x}) + b^i) \tag{2.29}$$

En la estrategia **uno contra uno** se construyen $k(k-1)/2$ clasificadores, donde cada uno es entrenado con datos de dos clases.

Usando la estrategia *uno contra uno*, la predicción se realiza mediante votos. Si $\text{sign}((\mathbf{w}^{ij})^\top \phi(\mathbf{x}) + b^{ij})$ dice que $\mathbf{x}$ está en la clase i-ésima, se añade un voto a la clase i-ésima, en caso contrario, se incrementa un voto a la clase j-ésima. Se predice $\mathbf{x}$ en la clase con mayor número de votos.

Hsu y Lin [19] concluyen que, aunque la estrategia *uno contra todos* implica construir menos clasificadores, *uno contra uno* es más adecuado para problemas prácticos pues produce mejores tasas de exactitud. En el método de este trabajo se utiliza SVM, que está implementado en la biblioteca Scikit-learn [20], la cual utiliza la estrategia *uno contra uno* para entrenar clasificadores multiclase.

Capítulo 3

Método

El objetivo de este trabajo es implementar un método de reconocimiento de escenas que sea una alternativa de bajo consumo de recursos computacionales. La función de frontera BOF es un descriptor que ha sido utilizado para obtener la posición relativa de un observador con respecto a un punto de referencia [1], además, BOF tiene la ventaja de que su obtención involucra pocos cálculos una vez que se han encontrado los contornos de los objetos. Estas condiciones incentivan a este trabajo a extender el uso del descriptor BOF para el reconocimiento de escenas, utilizándolo como método de extracción de características.

Tomando en cuenta el esquema general para la clasificación de imágenes (subsección 2.2.3), se propone un esquema que consiste en Extracción de características usando *BOF*, transformación de características usando *BoVW* y clasificación usando *SVM*. Para propósitos de este trabajo, a este esquema se le define como BOF-BoVW.

Para comparar el rendimiento del esquema BOF-BoVW propuesto, se entrena el siguiente esquema: Extracción de características usando *SIFT*, transformación de características usando *BoVW* y clasificación usando *SVM*, esquema que ya ha sido utilizado en el estado del arte, por ejemplo, en [16], [21], y [22]. Para propósitos de este trabajo, a este esquema se le define como SIFT-BoVW. SIFT es un método de extracción de características locales robusto, cuyos descriptores proporcionan invariancia a distorsiones afines, lo que significa que puede reconocer los mismos puntos característicos a pesar de haber cambios de perspectiva o puntos de vista, sin embargo, la obtención de estos puntos característicos involucra una alta cantidad de operaciones, especialmente si se compara con las que realiza BOF.

Dentro del estado del arte existen métodos de extracción de características locales que simplifican los cálculos que realiza SIFT, tales como SURF [23] y ORB [24], sin embargo, estos no se seleccionan en este trabajo porque presentan inconvenientes para su implementación. SURF es un método con patente vigente [25], mientras que ORB produce descriptores basados en cadenas binarias, mismos que no pueden ser utilizados para generar el vocabulario de palabras visuales mediante k-medias, pues este tipo de descriptores no pueden ser comparados utilizando la distancia euclidiana; la medida de distancia adecuada para comparar descriptores basados en cadenas binarias es la

distancia de Hamming.

La diferencia entre la propuesta BOF-BoVW y SIFT-BoVW radica en la etapa de extracción de características. Se espera que BOF tenga un menor costo computacional que SIFT, aunque, sacrificando el rendimiento de clasificación.

Cabe resaltar que SIFT y BOF funcionan de distinta forma, el primero describe puntos característicos, mientras que el segundo describe la forma de un objeto, además, SIFT también detecta dichos puntos característicos, mientras que BOF requiere previamente de la segmentación de una imagen y localización de contornos. Los detalles sobre la extracción de características BOF son descritos en la sección 3.2.

3.1. Conjunto de datos utilizado

El conjunto de datos utilizado en los experimentos es *Microsoft 7 scenes RGB-D dataset* [26], el cual es una colección de fotogramas RGB-D con la respectiva trayectoria de la cámara rastreada.

Como su nombre lo indica, los fotogramas pertenecen a 7 diferentes tipos de escenas, las cuales se presentan en la tabla 3.1.

Cada tipo de escena contiene varias secuencias, las cuales fueron grabadas a mano con una cámara Kinect RGB-D a una resolución de 640 x 480 pixeles. Cada secuencia es un flujo de fotogramas RGB-D, con una longitud de entre 500 y 1 000 fotogramas. Cada fotograma consiste en 3 archivos: color (imagen a color RGB de 24 bits), profundidad (imagen de profundidad de 16 bits) y pose (matriz 4x4 de cámara a mundo[1], en coordenadas homogéneas). Para este trabajo solo se ocupan la imagen de color en la implementación SIFT-BoVW y la imagen de profundidad en la implementación BOF-BoVW.

Los parámetros intrínsecos de la cámara son: centro óptico de (320,240) y longitud focal de (585,585).

[1]La matriz de cámara a mundo representa la transformación entre el sistema de coordenadas de la cámara y el sistema de coordenadas del mundo. Se trata de una matriz de tamaño 4x4 que determina las operaciones de traslación, rotación, escala y proyección perspectiva, necesarias para llevar a cabo la transformación de coordenadas. Usando esta matriz, un punto en el sistema de coordenadas de la cámara puede ser transformado al sistema de coordenadas del mundo. Ver capítulo 2 de [27].

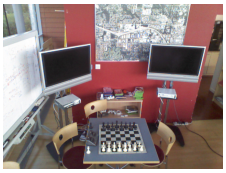
Escena	Descripción	Fotograma ejemplo
Chess (Ajedrez)	Escena que contiene una mesa con un tablero de ajedrez.	
Fire (Fuego)	Parte de un pasillo que contiene un par de extintores, una maceta y un letrero.	
Heads (Cabezas)	Un escritorio sobre el cual se encuentran varios objetos como dos monitores, tres modelos de cabezas, algunos documentos, un teclado, etc.	
Office (Oficina)	Oficina con 3 escritorios, cada uno con diferentes objetos.	
Pumpkin (Calabaza)	Área de descanso con algunos sillones, una cafetera, un lavatrastos, y, en el centro de la estancia, una calabaza.	
Red Kitchen (Cocina roja)	Estancia con un comedor de madera.	
Stairs (Escaleras)	Escaleras del edificio.	

Tabla 3.1: Tipos de escenas en *Microsoft 7 scenes RGB-D dataset*

3.2. Extracción de características BOF

Dentro del esquema BOF-BoVW, se propone que los objetos contenidos en las escenas sean lo suficientemente significativos como para representar a la imagen, y para poder distinguirlas entre distintos tipos de escenas. La idea es extraer los descriptores BOF de los objetos contenidos en un fotograma en particular, agruparlos en una representación usando la técnica de BoVW, y que esta sea la entrada del clasificador.

El problema de la técnica BOF-BoVW está en cómo encontrar y segmentar automáticamente los objetos contenidos en un fotograma. La detección de objetos[1] es un problema de visión computacional que por sí solo es tan extenso como la clasificación de imágenes, así que para resolver este problema se adquiere un enfoque más simple, que consiste en segmentar de acuerdo con capas en la nube de puntos. Cada una de estas capas contiene información de los objetos a diferentes niveles de profundidad, lo que facilita el hallazgo de los contornos de objetos. Entonces, para la extracción de características BOF se siguen los siguientes pasos:

1. A partir de la imagen de profundidad se genera la nube de puntos (puntos en el espacio $\mathbb{R}^3$) tal como se describe en la subsección 2.1.3. Esta nube de puntos es un esbozo del modelo 3D de la escena.

2. La nube de puntos se divide en capas de acuerdo con un intervalo de corte, o bien, de acuerdo con el número de capas deseado. Si se elige utilizar un intervalo de corte, se establece el tamaño de este, por el contrario, si se elige un número de capas, se establece ese número. Estos últimos parámetros se establecen antes de la extracción de características, y dependen del problema a resolver (son hiperparámetros[2]). Todos los puntos contenidos dentro de una capa son proyectados a un plano perpendicular al eje de alabeo de la cámara (del inglés, *roll axis*). Los ejes de la cámara se muestran en la figura 3.1.

3. Por cada capa generada en el paso anterior, se genera una imagen binaria, es decir, se discretiza el espacio. El espacio es dividido en rejillas de $m \times n$ pixeles. La imagen binaria conste de 1's en aquellas rejillas que contengan al menos un punto del espacio, 0's en donde no exista punto alguno.

[1] La detección de objetos es un área de la visión computacional que trata con la detección de instancias de objetos visuales de cierta clase (como humanos, animales, carros, etc.) en imágenes digitales. Las principales métricas en la detección de objetos son la exactitud (de clasificación y de localización) y la rapidez. Un análisis detallado y actualizado de esta área es [28].

