Trabajo Fin de Grado
Grado en Ingeniería Electrónica, Robótica y Mecatrónica

Clasificación de imágenes de eventos empleando descriptores basados en frecuencia

Autor: Francisco Javier Gañán Onieva
Tutores: José Ramiro Martínez de Dios
Raúl Tapia López

Dpto. de Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2021

Trabajo Fin de Grado
Ingeniería Electrónica, Robótica y Mecatrónica

Clasificación de imágenes de eventos empleando descriptores basados en frecuencia

Autor:
Francisco Javier Gañán Onieva

Tutores:
José Ramiro Martínez de Dios
Raúl Tapia López

Dpto. de Ingeniería de Sistemas y Automática
Grupo de Robótica, Visión y Control
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2021

Trabajo Fin de Grado: Clasificación de imágenes de eventos empleando descriptores basados en frecuencia

Autor: Francisco Javier Gañán Onieva

Tutores: José Ramiro Martínez de Dios
Raúl Tapia López

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

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2021

El Secretario del Tribunal

A la experiencia y sabiduría de Ramiro;

a la inteligencia e intuición de Raúl;

al trabajo de Juan Antonio,

y al apoyo de Isa.

Resumen

Las cámaras de eventos tienen un gran potencial en robótica y visión por computador. Sin embargo, su incorporación precisa del desarrollo de nuevas formas de procesar los eventos, pues estos presentan la información de forma fundamentalmente distinta a las cámaras tradicionales. En este trabajo se clasifican imágenes de eventos y se defiende la efectividad de emplear para ello descriptores basados en frecuencia.

El método propuesto comprende cuatro etapas: agrupación de eventos, descripción de la imagen de eventos, reducción de los datos y clasificación. En el presente trabajo de fin de grado se realiza una comparativa entre dos descriptores globales, evaluando sus resultados en términos de acierto en la clasificación, mediante máquinas SVM, y velocidad. También se estudian los casos en los que no se incluyen las etapas de descripción y/o reducción. Todo lo anterior se aplica para clasificar las imágenes de eventos generadas durante los vuelos de un robot ornitóptero en tres escenarios diferentes.

Abstract

Event cameras have great potential in robotics and computer vision. However, their incorporation requires the development of novel methods to process events, as they present information in a fundamentally different way compared with traditional cameras. This paper classifies event images and argues for the effectiveness of using frequency-based descriptors.

The proposed method comprises four stages: event clustering, event image description, data reduction and classification. In this Degree Final Project, a comparison is made between two global descriptors, evaluating their results in terms of classification accuracy, using Support Vector Machines, and speed. Cases in which the description and/or reduction stages are not included are also studied. All the above is applied to classify the event images generated during the flights of an ornithopter robot in three different scenarios.

Índice

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

1 INTRODUCCIÓN

Actualmente, la visión basada en eventos es un tema en investigación. El reto consiste en desarrollar formas de procesar la salida de estos nuevos sensores, conocidos como cámaras de eventos, que presentan numerosas ventajas en muchas aplicaciones. En robótica, concretamente, estas cámaras están en auge. Algunos de los mejores grupos de investigación, de diferentes universidades, están incorporándolas en sus robots y grandes empresas privadas también han invertido en estos nuevos sensores.

El presente trabajo de fin de grado presenta un método de clasificación de imágenes generadas a partir de eventos, y demuestra la efectividad de emplear descriptores basados en frecuencia en el proceso. Para probar la validez del método, se han clasificado los eventos generados durante diferentes vuelos de un ornitóptero, o robot aéreo de ala batiente (Figura 1.1). Además, el método desarrollado en este proyecto es válido para su implementación en tiempo real, ya que la duración de un ciclo, en el que sucede el proceso completo, para cada imagen, es del orden del milisegundo.

En concreto, el método desarrollado se centra en las etapas de reducción del descriptor de la imagen de eventos y en su clasificación. El esquema general, que incluye las distintas etapas o bloques del método, se muestra en la Figura 1.2.

Figura 1.1 Prototipo de ornitóptero equipado con una cámara de eventos DAVIS 346, desarrollado por la Universidad de Sevilla. Fuente: [25].

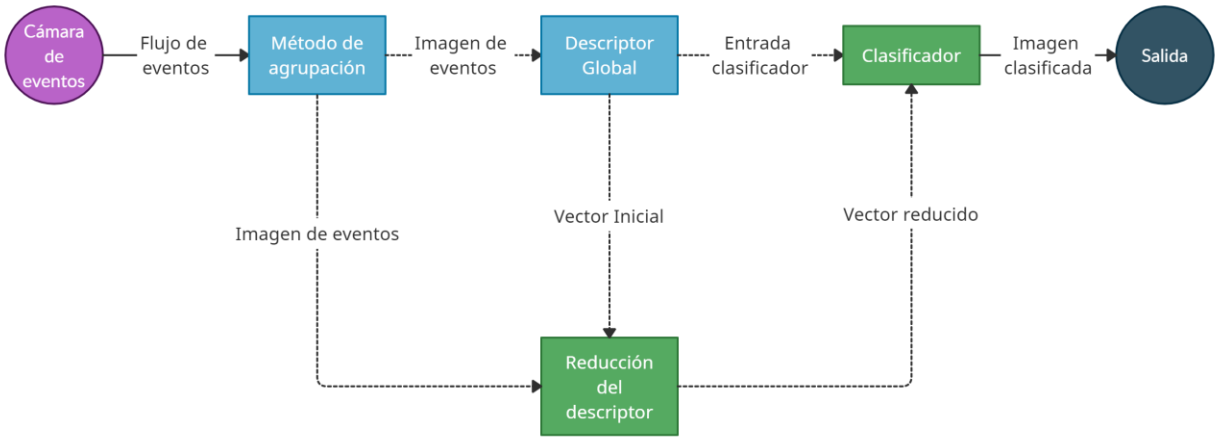

Figura 1.2 Esquema de las etapas desarrolladas en el método, desde que se genera la imagen de eventos hasta que se clasifica.

1.1 Objetivos

Los objetivos que persigue este proyecto son:

- Demostrar experimentalmente la efectividad de los descriptores globales basados en frecuencia, aplicados a clasificación de imágenes de eventos.
- Conseguir tiempos de ejecución adecuados para su implementación en tiempo real.
- Analizar los resultados de aplicar o no descripción global a la imagen y/o reducir la dimensión de la imagen o de su descriptor.
- Clasificar correctamente los eventos procedentes de la cámara de eventos a bordo del ornitóptero.

1.2 Contribución

En visión basada en eventos, se conocen pocos trabajos que incluyan descriptores basados en técnicas frecuenciales en el procesamiento de los eventos. Estos presentan muchas ventajas para procesarlos. En concreto, en este documento se resaltan aquellas que ofrecen mejoras en la clasificación, mediante comparativas.

La clasificación de los eventos es un tema en investigación, por lo que se desarrollan cada vez más métodos para cumplir esta misión. Además, las técnicas de aprendizaje automático cada vez son más potentes y su desarrollo está teniendo un gran impacto en la robótica. No obstante, muchas de las más novedosas requieren de hardware específico (como procesadores neuromórficos, aún en investigación y desarrollo) para poder ser implementadas o funcionar de manera eficiente. Estos dispositivos computacionales, además, no están diseñados para ser programados mediante el enfoque tradicional.

Un robot es un sistema complejo, en el que la clasificación es solo una pequeña parte, y necesita implementar y coordinar diferentes módulos para su correcto funcionamiento. Además, en robótica aérea, la carga útil del robot está muy limitada, por lo que es preciso que el dispositivo hardware seleccionado encuentre un equilibrio entre:

- Capacidad para alojar todos los módulos del sistema, permitiendo su coordinación, interacción, y correcto funcionamiento.
- Poca carga útil del robot aéreo.

Nuestro método propone un clasificador eficiente en memoria, rápido e implementable en dispositivos hardware tradicionales, lo que lo convierte en un interesante candidato a ser incorporado en robots aéreos.

1.3 Marco de realización

Este trabajo de fin de grado se ha realizado gracias a la colaboración de miembros del grupo GRVC (Grupo de Robótica, Visión y Control) de la Universidad de Sevilla, que han proporcionado los *datasets* analizados en este proyecto. Estos fueron generados en el marco del proyecto internacional GRIFFIN, y presentados en [26].

Hardware

Para la obtención de las imágenes de eventos utilizadas en este proyecto se empleó la cámara DAVIS346 (Figura 1.3).

Figura 1.3 Cámara de eventos DAVIS346.

Software

La implementación del método se ha llevado a cabo en dos lenguajes de programación, Python y MATLAB. Se han utilizado las versiones Python 3.9.6 y MATLAB R2020a. Los programas se encuentran disponibles en ambos lenguajes.

En Python se han utilizado las librerías OpenCV [27] y scikit-learn [28]. OpenCV es una librería *open source* y multiplataforma que ofrece numerosas funciones relacionadas con la percepción artificial. Scikit-learn permite implementar algoritmos de aprendizaje automático (*Machine Learning*) en Python.

En MATLAB se han utilizado diferentes *toolboxes* [29], como Image Processing Toolbox, que permite procesar, visualizar y analizar imágenes; o Statistics and Machine Learning Toolbox, que ofrece funciones para analizar y modelar datos con estadística y aprendizaje automático.

1.4 Estructura

El presente trabajo de fin de grado se compone de 6 capítulos. El Capítulo 2 hace una revisión del estado del arte de las distintas técnicas y métodos existentes en cada una de las etapas involucradas en este proyecto. En el Capítulo 3 se describen las técnicas y métodos escogidos en cada etapa. En el Capítulo 4 se realiza un análisis de los *datasets* disponibles, para extraer conclusiones importantes que permitan implementar el método más adecuado. En el Capítulo 5 se evalúan los resultados de los experimentos realizados, y se compara el desempeño de los diferentes descriptores en la clasificación. En el Capítulo 6 se presentan las conclusiones extraídas durante la realización de este proyecto, y se comentan posibles desarrollos futuros.

2 ESTADO DEL ARTE

En este capítulo se describirán las distintas áreas de investigación y técnicas involucradas en este trabajo, prestando especial atención a los métodos que se han escogido para implementar la solución.

Las cámaras de eventos son sensores en creciente auge y desarrollo, y el procesamiento de su salida, conocida como eventos, es un tema en investigación. En esta sección se explicará su funcionamiento y se mostrarán algunos prototipos comerciales. Se comentarán sus ventajas y aplicaciones. Además, se explicarán algunos de los métodos y técnicas empleados para el procesamiento de eventos, en actual desarrollo debido a su reciente aparición.

Los métodos de extracción de características (*features*), en imágenes tradicionales, se centran en identificar propiedades relevantes de las imágenes, mediante lo que conocemos como descriptores. Existen dos tipos de descriptores: locales y globales. En este capítulo se describirá brevemente cada tipo de descriptor, y se presentarán algunos descriptores globales.

Existen numerosos métodos para solucionar el conocido problema de clasificación. En este capítulo se introducirá el concepto de aprendizaje automático y se hará una distinción entre algoritmos de aprendizaje supervisado y no supervisado. También se describirán ejemplos de cada grupo de algoritmos, mencionando algunas aplicaciones.

2.1 Cámaras de eventos

Las cámaras de eventos son esencialmente distintas a las cámaras convencionales [1]. El hecho de que sean llamadas cámaras es porque también perciben ondas en el espectro electromagnético, concretamente en la franja de luz visible. La diferencia y característica fundamental de este sensor es que, en lugar de percibir imágenes a una velocidad fijada por un reloj externo, mide asíncrona e independientemente cambios de intensidad (de magnitud predefinida) en cada píxel. La salida de este sensor es una cadena de eventos, donde cada uno aporta información sobre:

- La coordenada $\boldsymbol{x}_k = (x_k, y_k)$ del píxel, en el plano de la imagen, donde sucedió el cambio;
- el instante de tiempo t_k;
- el signo (positivo o negativo) de cada cambio local de intensidad p_k.

Las cámaras de eventos ofrecen numerosas ventajas frente a las cámaras tradicionales:

- Alta resolución temporal: Los eventos son detectados y registrados con resolución de microsegundos. Esto hace que estos nuevos sensores no sufran el típico desenfoque por movimiento de las cámaras basadas en fotogramas, pudiendo detectar movimientos muy rápidos.
- Baja latencia: Cada píxel trabaja de forma independiente, por lo que en cuanto detecta un cambio, lo transmite. Esto hace que la latencia de las cámaras de eventos esté por debajo del milisegundo [1].
- Bajo consumo: Como las cámaras de eventos transmiten solo cambios de luminosidad, eliminando información redundante, la energía únicamente se emplea en procesar los píxeles que cambian su valor de intensidad.
- Gran rango dinámico: En imágenes, se refiere a la diferencia de intensidades entre la parte más clara y más oscura de la imagen. En estas nuevas cámaras, su rango es mayor de 120 dB, lo que excede con

creces los 60 dB de las mejores cámaras basadas en fotogramas [1]. Esta característica es importante en aplicaciones en las que el sensor debe adaptarse a estimulos tanto muy liminosos como muy oscuros.

Los diseños más comunes de cámaras de eventos son los que se implementan en: el sensor DVS [2], el sensor ATIS (Asynchronous Time Based Image Sensor) [3] y, el más extendido y reciente, el sensor DAVIS [4].

A continuación, se muestra, en la Figura 2.1, una comparación de las cámaras más populares y recientes. Es importante destacar que los números en ella son aproximados, debido a que las medidas no fueron resultado de un banco de pruebas común (por su reciente aparición).

	Supplier	iniVation			Prophesee				Samsung			CelePixel		Insightness
	Camera model	DVS128	DAVIS240	DAVIS346	ATIS	Gen3 CD	Gen3 ATIS	Gen 4 CD	DVS-Gen2	DVS-Gen3	DVS-Gen4	CeleX-IV	CeleX-V	Rino 3
Sensor specifications	Year, Reference	2008 [2]	2014 [4]	2017	2011 [3]	2017 [67]	2017 [67]	2020 [68]	2017 [5]	2018 [69]	2020 [39]	2017 [70]	2019 [71]	2018 [72]
	Resolution (pixels)	128 × 128	240 × 180	346 × 260	304 × 240	640 × 480	480 × 360	1280 × 720	640 × 480	640 × 480	1280 × 960	768 × 640	1280 × 800	320 × 262
	Latency (µs)	12μs @ 1klux	12μs @ 1klux	20	3	40 - 200	40 - 200	20 - 150	65 - 410	50	150	10	8	125μs @ 10lux
	Dynamic range (dB)	120	120	120	143	> 120	> 120	> 124	90	90	100	90	120	> 100
	Min. contrast sensitivity (%)	17	11	14.3 - 22.5	13	12	12	11	9	15	20	30	10	15
	Power consumption (mW)	23	5 - 14	10 - 170	50 - 175	36 - 95	25 - 87	32 - 84	27 - 50	40	130	-	400	20-70
	Chip size (mm^2)	6.3 × 6	5 × 5	8 × 6	9.9 × 8.2	9.6 × 7.2	9.6 × 7.2	6.22 × 3.5	8 × 5.8	8 × 5.8	8.4 × 7.6	15.5 × 15.8	14.3 × 11.6	5.3 × 5.3
	Pixel size (µm^2)	40 × 40	18.5 × 18.5	18.5 × 18.5	30 × 30	15 × 15	20 × 20	4.86 × 4.86	9 × 9	9 × 9	4.95 × 4.95	18 × 18	9.8 × 9.8	13 × 13
	Fill factor (%)	8.1	22	22	20	25	20	> 77	11	12	22	8.5	8	22
	Supply voltage (V)	3.3	1.8 & 3.3	1.8 & 3.3	1.8 & 3.3	1.8	1.8	1.1 & 2.5	1.2 & 2.8	1.2 & 2.8		1.8 & 3.3	1.2 & 2.5	1.8 & 3.3
	Stationary noise (ev/pix/s) at 25C	0.05	0.1	0.1	-	0.1	0.1	0.1	0.03	0.03		0.15	0.2	0.1
	CMOS technology (nm)	350	180	180	180	180	180	90	90	90	65/28	180	65	180
		2P4M	1P6M MIM	1P6M MIM	1P6M	1P6M CIS	1P6M CIS	BI CIS	1P5M BSI			1P6M CIS	CIS	1P6M CIS
	Grayscale output	no	yes	yes	yes	no	yes	no	no	no	no	yes	yes	yes
	Grayscale dynamic range (dB)	NA	55	56.7	130	NA	> 100	NA	NA	NA	NA	90	120	50
	Max. frame rate (fps)	NA	35	40	NA	NA	NA	NA	NA	NA	NA	50	100	30
Camera	Max. Bandwidth (Meps)	1	12	12	-	66	66	1066	300	600	1200	200	140	20
	Interface	USB 2	USB 2	USB 3		USB 3	USB 3	USB 3	USB 2	USB 3	USB 3			USB 2
	IMU output	no	1 kHz	1 kHz	no	1 kHz	1 kHz	no	no	1 kHz	no	no	no	1 kHz

Figura 2.1 Comparación de prototipos comerciales de cámaras de eventos. Fuente: [1].

En los experimentos, los eventos han sido generados mediante el modelo de cámara DAVIS346.

2.2 Procesamiento de eventos

Una de las cuestiones clave en el cambio de paradigma que suponen las cámaras de eventos es cómo extraer información significativa a partir de los eventos, para realizar tareas determinadas. Hoy en día, al ser un tema en auge y desarrollo, la pregunta es abierta, ya que sus diferentes respuestas son dependientes de la aplicación en concreto. Por consiguiente, los algoritmos para procesar eventos, que se están desarrollando actualmente, son un tema en investigación.

Sin embargo, es obvio que la latencia juega un papel muy importante en la forma en la que los eventos son procesados, por lo que se diferencian dos categorías de algoritmos en función del número de eventos que procesan simultáneamente [1]: métodos que operan evento a evento, donde el estado del sistema cambia con la llegada de cada evento, consiguiendo mínima latencia; y métodos que trabajan con grupos o paquetes de eventos, que introducen algo de latencia con respecto a los primeros. Sin embargo, esta distinción es sutil, ya que puede haber tanto métodos que trabajen con grupos de eventos que actualicen el estado del sistema con la llegada de cada evento singular; como métodos evento a evento que obtengan información a partir de grupos de eventos (ya que un solo evento aislado no aporta información significativa).

Por otro lado, otra forma de establecer una clasificación en el marco del procesamiento de eventos puede ser distinguir entre enfoques basados en un modelo o enfoques sin modelo (conceptos como *machine-learning*). Cada categoría presenta sus ventajas y desventajas, y las investigaciones actuales se centran en explorar las posibilidades que ofrece cada método.

2.3 Representación de los eventos

Para resolver una determinada tarea, los eventos son procesados y muchas veces transformados en representaciones alternativas que facilitan la extracción de información relevante. A continuación, se revisan algunas de las formas de representación más populares [1]:

- Eventos individuales: Cada evento $e_k = (\boldsymbol{x}_k, t_k, p_k)$ es usado por métodos evento a evento, como

filtros probabilísticos o SNNs (*Spiking Neural Networks*). Los métodos que usan esta representación disponen de información adicional, bien proveniente de eventos pasados o aportada por otros sensores, que fusionan con los eventos que van llegando de forma asíncrona.

- Paquetes de eventos: Los eventos $E = \{e_k\}_{k=1}^{N_e}$ que se producen en una ventana espaciotemporal son procesados juntos para dar lugar a una salida. Es crítico elegir el número de eventos N_e adecuado para satisfacer los supuestos de cada algoritmo, que son dependientes de la tarea.
- Imágenes de eventos o histogramas 2D: Los eventos en una ventana espacial, temporal o espaciotemporal son convertidos en una imagen bidimensional que puede servir de entrada para algoritmos de visión por computador tradicionales. Un inconveniente de esta representación es que las imágenes generadas son sensibles al número de eventos, aunque, aún así, han tenido un gran impacto en la literatura debido a que: Son una forma simple de convertir una cadena de eventos (información poco familiar) en una representación conocida (una imagen bidimensional) que aporta información sobre los contornos de la escena, que son las regiones más informativas en imágenes naturales; detectan la presencia y ausencia de eventos (o movimiento) en la escena; tienen una interpretación intuitiva; y son estructuras de datos compatibles con los algoritmos convencionales de visión por computador. Por todo lo anteriormente mencionado y algunas ventajas más, que se mencionan en capítulos posteriores, se ha decidido usar imágenes de eventos como punto de partida en este trabajo.
- Superficies temporales: Una superficie temporal es un mapa bidimensional donde cada píxel almacena una valor de tiempo, por ejemplo, el último instante en el que se produjo un evento en ese píxel [5]. De esta manera, los eventos son convertidos en una imagen cuya intensidad es función del historial de movimiento en cada píxel.
- Cuadrícula de píxeles volumétricos (*Voxel Grid*): Es un histograma espaciotemporal tridimensional de eventos, donde cada voxel (*volumetric pixel*) representa un píxel particular en un intervalo de tiempo. Esta representación preserva mejor la información temporal de los eventos que las superficies temporales, ya que no sobreescribe su marca de tiempo en la cuadrícula bidimensional.
- Conjunto de puntos 3D: Los eventos en una ventana espacio-temporal son considerados como puntos en el espacio tridimensional $(x_k, y_k, t_k) \in \mathbb{R}^3$.
- Conjunto de puntos en el plano de la imagen: Los eventos son tratados como un conjunto de puntos 2D en evolución. Es una representación popular en métodos de seguimiento de formas.

En la Figura 2.2 se representa un conjunto de eventos en base a sus coordenadas (x,y,t) y su polaridad.

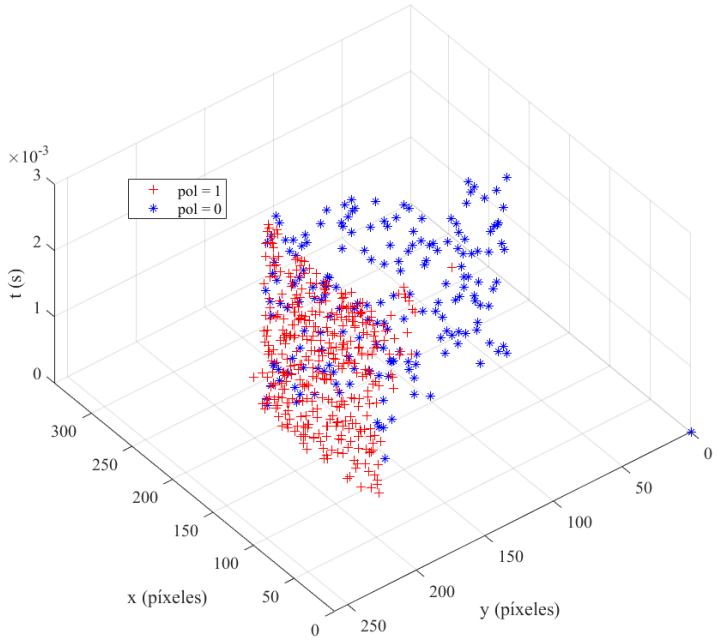

Figura 2.2 Representación espaciotemporal de eventos en función de su polaridad.

2.4 Métodos para procesar eventos

Los sistemas de procesamiento de eventos consisten en tres etapas: Preprocesamiento (adaptación de la entrada), procesamiento central (extracción de características y análisis) y postprocesamiento (generación de la salida). Los métodos de representación presentados en la Sección 2.4 suelen ocurrir en la etapa de preprocesamiento, aunque podrían utilizarse también en otras etapas [1]. Por otro lado, los métodos de procesamiento de eventos (comentados en la Sección 2.3) están fuertemente influenciados por la representación y por la plataforma hardware disponible. Se podría decir que estos tres factores son mutuamente influyentes. Por ejemplo, las representaciones de eventos más densas y sus correspondientes algoritmos suelen ser la mejor opción para procesadores estándar (como CPUs o GPUs), mientras que los métodos evento a evento son comunes en SNNs (*Spiking Neural Networks*) implementadas en hardware neuromórfico.

Nuestra elección ha sido procesar los eventos mediante un método que los agrupa en imágenes (etapa de preprocesamiento) y las caracteriza mediante descriptores globales (etapa de procesamiento central), siendo estos últimos la entrada de un clasificador (etapa de postprocesamiento).

2.5 Aplicaciones

Por todo lo anteriormente mencionado, estas cámaras tienen un gran potencial en muchas aplicaciones. Algunas de estas pueden ser [1]:

- Detección de esquinas y seguimiento: En la literatura, aparecen algunos métodos que combinan eventos y fotogramas [6]. También otros que detectan esquinas (*features* locales) y les hacen un seguimiento, mediante adaptación de métodos clásicos como Harris [7] o Fast [8] para eventos.
- Estimación del flujo óptico (*Optical Flow Estimation*).
- Reconstrucción 3D, monocular y estéreo.
- Estimación de la posición y SLAM.

- Odometría visual-inercial.
- Reconstrucción de imágenes.
- Segmentación.

Sin embargo, aunque las cámaras de eventos se usan en gran variedad de aplicaciones debido a su grandes ventajas, métodos y algoritmos que permitan explotar su potencial son actualmente un campo abierto a la investigación.

Concretamente, en robótica y visión por computador, y especialmente en escenarios difíciles, su baja latencia, alta velocidad y gran rango dinámico solucionan problemas como el emborronado (que sucede en cámaras convencionales con movimientos rápidos) o las malas condiciones de iluminación.

En nuestro método, se aplican algunos descriptores globales (como la transformada de Fourier) a las imágenes de eventos generadas, para finalmente clasificarlas. Sobre esta forma de proceder se tiene poca constancia en la literatura, pues es una idea reciente que está actualmente investigando y probando el grupo GRVC (*Robotics, Vision and Control Group*) de la Universidad de Sevilla.