[2] En un modelo de aprendizaje automático, los hiperparámetros son parámetros que nos son aprendidos por el modelo, sino que son establecidos manualmente antes del entrenamiento.

4. Se suaviza la imagen binaria para eliminar huecos causados por la baja resolución de la nube de puntos. El suavizado se realiza mediante la operación morfológica de cierre[1].

5. Por cada imagen binaria se buscan contornos cerrados que cumplan con que su área sea de al menos el 10 % del área de la imagen binaria.

6. Por cada contorno se extrae su BOF, tal como se describe en la subsección 2.3.2.1.

7. Todos los descriptores BOF extraídos se apilan y son asociados al fotograma.

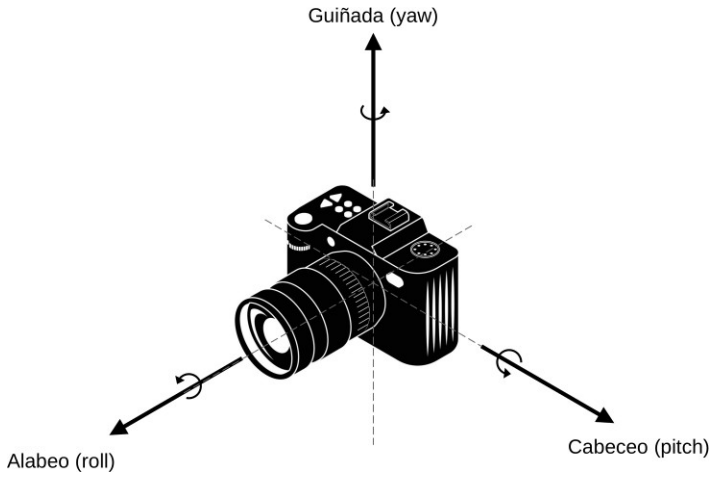


Figura 3.1: Ejes de la cámara [2]

Hay una consideración del método descrito que debe tenerse en cuenta. Existen casos en que la nube de puntos se encuentra vacía, o tiene muy pocos puntos como para formar cúmulos de puntos que describan objetos. En ambos casos no es posible seguir este método, pues no se están generando datos para obtener predicciones. Un ejemplo es un cuarto vacío y muy grande, o un pasillo sin muebles. Todos los fotogramas que no aportan información con contornos son descartados, y se sugiere utilizar otro método.

Retomando la imagen RGB-D mostrada en la figura 2.3, y para ilustrar el proceso descrito, se tiene la figura 3.2. En este ejemplo la nube de puntos fue dividida en 3 capas. En la figura 3.2 se muestran cada una de las capas proyectadas en un plano 2D y su correspondiente imagen binaria, además, en cada imagen binaria se muestran en rojo los contornos que cumplen con al menos el 10 % del área de la imagen completa, de los cuales se extraen sus descriptores BOF.

[1] La operación morfológica de cierre es una operación que permite cerrar pequeños agujeros en objetos de primer plano. Está definida como una operación de dilatación seguida de una operación de erosión. Ver capítulo 9 de [2]

[2] Figura propia. Ilustración de cámara obtenida de `https://www.vecteezy.com/`

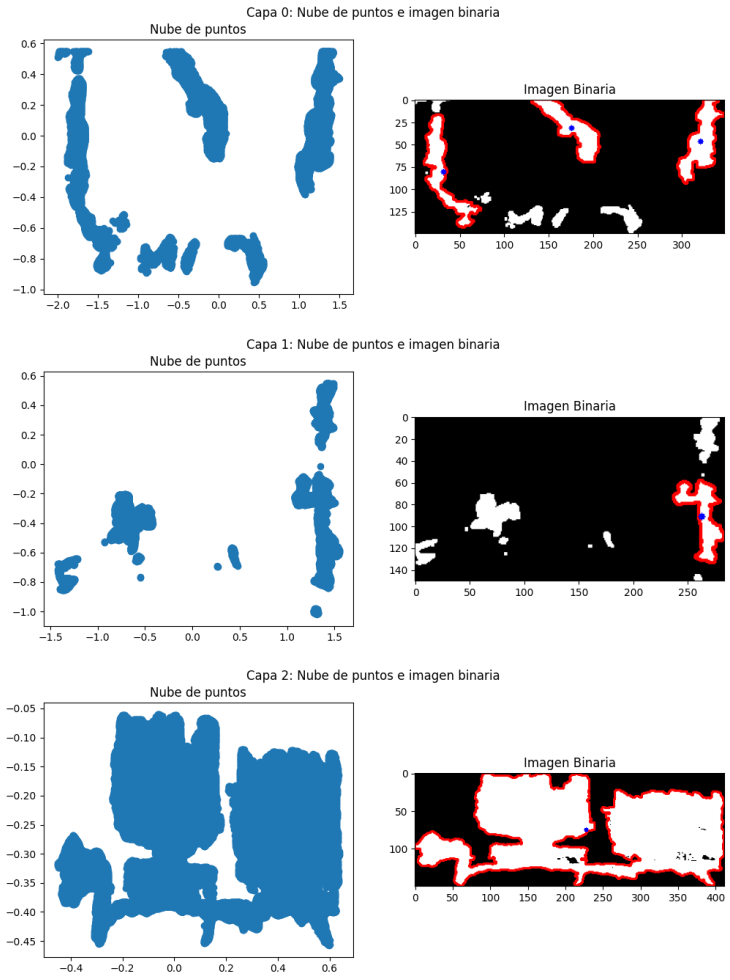


Figura 3.2: Capas extraídas de una nube de puntos 3D y su correspondiente imagen binaria

Cabe resaltar que, por cada capa extraída, se pueden obtener desde cero a varios contornos. Por consiguiente, por cada fotograma, se tienen de cero a varios contornos. De igual forma, por cada fotograma, se tienen de cero a varios descriptores BOF.

3.3. Implementación del método propuesto

Ambos métodos BOF-BoVW y SIFT-BoVW se entrenan y evalúan mediante el método de retención (ver subsección 2.5.1.1). Se utiliza una secuencia por tipo de escena. El conjunto de datos se divide en 75 % para entrenamiento y 25 % para prueba.

Como ya se mencionó, la única diferencia entre BOF-BoVW y SIFT-BoVW radica

en la etapa de extracción de características, por lo que la etapa de transformación de características y clasificación se realizan de la misma forma en ambos métodos.

En la etapa de extracción de características se extrae un conjunto (de tamaño variable) de descriptores por cada fotograma. Para descriptores BOF se extraen siguiendo los pasos que se muestran en la sección 3.2, mientras que para descriptores SIFT se sigue la metodología que propuso Lowe [10], y está descrita en la subsección 2.3.1. Para descriptores BOF se utilizan las imágenes de profundidad, mientras que para descriptores SIFT se ocupan las imágenes a color RGB, estas últimas transformadas a escala de grises debido a que la formulación de SIFT original contempla solo imágenes monocromáticas.

La etapa de transformación de características basada en BoVW se aplica siguiendo los pasos de la subsección 2.4.1: generación del vocabulario de palabras visuales y generación del histograma.

Cabe resaltar que la generación del vocabulario de palabras visuales se realiza con el conjunto de datos de entrenamiento, para desde el inicio evitar influencia de los datos de prueba, y tener una mayor confianza en las métricas de evaluación.

Una vez generadas las bolsas de palabras visuales de cada fotograma, estos vectores son utilizados como entradas al clasificador. Se utiliza SVM porque es un clasificador que se adapta correctamente a problemas multiclase, además que ha dado buenos resultados en otros trabajos, como en [16], [21], y [22].

Para poder entrenar el modelo SVM satisfactoriamente, los datos necesitan estar estandarizados. La estandarización de datos es una técnica usada en aprendizaje automático para transformar los datos en una escala común, de tal forma que las características sean comparables y puedan ser analizadas. Implica que, para cada característica de las muestras, substraer la media y dividir por la desviación estándar. La estandarización es importante porque algunos modelos son sensibles a las escalas de las características de entrada. De acuerdo con Pedregosa et al. [20], SVM (con kernel RBF) es uno de los modelos que asume que todas las características están centradas en cero y tienen varianza dentro del mismo orden. Si una característica tiene varianza mucho mayor que otras, esta puede dominar la función objetivo y hacer que el estimador no pueda predecir otras características correctamente.

Para encontrar los parámetros adecuados de SVM[1] se utiliza validación cruzada (ver subsección 2.5.1.2) con $k = 5$. En primera instancia se evalúa el mejor tipo de kernel, y posteriormente el mejor parámetro regularizador C. El espacio de parámetros de tipos de funciones kernel son: lineal, polinomial de grado 3, gaussiano (RBF) y sigmoide. El espacio de parámetros de regularizador C son quince valores uniformemente espaciados entre 0.01 y 10.