2.6 Descriptores Globales

Los descriptores, en visión por computador para imágenes tradicionales, pueden entenderse como datos extraídos a partir de imágenes, para caracterizarlas y/o extraer información relevante a partir de ellas. Se usan en la inmensa mayoría de aplicaciones que utilizan cámaras como sensor. Existen dos tipos bien diferenciados de métodos que usan descriptores, enfocando cada uno el problema de extracción de características (*features*) de forma distinta:

- Métodos locales: Analizan la imagen por zonas, generalmente el entorno de píxeles, extrayendo descriptores locales de la imagen. En imágenes basadas en fotogramas son bien conocidos y se usan en campos como visión estéreo y monocular. También son muy utilizados en aplicaciones como reconocimiento de formas, por ejemplo. En eventos, se han adaptado algunos descriptores (como el Harris o el FAST) que ya se usaban en imágenes tradicionales para extraer características locales de forma asíncrona. La adaptación consiste, básicamente, en considerar como entorno local del píxel una ventana espaciotemporal, en lugar de exclusivamente espacial para cada imagen. Además, existen trabajos que usan descriptores locales tradicionales en imágenes o representaciones 2D de eventos.
- Métodos globales: Se centran en extraer de la imagen un único descriptor global que la caracteriza. Poseen algunas ventajas frente a los descriptores locales mencionados anteriormente, como la reducción de memoria. En la literatura existen pocos trabajos que apliquen descriptores globales a imágenes de eventos, por lo que el método que se presenta en este trabajo es un terreno por explorar.

Seguidamente, se comentan algunos descriptores globales: [9]:

- HOG (*Histogram of Oriented Gradients*): Este método consiste en dividir la imagen en celdas y calcular el histograma de cada una de ellas. Se aplica tanto a celdas verticales como horizontales. El descriptor que genera es el conjunto de histogramas (tanto para celdas verticales como horizontales) concatenados, formando un vector [10]. En la Figura 2.3 se ilustra el funcionamiento de este método.

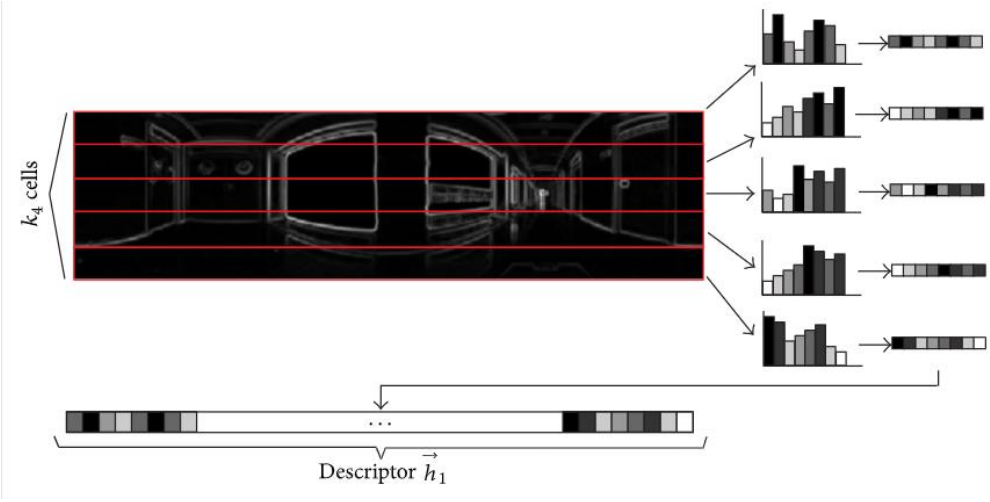

Figura 2.3 Ejemplo de la construcción del descriptor HOG en imágenes omnidireccionales. Fuente: [10].

- GIST: Este descriptor global intenta capturar la esencia de la escena [11]. El propósito de este método es desarrollar un vector que contenga información del entorno a diferentes escalas y con la información de los bordes de la imagen desde diferentes ángulos [9][10]. Para construir el descriptor se crean varias escalas de la misma imagen, se le aplican filtros de Gabor equiespaciados angularmente y se calcula el histograma de las celdas de la imagen, como en el método HOG. Finalmente, para construir el descriptor, se calculan las medias de intensidades de las celdas y se concatenan en un vector.
- Transformada discreta de Fourier (DFT): Se trata de aplicar la transformada de Fourier a la imagen, y considerar variaciones de esta como su descriptor. Es interesante porque aporta información sobre la distribución de frecuencias que tiene la imagen, lo cual puede ser conjuntamente característico en imágenes de un mismo escenario. También es interesante porque se pueden conseguir invarianzas a rotación y escala. La transformada de Fourier de una imagen, sin embargo, es una matriz compleja de las mismas dimensiones que la propia imagen. Esto puede suponer mayor carga computacional y/o más recursos consumidos en algunas aplicaciones, por lo que puede considerarse una reducción de las dimensiones de este descriptor mediante métodos como PCA. En [10] se utiliza la DFT unidimensional por filas en imágenes omnidireccionales, almacenándose la información relevante a frecuencias menores y pudiendo descartar las mayores (que normalmente corresponden a ruido). En la Figura 2.4 se muestra la magnitud de la DFT bidimensional aplicada a una imagen de eventos.

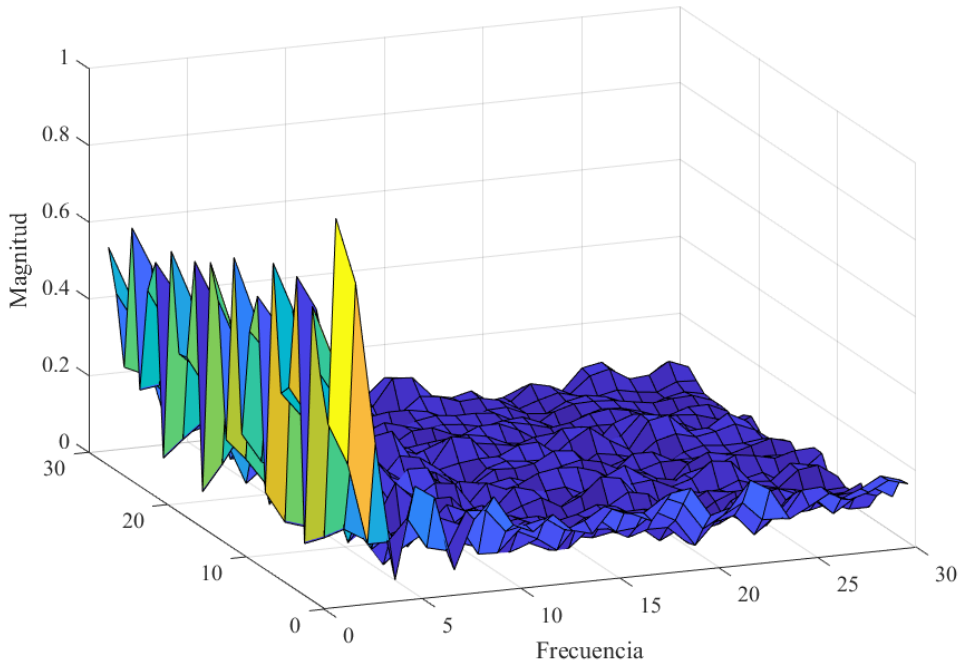

Figura 2.4 Magnitud de la DFT 2D aplicada a una imagen de eventos.

2.7 Clasificación

Introducción

El aprendizaje automático (*machine learning*) es una rama de la inteligencia artificial que permite que las máquinas (ordenadores y otros dispositivos con capacidad computacional) aprendan sin ser expresamente programadas para ello. Estas máquinas aprenden a extraer y detectar patrones y relaciones existentes en un conjunto de datos, que posteriormente pueden usarse para predecir comportamientos o tomar decisiones.

Algoritmos de aprendizaje automático dedicados a la clasificación son llamados clasificadores. Estos analizan un conjunto de datos llamados de entrenamiento, que emplean para aprender, agrupándolos finalmente en distintas clases. El objetivo final del clasificador es ser capaz de predecir la clase a la que pertenece una determinada muestra no incluida en el conjunto de entrenamiento.

En aprendizaje automático se distinguen dos tipos de algoritmos: No supervisados (no se dispone a priori de la clase a la que pertenecen los datos de entrenamiento) y supervisados (sí se conoce la clase a priori). En la Figura 2.5 se muestra un esquema de las distintas categorías de algoritmos de aprendizaje automático.

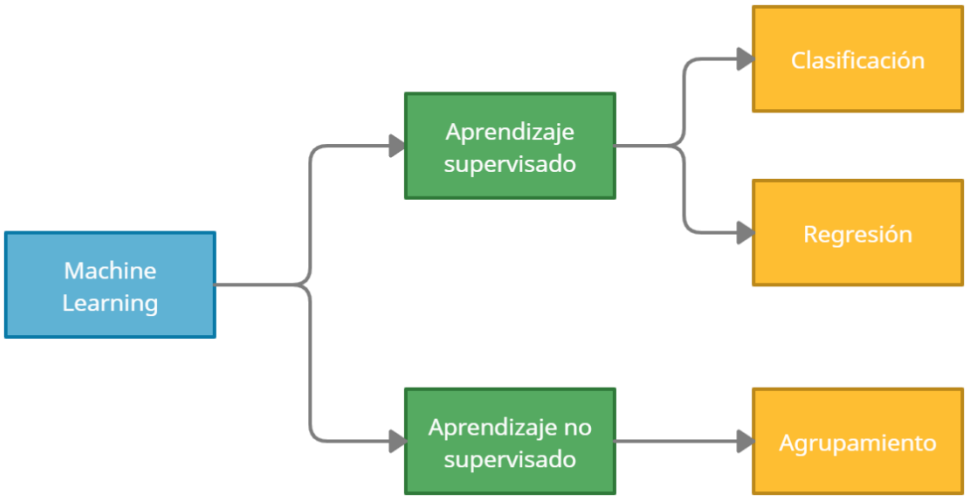

Figura 2.5 Esquema tipos de algoritmos de *Machine Learning*.

Aprendizaje no supervisado

En los métodos de aprendizaje no supervisado, un concepto importante es el agrupamiento. Se trata principalmente de encontrar una estructura o patrón en una colección de datos no categorizados [17]. Los algoritmos de agrupamiento procesan los datos y encuentran en ellos grupos naturales. Estos algoritmos pueden recibir como parámetro el número de grupos a detectar. Existen cuatro tipos de algoritmos de agrupamiento: Exclusivos (los datos se agrupan de tal manera que una muestra solo pueda pertenecer a un grupo, como el conocido K-means), aglomerativos, de solapamiento y probabilísticos. En [18] se comparan distintos tipos de algoritmos de agrupamiento.

En cuanto a clasificación se refiere, un algoritmo de agrupamiento puede ser utilizado como un clasificador. Una vez ha inferido las distintas clases (datos agrupados), puede predecir la categoría a la que pertenece cada nueva muestra, en función de medidas de similitud que dependen del algoritmo de agrupamiento empleado.

Aprendizaje supervisado

El aprendizaje supervisado es una técnica para deducir una función a partir de datos de entrenamiento. Estos últimos se componen de pares de objetos, donde una parte del par se corresponde con los datos de entrada y la otra con la clase a la que pertenecen, que es a priori conocida. La salida de la función puede ser un valor numérico (como en los problemas de regresión) o una etiqueta de clase (como en el problema de clasificación). El objetivo de los métodos de aprendizaje supervisado es crear una función con capacidad de generalización (a partir de los datos de entrenamiento, para situaciones no vistas previamente) y predicción (en el caso de los clasificadores, etiquetar la clase de una muestra no vista previamente).

A continuación, se comentan algunos métodos populares de aprendizaje supervisado, los cuales se revisan detenidamente en [21]:

Redes Bayesianas

Una red Bayesiana es un modelo grafo para representar las relaciones de probabilidad entre un conjunto de variables. Representa un conjunto de variables aleatorias y sus dependencias condicionales a través de un grafo acíclico dirigido (DAG) [22]. Uno de los problemas de las redes bayesianas es que su rendimiento decrece a medida que aumenta la dimensión de los datos de entrenamiento. Se usan en áreas como la clasificación de documentos o sistemas de diagnóstico médico [21].

SVM

Las máquinas de vectores de soporte (SVM) son un conjunto de algoritmos de aprendizaje supervisado, es decir, se conoce a priori la categoría o clase a la que pertenecen los datos de entrenamiento. Fueron desarrollados por Vladimir Vapnik y su equipo en los laboratorios AT&T [14] [15]. Estos algoritmos maximizan el margen entre

los patrones de entrenamiento y el límite de decisión, y precisan de datos de entrenamiento linealmente separables en un espacio de características determinado. Si los datos no son separables en el espacio de características original, se pueden emplear distintos tipos de funciones kernel, que se encargan implícitamente de transformar el espacio de características original a un espacio de características de dimensión superior, donde los datos sean linealmente separables.

k-NN

El algoritmo k-NN (*k-nearest neighbors*) es un método de clasificación no paramétrica. Asigna a un punto cualquiera (que se corresponde con una medida sin etiquetar) la clase del más cercano de un conjunto de puntos previamente etiquetados. Es un método que puede funcionar bien en aplicaciones donde los objetos pueden tener muchas etiquetas, y su correcto funcionamiento depende del parámetro K. En la Figura 2.6 puede observarse gráficamente la dependencia del algoritmo con K, ya que la muestra se consideraría de la clase A si se establece $K = 7$ y de la clase B si $K = 3$.

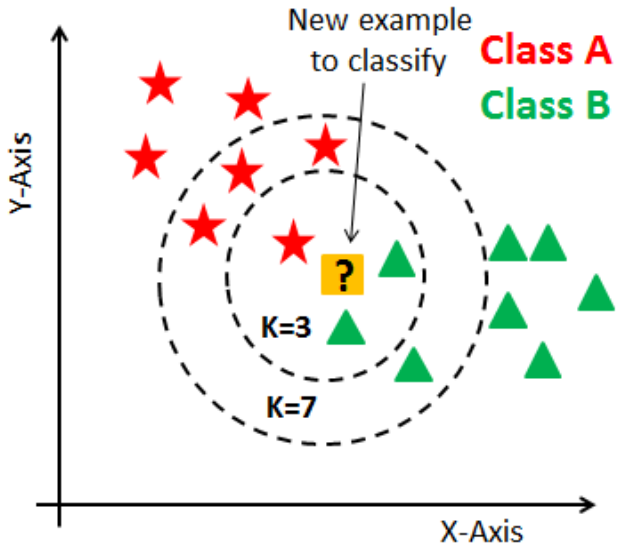

Figura 2.6 Dependencia del parámetro K en el algoritmo K-NN.

Redes neuronales

Las redes neuronales son dispositivos computacionales que se basan en la estructura de neuronas, método de procesamiento y capacidad del cerebro humano [21], aunque a menor escala. Esta técnica es aplicable a problemas donde las relaciones puedan ser no lineales o algo dinámicas. Proponen una alternativa a las técnicas convencionales que, a veces, se ven limitadas por fuertes supuestos como linealidad, independencia entre variables, etc. Existen varios tipos de redes neuronales: BPNN (*Back Propagation Neural Network*), PNN (*Probabilistic Neural Network*), RBNN (*Radial Basis Function Neural Network*), La clasificación anterior se establece en base al método que utilizan para entrenar la red. El Perceptrón es la forma más simple de red neuronal, empleada para clasificación de patrones linealmente separables. Consiste en una sola neurona con los coeficientes ajustados. El perceptrón multicapa es el clasificador basado en redes neuronales más utilizado. Es capaz de modelar funciones complejas y es robusto a entradas irrelevantes y ruido. Es también, en general, complicado de entrenar.

2.8 Reducción de datos

En numerosas aplicaciones conviene reducir la dimensión de un conjunto determinado de datos. En clasificación, por ejemplo, las características que definen el conjunto de medidas pueden ser demasiado numerosas, por lo que es necesario reducir el espacio de características inicial a uno de menor dimensión. Técnicas como ICA (*Indepent Components Analysis*) o PCA (*Principal Components Analysis*) son ampliamente

empleadas con este fin.

Algunos trabajos como [23] defienden la superioridad de ICA frente a PCA como método de extracción de características en tareas de reconocimiento visual de objetos basado en la apariencia. Otros como [24] argumentan que son igual de buenos. En [25] se argumenta que estos dos métodos son equivalentes bajo determinadas condiciones.

ICA

ICA es un método computacional que intenta expresar los datos originales utilizando vectores aleatorios estadísticamente independientes [26]. Las dos suposiciones clave que se hacen en ICA son que los vectores en los que se descomponen los datos originales son estadísticamente independientes y no Gaussianos.

PCA

PCA, o Análisis de Componentes Principales, es una técnica utilizada para describir un conjunto de datos en términos de nuevas variables (llamadas componentes principales) no correlacionadas. El método PCA hace uso de la matriz de covarianza muestral S del conjunto de datos a reducir, y se basa en que las direcciones ortogonales de máxima varianza muestral de los datos iniciales vienen dadas por los autovectores de la matriz S. Además, la varianza muestral en cada eje viene dada por el autovalor de S asociado a cada autovector.

3 DESARROLLO

En este capítulo se explicará el método completo. Se desarrollará cada una de las etapas presentadas en el Capítulo 2, prestando especial atención a los métodos que se han considerado en nuestro trabajo.

3.1 Método de agrupación

En el Capítulo 2, Sección 2.3, se muestran diversas formas de representación de eventos. Además, en el procesamiento de eventos (Sección 2.2), se distinguen dos métodos: métodos que operan evento a evento y métodos que trabajan con grupos o paquetes de eventos. En nuestro caso, se ha decidido trabajar con grupos de eventos, escogiendo, como representación de estos, imágenes binarias.

En imágenes de eventos, existen distintas posibilidades de formación y agrupación. Cada evento $e_k = (\boldsymbol{x}_k, t_k, p_k)$ aporta información espacial, temporal e indica el signo del cambio de intensidad que se ha producido; por lo que se presentan distintas posibilidades de formación de la imagen (según qué información se almacena en cada píxel de la imagen) y diferentes formas de agrupación de los eventos (como ventanas espaciales, temporales, o repetición de un número determinado de eventos en un mismo píxel).

En [9] se muestran distintos métodos de agrupación de eventos para la formación de imágenes. Los eventos, por su naturaleza, aportan información sobre el movimiento de la escena. Esta se encuentra en los instantes de tiempo en los que se registran los diferentes eventos (velocidad del movimiento) y en la polaridad de cada cambio de intensidad (sentido del movimiento). Este trabajo, sin embargo, se centra en la clasificación de las distintas imágenes formadas a partir de eventos, sin tener en cuenta el movimiento relativo entre cámara y escenario. Es por ello por lo que se consideran únicamente imágenes binarias, en las cuales se registra un valor alto si se produjo un evento en ese píxel y un valor bajo si no.

Las distintas imágenes de este proyecto se agrupan en su mayoría mediante ventanas temporales adaptativas, para que cada imagen se forme a partir de un número fijo (N) de eventos. Además, se estudia el caso en el que las imágenes se forman mediante intervalos temporales fijos, atendiendo a los resultados que se obtienen en la clasificación. En la Figura 3.1 se muestran una imagen binaria de cada escenario a clasificar, formada con $N = 7000$ eventos.

Figura 3.1 Imágenes binarias formadas con N=7000 eventos.

Aunque considerar imágenes binarias de un determinado número de eventos pueda parecer a priori una elección simple, nos aporta naturalmente, y a gran velocidad, información sobre los contornos de imágenes, que son las regiones más informativas de estas. Adicionalmente, considerar un número fijo de eventos en la formación de imágenes hace que nuestro método no sufra del problema de escala[1] en eventos, aunque se proponen e implementan soluciones para hacer nuestro método invariante a escala, en función del descriptor usado y del clasificador empleado. También se mantiene la naturaleza asíncrona en el procesamiento de los eventos (aunque aumenta la latencia, proporcionalmente con N), ya que los intervalos de tiempo en los que se forman las imágenes no son fijos.

Cabe resaltar, además, la importancia de la correcta elección del parámetro N, que depende fundamentalmente de la aplicación. A nivel intuitivo y de forma general, se podría afirmar que a mayor N, mayor tiempo de formación de las imágenes de eventos y mejores resultados se obtienen en la clasificación. En el Capítulo 5 se mostrará, en base a resultados experimentales, la dependencia de este parámetro N en el funcionamiento de nuestro método.

3.2 Eventos para el estudio

En este trabajo se ha desarrollado y probado el método con dos *datasets* de eventos diferentes, obtenidos ambos mediante una cámara DAVIS346. Las imágenes se han formado sobre un plano de 260×346 píxeles, que son las dimensiones conocidas del sensor.

El primer *dataset* de eventos se corresponde con los eventos que producen, respectivamente, un cuadrado y un triángulo. De estos conjuntos de eventos pueden extraerse imágenes binarias de contornos de cuadrados y triángulos, desplazados y con diferente tamaño (cambios de escala en imágenes tradicionales) entre sí. En la Figura 3.2 se representa el vector de tiempos para cada figura, de todos los eventos que se han producido. Como se puede observar, el vector de tiempos, para 18 segundos, en el caso del triángulo, registra más de 6 millones de eventos, que es una clara muestra de la alta resolución temporal que presentan estas cámaras. Sin embargo, en nuestro método se ha trabajado, en la mayoría de los casos, con imágenes creadas con 7000 eventos, por lo que la frecuencia con la que se forman las imágenes es del orden de 100 Hz.

En este primer *dataset* se dispone inicialmente de eventos correspondientes a triángulos y cuadrados desplazados y de diferente tamaño, aunque, para considerar también el caso de figuras rotadas, las imágenes creadas a partir de los eventos han sido giradas un ángulo aleatorio, y se ha generado un *dataset* más para la etapa de clasificación. Cabe señalar que algunas de estas figuras rotadas no se incluyen completamente en el plano de la imagen, como se muestra en la Figura 3.4.

El segundo *dataset* [27] ha sido proporcionado por el grupo GRVC de la Universidad de Sevilla, y es parte del material complementario del artículo [28].

En él se encuentran los eventos generados durante 6 trayectorias de un robot ornitóptero en escenarios reales, hasta que aterriza en condiciones de seguridad. Las trayectorias se generan en tres escenarios diferentes, que se denominan: Hills, Soccer y Testbed. Se registran tres trayectorias en cada escenario. El objetivo de la clasificación será determinar con qué escenario se corresponden las imágenes, que podrán pertenecer a cualquiera de los 6 vuelos. En la Tabla 3.1 se muestran los tiempos de vuelo de cada trayectoria, así como el número de eventos generados.

[1] La escala en imágenes de eventos hace referencia al número de eventos que forman la imagen. Conseguir invarianza a este tipo de escala puede ser un problema que resolver en muchas aplicaciones.

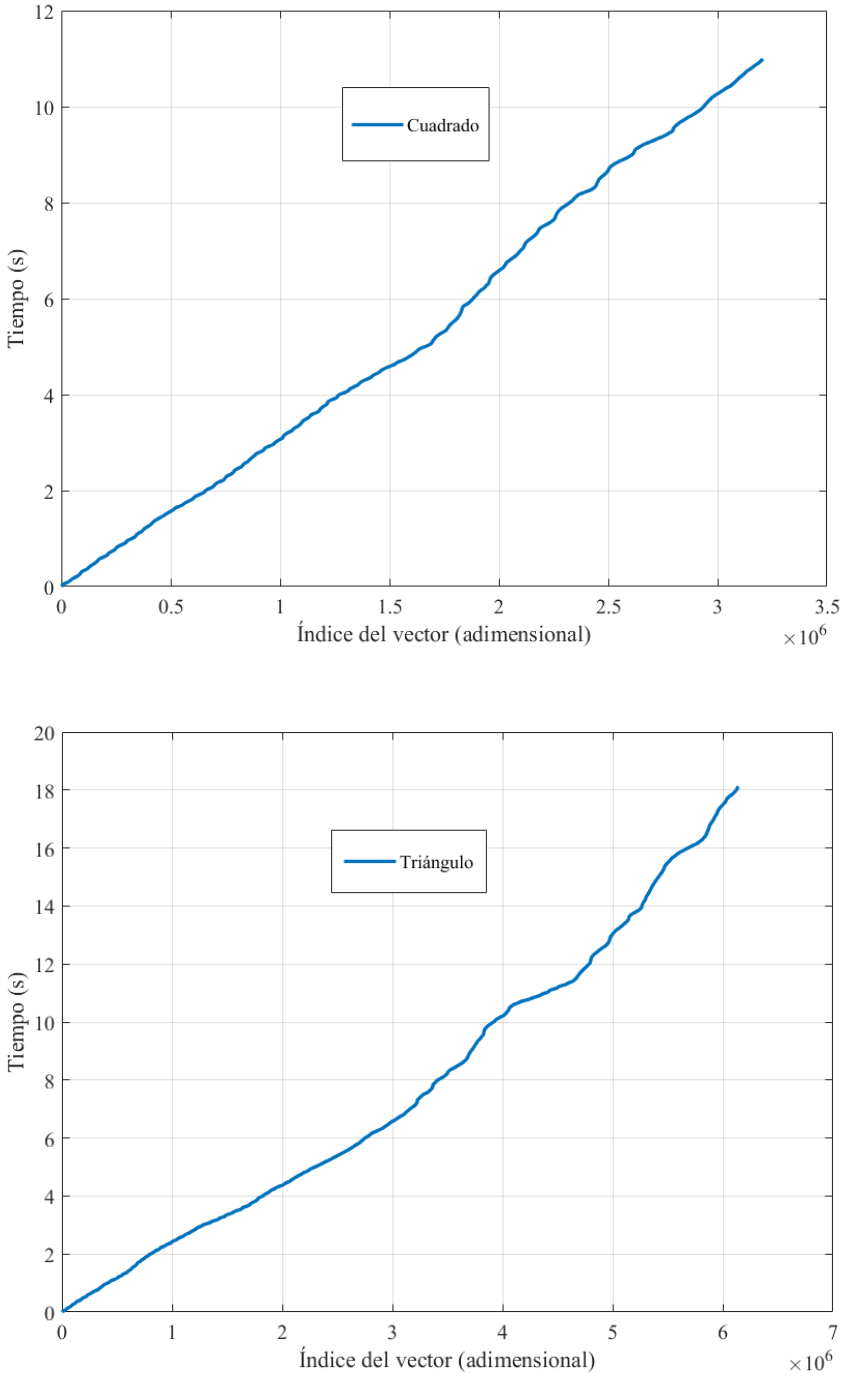

Figura 3.2 Vector tiempo generado en los eventos para cada figura.