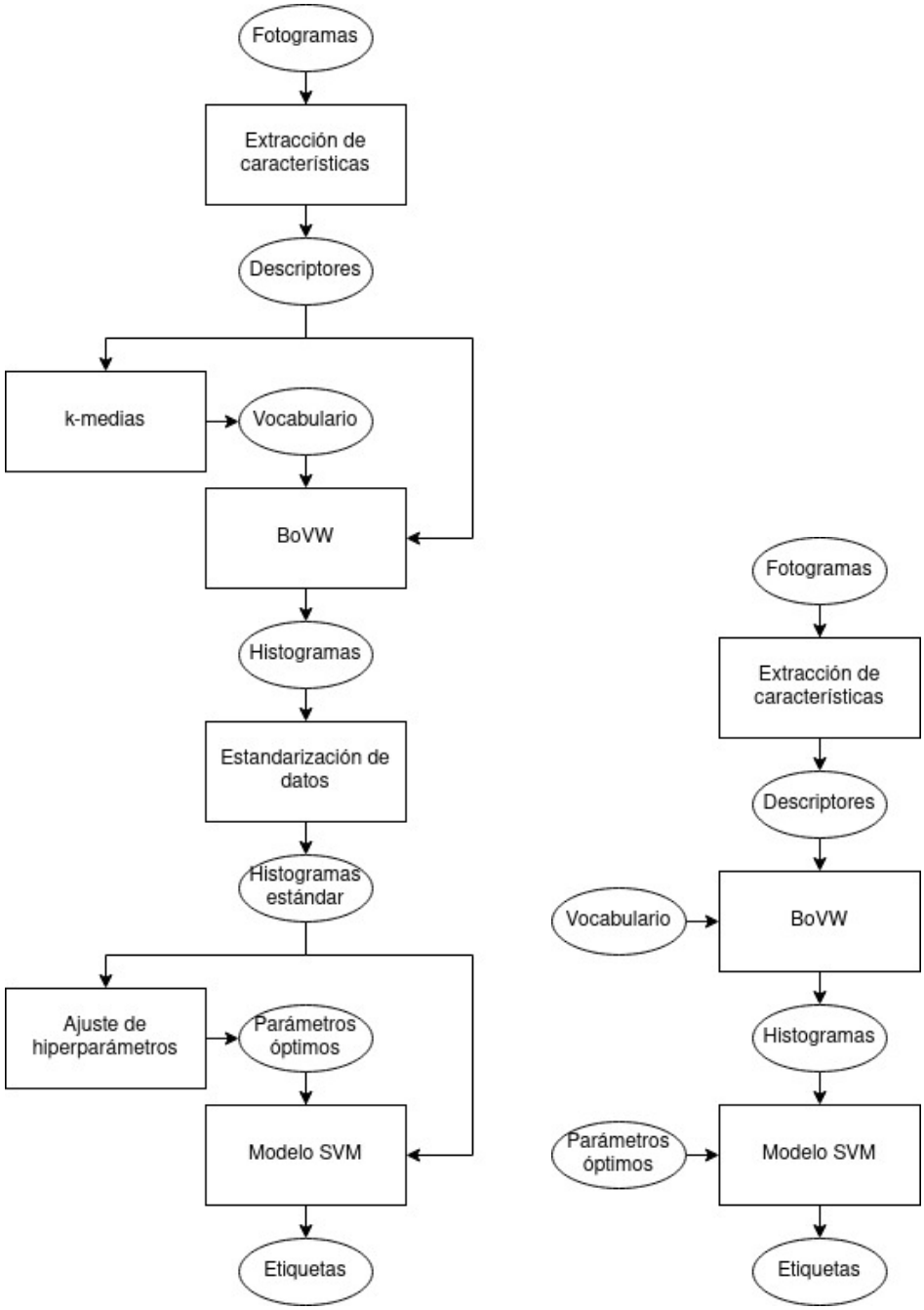
Por último, los modelos entrenados se evalúan en una secuencia distinta a la que se usó para el entrenamiento (evaluación distinta al método de retención que se mencionó inicialmente), esto para valorar su eficacia generalizando las escenas. Todos los resultados son mostrados en el capítulo 4.

[1] Al proceso para encontrar los hiperparámetros que ofrecen el mejor rendimiento de clasificación de un modelo de aprendizaje automático se le conoce como *ajuste de hiperparámetros*.

3.4. Diagrama del método propuesto

Una representación gráfica de los métodos BOF-BoVW y SIFT-BoVW descritos anteriormente es la que se tiene en la figura 3.3. En la figura 3.3a se representa la *fase de entrenamiento*. Como entrada a todo el esquema se tienen los fotogramas de entrenamiento, a partir de los cuales la etapa de extracción de características genera un conjunto de descriptores. Los descriptores posteriormente entran al algoritmo de k-medias para generar el vocabulario de palabras visuales. El vocabulario de palabras visuales es necesario para generar los histogramas a partir de los descriptores de entrenamiento. Los histogramas deben pasar por una etapa de estandarización de datos para después ser la entrada del ajuste de hiperparámetros (que utiliza validación cruzada para encontrar los parámetros óptimos). Una vez obtenidos los parámetros óptimos, estos se usan para generar el modelo SVM mejor evaluado. Por último, el modelo SVM óptimo toma como entrada los histogramas estandarizados de fotogramas para poder generar sus etiquetas correspondientes.

Una vez entrenado el modelo, para generar nuevas predicciones no se requieren realizar todas las etapas, y así se representa en la figura 3.3b. En la *fase de nuevas predicciones*, la extracción de características toma como entrada los nuevos fotogramas y genera sus correspondientes descriptores, los cuales BoVW utiliza para generar sus histogramas, que a su vez son una representación de los fotogramas originales. Posteriormente estos histogramas son los que se utilizan como entrada para que el modelo SVM genere su correspondiente predicción. Cabe resaltar que, para las nuevas predicciones, la etapa de estandarización de datos sigue presente, pero se omite en el diagrama por simplicidad. El diagrama de la figura 3.3b es justamente la implementación del esquema general para la clasificación de imágenes presentado en la subsección 2.2.3.



(a) Fase de entrenamiento **(b)** Fase de nuevas predicciones

Figura 3.3: Diagrama del método de reconocimiento de escenas propuesto

3.5. Herramientas de programación utilizadas

La implementación de los métodos BOF-BoVW y SIFT-BoVW, así como su evaluación, se realizaron utilizando Python 3.9. Para la mayoría de las etapas se utilizaron paquetes ya existentes, con excepción de la etapa de extracción de características BOF (sección 3.2). Para esta etapa se desarrolló un módulo que toma como entrada una nube de puntos (un fotograma), y retorna una lista de descriptores BOF.

A continuación, se presenta una lista de todos los paquetes de Python utilizados en el proyecto:

- NumPy (versión 1.23.5) [29]. Para funciones matemáticas y el tratamiento de arreglos multidimensionales.
- OpenCV (versión 4.5.5) [30]. Para el procesamiento digital de imágenes, así como para la extracción de características locales SIFT.
- Open3D (versión 0.16.0) [31]. Para la transformación, lectura y almacenamiento de nubes de puntos.
- SciPy (versión 1.9.3) [32]. Para la implementación de k-medias.
- Scikit-learn (versión 1.1.1). Para la división de datos en el método de retención, estandarizador de datos, implementación de SVM, validación cruzada, matriz de confusión y reporte de métricas de clasificación.
- Matplotlib (versión 3.5.2) [33]. Para la visualización de gráficas e imágenes.
- Jupyter Notebook (versión 6.4.11) [34]. Aplicación web para la creación de documentos con código Python interactivo, en los que se pueden guardar resultados, visualizaciones y texto descriptivo.
- Joblib (versión 1.1.0). Para el almacenamiento y carga de modelos entrenados.
- Tqdm (versión 4.64.0). Para la visualización de una barra de progreso en procesos largos.

El código desarrollado está disponible en el repositorio de GitHub `https://github.com/masues/ScenesRecognition-AYUDAME`.

Capítulo 4

Análisis de Resultados

4.1. Resultados de clasificación

Para los resultados obtenidos se utilizó el conjunto de datos *Microsoft 7 scenes RGB-D dataset* descrito en la sección 3.1. Se ocupan las primeras dos secuencias, abreviadas como *seq1* y *seq2*. Cada una contiene 6 500 fotogramas RGB-D, 1 000 fotogramas para cada una de las primeras 6 clases y 500 fotogramas para la clase escaleras (*stairs*).

4.1.1. BOF-BoVW

Para la extracción de descriptores BOF se utilizó el hiperparámetro de 3 capas de corte. Para la secuencia *seq1* se encontraron 19 244 descriptores BOF en total.

En primera instancia, se entrena el modelo utilizando el método de retención con la secuencia *seq1*, 75 % datos para entrenamiento y 25 % para prueba. Después de la división de datos, se tienen 4 875 fotogramas de entrenamiento, de los cuales se extrajeron un total de 14 484 descriptores BOF. Se tienen también 1 625 fotogramas de prueba, de los cuales se extrajeron 4 760 descriptores BOF.

Para la generación del vocabulario de palabras visuales se utilizan 256 centroides, lo que equivale a aproximadamente 1.8 % del número de descriptores de entrenamiento.