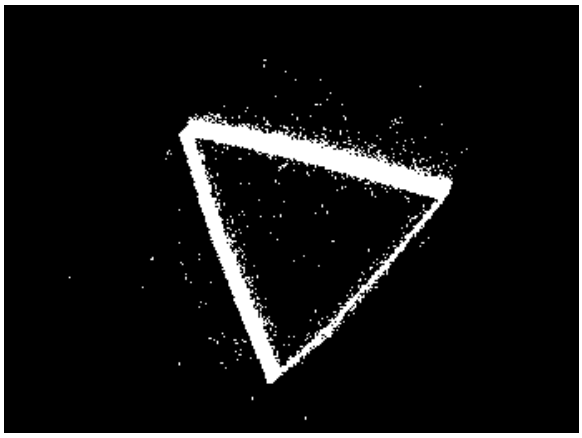

Figura 3.4 Triángulo rotado completo en el plano de la imagen, caso favorable.

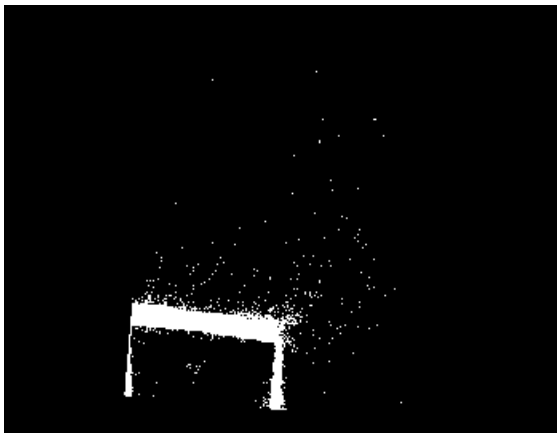

Figura 3.3 Cuadrado rotado incompleto en el plano de la imagen, caso desfavorable.

Tabla 3.1 Tiempos de vuelo y eventos en escenarios.

Trayectorias	**Tiempo de vuelo** (s)	**Número de eventos**
Hills 1	41	15996120
Hills 2	43	15350899
Soccer 1	33	14023297
Soccer 2	26	13921449
Testbed 1	28	25608623
Testbed 2	19	16677041

3.3 Descriptores globales empleados

Los descriptores que se han probado en este proyecto son:

Imagen Vectorizada

El descriptor global es la propia imagen vectorizada, o lo que es lo mismo, no se está empleando ningún descriptor sobre la imagen. Se incluye en este apartado porque, en capítulos posteriores, se evaluará junto al resto de descriptores en términos de clasificación y reducción de su dimensión.

En nuestro caso, se tiene un sensor de 260×346 píxeles. Por tanto, el vector descriptor tendrá 89960 elementos. Este número puede ser demasiado elevado en algunas aplicaciones, por lo que se considerará también reducir las componentes mediante PCA. Este vector es simple y muy rápido de obtener, y funciona bien bajo algunas condiciones que se presentan en el Capítulo 5. No presenta invarianza a traslación, escala o rotación de las imágenes de forma intrínseca, aunque podría funcionar para la clasificación de imágenes que varíen entre ellas de esta forma, si el clasificador es adecuado. Por otro lado, es sensible al número de eventos que contenga la imagen, por lo que no es invariante a escala en eventos.

Transformada discreta de Fourier (DFT)

En nuestro caso se considera inicialmente la transformada bidimensional de Fourier de la imagen binaria de eventos. En una segunda etapa se le calcula la magnitud a la DFT. Finalmente, se vectoriza esta magnitud,

resultando en un vector descriptor de 89960 componentes, que puede reducirse a la mitad (44980 componentes) en la mayoría de los casos. En la Figura 3.5 se muestran las magnitudes de las DFT 2D de un cuadrado y un triángulo sin rotar, donde se observa la simetría que presenta, por lo que puede este descriptor reducirse a la mitad de las componentes. Además, en la mayoría de las imágenes, pueden descartarse componentes de alta frecuencia, ya que se corresponden normalmente con ruido de la imagen. Otra posibilidad es emplear PCA para su reducción, que es la opción que se considera en este trabajo.

Es conocido, por las propiedades de la transformada de Fourier, que si se desplaza una señal en el tiempo la magnitud no se ve afectada. Esto, traducido a la DFT bidimensional de imágenes, quiere decir que la magnitud de la transformada de Fourier es invariante a traslación, como se explica en [9]. Además, se puede considerar que la magnitud de la DFT presenta invarianza a escala (al menos de forma aproximada), ya que una figura, con respecto a otra igual pero más pequeña, en el plano de la imagen, tiene los mismos contornos, sólo que estos están más alejados del centro de la imagen. Esto equivale, si se desprecia la diferencia de longitud de los contornos, a una "traslación" de la figura en el plano de la imagen. Esta aproximación es válida para cambios no demasiado grandes de escala. En la Figura 3.7 puede apreciarse como para dos triángulos de diferente tamaño (Figura 3.6) la forma de la magnitud de la DFT no cambia significativamente. Finalmente, la DFT no presenta (de forma directa) invarianza a rotación. Sin embargo, si quisiéramos conseguir invarianza a rotación con respecto a un eje perpendicular al plano de la imagen, podríamos referir la imagen a coordenadas polares y la invarianza a traslación se transformaría en invarianza a rotación [9]. Cabe destacar que la magnitud de la DFT no es invariante al número de eventos que forman la imagen. Para conseguir invarianza frente a escala en eventos, se normaliza la magnitud de la DFT, dividiendo todos sus componentes entre el valor de continua (valor de la DFT 2D para frecuencia igual a cero en cada eje), que coincide con el número de píxeles a valor alto en la imagen. Esta normalización es necesaria, por ejemplo, en caso de que se formen imágenes de eventos mediante intervalos temporales.

V-ceils

Este descriptor se ha pensado en este trabajo. Funciona bien para imágenes binarias de eventos, sobre todo para distinguir imágenes en las que la dispersión de los eventos por zonas de la imagen es una característica diferenciadora. Esto puede pasar en escenarios reales, como es el caso del *dataset* de vuelos de nuestro caso de aplicación.

Consiste en dividir la imagen de eventos en N celdas. Este número resulta de multiplicar el número de celdas por fila y por columna, que son parámetros escogidos experimentalmente. En cada celda se calcula la matriz de covarianza de los eventos contenidos en ella, considerando su posición (x, y) respecto a los ejes horizontal y vertical de la cámara, locales para cada celda. De esta matriz simétrica 2×2 se calculan sus autovalores y autovectores. Los autovalores representan la varianza que tienen los eventos en las direcciones (perpendiculares entre sí) de máxima varianza. Estas direcciones vienen dadas por los autovectores correspondientes a cada autovalor. Al autovector correspondiente a la dirección de máxima varianza se le calcula el ángulo que forma con el eje horizontal de la imagen. Este ángulo, junto a los autovalores calculados previamente, forma parte del descriptor. Es decir, cada celda aporta individualmente 3 componentes al descriptor, resultando en $3 \times N$ componentes.

Además, para tener en cuenta las relaciones entre las diferentes celdas, se calcula una nueva matriz de covarianza global. Esta se forma entendiendo cada celda como un punto, de coordenadas las varianzas en x, y y cruzada (xy), de cada celda, además del ángulo que forma el autovector de mayor varianza con el eje horizontal (nuevamente en cada celda). Se construye, a partir de las N celdas, con 4 características cada una, la matriz de covarianza global 4×4. De ella se extraen 10 componentes, los 4 de la diagonal y los 6 términos cruzados no repetidos, ya que la matriz es simétrica. Se consideran también parte del descriptor. Finalmente, queda un descriptor de $3 \times N + 10$ componentes. En nuestro trabajo se ha decidido dividir la imagen en 10 celdas por fila y 10 celdas por columna, por lo que este descriptor resulta en 310 componentes.

Además del número de celdas, se ha definido otro parámetro experimental (K) en este método, que sirve para no considerar la dispersión de las celdas con un número de eventos menor que K. Es útil para eliminar el ruido, y depende principalmente del número de eventos que vayan a contener las imágenes y de la dispersión media que presenten. En nuestro caso se ha escogido K = 3 para triángulos y cuadrados, K = 10 para escenarios con 7000 eventos por imagen y K=2 para escenarios con 100 eventos por imagen.

La ventaja fundamental de este descriptor es que presenta invarianza a escala en eventos, ya que con un número

pequeño de estos es capaz de evaluar la dispersión de la imagen, considerando como se distribuyen los eventos en cada celda y, posteriormente, correlacionando esta distribución local para la imagen completa. Además, presenta invarianza parcial a traslación, escala y rotación, debido a los 10 componentes de la matriz de covarianza global para las N celdas. Se podría, además, deducir información sobre el ángulo de rotación que presenta la imagen, gracias al ángulo que forman las direcciones de mayor varianza en cada celda.

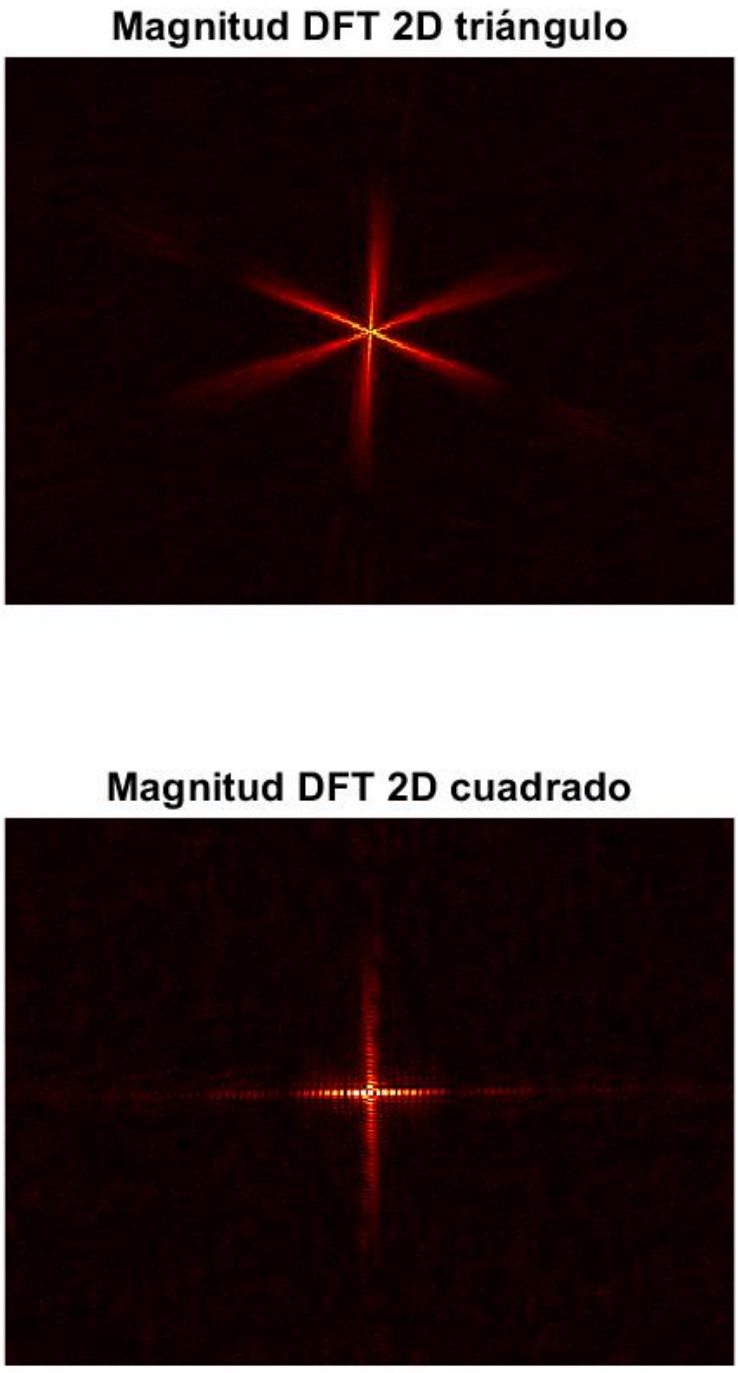

Figura 3.5 DFT 2D de un triángulo y un cuadrado.

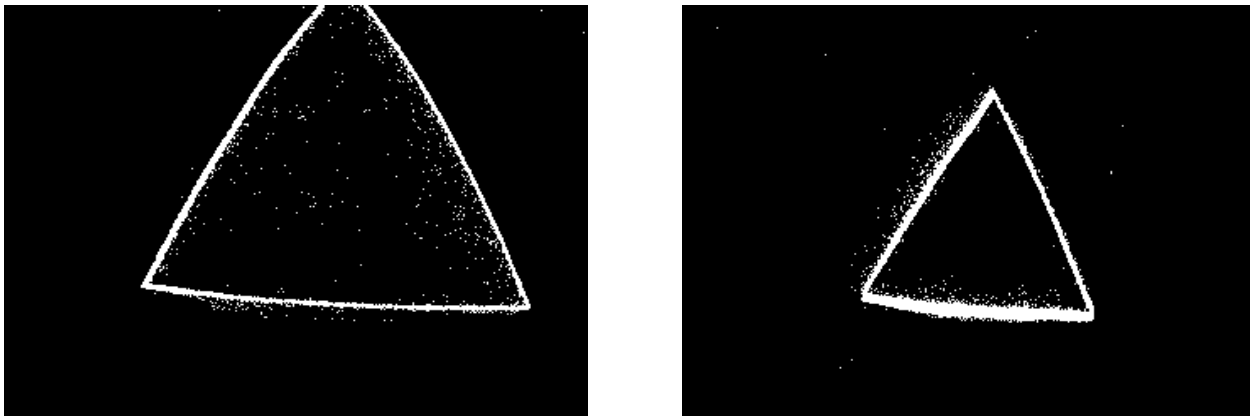

Figura 3.6 Triángulos de distinto tamaño.

A continuación, en la Tabla 3.2 se muestra una comparativa de los distintos descriptores, en función de las invarianzas que presentan y el tiempo que tardan en calcularse para una imagen:

Tabla 3.2 Comparativa de los descriptores globales en función de invarianzas y tiempos de generación[2].

Descriptor	**Invarianza a traslación**	**Invarianza a escala**	**Invarianza a rotación**	**Invarianza a escala en eventos**	**Tiempo de generación** (ms)
Imagen vectorizada	No	No	No	No	0.0008
DFT	Sí	Sí	En coordenadas polares	Si se normaliza	1
V-ceils	Parcial	Parcial	Parcial	Sí	5

[2] Los tiempos de generación han sido medidos en un procesador Intel core i7 10th Gen.

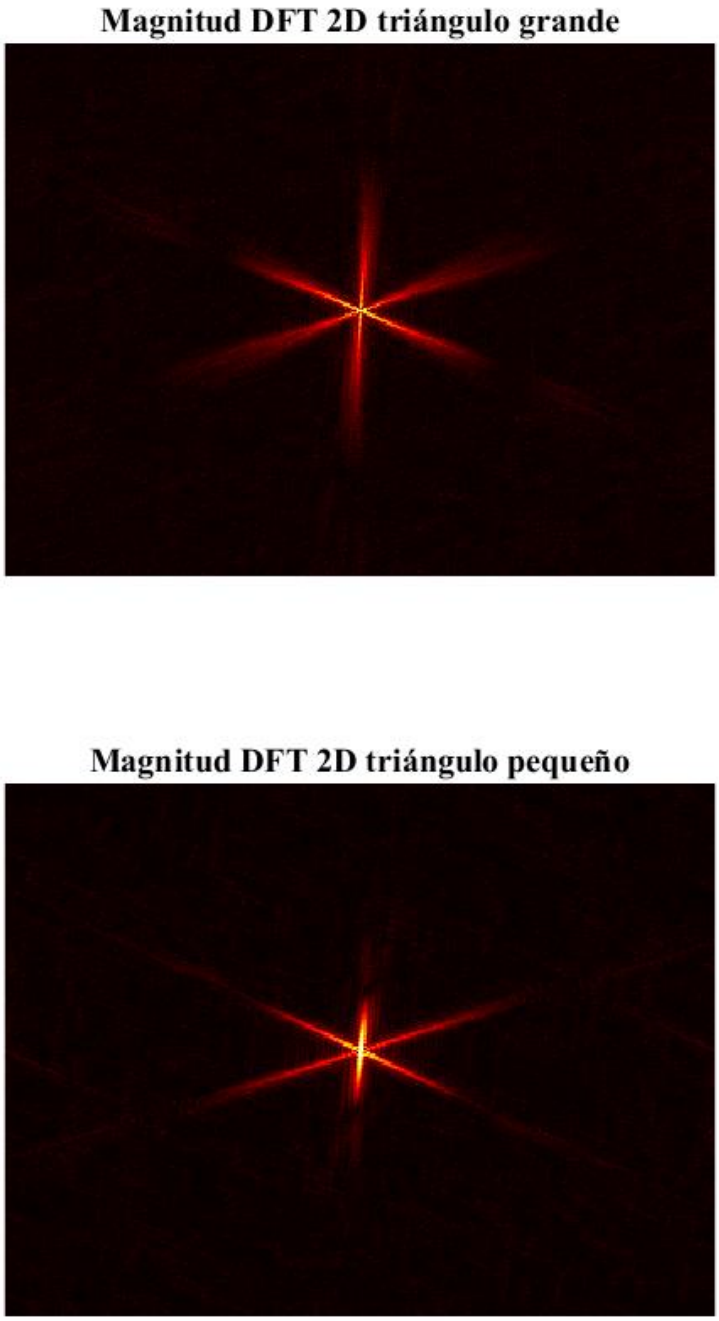

Figura 3.7 DFT 2D de dos triángulos de distinto tamaño.

3.4 Clasificación

En esta sección se explica el conjunto de clasificadores SVM (*Support Vector Machines*) y el algoritmo de *clustering* K-means, que han sido probados durante el desarrollo de este trabajo. Sin embargo, en capítulos posteriores se presentan únicamente resultados para SVM, ya que en esta aplicación concreta se obtienen resultados mucho mejores para este método de aprendizaje supervisado. En aplicaciones similares a la de este proyecto, con un esquema general parecido al presentado en la introducción, pero en las que no se dispone de datos de entrenamiento etiquetados, debe emplearse un método de aprendizaje no supervisado.

K-means

K-means es un algoritmo no supervisado de agrupamiento. En este tipo de métodos no se dispone a priori de información sobre la categoría o clase a la que pertenecen los datos de entrenamiento.

Se cuenta inicialmente con un conjunto de objetos o muestras a clasificar, cada uno con su respectivo conjunto de características, siendo la finalidad del algoritmo agrupar estos objetos en diferentes clases separables.

El algoritmo K-means, en particular, encuentra el centroide de los puntos n-dimensionales (siendo n el número de características de cada elemento a clasificar) pertenecientes a cada clase agrupada. En la Figura 3.8 se ilustra el funcionamiento del agrupamiento K-means, agrupando los datos en 3 conjuntos diferentes.

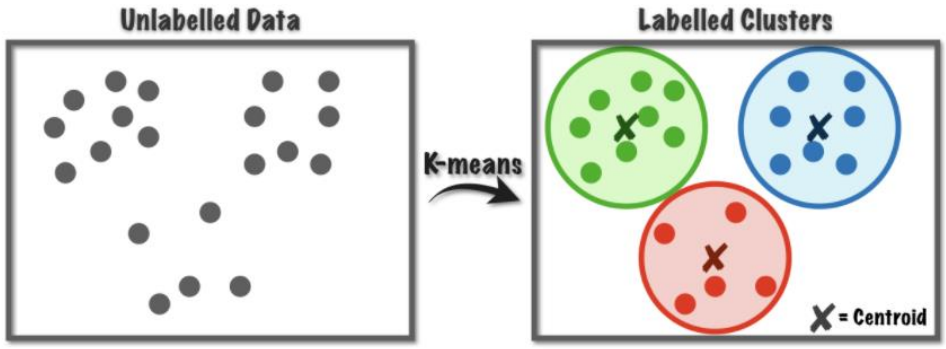

Figura 3.8 Explicación gráfica del agrupamiento K-means. Fuente: [13].

Algoritmo

Sea K el número de clases (y centroides), N el numero de objetos a clasificar y $\boldsymbol{n_i}$ el número de puntos más cercanos al centroide i, con $i \in [1, K]$ y $\sum_{i=1}^{K} \boldsymbol{n_i} = N$.

Código 4.1 Algoritmo K-means

```
Inicializar K centroides aleatorios
MIENTRAS centroides anteriores != nuevos centroides
  Calcular las N distancias de los N puntos al centroide más cercano
  Calcular los K nuevos centroides de n_i puntos asociados a cada centroide anterior
FIN MIENTRAS
```

SVM

En la Figura 3.9 se muestra un conjunto de patrones en un espacio bidimensional de características, donde la recta (que sería un hiperplano en el caso n-dimensional) es el límite de decisión, dividiendo los datos de entrenamiento en 2 clases. Como puede observarse, se maximiza el canal de separación y se minimizan los

errores de clasificación cometidos.

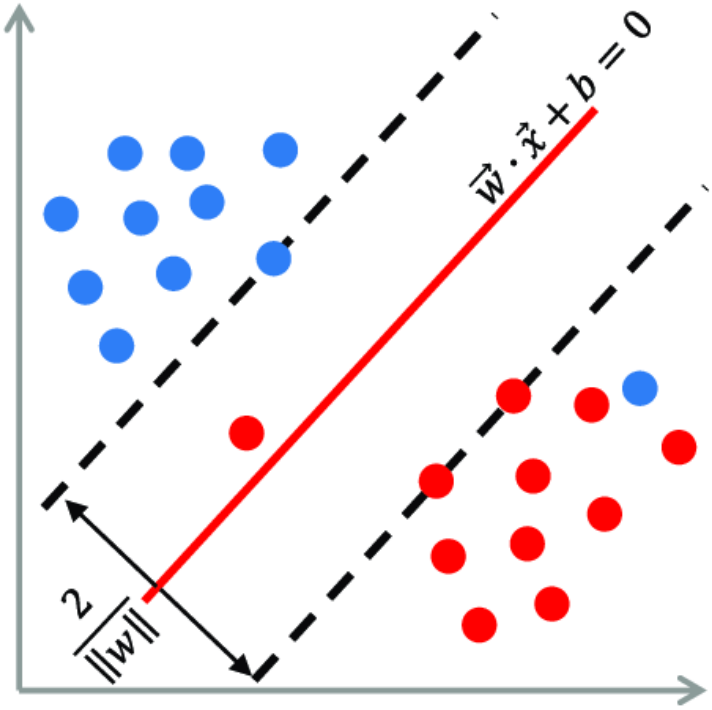

Figura 3.9 SVM en espacio bidimensional linealmente cuasiseparable. Fuente: [16].

Explicación

Sea $\boldsymbol{X} \in \mathbb{R}^{M \times N}$ el conjunto de datos de entrenamiento, siendo M el número de medidas y N el número de características.

Sea $\boldsymbol{w}$ un vector de longitud N, perpendicular al hiperplano que separa ambas clases y con el origen situado en el origen del espacio de características; b una constante; y $\boldsymbol{u}$ un elemento a clasificar (vector de longitud N). Sean también $\boldsymbol{A}$ y $\boldsymbol{B}$ las dos clases a la que pertenecen los datos del problema.

Se define $\boldsymbol{y}$, de longitud M, como: $y_i = 1 \leftrightarrow \boldsymbol{x}_i \in \boldsymbol{A}$ y $y_i = -1 \leftrightarrow \boldsymbol{x}_i \in \boldsymbol{B}$, $\forall i \in [1, M]$. Para todos los datos de entrenamiento debe cumplirse [14]:

$$y_i \cdot (\boldsymbol{x}_i \cdot \boldsymbol{w} + b) - 1 \geq 0, \qquad \forall i \in [1, M] \tag{3.1}$$

La anchura del canal de separación (U), la cual buscamos maximizar, es [15]:

$$U = \frac{2}{\|\boldsymbol{w}\|} \tag{3.2}$$

La función para minimizar en SVM es, en la fase de entrenamiento, tras considerar distintas restricciones y aplicar el método de los multiplicadores de Lagrange [14]:

$$L = \sum_{i=1}^{M} \alpha_i - \frac{1}{2} \sum_{i=1}^{M} \sum_{j=1}^{M} \alpha_i \cdot \alpha_j \cdot y_i \cdot y_j \cdot \boldsymbol{x}_i \cdot \boldsymbol{x}_j \tag{3.3}$$

Esta función puede ser optimizada mediante diferentes herramientas de análisis numérico, como las mencionadas en [14]. Observando 2.4, se aprecia que, finalmente, el problema de optimización depende únicamente del producto escalar entre pares de medidas ($\boldsymbol{x}_i \cdot \boldsymbol{x}_j$), lo cual es una gran ventaja debido a la gran velocidad computacional con la que puede realizarse esta optimización.

La regla de decisión, considerando una muestra $\boldsymbol{u}$ clasificada como perteneciente a la clase $\boldsymbol{A}$, queda finalmente:

$$\sum_{i=1}^{M} \alpha_i \cdot y_i \cdot \boldsymbol{x}_i \cdot \boldsymbol{u} + b \geq 0 \tag{3.4}$$

Se observa nuevamente, ahora en la etapa de validación, que la regla de decisión únicamente depende del producto escalar entre pares de medidas. Esto es beneficioso debido a la sencillez matemática y consecuente velocidad computacional de la función de decisión.