En la etapa de clasificación y usando validación cruzada se encontró que los parámetros óptimos del clasificador SVM son: **regulador** $C = 3.58$ **y una función kernel RBF.**

La matriz de confusión[1], usando como datos de prueba el 25 % de la secuencia *seq1* se muestra en la figura 4.1, mientras que un reporte detallado de clasificación se muestra en la tabla 4.1 .

[1] La obtención de las métricas de evaluación se detalla en la subsección 2.5.1.3

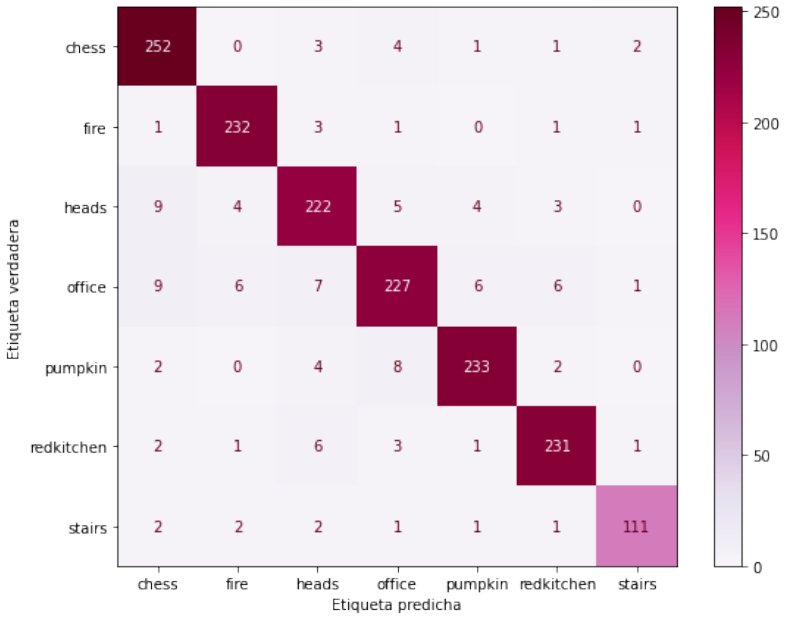


Figura 4.1: Matriz de confusión para BOF-BoVW usando método de retención con el 25 % de datos de prueba

	Precisión	Exhaustividad	Valor F1	Soporte
chess	0.91	0.96	0.93	263
fire	0.95	0.97	0.96	239
heads	0.90	0.90	0.90	247
office	0.91	0.87	0.89	262
pumpkin	0.95	0.94	0.94	249
redkitchen	0.94	0.94	0.94	245
stairs	0.96	0.93	0.94	120
Exactitud			0.93	1 625
Macro promedio	0.93	0.93	0.93	1 625
Macro promedio ponderado	0.93	0.93	0.93	1 625

Tabla 4.1: Reporte de clasificación para BOF-BoVW usando método de retención con el 25 % de datos de prueba. Valores redondeados a 2 decimales.

Para estimar la capacidad de generalización del modelo obtenido se evalúa utilizando como datos de prueba la secuencia *seq2*. Debido a que algunos fotogramas en la secuencia *seq2* estaban vacíos, solo se utilizaron 6 461. De estos fotogramas se extrajeron un total de 19 123 descriptores BOF.

La matriz de confusión, usando como datos de prueba la secuencia *seq2* se muestra en la figura 4.2, mientras que un reporte detallado de clasificación se muestra en la tabla 4.2.

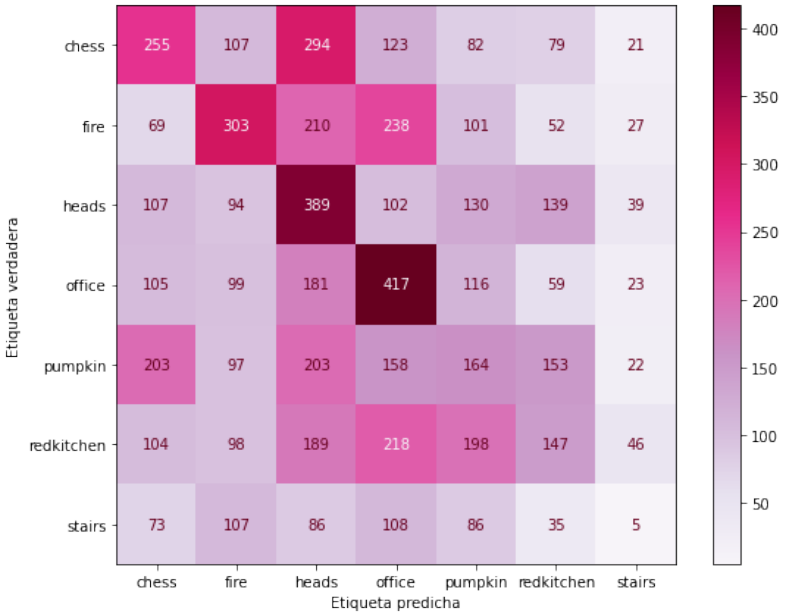


Figura 4.2: Matriz de confusión para BOF-BoVW usando la secuencia *seq2* como datos de prueba

	Precisión	Exhaustividad	Valor F1	Soporte
chess	0.28	0.27	0.27	961
fire	0.33	0.30	0.32	1 000
heads	0.25	0.39	0.30	1 000
office	0.31	0.42	0.35	1 000
pumpkin	0.19	0.16	0.17	1 000
redkitchen	0.22	0.15	0.18	1 000
stairs	0.03	0.01	0.01	500
Exactitud			0.26	6 461
Macro promedio	0.23	0.24	0.23	6 461
Macro promedio ponderado	0.24	0.26	0.25	6 461

Tabla 4.2: Reporte de clasificación para BOF-BoVW usando la secuencia *seq2* como datos de prueba. Valores redondeados a 2 decimales.

4.1.2. SIFT-BoVW

Se realizan los mismos pasos que para BOF-BoVW. Primero se entrena el modelo utilizando el método de retención con la secuencia *seq1*, 75 % datos para entrenamiento y 25 % para prueba. Después de la división de datos, se tienen 4 875 fotogramas de entrenamiento, de los cuales se extrajeron un total de 3 719 635 descriptores SIFT. Se tienen también 1 625 fotogramas de prueba, de los cuales se extrajeron 1 237 695 descriptores SIFT.

El número de descriptores que extrae el método SIFT es muy elevado, por lo que es necesario incrementar el tamaño del del vocabulario de palabras visuales, en este caso, utilizando un total de 1 024 centroides, lo que equivale a aproximadamente 0.03 % del número de descriptores de entrenamiento. De igual forma, por la gran cantidad de descriptores, el algoritmo de k-medias para la generación del vocabulario converge muy lentamente; para evitar este problema se toman 150 000 descriptores de forma aleatoria y estos son los que se utilizan para la generación del vocabulario. Haciendo este cambio, los 1 024 centroides ahora equivalen a aproximadamente 0.7 % del número de descriptores de entrada a k-medias.

En la etapa de clasificación y usando validación cruzada se encontró que los parámetros óptimos del clasificador SVM son: **regulador** $C = 0.01$ **y una función kernel lineal.**

La matriz de confusión, usando como datos de prueba el 25 % de la secuencia *seq1* se muestra en la figura 4.3, mientras que un reporte detallado de clasificación se muestra

en la tabla 4.3.

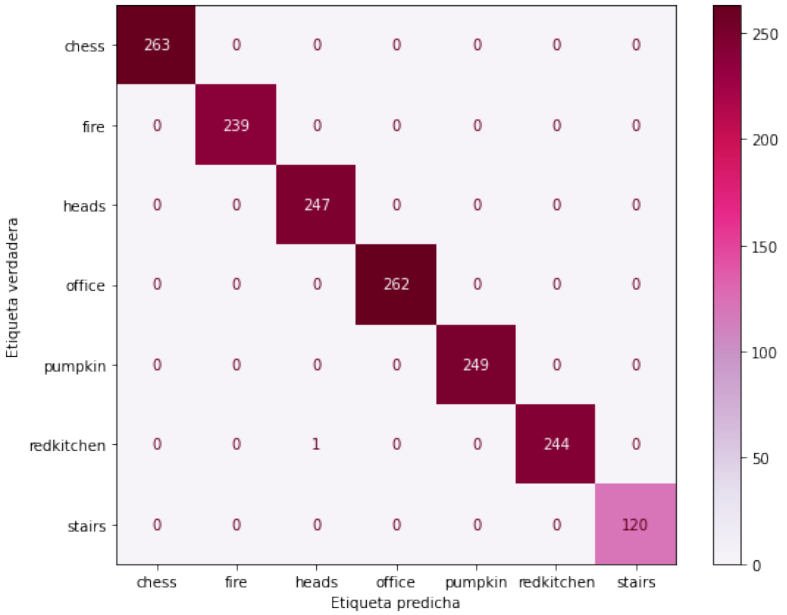


Figura 4.3: Matriz de confusión para SIFT-BoVW usando método de retención con el 25 % de datos de prueba

	Precisión	Exhaustividad	Valor F1	Soporte
chess	1.00	1.00	1.00	263
fire	1.00	1.00	1.00	239
heads	1.00	1.00	1.00	247
office	1.00	1.00	1.00	262
pumpkin	1.00	1.00	1.00	249
redkitchen	1.00	1.00	1.00	245
stairs	1.00	1.00	1.00	120
Exactitud			1.00	1 625
Macro promedio	1.00	1.00	1.00	1 625
Macro promedio ponderado	1.00	1.00	1.00	1 625

Tabla 4.3: Reporte de clasificación para SIFT-BoVW usando método de retención con el 25 % de datos de prueba. Valores redondeados a 2 decimales.

De igual forma que con BOF-BoVW, se evalúa el modelo utilizando como datos de prueba la secuencia *seq2*, con el objetivo de determinar su capacidad de generalización. De los 6 500 fotogramas de la secuencia *seq2* se extrajeron un total de 4 840 409 descriptores.

La matriz de confusión, usando como datos de prueba la secuencia *seq2* se muestra en la figura 4.4, mientras que un reporte detallado de clasificación se muestra en la tabla 4.4.

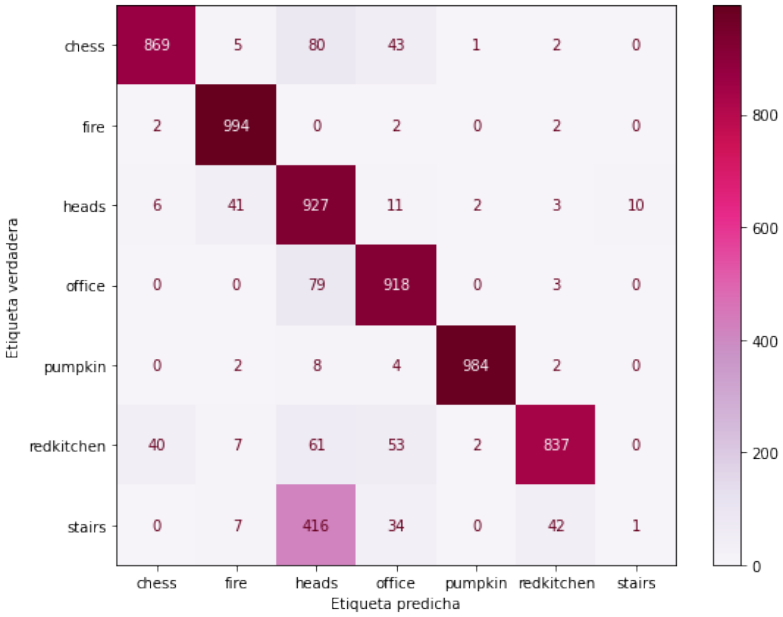


Figura 4.4: Matriz de confusión para SIFT-BoVW usando la secuencia *seq2* como datos de prueba

	Precisión	Exhaustividad	Valor F1	Soporte
chess	0.95	0.87	0.91	1 000
fire	0.94	0.99	0.97	1 000
heads	0.59	0.93	0.72	1 000
office	0.86	0.92	0.89	1 000
pumpkin	0.99	0.98	0.99	1 000
redkitchen	0.94	0.84	0.89	1 000
stairs	0.09	0.00	0.00	500
Exactitud			0.85	6 500
Macro promedio	0.77	0.79	0.77	6 500
Macro promedio ponderado	0.82	0.85	0.82	6 500

Tabla 4.4: Reporte de clasificación para SIFT-BoVW usando la secuencia *seq2* como datos de prueba. Valores redondeados a 2 decimales.

4.2. Resultados de rendimiento en el tiempo

En esta sección se evalúa el tiempo que tarda en ejecutar cada uno de los procesos que involucran el método. Las mediciones de tiempo se realizan en 2 plataformas distintas:

1. **Computadora 1**: Laptop con procesador Intel Core i5-1035G1 @ 3.60 GHz (Cuatro núcleos), 8 GB de RAM, S.O. (sistema operativo) Fedora Linux 37 (x86_64).

2. **Computadora 2**: Raspberry Pi 4 con procesador Cortex-A72 (ARM v8) 64-bit SoC @ 1.8GHz (Cuatro núcleos), 4G de RAM, S.O. Ubuntu 20.04.6 LTS (aarch64).

Las mediciones con la computadora 2 se realizan porque se espera que el método pueda ser implementado en una plataforma de factor reducido.

Se presentan dos tipos de resultados, tiempo de uso de CPU y tiempo real transcurrido. El primero consiste en la suma de los tiempos de Usuario y de Kernel[1], y se refiere al tiempo de CPU que ha utilizado el proceso. En procesadores multinúcleo, el tiempo de CPU toma en cuenta el tiempo utilizado por el proceso en cada uno de los núcleos por separado. El segundo es el tiempo real que ha transcurrido desde el inicio del proceso hasta la finalización de este; es básicamente un temporizador que no toma en cuenta el tiempo requerido por cada núcleo. Dependiendo del proceso estas medidas de tiempo pueden diferir debido a la existencia de los procesadores multinúcleo, en donde se podría paralelizar el proceso, y permitir concluirlo en un menor tiempo a que si se tuviera un solo núcleo (en este caso, el tiempo real sería menor que el tiempo de CPU).

Se realiza una comparación entre el método BOF-BoVW y SIFT-BoVW. Los procesos que se evaluaron son los siguientes:

1. Extracción de descriptores de un fotograma. Valor promedio obtenido de 10 ejecuciones sobre el mismo fotograma.

2. Extracción de descriptores de múltiples fotogramas. 1 000 fotogramas procesados. Para el caso de BOF-BoVW se saltaron 39 fotogramas porque estaban vacíos.

3. Generación del vocabulario de palabras visuales. Para el caso de BOF-BoVW es computado sobre 14 484 muestras con 256 centroides, mientras que para el caso de SIFT-BoVW es computado sobre 150 000 muestras con 1 024 centroides.

[1]Los modos de usuario y de kernel se refieren a la forma en la que operan los procesos a nivel de sistema operativo. El modo de usuario tiene acceso restringido a los recursos de hardware de la computadora, mientras que el modo de kernel tiene acceso privilegiado. La existencia de los modos de usuario y de kernel agrega una capa de seguridad a la ejecución de procesos asegurando que estos no interfieran entre sí.

4. Generación de los histogramas. Se realiza la medición de la generación de los histogramas de 4 875 muestras.

5. Ajuste del parámetro "función kernel" por validación cruzada. Espacio de parámetros de 4 valores.

6. Ajuste del parámetro "regulador C" por validación cruzada. Espacio de parámetros de 10 valores.

7. Entrenamiento del modelo usando parámetros definidos. Ambos SVM y con los parámetros que se mencionaron en la sección *Resultados de clasificación* 4.1.

8. Clasificación (SVM). Se realiza la medición de la clasificación de 4 875 muestras utilizando el modelo SVM entrenado en el punto 7.

9. Total fase de entrenamiento. Suma de los resultados obtenidos en los puntos 2 a 8 explicados previamente, mismos que equivalen a la fase de entrenamiento mencionada en la sección *Diagrama del método propuesto* 3.4.

10. Total fase de nuevas predicciones. Suma de los resultados obtenidos en los puntos 2, 4 y 8 explicados previamente, mismos que equivalen a la fase de nuevas predicciones mencionada en la sección *Diagrama del método propuesto* 3.4.

Las tablas comparativas poseen una columna llamada "Incremento", la cual contiene el incremento porcentual del tiempo requerido que se tiene para SIFT-BoVW, tomando como valor inicial el de BOF-BoVW. Se calcula con la fórmula (4.1). Los valores en esta columna están redondeados.

$$\text{incremento} = \frac{\text{Valor SIFT-BoVW} - \text{Valor BOF-BoVW}}{\text{Valor BOF-BoVW}} \times 100 \tag{4.1}$$

Los resultados obtenidos en la computadora 1 se muestran en la tabla 4.5, mientras que los resultados obtenidos en la computadora 2 se muestran en la tabla 4.6. Los valores en ambas tablas están redondeados a 2 decimales, con excepción de la primera fila "Extracción de descriptores de un fotograma", en la que están redondeados a 3 decimales.