Consideraciones adicionales

Si los datos de entrenamiento no pueden separarse sin error, como en la mayoría de los casos reales, se introducen algunas variables no negativas $\xi_i \geq 0, \forall i \in [1, M]$, que se corresponden con los errores de los patrones incorrectamente clasificados [15]. Se puede ahora minimizar el término

$$\Phi(\xi) = \sum_{i=1}^{M} \xi_i^{\sigma} \tag{3.5}$$

para un pequeño $\sigma > 0$, sujeto a las restricciones

$$y_i \cdot (\boldsymbol{x}_i \cdot \boldsymbol{w} + b) \geq 1 - \xi_i, \qquad \forall i \in [1, M] \tag{3.6}$$

$$\xi_i \geq 0, \qquad \forall i \in [1, M] \tag{3.7}$$

Para un σ suficientemente pequeño, el término descrito en 4.5 describe el número de errores de entrenamiento. Minimizándolo, se encuentra el subconjunto mínimo de errores de entrenamiento. Si este subconjunto de patrones incorrectos es excluido del conjunto de entrenamiento, podemos separar el resto de los patrones del conjunto de entrenamiento sin error.

Además, en el problema descrito hasta ahora, se construye el hiperplano (que separa los datos de entrenamiento) en el espacio de características de entrada [15]. Existen, sin embargo, conjuntos de entrenamiento que no son linealmente separables en el espacio de características original (de dimensión N). Para poder aplicar SVM a estos conjuntos, se debe transformar el espacio de características original en un nuevo espacio de características (de dimensión $D > N$) donde puedan separarse linealmente mediante un hiperplano. Formalmente, el vector $\boldsymbol{x}$ de entrada, de dimensión N, correspondiente a cada patrón de entrenamiento, debería ser previamente transformado en un espacio de características D-dimensional, mediante la transformación ϕ:

$$\phi: \mathbb{R}^N \rightarrow \mathbb{R}^D \tag{3.8}$$

La función de decisión quedaría:

$$f(\boldsymbol{x}) = \sum_{i=1}^{M} \alpha_i \cdot y_i \cdot \phi(\boldsymbol{x}_i) \cdot \phi(\boldsymbol{x}) + b \tag{3.9}$$

Que, como se puede observar en la ecuación 4.9, para un vector desconocido $\boldsymbol{x}$, depende únicamente del producto escalar de los vectores transformados ($\phi(\boldsymbol{x}_i) \cdot \phi(\boldsymbol{x})$).

Considerando formas generales del producto escalar en un espacio de Hilbert [15], se define, para dos vectores $\boldsymbol{u}$ y $\boldsymbol{v}$:

$$K(\boldsymbol{u}, \boldsymbol{v}) \equiv \phi(\boldsymbol{u}) \cdot \phi(\boldsymbol{v}) \tag{3.10}$$

Donde $K(\boldsymbol{u}, \boldsymbol{v})$ se conoce como la función kernel (*kernel function*). Gracias a ella, como este problema se basa en el producto escalar de patrones, no se necesita saber la transformación al nuevo espacio de características.

Basta con conocer la función kernel.

Algunas de estas funciones kernel se presentan en [15], como las Funciones Potenciales:

$$K(\boldsymbol{u}, \boldsymbol{v}) = e^{-\left(\frac{|\boldsymbol{u}-\boldsymbol{v}|}{\sigma}\right)} \tag{3.11}$$

También, para construir un clasificador polinómico de grado d, se puede usar la siguiente función kernel:

$$K(\boldsymbol{u}, \boldsymbol{v}) = (\boldsymbol{u} \cdot \boldsymbol{v} + 1)^d \tag{3.12}$$

Usando diferentes productos escalares $K(\boldsymbol{u}, \boldsymbol{v})$, se pueden construir diferentes máquinas de aprendizaje SVM [15].

Para la clasificación de las distintas imágenes de eventos que se tienen en nuestro trabajo, teniendo en cuenta los descriptores usados y la aplicación o no de PCA (además del número de componentes principales seleccionados y su distribución en el espacio de características, si se aplica), se han considerado los kernels mencionados anteriormente, empleando el más conveniente en cada caso. La forma de elegir el kernel es experimental, y se ve en detalle en el Capítulo 4. La dependencia entre el kernel empleado y el éxito del clasificador, considerando PCI y tiempo de validación, se detalla en el Capítulo 5.

3.5 Reducción del descriptor

En esta sección se describe el algoritmo PCA, que se emplea para reducir la dimensionalidad de las características correspondientes a un conjunto de muestras determinado. En nuestro método se usa PCA para reducir la longitud del descriptor global que caracteriza las distintas imágenes.

Explicación

Sea $\boldsymbol{X} \in \mathbb{R}^{N \times D}$ un conjunto de datos, donde N es el número de muestras y D el número de características de cada muestra.

Se define la media muestral, que se denota como $\overline{\boldsymbol{X}}$, y cuya expresión es:

$$\overline{\boldsymbol{X}} = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{X}_i \tag{3.13}$$

Donde $\boldsymbol{X}_i$ son los vectores de dimensión D correspondientes a las filas de la matriz de datos $\boldsymbol{X}$.

Se define ahora, a partir de la media muestral, la matriz de covarianza muestral ($\boldsymbol{S}$), que responde a la ecuación:

$$\boldsymbol{S} = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{X}_i \, \boldsymbol{X}_i{}^T - \overline{\boldsymbol{X}} \, \overline{\boldsymbol{X}}^T \tag{3.14}$$

Se tiene el objetivo de encontrar los vectores $\boldsymbol{u}_j$, de dimensión D, tales que maximicen las varianzas muestrales de los vectores $\boldsymbol{X}_i$ proyectados sobre las direcciones de los vectores $\boldsymbol{u}_j$. En este caso, no es necesario emplear métodos de optimización para la búsqueda de tales vectores $\boldsymbol{u}_j$, ya que el álgebra lineal aporta una solución eficiente, que es la base de la técnica PCA y hace que su uso sea tan popular:

Los autovectores $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_D$ de la matriz $\boldsymbol{S}$, correspondientes a los autovalores $\lambda_1, \lambda_2, \ldots, \lambda_D$, ordenados en sentido decreciente, indican las direcciones (ortogonales entre sí) de máxima varianza muestral del conjunto de datos inicial ($\boldsymbol{X}$). Además, $\lambda_1, \lambda_2, \ldots, \lambda_D$ se corresponden con las varianzas muestrales de los datos ($\boldsymbol{X}$) proyecctados sobre las direcciones de los autovectores $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_D$.

La técnica PCA suele emplearse para reducir la dimensión D del espacio original de características a un nuevo valor K. Los K autovectores se agrupan en $\boldsymbol{P}_K = (\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_K) \in \mathbb{R}^{D \times K}$ y se denominan Componentes Principales (*Principal Components*).

Código 2.2 Algoritmo PCA

Cargar los datos $\boldsymbol{X}$ de entrada

Calcular la matriz de covarianza $\boldsymbol{S}$

Hallar los autovectores y autovalores de $\boldsymbol{S}$

Elegir $\boldsymbol{K} < \boldsymbol{D}$ y establecer los componentes principales $\boldsymbol{P_K}$

Calcular los datos transformados como $\boldsymbol{Y_i} = \boldsymbol{P_K^T X_i} \in \mathbb{R}^{\boldsymbol{K}}, \forall \boldsymbol{i} \in [\boldsymbol{1}, \boldsymbol{M}]$

Consideraciones adicionales

La elección del parámetro K es crítica y muy dependiente de la aplicación. Si se quiere visualizar la dispersión del conjunto de datos original, estimada mediante la varianza en este caso, K debe ser igual a 2 ó 3. Sin embargo, si se pretende buscar un correcto equilibrio entre la información que se mantiene (porcentaje de varianza asociada a los K Componentes Principales respecto de la varianza total) y el número de autovectores que no se tienen en cuenta ($N - K$), una buena opción es hacer uso de lo que se conoce como *Scree Plot*. Como se observa en la Figura 3.10, en el eje horizontal se representan los Componentes Principales (autovectores) y en el eje vertical la varianza, en por unidad, asociada a cada Componente Principal, que se corresponde con los autovalores.

En el Capítulo 4 se muestra el análisis PCA para los distintos descriptores y *datasets* empleados, analizando la dependencia del parámetro K con el tipo de kernel empleado. En el Capítulo 5 se muestra la dependencia del parámetro K con la velocidad de validación del algoritmo.

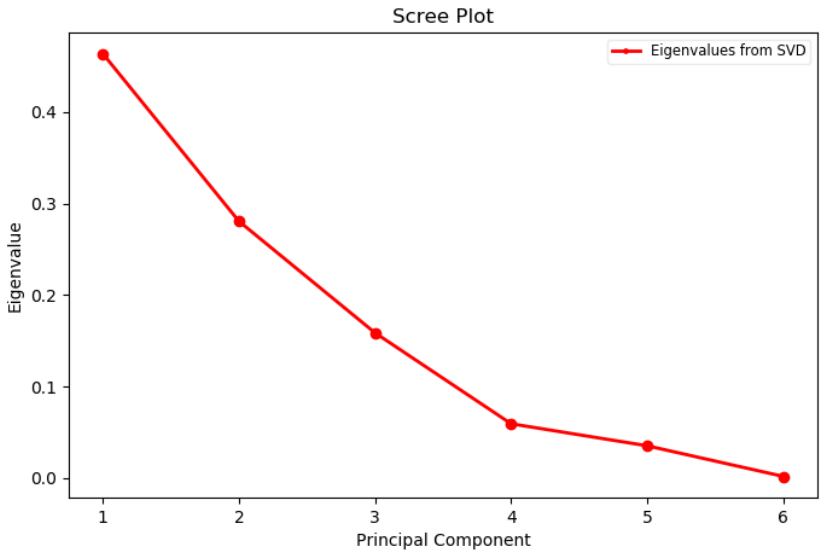

Figura 3.10 *Scree Plot* para 6 componentes principales.

4 ANÁLISIS E IMPLEMENTACIÓN

La implementación de este trabajo está muy ligada al análisis de los datos experimentales de los que se disponen. El objetivo de este capítulo es mostrar que el descriptor empleado, el *dataset* de eventos que se pretende clasificar, el número de componentes principales seleccionados (si se considera una etapa de reducción de los datos de entrenamiento) y el tipo de kernel para SVM son factores mutuamente influyentes. En este capítulo se revisan los distintos descriptores probados y se estudia la dispersión de los datos (correspondientes a cada *dataset*) mediante PCA.

4.1 Conjuntos de entrenamiento y validación

Las imágenes de este proyecto han sido generadas mediante ventanas temporales adaptativas (de un número de eventos fijo) o mediante ventanas temporales fijas (y número de eventos variable). Sea cual fuere el método empleado para generar las imágenes binarias, se han considerado en todos los *datasets* de estudio 6000 imágenes de entrenamiento y 1500 de validación. Las imágenes han sido generadas, para cada escena, de la siguiente forma:

Sea T el total de imágenes (de entrenamiento o validación) a generar en cada escena, D el número de eventos a considerar en cada imagen (si se considera fijo), N el número de eventos totales generados en cada escena, $\boldsymbol{x}_{i+k}$ la posición del evento $i + k$, en el plano de la imagen, y $\boldsymbol{f}_j$ la imagen j, $\forall\, j \in [1, T]$.

Código 4.1 Algoritmo de generación de imágenes aleatorias para cada escena.

```
j=1
MIENTRAS número de imágenes generadas < T
  Inicializar f_j como matriz nula
  Generar número aleatorio k ∈ [1, N − D]
  i=1
  MIENTRAS i < D
    f_j(x_{i+k}) = 1
    i++
  FIN MIENTRAS
  j++
FIN MIENTRAS
```

En cada *dataset* de eventos se ha considerado el mismo número de imágenes para cada escena (o cada clase).

4.2 *Dataset* de triángulos y cuadrados sin rotar

En primer lugar, se considera un *dataset* sencillo de clasificar, compuesto por eventos correspondientes a triángulos y cuadrados, trasladados y a diferente escala. Las imágenes se han generado con un número fijo de eventos N=7000. Se evalúa la dispersión de los datos generados por los diferentes descriptores, mediante análisis PCA, lo cual es útil para elegir el tipo de kernel en el clasificador SVM. Además, analizando estos datos sencillos de clasificar (y posteriores más complejos), puede deducirse, por comparación, el nivel de cada tipo de invarianza que presenta cada descriptor. En nuestro caso de aplicación, por ejemplo, la clasificación de estos patrones sencillos puede ser útil para la detección de *landmarks* por parte del robot ornitóptero en tiempo real. En las Figuras 4.1 y 4.2, se muestran imágenes correspondientes a cada clase, donde puede apreciarse que, a pesar de ser figuras sencillas, en muchas ocasiones no se cierran del todo y sufren del ruido que añade el sensor.

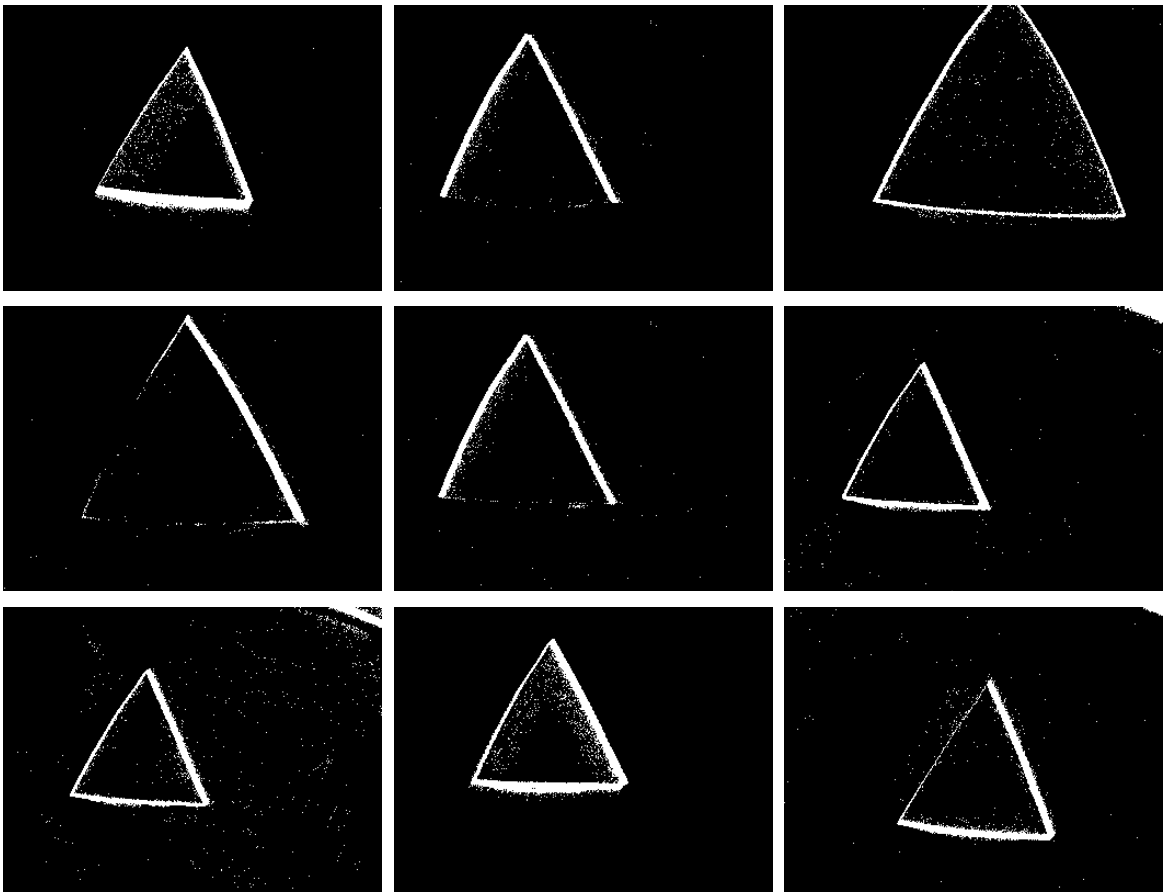

Figura 4.1 Triángulos trasladados y de diferente tamaño.

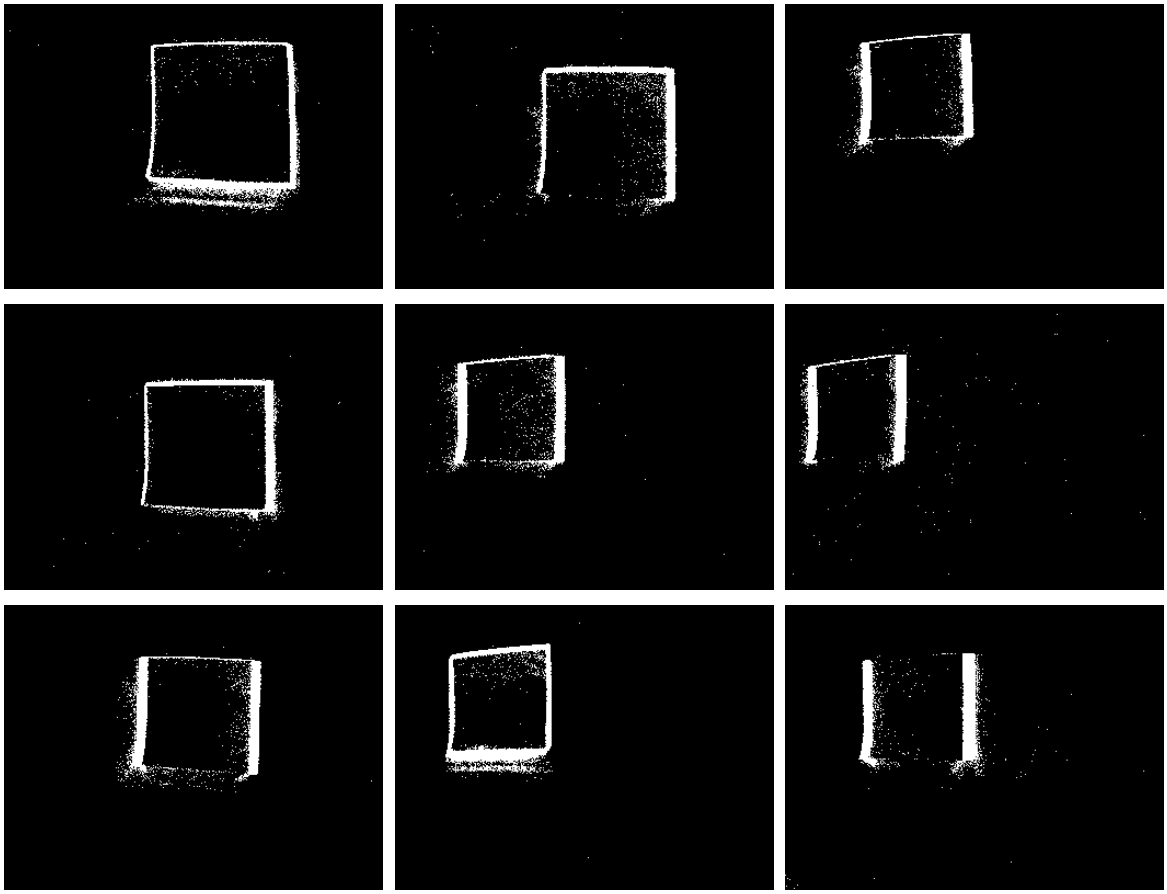

Figura 4.2 Cuadrados trasladados y de diferente tamaño.

A continuación, se evalúa la dispersión de los prototipos de entrenamiento generados mediante cada descriptor, mediante PCA, que serán matrices datos de $6000 \times D$, siendo D el número de elementos de cada vector descriptor.

Imagen vectorizada

Este descriptor tiene 89960 componentes. Esto puede o no ser un número demasiado elevado. En nuestra aplicación concreta, por ejemplo, podría ser interesante que el procesamiento a bordo del ornitóptero fuese mínimo y la clasificación de los distintos datos se realice en la estación de tierra. En esta situación, el ornitóptero necesita enviar el descriptor generado a la estación y, si este es demasiado extenso, podría haber problemas de latencia, además de incurrir en una mayor probabilidad de cometer errores en la transmisión. Además, en algunas ocasiones, reducir el número de componentes principales disminuye el tiempo de validación e incluso mejora el PCI (Porcentaje de Clasificaciones Incorrectas) del clasificador adaptando la función kernel del SVM a los datos reducidos.

En la Figura 4.3 se representa la varianza correspondiente a cada componente principal. Este gráfico es útil para escoger el número de componentes principales en el que proyectar los datos y reducir la dimensión del espacio de características. En este caso, reducir el espacio de características a 3, 5, 15 o 22 componentes, parece, a priori una opción interesante para el estudio.

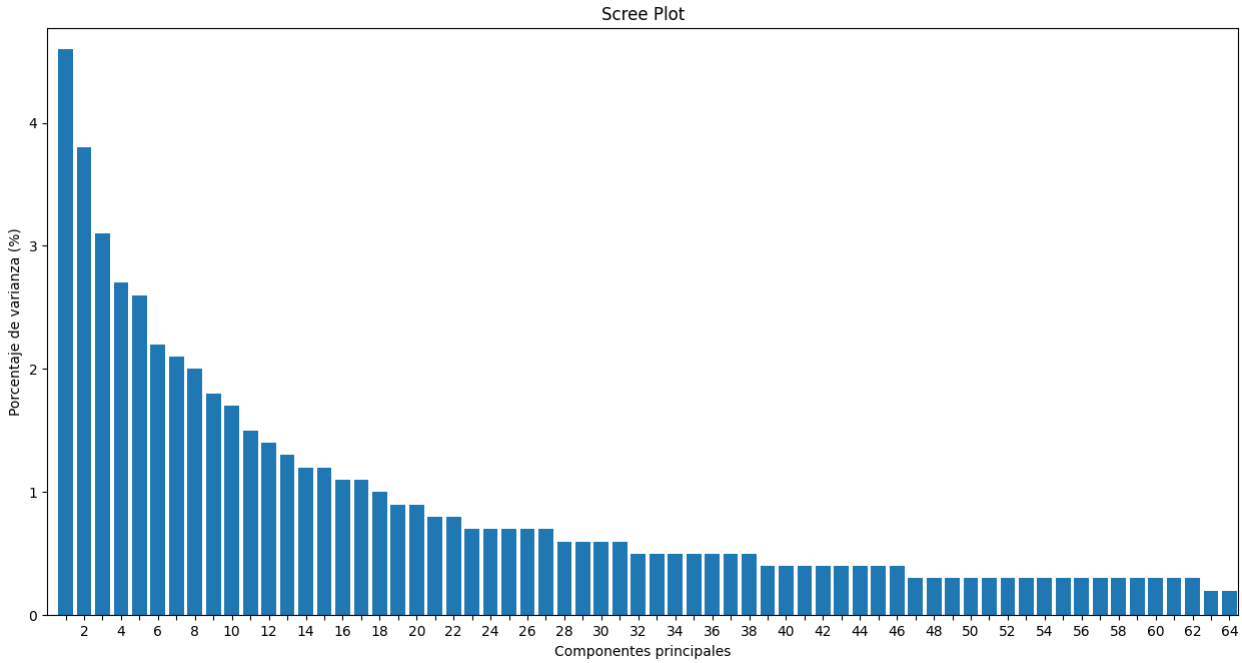

Figura 4.3 *Sree plot* de los prototipos generados vectorizando la imagen, para triángulos y cuadrados sin rotar.

En la Figura 4.4 se representan los datos proyectados sobre las 3 direcciones de mayor varianza muestral. Como puede observarse, los datos no son linealmente separables. Además, no están cerca de serlo, por lo que, en este caso, puede intuirse que no serán linealmente separables proyectados sobre pocos componentes principales, por lo que un kernel lineal no dará buenos resultados en estas circunstancias. Las funciones kernel potenciales suelen funcionar bien en muchos casos. En este en concreto, la transformación al nuevo espacio de características no es evidente, por lo que se prueba a clasificar con una función potencial, concretamente el conocido kernel RBF (*Radial Basis Function* kernel).

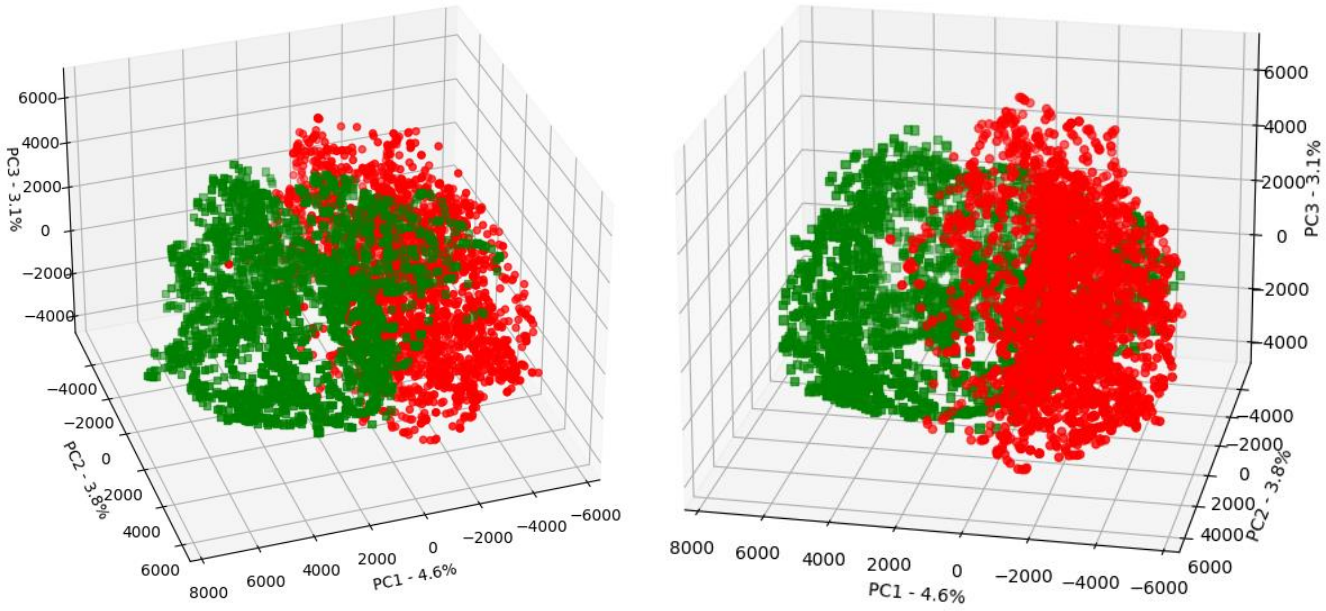

Figura 4.4 Visualización de los prototipos generados vectorizando la imagen, proyectados sobre 3 componentes.