	BOF-BoVW		SIFT-BoVW		Incremento	
	CPU [s]	Real [s]	CPU [s]	Real [s]	CPU [%]	Real [%]
Extracción de descriptores de un fotograma	0.088	0.088	0.548	0.078	522	-11
Extracción de descriptores de múltiples fotogramas	79.67	75.55	461.56	71.73	479	-5
Generación del vocabulario de palabras visuales	64.26	8.36	4 763.76	752.93	7 313	8 905
Generación del histograma	5.01	0.78	206.08	27.83	4 012	3 453
Ajuste del parámetro "función kernel" por validación cruzada	29.17	29.32	94.90	95.59	225	226
Ajuste del parámetro "regulador C" por validación cruzada	76.25	76.56	315.46	317.20	314	314
Entrenamiento del modelo usando parámetros definidos	1.25	1.26	1.93	1.94	54	55
Clasificación	2.28	2.29	2.20	2.21	-4	-3
Total fase de entrenamiento	257.89	194.12	5 845.90	1 269.43	2 167	554
Total fase de nuevas predicciones	86.96	78.62	669.84	101.77	670	29

Tabla 4.5: Resultados de rendimiento en el tiempo en la computadora 1

	BOF-BoVW		SIFT-BoVW		Incremento	
	CPU [s]	Real [s]	CPU [s]	Real [s]	CPU [%]	Real [%]
Extracción de descriptores de un fotograma	0.296	0.310	0.559	0.288	89	-7
Extracción de descriptores de múltiples fotogramas	273.50	300.46	522.22	271.57	91	-10
Generación del vocabulario de palabras visuales	108.97	27.32	7 575.42	2 296.57	6 852	8 307
Generación del histograma	5.01	2.95	274.36	68.68	5 380	2 231
Ajuste del parámetro "función kernel" por validación cruzada	245.91	246.48	503.13	504.23	105	105
Ajuste del parámetro "regulador C" por validación cruzada	753.19	754.87	1 754.26	1 758.11	133	133
Entrenamiento del modelo usando parámetros definidos	13.92	13.95	10.39	10.41	-25	-25
Clasificación	15.60	15.64	12.34	12.36	-21	-21
Total fase de entrenamiento	1 416.10	1 361.66	10 652.11	4 921.93	652	261
Total fase de nuevas predicciones	294.11	319.04	808.92	352.61	175	11

Tabla 4.6: Resultados de rendimiento en el tiempo en la computadora 2

Por último, cabe resaltar que las mediciones de tiempo dependen completamente de la configuración de la computadora sobre las que se ejecuten, incluyendo sus procesos activos. Esto implica que los resultados de tiempo presentados en este trabajo no son repetibles, sin embargo, sirven como una guía para entender y comparar la cantidad de operaciones que se realizan por cada etapa del método.

4.3. Resultados de rendimiento en almacenamiento

Se muestran los resultados de la cantidad de almacenamiento que ocupan los diferentes archivos generados por el método propuesto y se comparan entre BOF-BoVW y SIFT-BoVW. Los archivos generados son:

1. Descriptores. Todos los descriptores extraídos de la secuencia *seq2* (6 500 fotogramas). Para el caso de BOF-BoVW, se saltaron 39 fotogramas porque estaban

vacíos.

2. Vocabulario de palabras visuales. Para el caso de BOF-BoVW, 256 palabras de tamaño 180; mientras que para SIFT-BoVW, 1 024 palabras de tamaño 128.

3. Modelo entrenado. Ambos SVM y con los parámetros que se mencionaron en la sección *Resultados de clasificación* 4.1, además, la medición incluye al estandarizador de datos (mencionado en la sección *Implementación del método propuesto* 3.3).

Al igual que con las tablas comparativas para rendimiento en el tiempo, se presenta el incremento porcentual del tamaño de los archivos para SIFT-BoVW, tomando como valor inicial la cantidad de almacenamiento que requiere BOF-BoVW.

El espacio de almacenamiento de descriptores revela la cantidad de información que los descriptores conservan de las imágenes originales, y a su vez, permite observar la cantidad de compresión de datos que se ha realizado.

El vocabulario de palabras visuales y el modelo se ocupan para poder realizar nuevas predicciones.

Los resultados de almacenamiento se muestran en la tabla 4.7.

	BOF-BoVW	SIFT-BoVW	Incremento
Descriptores	27.767 MB	2.479 GB	6 716 %
Vocabulario de palabras visuales	368.768 kB	524.416 kB	42 %
Modelo entrenado	5.540 MB	10.890 MB	97 %

Tabla 4.7: Resultados de rendimiento en almacenamiento

4.4. Discusión de los resultados

El objetivo general del trabajo de investigación es la implementación de un método de reconocimiento de escenas usando un descriptor liviano con el propósito de ser una alternativa de bajo coste computacional a los métodos de extracción de características locales existentes en el estado del arte. BOF-BoVW es la alternativa de bajo coste computacional y es comparado con SIFT-BoVW que es una de las opciones que existen en el estado del arte.

4.4.1. Resultados de clasificación

Los resultados de BOF-BoVW usando el método de retención con el 25 % de datos de prueba, muestran una exactitud de clasificación del 93 %, con un valor F1 macro

promedio ponderado del 93 % (expuestos en la tabla 4.1), así mismo, la matriz de confusión (figura 4.1) muestra pocas predicciones falsas. Esto en un principio indica que se tiene un modelo altamente preciso, sin embargo, los resultados cambian abruptamente cuando se evalúa el modelo con la secuencia *seq2*, en donde la exactitud de clasificación cae a 26 %, y el valor F1 macro promedio ponderado a 25 % (expuestos en la tabla 4.2), confirmando este resultado con la matriz de confusión (figura 4.2) que muestra un alto número de predicciones falsas. La diferencia de resultados entre ambos métodos de evaluación puede ser un indicador de que el modelo está sobreajustado, pues, aunque la evaluación con el método de retención se ha realizado con datos no vistos por el modelo y esta presenta buenos resultados, no se ha logrado una generalización capaz de obtener una predicción adecuada de secuencias distintas de la misma escena.

La evaluación de SIFT-BoVW usando el método de retención con el 25 % para datos de prueba indica un modelo perfecto, logrando un 100 % en todas las métricas que muestra el reporte de clasificación (tabla 4.3), además, en la matriz de confusión (figura 4.3) se observa que existe una sola predicción falsa. Igualmente se evalúa a SIFT-BoVW con la secuencia *seq2*, en donde las métricas también caen, pero no tan abruptamente como en BOF-BoVW. En este caso se tiene una exactitud de clasificación del 85 %, con un valor F1 macro promedio ponderado del 82 % (mostrados en la tabla 4.4). La matriz de confusión (figura 4.4) muestra que para la mayoría de las categorías de escenas se está teniendo una clasificación correcta, con excepción de la categoría "*stairs*", donde se confunde principalmente con la categoría "*heads*", lo cual puede deberse a que la categoría "*stairs*" tiene la mitad del soporte que el resto, y revela que el desequilibrio de clases afecta al aprendizaje de las clases minoritarias. La combinación de los resultados en ambas evaluaciones hace a SIFT-BoVW un buen modelo, que incluso generaliza a diferentes secuencias de la misma escena.

4.4.2. Resultados de rendimiento en el tiempo y almacenamiento

Se espera que BOF-BoVW sea además una alternativa de bajo coste computacional frente a SIFT-BoVW, es por ello por lo que se realiza una evaluación del rendimiento en el tiempo y almacenamiento de ambos métodos.

En este análisis se refiere siempre a los resultados de la computadora 1 (tabla 4.5), puesto que los de la computadora 2 (tabla 4.6) siguen la misma tendencia, y cambia en proporción a la capacidad de cómputo de la propia computadora.

Para la evaluación de tiempo "extracción de descriptores de un fotograma", en el tiempo de CPU, se muestra un amplio mejor rendimiento de BOF con aproximadamente 88 ms, mientras que SIFT se va a los 548 ms, lo cual implica un incremento de aproximadamente el 522 %. Si se toma en cuenta el tiempo real no existe mucha diferencia, 88 ms de BOF contra 78 ms de SIFT, de hecho, hay un decremento del 11 %. Este último resultado se debe a la implementación optimizada de SIFT en OpenCV [30].

Un resultado similar ocurre con la evaluación de tiempo "extracción de descriptores de múltiples fotogramas", con 01:20 (1 minuto con 20 segundos) para BOF y 07:42 para

SIFT en tiempo de CPU, lo que representa un incremento del 579 %. Estos resultados demuestran una amplia ventaja en el ahorro de recursos por parte de BOF frente a SIFT.

La eficiencia de BOF se debe en gran parte a la cantidad de descriptores que se extraen por fotograma. En promedio, para BOF se tienen 3 descriptores por fotograma, mientras que para SIFT se tienen 763 descriptores por fotograma. Eso sumado a la cantidad de operaciones que realiza SIFT para obtener cada descriptor frente a las que realiza BOF.

Siguiendo con la evaluación de tiempo "generación del vocabulario de palabras visuales", se tiene 01:04 para BOF-BoVW, mientras que 1:19:24 (1 hora, 19 minutos, 24 segundos) para SIFT-BoVW en tiempo de CPU, lo que representa un incremento exorbitante del 7 313 %, siendo también la etapa que mayor tiempo toma en todo el método. Este resultado se debe principalmente a la cantidad de puntos que procesa el algoritmo k-medias; para BOF-BoVW 14 484 muestras con 256 centroides, mientras que para SIFT-BoVW, 150 000 muestras con 1 024 centroides. Afortunadamente, la implementación de k-medias está optimizada [32] y el tiempo real para ambos casos es bastante menor, resaltando que, a pesar de ello, los 12:33 reales de SIFT-BoVW sigue siendo un tiempo alto por considerar en el marco de trabajo.

Para la evaluación de tiempo "generación de los histogramas", se tienen 5 s para BOF-BoVW, mientras que 03:26 para SIFT-BoVW en tiempo de CPU, lo que representa un incremento del 4 012 %, igualmente enorme como en la evaluación anterior. De forma similar, se debe a la cantidad de centroides y descriptores involucrados en la generación de los histogramas.

Las diferencias continúan en las etapas de ajuste de parámetros por validación cruzada; en el ajuste de la función kernel, se tienen 29 s contra 01:35 en tiempo de CPU para BOF-BoVW y SIFT-BoVW respectivamente, con un incremento del 225 %, mientras que para el ajuste del regulador C se tiene 01:16 contra 05:15 en tiempo de CPU para BOF-BoVW y SIFT-BoVW respectivamente, con un incremento de 314 %. En este caso podría deberse al tamaño de los datos de entrada a SVM y la distribución de estos, siendo vectores de tamaño 256 para BOF-BoVW, mientras que de tamaño 1 024 para SIFT-BoVW.

En la evaluación del entrenamiento de SVM con parámetros definidos se tiene 1.25 s contra 1.93 s en tiempo de CPU para BOF-BoVW y SIFT-BoVW respectivamente, con un incremento del 54 %.

En la evaluación del tiempo de clasificación se tienen resultados similares para ambos métodos, 2.28 s contra 2.20 s en tiempo de CPU para BOF-BoVW y SIFT-BoVW respectivamente, teniendo un decremento de apenas 4 %.

Para hacer una comparación que resuma los resultados de rendimiento en el tiempo, se hace la suma para las fases de entrenamiento y de nuevas predicciones (fases explicadas con mayor detalle en la sección *Diagrama del método propuesto* 3.4).

En cuanto al total de la fase de entrenamiento, se tiene, en tiempo de CPU, 04:18 para BOF-BoVW, contra 1:37:26 para SIFT-BoVW, lo que equivale a un incremento del 2 167 %. En tiempo real se tiene 03:14 para BOF-BoVW, contra 21:09 de SIFT-BoVW,

siendo un incremento del 554 %. Cabe resaltar que, aunque la fase de entrenamiento se realiza una sola vez, la diferencia entre tiempos de ejecución es bastante importante.

Para la fase de nuevas predicciones, en tiempo de CPU, se tiene 01:27 para BOF-BoVW, contra 11:10 para SIFT-BoVW, lo que equivale a un incremento del 670 %. En tiempo real se tiene 01:19 para BOF-BoVW, contra 01:42 de SIFT-BoVW, siendo un incremento del 29 %. La fase de nuevas predicciones es la fase que se llevará a cabo con mayor frecuencia, y en esta, BOF-BoVW resulta tener ventaja incluso tomando en cuenta el tiempo real.

En cuanto a los resultados de almacenamiento (tabla 4.7), para el vocabulario de palabras visuales se tiene un incremento del 42 % de BOF-BoVW frente a SIFT-BoVW, sin embargo, es despreciable pues está en el orden de los kB, de igual forma para el modelo entrenado, con un incremento del 97 % de BOF-BoVW frente a SIFT-BoVW, pero en este caso en el orden de los MB. La diferencia significativa está en el tamaño de los descriptores, pues para BOF-BoVW se tiene 27.767 MB, mientras que para SIFT-BoVW se dispara a 2.479 GB, diferencia exorbitante que debe tenerse en cuenta, especialmente en computadoras de factor reducido, pues en caso de necesitar cargarse en memoria se requerirían de al menos 3 GB disponibles para las muestras presentadas.

Todos los resultados en su conjunto introducen una solución de compromiso (en inglés, *trade-off*), en donde debe tenerse en cuenta si se requiere una clasificación exacta y confiable a costa de sacrificar recursos de cómputo (en este caso, elegir SIFT-BoVW), o bien, cuidar al máximo los recursos computacionales a costa de tener una clasificación con una mayor tasa de error (en este caso, elegir BOF-BoVW).

Capítulo 5

Conclusiones

En este trabajo se desarrolló un método para el reconocimiento de escenas que prioriza el ahorro de recursos computacionales, mismo que fue nombrado como BOF-BoVW debido a sus dos principales etapas: extracción de características usando BOF y transformación de características usando BoVW. La etapa de extracción de características BOF se realizó a partir de nubes de puntos de las escenas, nubes de puntos que son generadas a partir de imágenes de profundidad, las cuales a su vez son obtenidas mediante sensores RGB-D.

SIFT es un método extracción de características locales robusto que genera descripciones de puntos relevantes dentro de una imagen, sin embargo, es un método que implica un alto gasto de recursos computacionales. En el estado del arte se ha demostrado que los métodos de extracción de características locales aportan buenos resultados de clasificación, por ejemplo, en [16], [21], y [22]. Se propuso el método SIFT-BoVW para comparar los resultados de clasificación y rendimiento obtenidos con BOF-BoVW. La única diferencia entre los métodos comparados es la etapa de extracción de características, pues ambos aplican BoVW en la transformación de características y SVM en la clasificación.

Uno de los aportes más importantes de este trabajo al estado del arte es justamente la etapa de extracción de características BOF. Otros autores han utilizado el descriptor BOF principalmente para la descripción figuras que están bien definidas (por ejemplo, en [12] se utiliza para detectar un conjunto de piezas de ensamblaje, mientras que en [1] se detecta una portería), por el contrario, en este trabajo la cantidad de figuras descritas es muy dispersa, ya que se buscó identificar a los diferentes objetos que están presentes en cada uno de los fotogramas de las escenas. Para resolver este reto se recurrió a realizar cortes en la nube de puntos de tal forma que cada corte sea una capa que contiene la segmentación de los objetos que están presentes en la escena. Una vez detectados los objetos, se extraen sus contornos y su descripción tal cual otros autores lo han realizado.

Utilizando el método de retención (ver subsección 2.5.1.1), con 25 % de datos de prueba, se obtuvo una exactitud de clasificación del 93 % para BOF-BoVW, mientras que de 100 % para SIFT-BoVW, sin embargo, cuando se evalúan con secuencias distintas

a las de entrenamiento, se obtiene una exactitud de clasificación del 26 % y 85 % para BOF-BoVW y SIFT-BoVW respectivamente.

Realizando mediciones del tiempo que tarda en ejecutar la fase de entrenamiento, en tiempo de uso de CPU (ver sección 4.2), se encontró que SIFT-BoVW resulta en un incremento del 2 167 % con respecto a BOF-BoVW. En tiempo real y para la misma fase, SIFT-BoVW resulta en un incremento del 554 % con respecto a BOF-BoVW.

Para el tiempo que toma la fase de nuevas predicciones, en tiempo de uso de CPU, se encontró que SIFT-BoVW resulta en un incremento del 670 % con respecto a BOF-BoVW. En tiempo real y para la misma fase, SIFT-BoVW resulta en un incremento del 29 % con respecto a BOF-BoVW.

Otro resultado a destacar es el espacio de almacenamiento de los descriptores, pues SIFT-BoVW resulta en un incremento del 6 716 % con respecto a BOF-BoVW.

En cuanto a la comparación de los métodos, los resultados indican que SIFT-BoVW es mucho mejor generalizando a secuencias diferentes de las mismas categorías de escenas, pero cuya etapa de extracción de características resulta en un alto costo tanto en tiempo como en almacenamiento. Por otro lado, el método BOF-BoVW no es capaz de generalizar a secuencias diferentes, pero presenta buenos resultados detectando fotogramas dentro de un recorrido similar al de entrenamiento, además, su principal ventaja es que permite un ahorro de recursos computacionales bastante importante.

La hipótesis de investigación (ver sección 1.4) se comprobó, pues BOF-BoVW resulta en una alternativa de bajo coste computacional frente a SIFT-BoVW. Al mismo tiempo se introduce una solución de compromiso, en la que se somete a evaluación la importancia de la exactitud de clasificación frente al ahorro de recursos de cómputo, pues cada uno de los métodos comparados sacrifica un aspecto por mejorar otro.

Una de las limitaciones que presentó este trabajo fue encontrar un conjunto de datos adecuado que permitiera comparar los métodos de reconocimiento de escenas presentados. Se utilizó el conjunto de datos *Microsoft 7 scenes RGB-D dataset* [26] porque incorpora la información de profundidad y las clases que incluye presentan baja ambigüedad de etiquetado (ver subsección 2.2.2). No obstante, presenta algunos problemas, como el que las secuencias se hayan grabado a mano, pues no es consistente el ángulo desde el que se capturan los objetos. Si el ángulo de la cámara fuera constante podrían eliminarse capas de puntos que introducen ruido, como lo son el techo o el piso. Otro problema que presenta el conjunto de datos es el desbalanceo de la clase "*stairs*", pues afecta en el aprendizaje de las clases minoritarias.