DFT

Este descriptor tiene 44980 componentes, ya que se ha eliminado la mitad de la DFT porque presenta simetría. No se le aplica reducción de altas frecuencias. También puede ser conveniente una reducción del número de componentes.

En la Figura 4.5 puede observarse como, en este caso, la varianza de los datos es mucho mayor en los primeros componentes principales. Además, la Figura 4.6 muestra como los datos son linealmente separables proyectados sobre el primer componente principal (eje horizontal). En este caso, el clasificador SVM con kernel lineal puede recibir como datos de entrenamiento un vector de muestras, y separarlas perfectamente.

Estos datos son tan fácilmente separables por las invarianzas a traslación y escala que presenta el descriptor DFT, que se comentan en la Sección 3.3 y se analizan en detalle en [9].

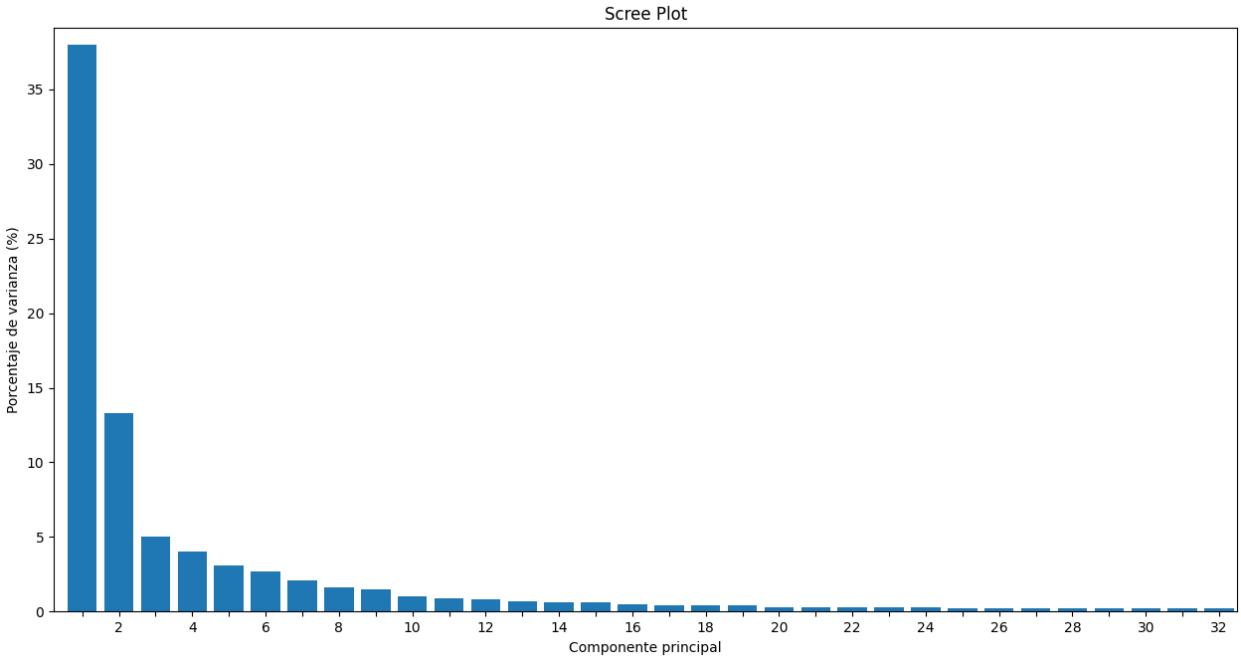

Figura 4.5 *Scree plot* de los prototipos generados mediante la DFT, para triángulos y cuadrados sin rotar.

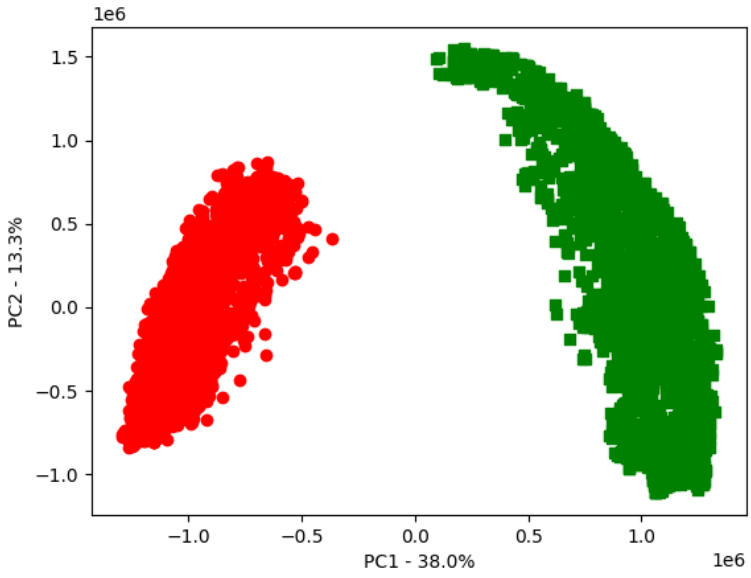

Figura 4.6 Visualización de los prototipos generados por la DFT, proyectados sobre 2 componentes principales, para triángulos y cuadrados sin rotar.

V-ceils

Este descriptor se compone de 310 componentes. La reducción de componentes no es tan interesante como en los casos anteriores, ya que este descriptor es un vector de mucha menor dimensión. Además, por construcción, la mayor parte de la información que extrae este descriptor es sobre la dispersión de los eventos en celdas de la imagen. No ha sido diseñado para ser aplicado a figuras geométricas simples, donde los eventos se distribuyen en los contornos de estas, a excepción del ruido. Sin embargo, para este caso de clasificación sencillo, funciona correctamente porque las figuras no aparecen rotadas y V-ceils aporta información sobre la dirección de los contornos presentes en cada celda.

En la Figura 4.7 puede observarse como la varianza de los primeros componentes principales (sobre todo el primero) es significativamente mayor que la del resto. Cabe señalar que es una varianza porcentual, con respecto a la varianza total de los datos que se generan con este método, por lo que esto no quiere decir que los datos sean muy separables con este descriptor particular. En la Figura 4.8 puede observarse como los datos no son linealmente separables con pocos componentes, lo que sí ocurre con la DFT.

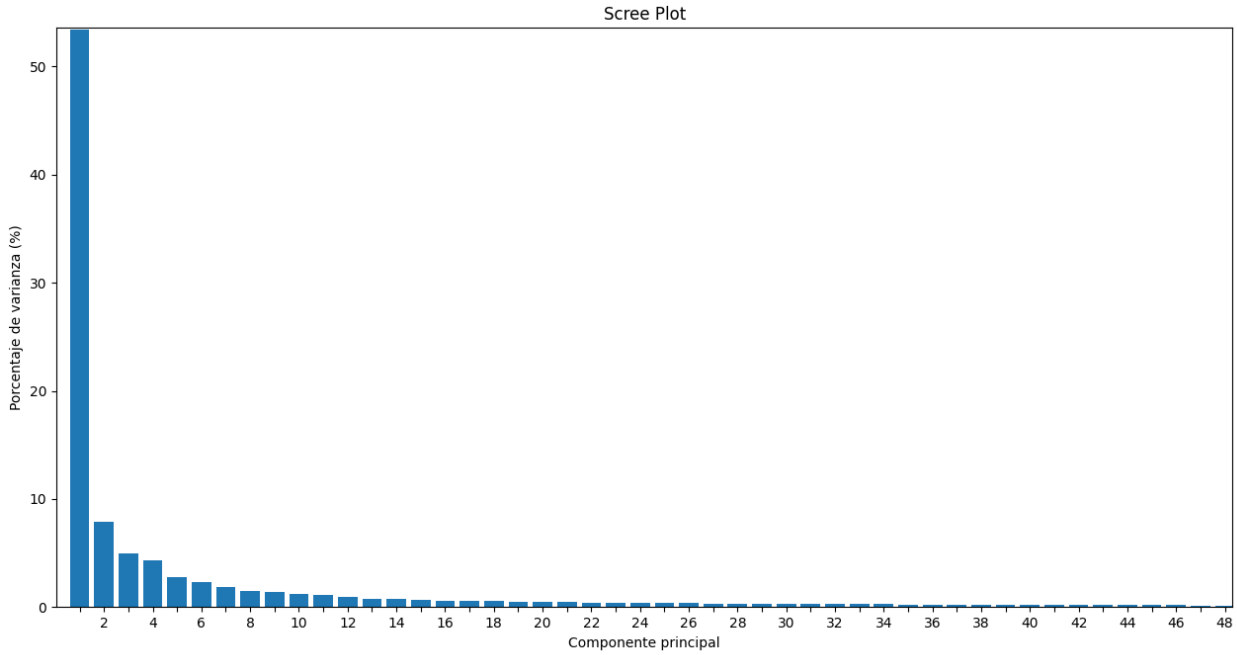

Figura 4.7 *Scree plot* de los prototipos generados mediante V-ceils, para triángulos y cuadrados sin rotar.

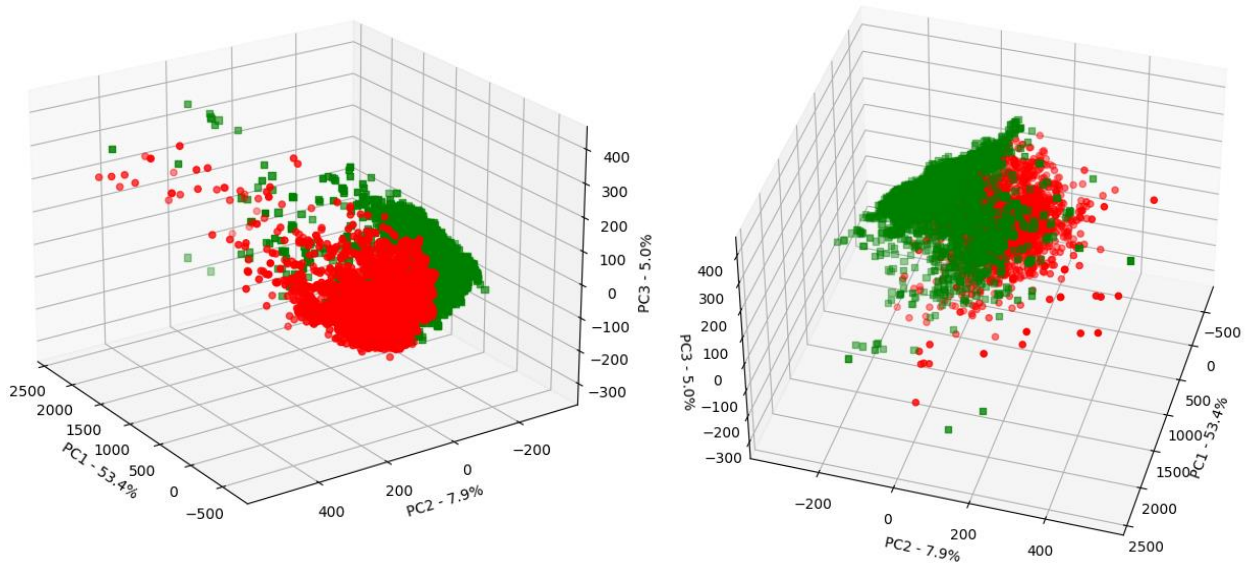

Figura 4.8 Visualización de los prototipos generados mediante V-ceils, proyectados sobre 3 componentes principales, para triángulos y cuadrados sin rotar.

4.3 *Dataset* de triángulos y cuadrados rotados

Se repite ahora el análisis anterior para el mismo conjunto de imágenes, pero rotadas cada una un ángulo aleatorio. En este caso algunas imágenes aparecen cortadas o fuera del plano de la imagen, lo que a priori es una dificultad añadida a la clasificación. En las Figuras 4.9 y 4.10 se muestran imágenes correspondientes a cada clase.

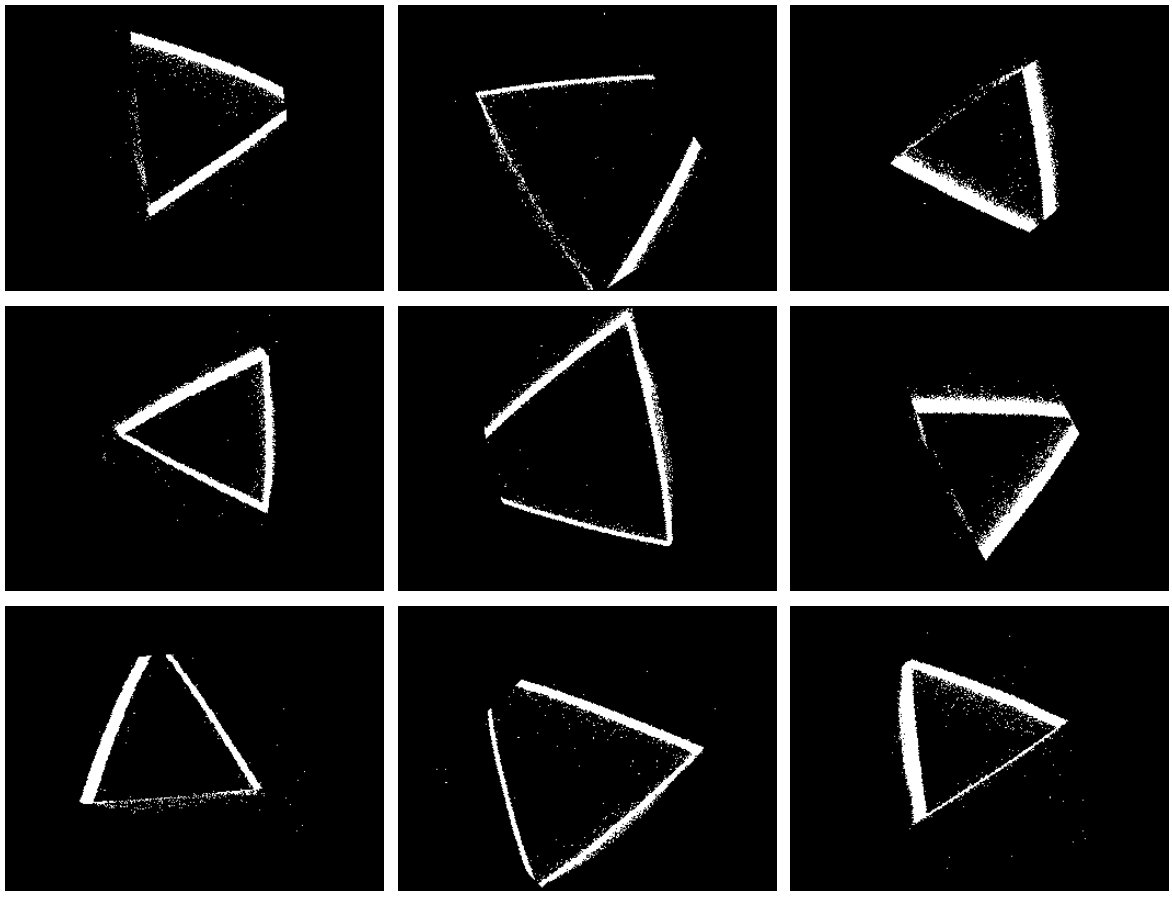

Figura 4.9 Triángulos trasladados, rotados y de diferente tamaño.

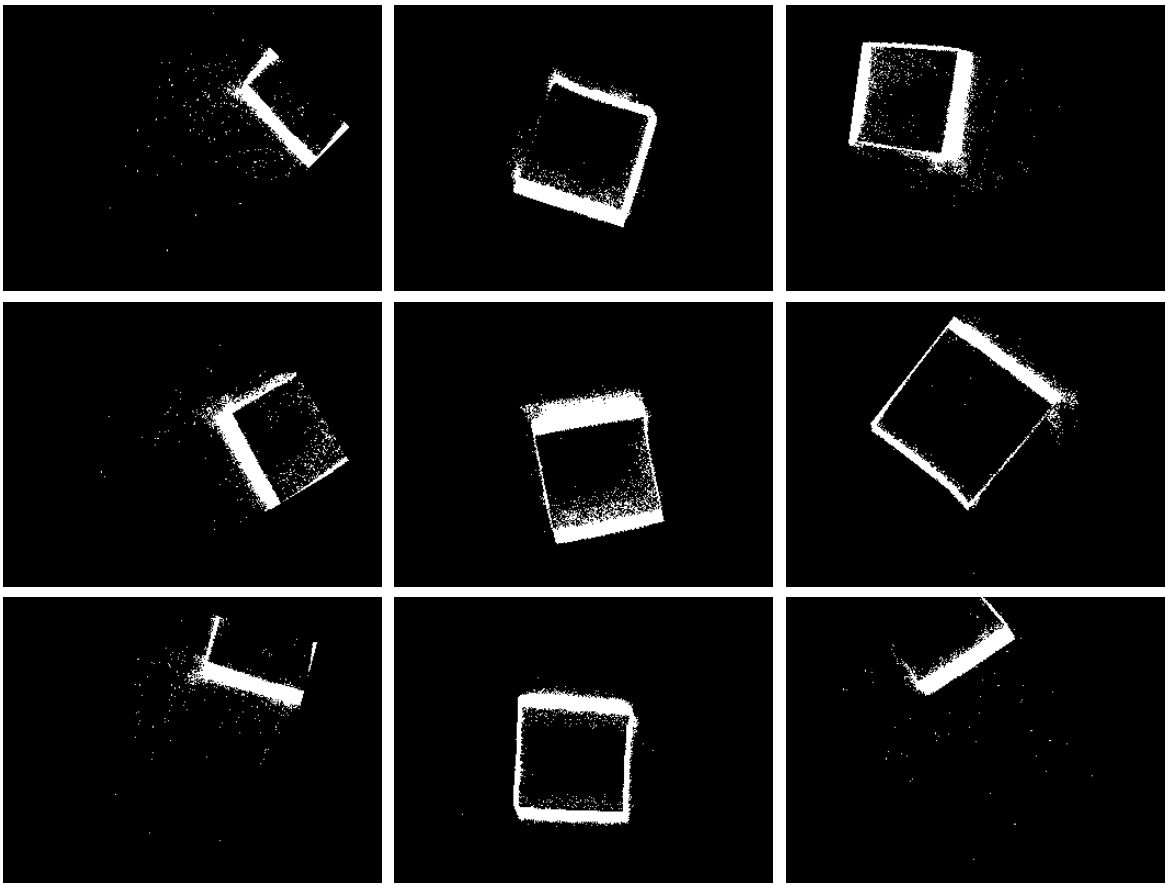

Figura 4.10 Cuadrados trasladados, rotados y de diferente tamaño.

Imagen vectorizada

Volvemos a estudiar la dispersión y separabilidad de los datos generados mediante este descriptor, ahora para el caso en el que las figuras están rotadas.

En la Figura 4.11 se representa la varianza muestral correspondiente a cada componente principal. Como puede observarse, la varianza de los primeros componentes principales es inferior al caso de triángulos y cuadrados sin rotar, lo que a priori parece una dificultad para reducir el número de componentes principales.

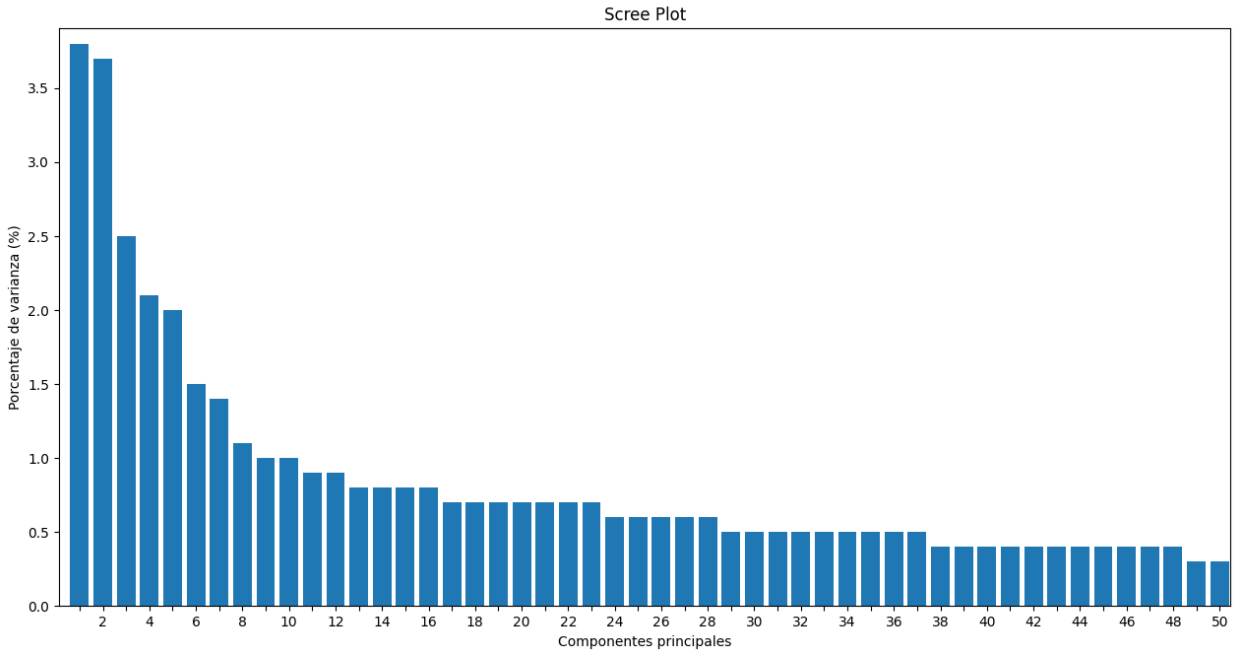

Figura 4.11 *Scree plot* de los prototipos generados vectorizando la imagen, para triángulos y cuadrados rotados.

En la Figura 4.12 se representan los datos proyectados sobre las 3 direcciones de mayor varianza muestral. Puede observarse nuevamente que los datos no son linealmente separables expresados en términos de los 3 primeros componentes principales. De hecho, los datos son separables mediante lo que parece una esfera, ya que los puntos verdes rodean a los rojos. Podría aplicarse un kernel a medida para hacer que los datos fuesen separables mediante SVM. Este kernel no es más que el que se corresponde con la siguiente transformación de 3 a 4 dimensiones: $f(x, y, z) = [x, y, z, (x^2 + y^2 + z^2)]$. Sin embargo, implementar este kernel no es eficiente computacionalmente (hace la clasificación demasiado lenta) en el caso del SVM, lo que es determinante en nuestra aplicación. Kernels ampliamente usados, como los polinómicos o RBF, sí son eficientes y conviene probarlos en este caso. De todas formas, como se verá a continuación, la DFT vuelve a proporcionar resultados mucho mejores en este *dataset*, por lo que tampoco es demasiado interesante, en nuestra aplicación, conseguir buenos resultados con la imagen vectorizada.

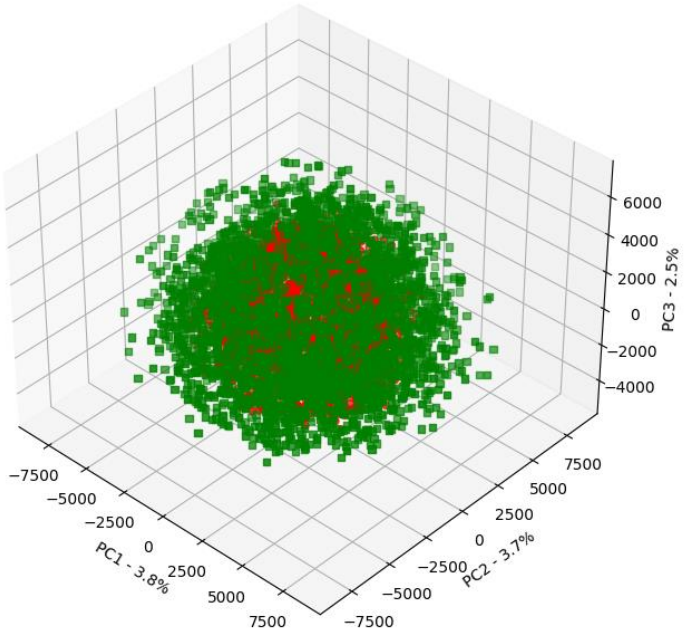

Figura 4.12 Visualización de los prototipos generados vectorizando la imagen, proyectados sobre 3 componentes principales, para triángulos y cuadrados rotados.

DFT

Se vuelve a analizar, ahora para triángulos y cuadrados rotados, el rendimiento de la DFT. En este caso, se aplica PCA sobre las imágenes descritas con la magnitud de la DFT completa, es decir, con 89960 componentes. No se reduce el descriptor a la mitad porque la simetría que presenta es útil para la reducción de componentes, en este caso particular. Este efecto puede apreciarse comparando las Figuras 4.14 y 4.15, observando como en la primera los datos son separables y en la segunda no.

En la Figura 4.13 se observa el *Scree Plot*, que en principio parece que permitirá una reducción de los componentes principales. En la Figura 4.14 se muestran los datos proyectados sobre los dos componentes principales de mayor varianza muestral. Como puede observarse, no son linealmente separables, aunque lo serían si se le aplica una transformación lineal de 2 a 3 dimensiones de tipo paraboloide: $f(x, y) = [x, y, (x^2 + y^2)]$. La función kernel que se corresponde con esta transformación es computacionalmente lenta, por lo que en este caso se aplica una función kernel polinómica de grado 2, que es muy parecida.

La DFT es claramente superior como descriptor frente a la imagen vectorizada en triángulos y cuadrados rotados. Aunque no presenta invarianza a rotación, sí las presenta a traslación y escala. En este caso, la invarianza a rotación se consigue en el clasificador, entrenándolo con imágenes rotadas. Esta clara victoria de la DFT en este *dataset* se presenta en el Capítulo 5.

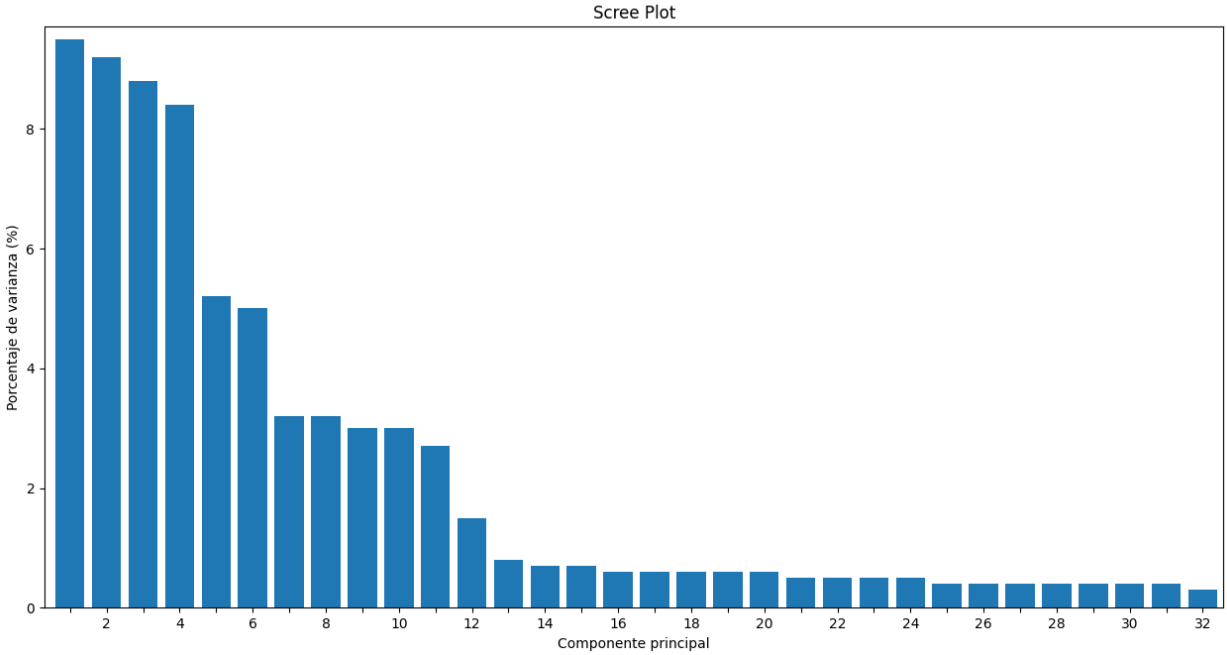

Figura 4.13 *Scree plot* de los prototipos generados mediante la DFT, para triángulos y cuadrados rotados.

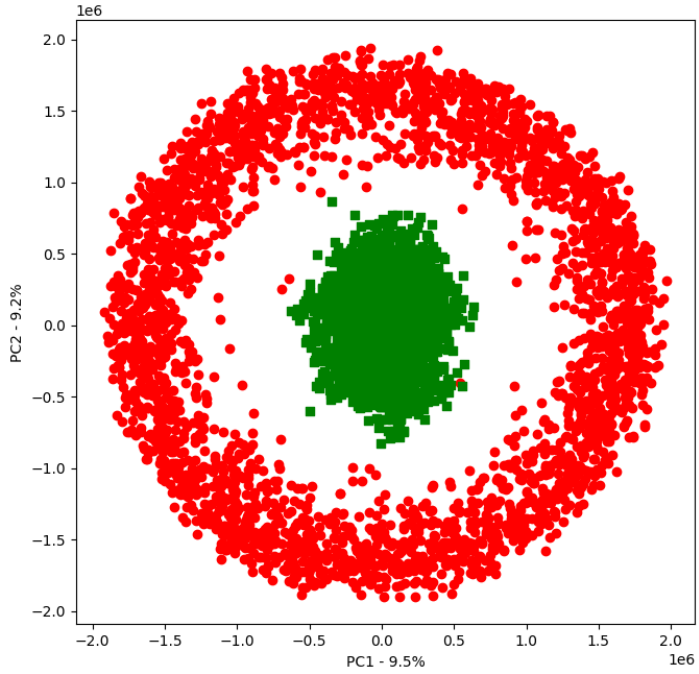

Figura 4.14 Visualización de los prototipos generados por la DFT, proyectados sobre 2 componentes principales, para triángulos y cuadrados rotados y descriptor simétrico completo.

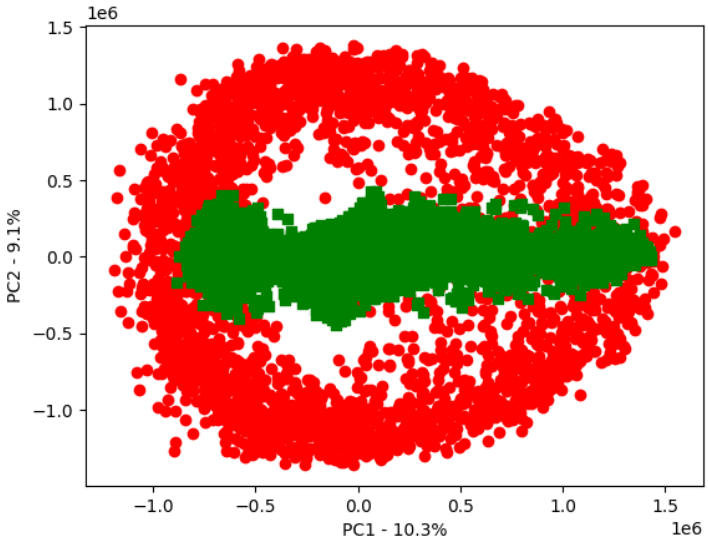

Figura 4.15 Visualización de los prototipos generados por la DFT, proyectados sobre 2 componentes principales, para triángulos y cuadrados rotados y descriptor reducido a la mitad.

V-ceils

En triángulos y cuadrados rotados, este descriptor es un candidato aún peor que en el caso anterior. Las figuras aparecen rotadas, por lo que la dirección de máxima varianza de cada celda no aporta información significativa.

En la Figura 4.16 se puede comprobar lo mismo que en el caso anterior, mucha información de este descriptor es prescindible para este caso. En la Figura 4.17 se confirma que estos datos no son separables con pocos componentes. En el Capítulo 5 se observarán los resultados.

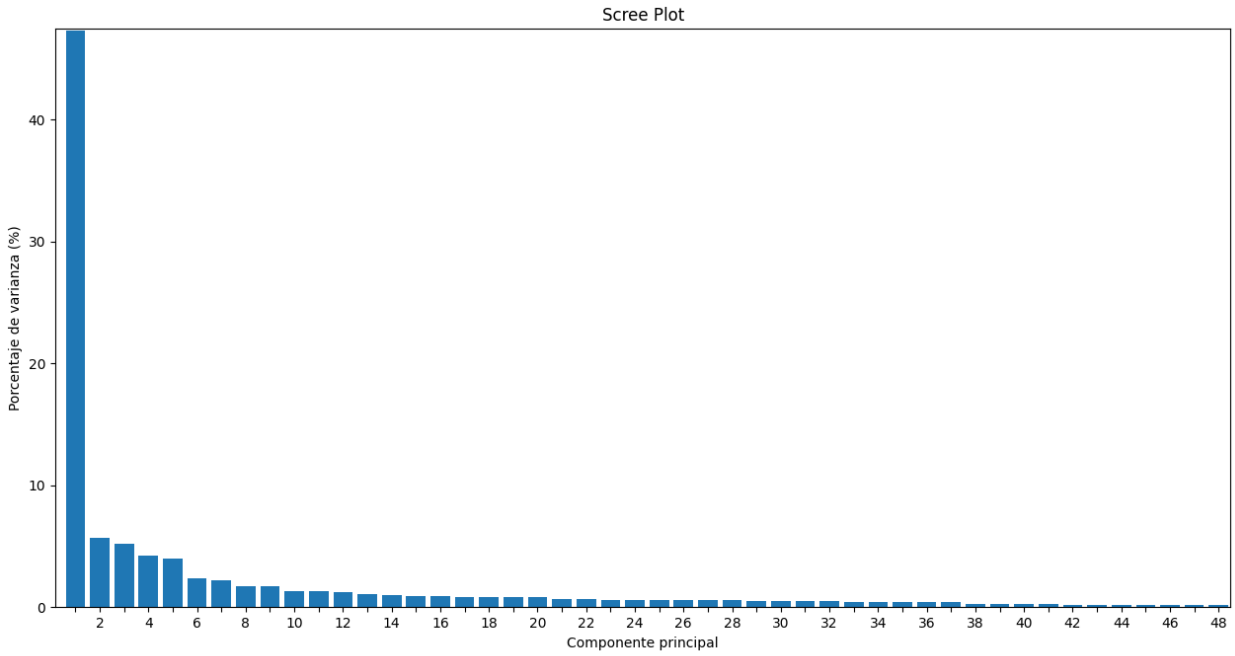

Figura 4.16 *Scree plot* de los prototipos generados mediante V-ceils, para triángulos y cuadrados sin rotar.

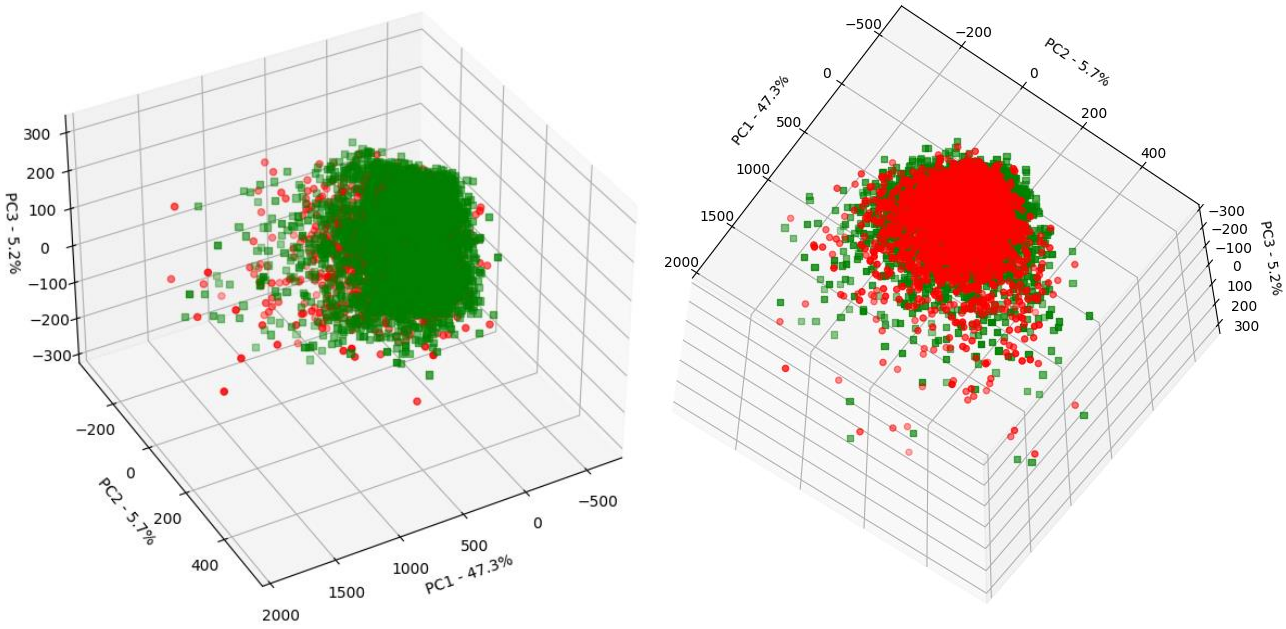

Figura 4.17 Visualización de los prototipos generados mediante V-ceils, proyectados sobre 3 componentes principales, para triángulos y cuadrados sin rotar.

4.4 *Dataset* de escenarios reales

Consideramos ahora el *dataset* para los escenarios de vuelo del robot ornitóptero. Las imágenes analizadas en este capítulo se han generado con un número fijo de eventos N=7000, para un total de 6 vuelos. Aunque se dispone de 9 vuelos en total (3 de cada clase) se ha optado por reservar uno en cada escenario para probar, en el Capítulo 5, la robustez del método. En esta sección se evalúa la dispersión de los datos, al igual que en las anteriores, para los descriptores considerados. En el Capítulo 5 se analizan los resultados obtenidos, enfocados a realizar una buena clasificación de los distintos escenarios en tiempo real. En las Figuras 4.18, 4.19 y 4.20, se muestran imágenes pertenecientes a los distintos escenarios. Cabe señalar que, incluso para un humano, resultaría difícil en ciertos casos reconocer a qué escenario pertenecen algunas imágenes. En estas figuras se muestran solo algunas de las imágenes, por lo que no puede apreciarse plenamente la dificultad de la clasificación. Los distintos vuelos describen las trayectorias del ornitóptero hasta que aterriza en condiciones de seguridad, y las maniobras realizadas dependían principalmente de las condiciones del viento [26].

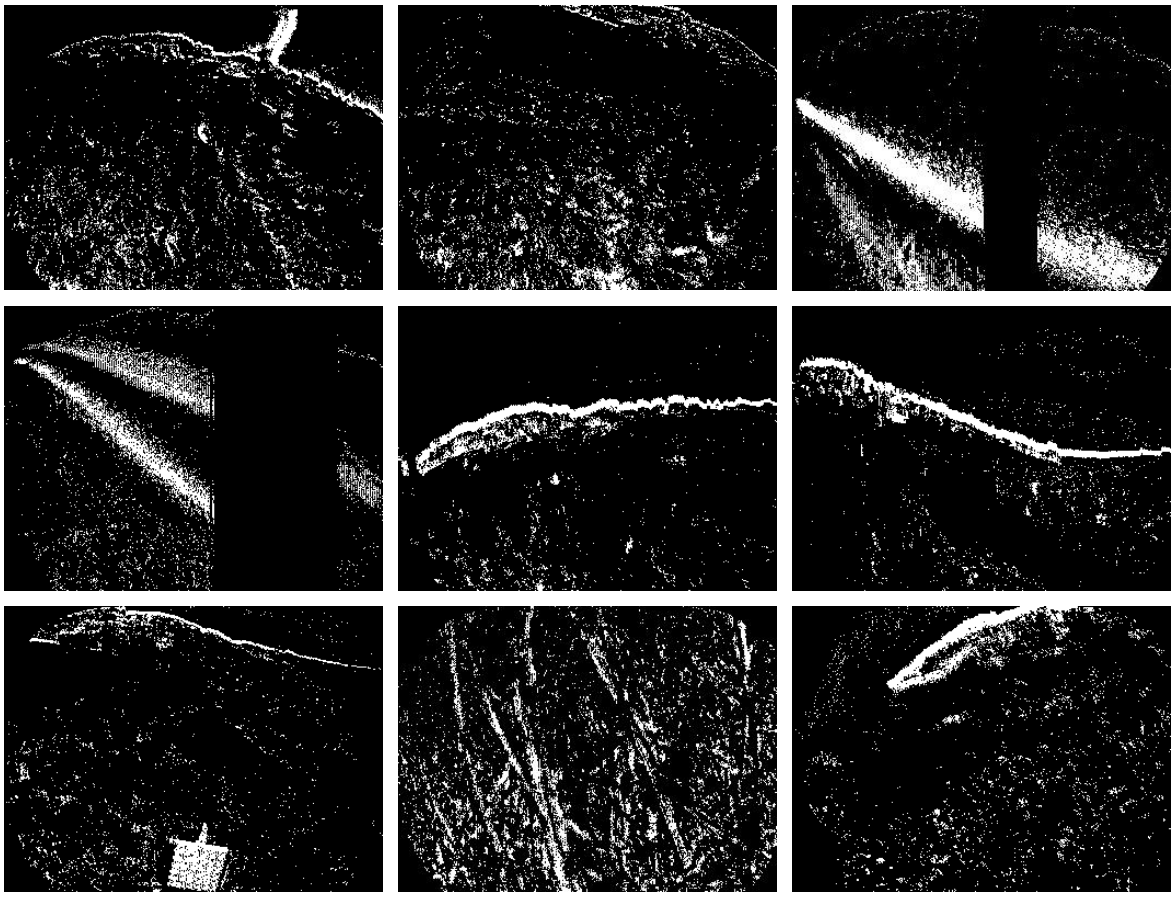

Figura 4.18 Imágenes binarias creadas a partir de los eventos en Hills.

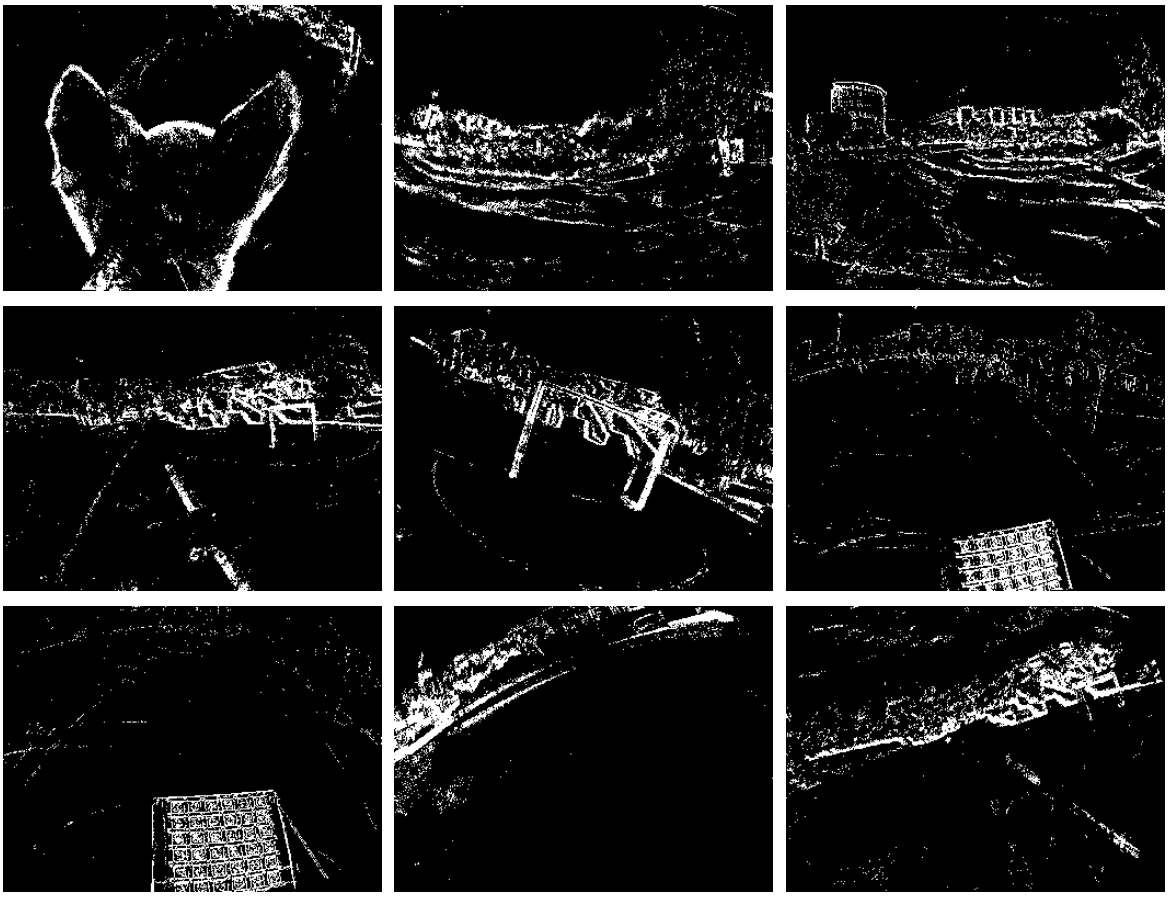

Figura 4.19 Imágenes binarias creadas a partir de los eventos en Soccer.

En lo que sigue a este capítulo, se hablará de que se implementa un clasificador SVM con una función kernel determinada, para clasificar los escenarios de las tres clases, cuando en realidad se implementan internamente tres SVM en un mismo clasificador. El algoritmo SVM clásico funciona únicamente para separar los datos en dos clases mediante un hiperplano, pero se han desarrollado variantes de este para que sea aplicable al caso en el que se tienen *N* clases. Las estrategias más empleadas para solucionar este problema son: *one-versus-one*, *one-versus-all* y DAGSVM (*Directed Acyclic Graph SVM*).

One-versus-one

Esta estrategia consiste en generar un total de $N(N-1)/2$ SVMs, comparando todos los posibles pares de clases. Para generar una predicción se emplean todos los clasificadores, y se registra el número de veces que la muestra es asignada a cada una de las clases. Se considera que la observación pertenece a la clase que ha sido asignada más veces. Este método de SVM multiclase es adecuado cuando no se tienen muchas clases, porque para un número *N* elevado de clases el número de máquinas SVM necesarias se dispara.

One-versus-all

En este caso se construyen *N* SVMs, cada uno comparando una de las clases con las restantes. Para realizar una predicción, se asigna la muestra a la clase para la cual la predicción resulte positiva. Esta aproximación no es demasiado buena, ya que puede darse el caso en el que una muestra sea etiquetada con dos clases. Además, se entrena el clasificador de forma no balanceada, es decir, con distinto número de muestras para las dos subclases generadas en cada SVM.

DAGSVM

Este método es una mejora del *one-versus-one*. La estrategia que sigue es la misma, pero se consigue reducir su tiempo de ejecución eliminando comparaciones innecesarias gracias al empleo de un DAG (*Directed Acyclic*

Graph).

En nuestro método se ha decidido emplear la estrategia *one-versus-one*, ya que se tienen únicamente tres clases.

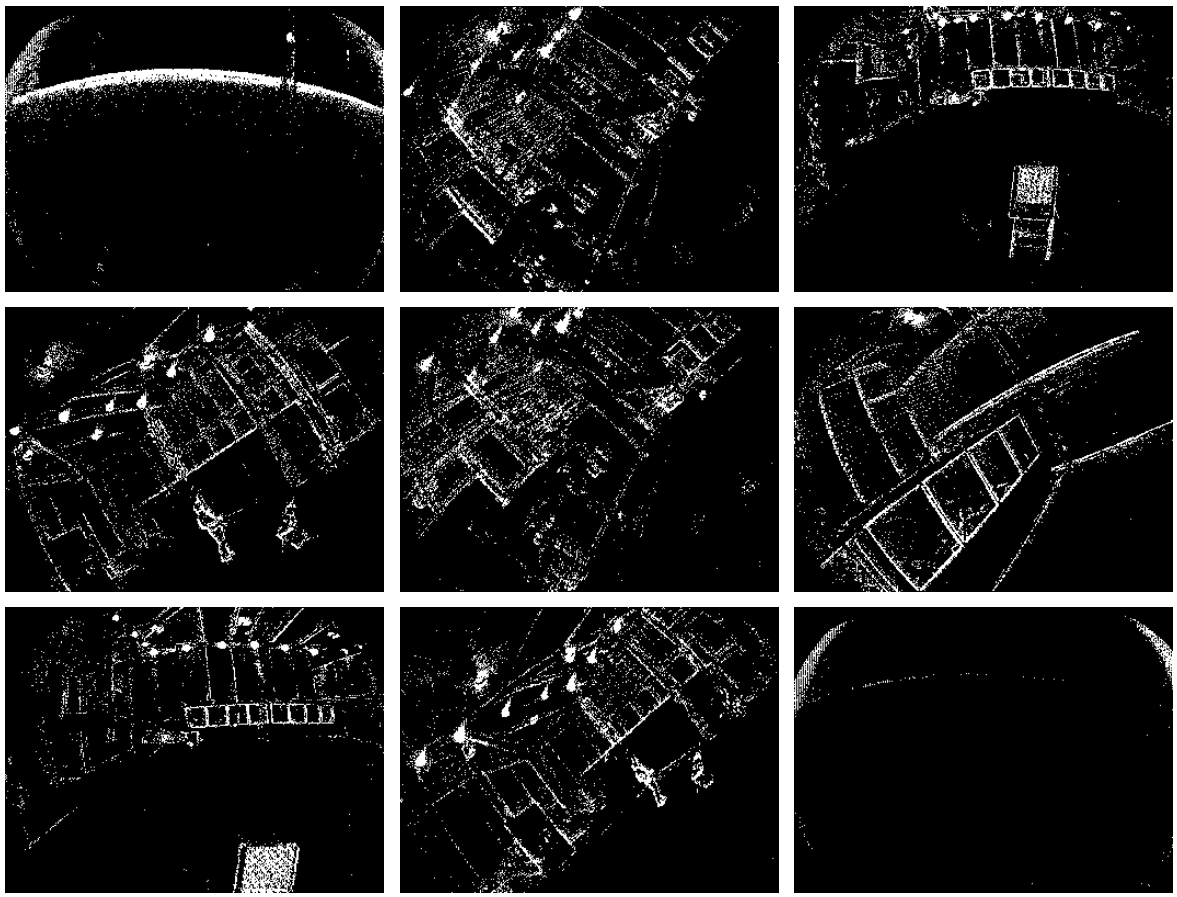

Figura 4.20 Imágenes binarias creadas a partir de los eventos en Testbed.