Los objetivos, general y específicos (ver sección 1.3) se cumplieron satisfactoriamente.

Se concluye que el método BOF-BoVW puede resultar útil en escenarios en dónde se tienen recursos de cómputo limitados, en dónde, además se realizan recorridos o secuencias similares.

5.1. Trabajo a futuro

A partir de los resultados obtenidos se han detectado áreas de oportunidad que pueden ayudar a mejorar el método:

1. **La creación de un conjunto de datos RGB-D adecuado**. Realizar las grabaciones de secuencias en clases de escenas bien definidas, con un número de muestras balanceado entre clases, y con una plataforma móvil con el objetivo de que el ángulo de captura sea constante y poder eliminar puntos irrelevantes del techo y el piso. Posteriormente evaluar los métodos BOF-BoVW y SIFT-BoVW bajo este nuevo conjunto de datos.

2. **Comparar los métodos con uno basado en Aprendizaje Automático Profundo**. En los últimos años se ha tenido un amplio desarrollo de las redes neuronales profundas, y su uso ha acaparado los métodos de aprendizaje automático gracias a que presenta resultados de clasificación excelentes. La principal desventaja del aprendizaje automático profundo es que se necesitan datos masivos para su entrenamiento. Sería importante realizar una comparación de métricas de clasificación y rendimiento en el tiempo y almacenamiento para evaluar la conveniencia de cada uno de los métodos.

3. **Comparar los métodos de extracción de características propuestos con aquellos existentes en el estado del arte que son eficientes en tiempo y almacenamiento**. Realizar la comparación de BOF, con los métodos de extracción de características locales SURF y ORB, pues estos últimos fueron concebidos como una simplificación de SIFT, que mejoran la eficiencia de recursos computacionales. En este trabajo no se seleccionaron estos métodos porque SURF es un método con patente vigente [25], y ORB está basado en cadenas binarias [24], lo que complica la implementación de k-medias para la generación del vocabulario de palabras visuales, pues una cadena binaria no puede ser comparada utilizando la distancia euclidiana.

4. **Optimización de la extracción de características BOF y paralelización de procesos**. Buscar alternativas dentro de la implementación del método de extracción de características BOF para mejorar aún más su rendimiento en el tiempo. Entre estas mejoras se sugiere utilizar un lenguaje de programación compilado que permita la ejecución de los procesos con mayor rapidez que el utilizado en este proyecto. También se sugiere encontrar e implementar aquellos procesos que se pueden realizar en paralelo para aprovechar al máximo las capacidades de los procesadores multinúcleo modernos.

5. **Implementación de la etapa de extracción de características BOF en un esquema de SLAM Visual**[1]. Los resultados de rendimiento en tiempo y almacenamiento de BOF hacen atractivo intentar su implementación en alguna de las etapas de SLAM Visual, pues la optimización de los sistemas de navegación autónoma es una tarea aún vigente.

[1]SLAM (del inglés, *Simultaneous Localization and Mapping*) se refiere al método que utiliza un robot o cuerpo rígido móvil, equipado con un sensor específico, para estimar su movimiento y construir un modelo (algún tipo de descripción) del ambiente que lo rodea sin necesidad de información previa. Si el sensor que utiliza es una cámara, al método se le conoce como SLAM Visual. Ver prefacio de [9].

Bibliografía

[1] V. Lomas-Barrie, M. Peña-Cabrera, and J. Durán-Ortega, "Determining humanoid soccer player position on goal detection," in *Proceedings on the International Conference on Artificial Intelligence (ICAI)*. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2015, p. 61.

[2] R. C. Gonzalez and R. E. Woods, *Digital image processing*. Pearson/Prentice Hall.

[3] A. Kadambi, A. Bhandari, and R. Raskar, *3D Depth Cameras in Vision: Benefits and Limitations of the Hardware*. Cham: Springer International Publishing, 2014, pp. 3–26. [Online]. Available: https://doi.org/10.1007/978-3-319-08651-4_1

[4] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[5] E. Pérez-Careta, J.-R. Guzmán-Sepúlveda, J.-M. Lozano-García, M. Torres-Cisneros, and R. Guzmán-Cabrera, "Clasificación de imágenes médicas mediante aprendizaje automático. (spanish)." *DYNA - Ingeniería e Industria*, vol. 97, no. 1, pp. 35 – 38.

[6] L. Xie, F. Lee, L. Liu, K. Kotani, and Q. Chen, "Scene recognition: A comprehensive survey," *Pattern Recognition*, vol. 102, p. 107205, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S003132032030011X

[7] L. Wang, S. Guo, W. Huang, Y. Xiong, and Y. Qiao, "Knowledge guided disambiguation for large-scale scene classification with multi-resolution CNNs," *IEEE Transactions on Image Processing*, vol. 26, no. 4, pp. 2055–2068, 2017.

[8] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[9] X. Gao and T. Zhang, *Introduction to Visual SLAM : From Theory to Practice*. Springer Nature Singapore.

[10] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, Nov 2004. [Online]. Available: https://doi.org/10.1023/B:VISI.0000029664.99615.94

[11] ——, "Method and apparatus for identifying scale invariant features in an image and use of same for locating an object in an image," Mar 2004, US Patent No. US6711293B1.

[12] M. Peña-Cabrera, I. Lopez-Juarez, R. Rios-Cabrera, and J. Corona-Castuera, "Machine vision approach for robotic assembly," *Assembly Automation*, vol. 25, no. 3, pp. 204–216, Jan 2005. [Online]. Available: https://doi.org/10.1108/01445150510610926

[13] M. Peña-Cabrera, I. Lopez-Juarez, and R. Rios-Cabrera, "A learning approach for on line object recognition tasks," in *Proceedings of the Fifth Mexican International Conference in Computer Science, 2004. ENC 2004.*, 2004, pp. 242–248.

[14] D. Todorovic, "Gestalt principles," *Scholarpedia*, vol. 3, no. 12, p. 5345, 2008, revision #91314.

[15] M. Peña-Cabrera, M. Ontiveros, R. Osorio, G. Lefranc, and V. Lomas, "Contourn descriptor generator algorithm implemented in embedded system," in *2017 CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON)*, 2017, pp. 1–8.

[16] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *Workshop on statistical learning in computer vision, ECCV*, vol. 1, no. 1-22. Prague, 2004, pp. 1–2.

[17] Z.-H. Zhou, *Machine Learning*. Springer Nature Singapore.

[18] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, may 2011. [Online]. Available: https://doi.org/10.1145/1961189.1961199

[19] C.-W. Hsu and C.-J. Lin, "A comparison of methods for multiclass support vector machines," *IEEE Transactions on Neural Networks*, vol. 13, no. 2, pp. 415–425, 2002.

[20] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[21] Y.-q. LI, B.-d. XU, and H.-d. SHENG, "Bag-of-visual-words model for image classification based on spatial semantic distribution," in *2018 International Conference on Security, Pattern Analysis, and Cybernetics (SPAC)*, 2018, pp. 42–46.

[22] J. Gangrade, J. Bharti, and A. Mulye, "Recognition of indian sign language using ORB with bag of visual words by kinect sensor," *IETE Journal of Research*, vol. 68, no. 4, pp. 2953–2967, 2022. [Online]. Available: https://doi.org/10.1080/03772063.2020.1739569

[23] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-Up Robust Features (SURF)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008, similarity Matching in Computer Vision and Multimedia. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1077314207001555

[24] 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.

[25] R. Funayama, H. Yanagihara, L. Van Gool, T. Tuytelaars, and H. Bay, "Robust interest point detector and descriptor," Mar 2014, US Patent No. US8670619B2.

[26] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon, "Scene coordinate regression forests for camera relocalization in RGB-D images," in *Proc. Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2013. [Online]. Available: https://www.microsoft.com/en-us/research/publication/scene-coordinate-regression-forests-for-camera-relocalization-in-rgb-d-images/

[27] B. Jähne, *Digital Image Processing: Concepts, Algorithms, and Scientific Applications*. Springer Berlin Heidelberg, 1993.

[28] Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," *Proceedings of the IEEE*, pp. 1–20, 2023.

[29] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. Fernández del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, p. 357–362, 2020.

[30] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[31] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," *arXiv:1801.09847*, 2018.

[32] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors,

"SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[33] J. D. Hunter, "Matplotlib: A 2D graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.

[34] T. Kluyver, B. Ragan-Kelley, F. Pérez, B. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. Hamrick, J. Grout, S. Corlay, P. Ivanov, D. Avila, S. Abdalla, C. Willing, and J. development team, "Jupyter notebooks - a publishing format for reproducible computational workflows," in *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, F. Loizides and B. Scmidt, Eds. Netherlands: IOS Press, 2016, pp. 87–90. [Online]. Available: https://eprints.soton.ac.uk/403913/