Imagen vectorizada

A continuación, se analizan los prototipos generados vectorizando la imagen mediante PCA.

En la Figura 4.21 se observa que los primeros componentes principales no representan un gran porcentaje de la varianza de los datos. En principio, no se podrá llegar a reducir demasiado el espacio de características de los datos a clasificar.

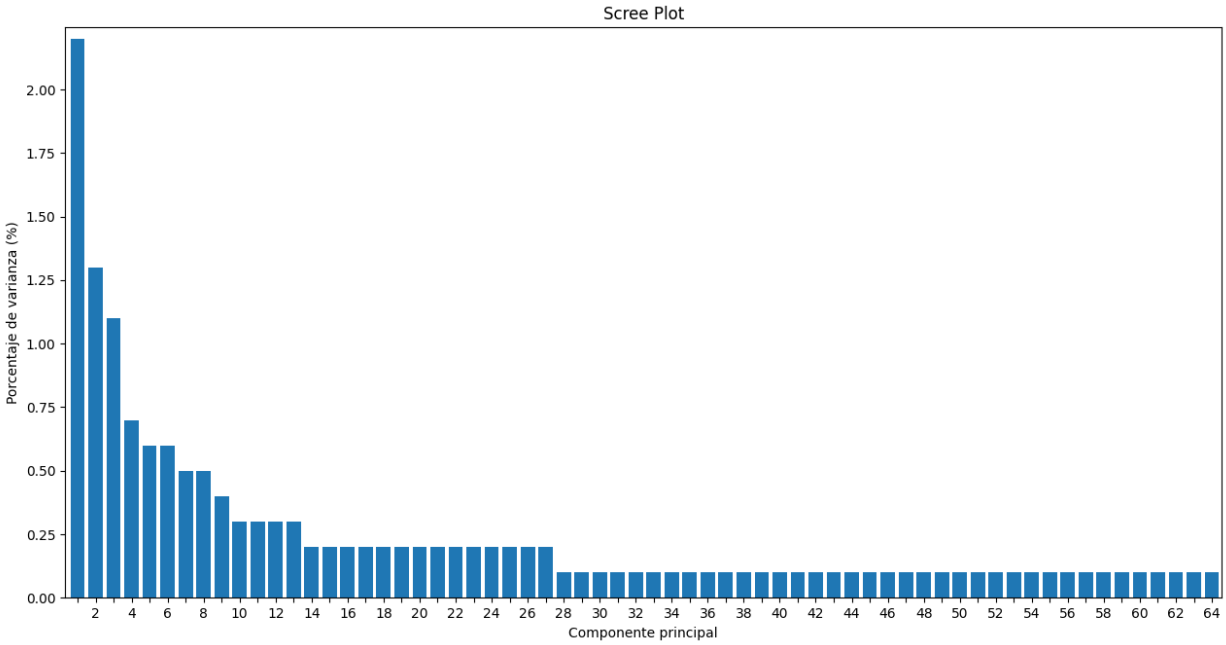

Figura 4.21 *Scree plot* de los prototipos generados vectorizando la imagen, para escenarios.

En la Figura 4.22 se representan los datos de entrenamiento correspondientes a cada escenario, dos a dos, proyectados sobre las direcciones de los tres componentes principales de mayor varianza muestral. Se representan dos a dos debido a que nuestro clasificador SVM utiliza la estrategia *one-versus-one*. En el caso de Hills y Soccer parece que los datos tienden a ser linealmente cuasiseparables en una dimensión mayor, mientras que en los dos otros casos es obvio que no lo son. Por todo lo anterior, en nuestra aplicación se han probado funciones kernel polinómicas de distinto grado y el kernel RBF, para reducir el número de componentes.

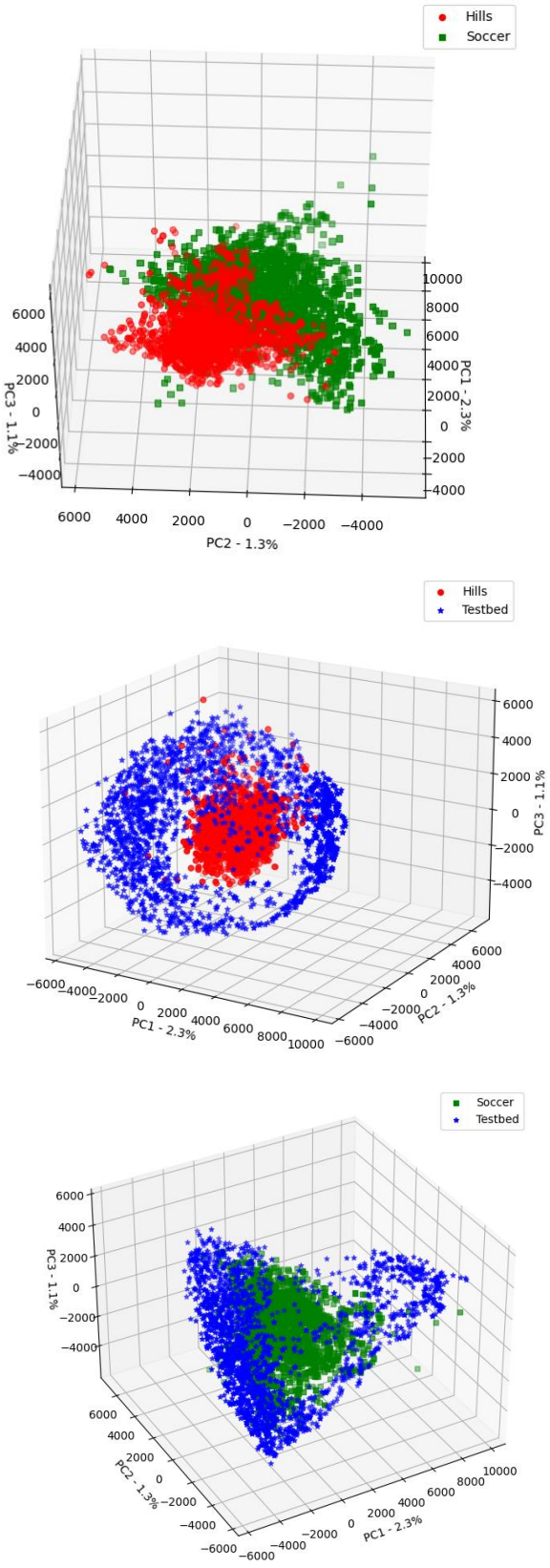

Figura 4.22 Visualización de los datos de los escenarios separados 2 a 2, proyectados sobre los 3 primeros componentes principales.

DFT

En este apartado, se analizan los prototipos generados con el descriptor DFT mediante PCA, para la DFT con 44980 componentes. Se ha reducido a la mitad porque se ha comprobado experimentalmente que, para este caso particular, la simetría que presenta el descriptor no aporta mejoras significativas en cuanto a PCI (es aproximadamente un 0.5% menor con el descriptor completo). Además, en la clasificación de escenarios en vuelos de un ornitóptero, el tiempo de clasificación juega un papel importante. Este se reduce considerablemente (es casi la mitad) si se emplea la mitad simétrica de la magnitud de la DFT.

En la Figura 4.23 se observa que los primeros componentes principales representan mayor porcentaje de la varianza total que en el caso de la imagen vectorizada. Es por esto por lo que la DFT parece, a priori, un descriptor más adecuado para realizar una reducción PCA del espacio de características.

En la Figura 4.24 se representan dos a dos las muestras de cada escenario proyectadas. En este caso, los datos tampoco son linealmente separables con tres componentes. El caso más desfavorable se presenta entre Soccer y Testbed. Se necesita un número mayor de componentes principales para poder reducir su dimensión. Para este caso se ha probado a clasificar mediante SVM con kernel lineal y RBF.

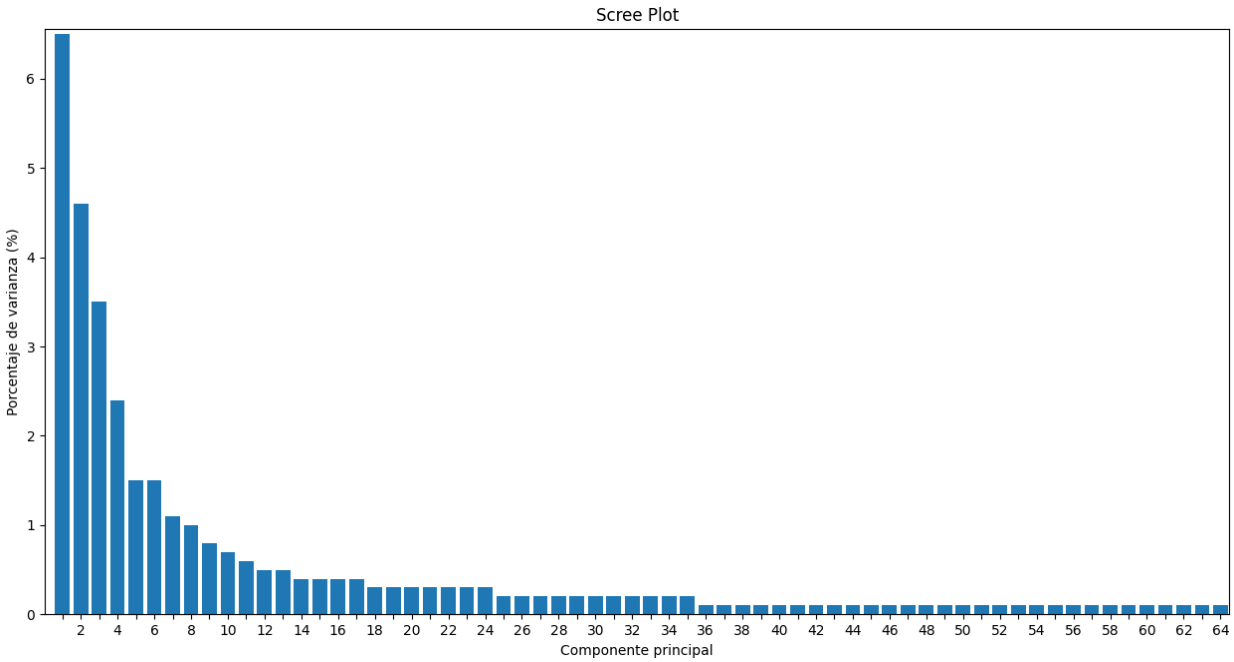

Figura 4.23 *Scree plot* de los prototipos generados por la DFT, para escenarios.

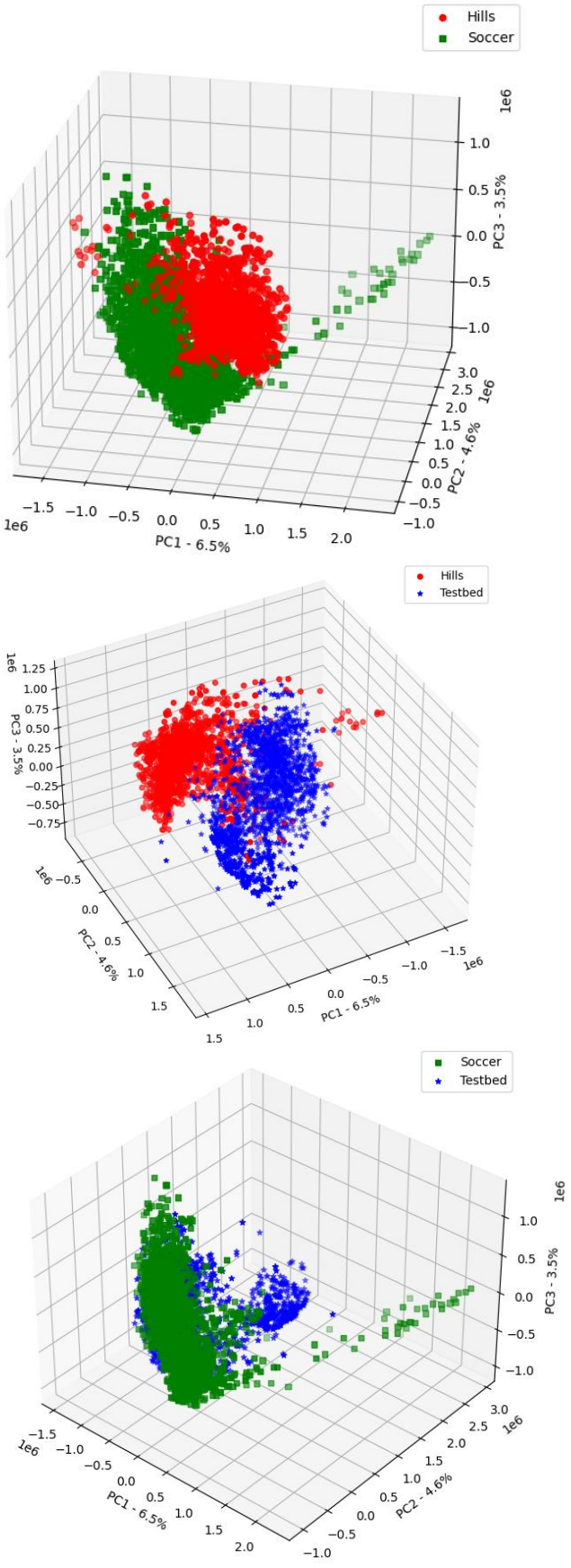

Figura 4.24 Visualización de los datos de los escenarios separados 2 a 2, proyectados sobre los 3 primeros componentes principales.

V-ceils

En este apartado, se analizan los prototipos generados con el descriptor V-ceils mediante PCA.

En la Figura 4.25 se muestra el *Scree Plot* de los datos, donde parece, a priori, que reducir el descriptor a 6, 12, 20 o 26 componentes son buenas opciones, ya que se produce una disminución de la varianza que representan los componentes principales a partir de ese punto.

En la Figura 4.26 se representan dos a dos las muestras de cada escenario proyectadas. En este caso, los datos tampoco son linealmente separables con tres componentes. Se necesita un número mayor de componentes principales para realizar una clasificación adecuada. Para este caso se ha probado a clasificar mediante SVM con kernel lineal y polinómicos.

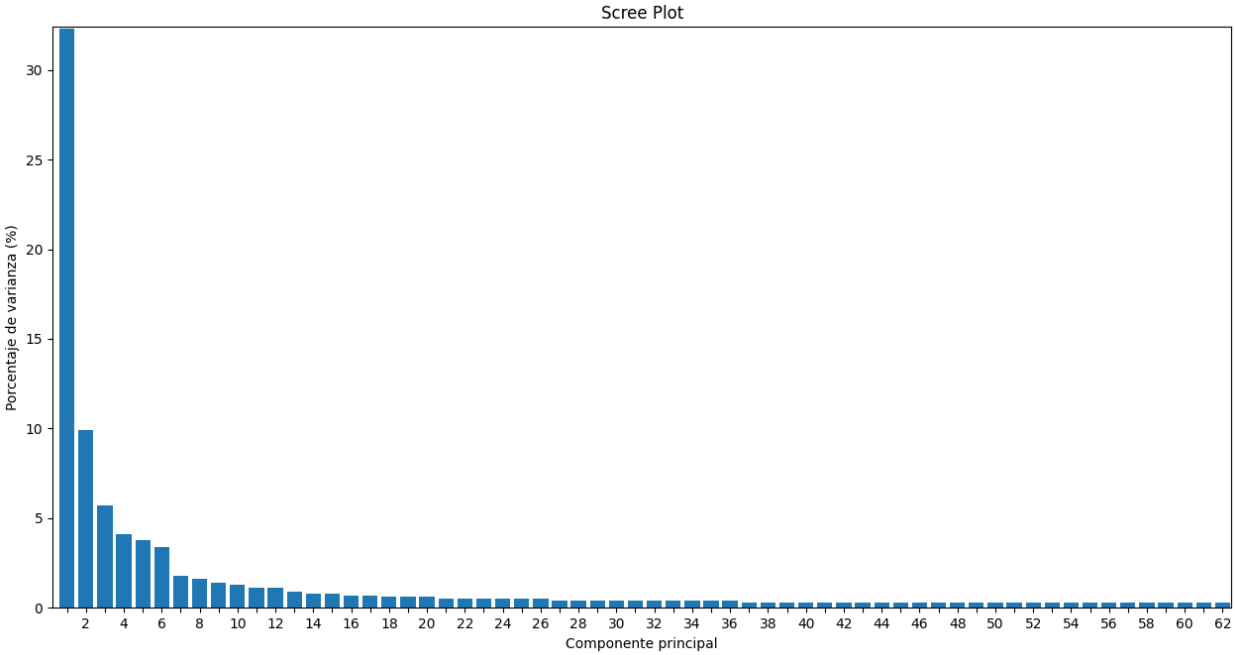

Figura 4.25 *Scree plot* de los prototipos generados por V-ceils, para escenarios.

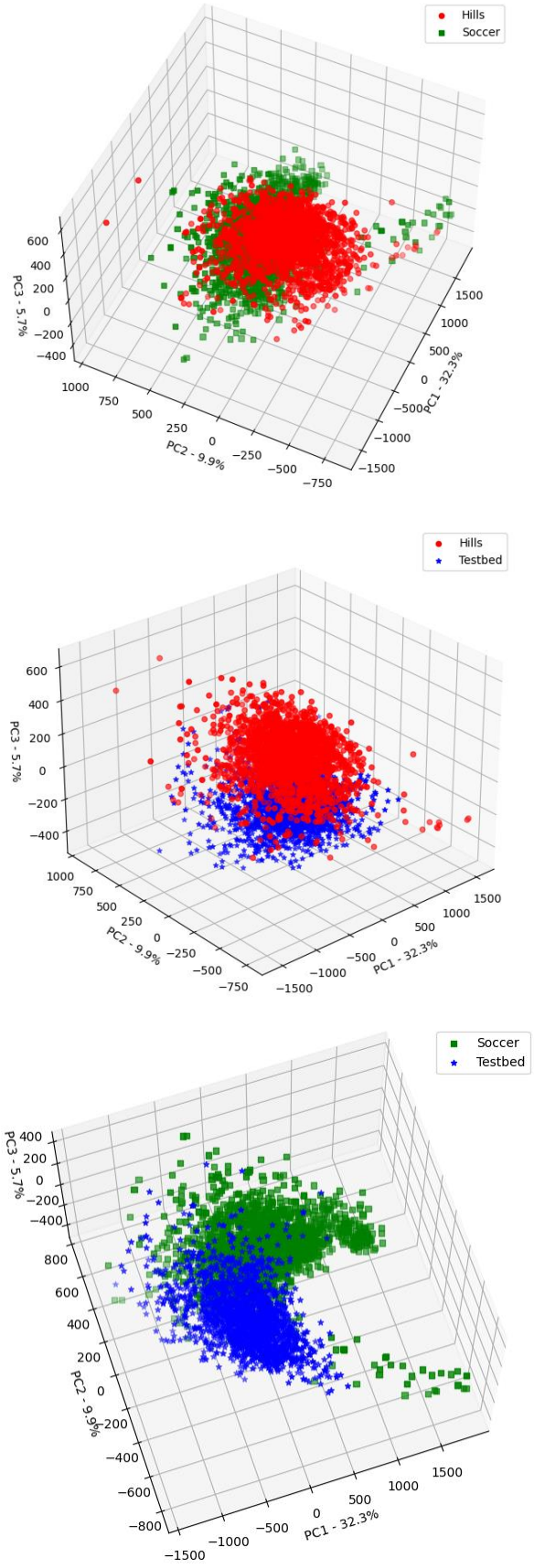

Figura 4.26 Visualización de los datos de los escenarios separados 2 a 2, proyectados sobre los 3 primeros componentes principales.

5 EXPERIMENTOS

El objetivo de este capítulo es argumentar la validez de nuestro método para clasificar imágenes binarias de eventos. También se destaca su posible aplicación en tiempo real.

Se mostrarán los experimentos realizados. Se evaluarán los distintos descriptores probados en cada *dataset*, en términos de PCI (Porcentaje de Clasificaciones Incorrectas) y tiempo de validación. Se entenderá por tiempo de validación el tiempo transcurrido desde el instante en el que se generó la imagen hasta el instante en el que el clasificador indica la clase predicha, lo cual incluye los tiempos de generación del descriptor, proyección al espacio reducido de características (si se aplica PCA) y predicción del clasificador. Se considerarán imágenes generadas con un mismo número de eventos, que no sufren del conocido problema de escala en eventos; e imágenes generadas en intervalos de tiempo fijo, consecuentemente afectadas por contener diferente número de eventos.

5.1 Experimentos para triángulos y cuadrados sin rotar

En esta sección se muestran los resultados de los descriptores para el caso más sencillo de clasificación, que son los eventos generados en triángulos y cuadrados sin rotar. Únicamente se analizan imágenes generadas con un número de eventos fijos, ya que este *dataset* es sencillo de clasificar y la escala en eventos se hace más notable en escenarios más complejos. Este efecto se analiza en la Sección 5.3.

En la Tabla 5.1 se comparan los resultados obtenidos en función del descriptor empleado, sin aplicación de PCA. Puede observarse como todos realizan una clasificación perfecta, aunque el tiempo de la imagen vectorizada es inferior.

En la Tabla 5.2, sin embargo, se aplica la técnica PCA para reducir los datos. En este caso, se hace notar la superioridad de la DFT frente a los otros dos métodos, ya que con solo un escalar como descriptor realiza una clasificación perfecta. Los datos son linealmente separables en un espacio unidimensional, lo que es una prueba clara de las invarianzas a traslación y escala que presenta la DFT.

Tabla 5.1 Resultados para clasificación de triángulos y cuadrados sin rotar, sin reducción de componentes.

Descriptor	**Número de eventos**	**Número de componentes**	**PCA**	**Kernel**	**PCI** (%)	**Tiempo de validación por imagen** (ms)
Imagen vectorizada	7000	89960	No	Lineal	0	< 1
DFT	7000	44980	No	Lineal	0	< 2
V-ceils	7000	310	No	Lineal	0	< 8

Tabla 5.2 Resultados para clasificación de triángulos y cuadrados sin rotar, con reducción de componentes.

Descriptor	**Número de eventos**	**Número de componentes**	**PCA**	**Kernel**	**PCI** (%)	**Tiempo de validación por imagen** (ms)
Imagen vectorizada	7000	15	Sí	RBF	< 1	< 6
DFT	7000	1	Sí	Lineal	0	< 2
V-ceils	7000	14	Sí	Polinómico de grado 2	< 0.5	< 8

5.2 Experimentos para triángulos y cuadrados rotados

Se analizan en esta sección los resultados para el *dataset* de triángulos y cuadrados rotados. En las tablas se presentan los mejores resultados obtenidos con cada descriptor, tras numerosas pruebas con distintos kernels y diferente número de componentes principales en el caso de reducción PCA.

En la Tabla 5.3 se comparan los resultados obtenidos en función del descriptor empleado, sin aplicación de PCA. La DFT es la mejor en este dataset, en cuanto a tiempo y PCI.

En la Tabla 5.4, se aplica la técnica PCA para reducir los datos. En este caso, la superioridad de la DFT es aún mayor frente al resto de métodos, ya que con solo dos componentes principales realiza una clasificación perfecta. Los datos son linealmente separables en un espacio de características bidimensional. En estos casos, en los que se necesita clasificar imágenes con invarianza a rotación, traslación y escala, la DFT es la mejor opción, ya que presenta invarianza a traslación y escala. Además, en la Tabla 5.4, puede apreciarse como V-ceils no responde bien a una reducción de sus componentes. Esto se debe a que es un descriptor de menor dimensión que el resto, y por ello ya presenta de forma intrínseca una reducción considerable de la imagen.

Como se intuyó en el Capítulo 4, V-ceils no presenta buenos resultados para este caso concreto, ya que no ha sido diseñado para este tipo de *datasets*, en el que ambas clases presentan una dispersión similar de los eventos en la imagen. Vectorizar la imagen no es una buena opción si se quieren conseguir invarianzas.

Tabla 5.3 Resultados para clasificación de triángulos y cuadrados rotados, sin reducción de componentes.

Descriptor	**Número de eventos**	**Número de componentes**	**PCA**	**Kernel**	**PCI** (%)	**Tiempo de validación por imagen** (ms)
Imagen vectorizada	7000	89960	No	RBF	< 7	< 500
DFT	7000	44980	No	Lineal	< 3	< 3
V-ceils	7000	310	No	Polinómico de grado 2	< 5	< 7

Tabla 5.4 Resultados para clasificación de triángulos y cuadrados rotados, con reducción de componentes.

Descriptor	Número de eventos	Número de componentes	PCA	Kernel	PCI (%)	Tiempo de validación por imagen (ms)
Imagen vectorizada	7000	23	Sí	Polinómico de grado 3	< 5	< 1
DFT	7000	2	Sí	Polinómico de grado 2	0	< 5
V-ceils	7000	20	Sí	Polinómico de grado 2	< 9	<10

5.3 Experimentos para escenarios

En esta sección se presentan más resultados que en las secciones anteriores, ya que son de mayor interés debido a la complejidad de la clasificación.

Se muestra inicialmente una comparativa de los diferentes descriptores, evaluados en imágenes generadas con número fijo de eventos. Se considera el caso de vuelos entrenados y desconocidos. Se habla de vuelos entrenados cuando se ha realizado la validación con imágenes de los vuelos 1 y 2, que son con los que se ha entrenado el clasificador; y de vuelos desconocidos cuando se ha validado con imágenes de vuelos desconocidos para el clasificador. Es importante recalcar que el conjunto de imágenes de entrenamiento y validación se ha separado en los dos casos. Las imágenes se han generado de forma aleatoria para los eventos de cada vuelo, y se han separado los conjuntos de entrenamiento y validación, como se explica en la Sección 4.1. Sin embargo, sería razonable dudar del correcto funcionamiento del método en escenarios tan complejos y variables con el tiempo, ya que, como se comenta en la Sección 4.4, las imágenes generadas en cada vuelo pueden variar significativamente. No obstante, en esta sección se confirma la validez de nuestro método, especialmente con el uso de la DFT como descriptor, para vuelos desconocidos del ornitóptero.

Podría ocurrir que, en algunas aplicaciones, generar imágenes de eventos a frecuencia constante fuera un requisito. En robótica, el clasificador sería una parte de un sistema mucho más complejo, que puede necesitar que cada imagen se genere en un determinado número de ciclos de reloj. También podrían combinarse las dos formas de generación de imágenes (fijando el número de eventos o el intervalo de tiempo), considerando un número de eventos fijo por imagen si no se supera un intervalo de tiempo predeterminado. En cualquier caso, es interesante evaluar la validez de cada descriptor frente a imágenes generadas para un intervalo de tiempo fijo. En nuestro caso, se ha decidido considerar imágenes generadas cada 5 ms. Como se dijo en capítulos anteriores, esto evalúa la invarianza ante escala en eventos de cada descriptor, lo que puede ser determinante en algunas aplicaciones.

Clasificación de imágenes con número fijo de eventos

En este apartado se presentan resultados para imágenes generadas con un número fijo de eventos.

La Tabla 5.5 muestra los resultados de la validación de imágenes extraídas en vuelos conocidos, para los diferentes descriptores y sin aplicación de PCA. Si no se considera la etapa de reducción de componentes, todos los descriptores obtienen un PCI inferior al 2%. En el caso de V-ceils, el tiempo de validación es superior al resto, debido a que el tiempo de formación de este descriptor es mayor. Es importante destacar que los resultados de la Tabla 5.5 son para imágenes generadas con un número considerable de eventos (7000), por lo que la imagen vectorizada funciona correctamente, e incluso en menos tiempo que la DFT para este caso particular. Sin embargo, en muchas aplicaciones puede ser conveniente tomar imágenes de eventos a mayor frecuencia, es decir, con menor número de eventos por imagen. En la Tabla 5.7 se muestran, para vuelos entrenados, los

resultados de los distintos descriptores para imágenes formadas con 100, 500 y 1000 eventos. En este caso, la imagen vectorizada funciona peor que los otros dos métodos, dejando de funcionar en el caso extremo de 100 eventos por imagen. En la Figura 5.4 se muestran algunas imágenes generadas con 1000, 500 y 100 eventos. Es interesante como para imágenes de solo 100 eventos la DFT y V-ceils funcionan relativamente bien.

En la Figura 5.1 se muestran los resultados de los diferentes descriptores, en cuanto a PCI, tras aplicarle reducción PCA, para los vuelos entrenados. En el eje horizontal se representa el número de componentes resultantes tras la reducción, y en el eje vertical el PCI obtenido por cada descriptor reducido. En este caso, se observa como la DFT es la que ofrece, con diferencia, mejores resultados. Puede reducir el descriptor a un número de componentes menor que el resto, obteniendo en cualquier caso un PCI inferior. Por ejemplo, para 23 componentes, tiene un PCI de aproximadamente 0.5%, que es mejor resultado que el que ofrece sin reducción de componentes. Adicionalmente, en la Figura 5.3, puede comprobarse como la DFT presenta tiempos menores de validación para los descriptores reducidos.

La Tabla 5.6 es análoga a la tabla 5.5, pero para vuelos desconocidos. Puede observarse la robustez del método, que sigue funcionando en este caso, obteniendo PCIs inferiores al 9% en cada descriptor no reducido.

La Figura 5.2 es análoga a la 5.1, pero para vuelos descocidos. La DFT vuelve a ser la ganadora en este caso, y se confirma la validez de su aplicación en vuelos desconocidos.

Las tablas 5.8, 5.9 y 5.10 son las matrices de confusión de los mejores resultados de cada descriptor en vuelos desconocidos. La Tabla 5.8 muestra los resultados para kernel RBF y reducción del descriptor DFT a 25 componentes. La Tabla 5.9 presenta los resultados para kernel RBF y reducción de la imagen vectorizada a 27 componentes. La Tabla 5.10 expresa los resultados de V-ceils, con kernel polinómico de grado 2 y sin reducción de componentes.

Tabla 5.5 Resultados para clasificación de los vuelos entrenados, sin reducción de componentes.

Descriptor	**Número de eventos**	**Número de componentes**	**PCA**	**Kernel**	**PCI** (%)	**Tiempo de validación por imagen** (ms)
Imagen vectorizada	7000	89960	No	Lineal	< 2	< 0.5
DFT	7000	44980	No	Lineal	< 2	< 1
V-ceils	7000	310	No	Polinómico de grado 2	< 2	< 10

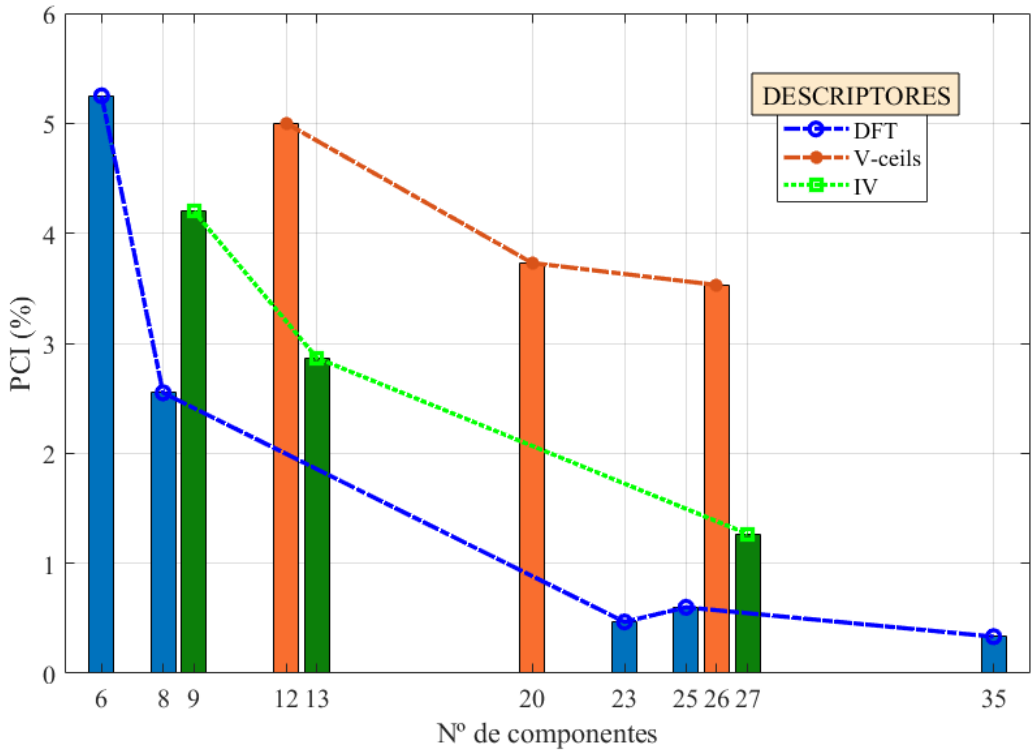

Figura 5.1 Representación de cada descriptor en función del número de componentes principales y PCI, evaluados en los vuelos entrenados.

Tabla 5.6 Resultados para clasificación de vuelos desconocidos, sin reducción de componentes.

Descriptor	**Número de eventos**	**Número de componentes**	**PCA**	**Kernel**	**PCI** (%)	**Tiempo de validación por imagen** (ms)
Imagen vectorizada	7000	89960	No	Lineal	< 7	< 0.5
DFT	7000	44980	No	Lineal	< 8	< 1
V-ceils	7000	310	No	Polinómico de grado 2	< 9	< 10

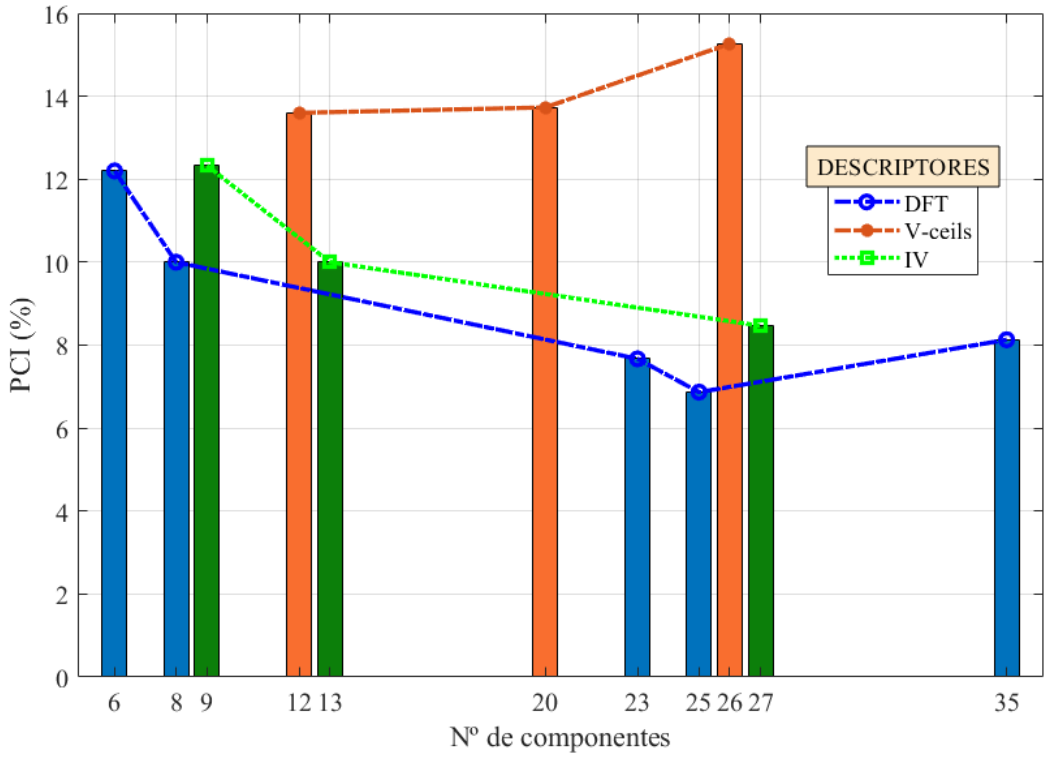

Figura 5.2 Representación de cada descriptor en función del número de componentes principales y PCI, evaluados en vuelos desconocidos.

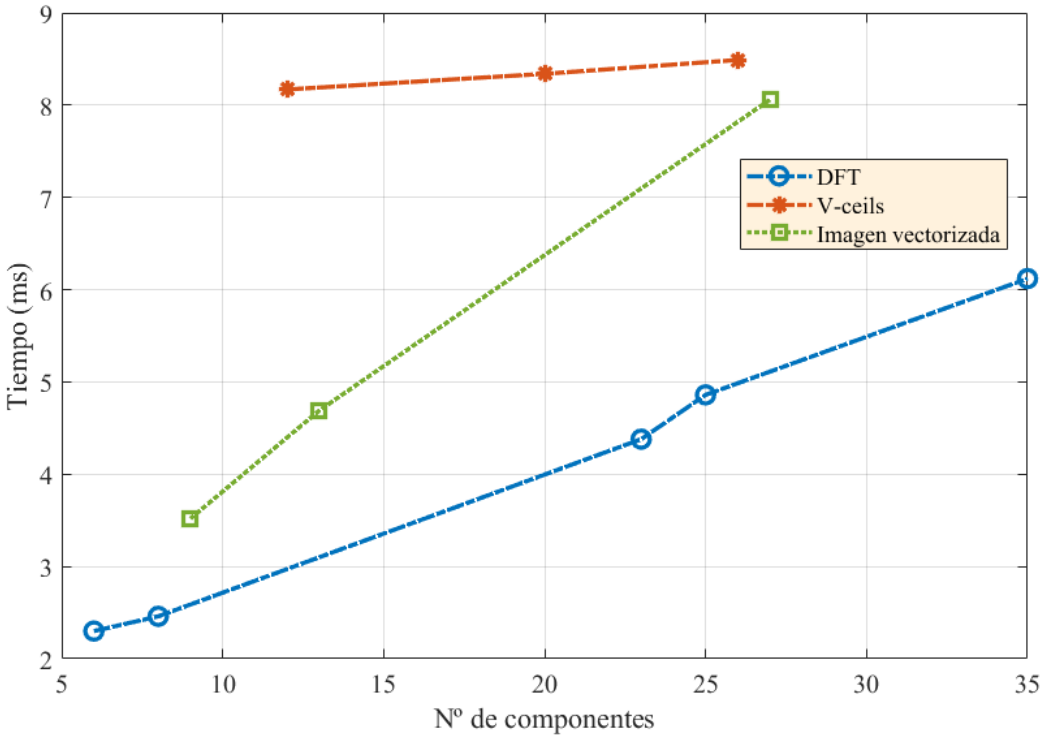

Figura 5.3 Representación del tiempo de validación de una sola imagen frente al número de componentes tras la reducción, para cada descriptor.

Tabla 5.7 Resultados para clasificación de los vuelos entrenados, con menor número de eventos.

Descriptor	Número de eventos	Número de componentes	PCA	Kernel	PCI (%)
Imagen vectorizada	1000	27	Sí	RBF	< 6
	500				< 9
	100				42
DFT	1000	23	Sí	RBF	< 3
	500				< 4
	100				<15
V-ceils	1000	310	No	Polinómico de grado 2	< 5
	500				< 6
	100				<15

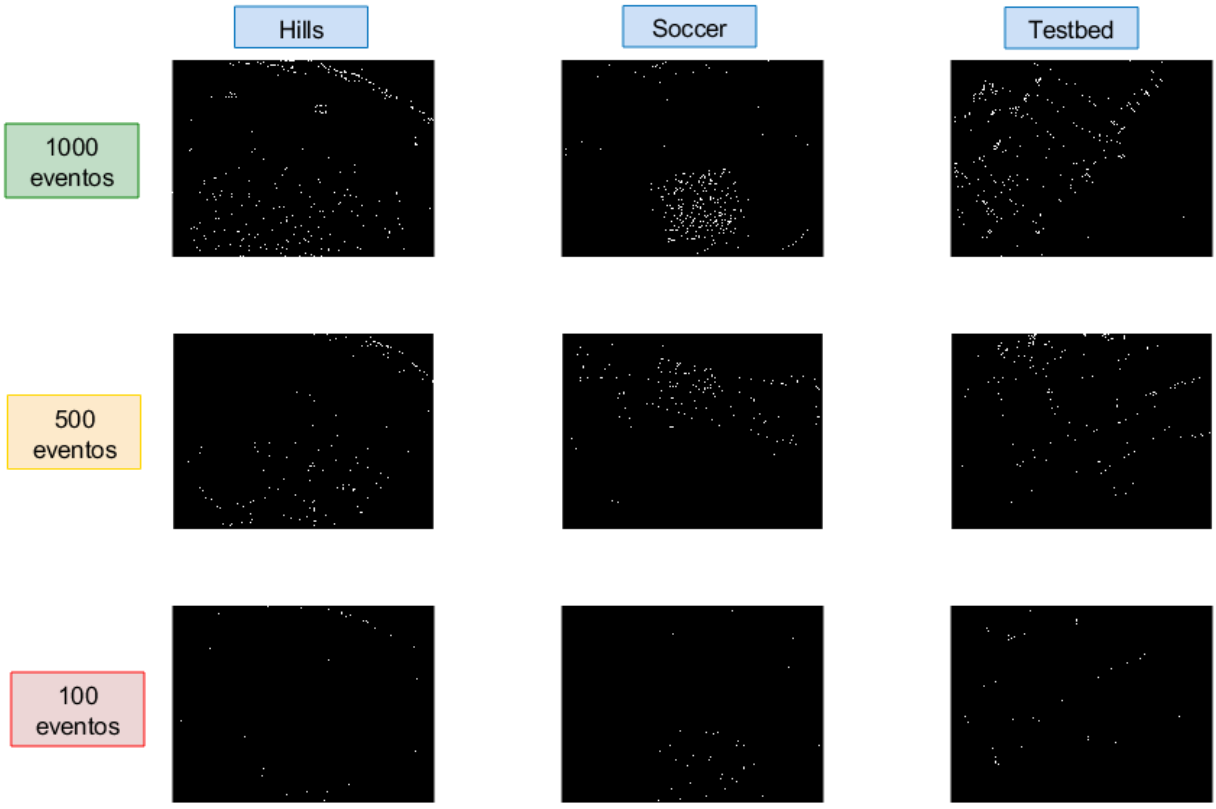

Figura 5.4 Imágenes en escenarios para número de eventos progresivamente menor.

Tabla 5.8 Matriz de confusión para DFT en vuelos desconocidos.

	Clase real			
Clase predicha		**Testbed**	**Hills**	**Soccer**
	Testbed	491	2	7
	Hills	11	470	19
	Soccer	7	57	439

Tabla 5.9 Matriz de confusión para imagen vectorizada en vuelos desconocidos.

	Clase real			
Clase predicha		**Testbed**	**Hills**	**Soccer**
	Testbed	497	0	3
	Hills	19	451	30
	Soccer	16	59	425

Tabla 5.10 Matriz de confusión para V-ceils en vuelos desconocidos.

	Clase real			
Clase predicha		**Testbed**	**Hills**	**Soccer**
	Testbed	458	26	16
	Hills	12	467	21
	Soccer	22	47	431

Clasificación de imágenes generadas con intervalo de tiempo fijo

Se consideran ahora imágenes generadas cada 5 milisegundos, para estudiar la invarianza a escala en eventos de cada descriptor evaluando el PCI. En la Tabla 5.11 se muestran los resultados para los distintos descriptores. En este caso, la DFT debe normalizarse para conseguir esta invarianza. Para ello, solo es necesario dividir la magnitud por su primera componente, que es el número de eventos que tiene la imagen. En esta tabla no se muestran los tiempos de validación, pues no varían significativamente con respecto al caso anterior y se mantiene la relación entre los distintos descriptores. Teniendo en cuenta el tiempo de validación, número de componentes y PCI obtenido, la DFT vuelve a ser la clara vencedora en esta comparativa.

En la Figura 5.5 se muestran dos imágenes de cada escenario, generadas en intervalos de tiempo de 5 ms, donde puede apreciarse el efecto de la escala en eventos y la consecuente dificultad añadida en la clasificación.

Tabla 5.11 Resultados para clasificación de los vuelos entrenados, con intervalo de tiempo fijo y consecuente número de eventos variable.

Descriptor	**Intervalo de tiempo** (ms)	**Número de componentes**	**PCA**	**Kernel**	**PCI** (%)
Imagen vectorizada	5	27	Sí	RBF	< 9
DFT	5	25	Sí	RBF	< 4
V-ceils	5	310	No	Polinómico de grado 2	< 4

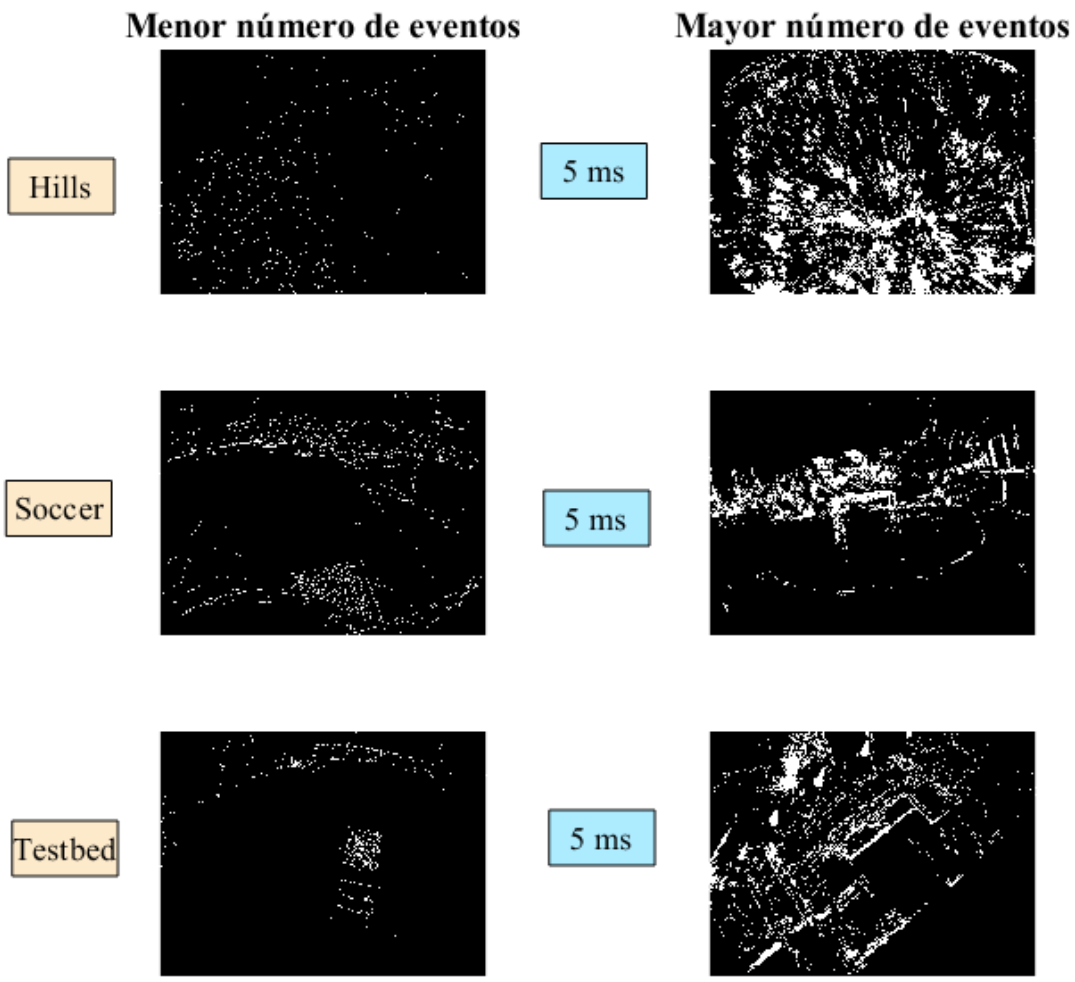

Figura 5.5 Imágenes generadas en escenarios con intervalos de tiempo fijo y consecuente número de eventos variable.

6 CONCLUSIONES Y DESARROLLO FUTURO

En este capítulo se presentan las conclusiones a las que se ha llegado durante el desarrollo de este proyecto y se comentan posibles desarrollos e investigaciones futuras. Se concluye, en base al análisis realizado y a los resultados obtenidos, la versatilidad y ventajas de emplear imágenes de eventos y descriptores basados en técnicas frecuenciales, frente a otros. También las ventajas de usar máquinas SVM para la clasificación. Además, se argumenta la aptitud del método para su aplicación en tiempo real.

6.1 Ventajas

El empleo de cámaras de eventos presenta muchas ventajas en numerosas aplicaciones, pero el potencial de estas no está todavía totalmente desbloqueado debido a su reciente aparición. En este trabajo se ha procesado su salida agrupando los eventos en imágenes binarias, mediante las cuales se genera un descriptor global que es la entrada de un clasificador.

El objetivo último de nuestro proyecto es clasificar eventos, y el método que se ha empleado presenta las siguientes ventajas en las etapas de preprocesamiento (agrupación de eventos en imágenes binarias), procesamiento central (descripción de las imágenes) y postprocesamiento (opcional reducción de componentes mediante PCA y empleo de SVM).

Ventajas de agrupar los eventos en imágenes binarias

Agrupar los eventos en imágenes binarias permite adaptar, e incluso aplicar directamente, algoritmos de visión por computador tradicionales a imágenes de eventos. También otorga información muy relevante, ya que proporciona los contornos de la imagen de forma natural y en intervalos cortos de tiempo. Además, frente a imágenes tradicionales, presentan la mayoría de las ventajas que tienen las cámaras de eventos y que se comentan en el Capítulo 2. Por ejemplo, no sufren de desenfoque por movimiento. La resolución temporal, en este caso, podría verse algo reducida con respecto al procesamiento de los eventos de forma individual o en pequeños grupos. Sin embargo, en el Capítulo 5 se muestra como para imágenes formadas con únicamente 100 eventos, mediante un buen descriptor global y un buen clasificador, se tienen resultados que pueden ser aceptables en algunas aplicaciones.

Ventajas de emplear descriptores globales

El empleo de descriptores globales frente a descriptores locales, en imágenes de eventos, no se compara en este trabajo. Sin embargo, el empleo de descriptores globales, normalmente, no sufre del ruido que introducen las cámaras de eventos, lo que es crítico para algunos descriptores locales en eventos. Este problema, en descriptores locales se soluciona filtrando la imagen de diferentes formas. En [9] aparece información sobre el empleo de diferentes filtros. De cualquier forma, el uso de filtros para las imágenes y su posterior descripción local incurriría en un tiempo demasiado alto para la clasificación en tiempo real, para la que nuestro método sí es apto.

Ventajas de emplear descriptores basados en técnicas frecuenciales

Uno de los objetivos de este proyecto ha sido demostrar la validez de la DFT como descriptor global de la imagen. Se ha comparado frente a un descriptor que caracteriza la imagen mediante la dispersión de los eventos (V-ceils) y frente a no considerar descriptor (lo que hemos llamado imagen vectorizada), en términos de clasificación.

Si comparamos V-ceils y la DFT en términos de descripción, la DFT es superior en la mayoría de los casos. Como se comenta en el Capítulo 3 y se demuestra experimentalmente en capítulos posteriores, presenta invarianzas a traslación, escala y escala en eventos. Además, mediante el empleo de coordenadas polares, se podría conseguir invarianza a rotación. Esto es muy útil si, como en la inmensa mayoría de aplicaciones robóticas, además de clasificar imágenes, se necesita extraer otro tipo de información de la escena (en conjunto con otros sensores), como la posición relativa del robot.

Si comparamos la DFT con el descriptor V-ceils o con la imagen vectorizada, en términos de clasificación, la primera es clara vencedora en la gran mayoría de situaciones, como se presenta en el Capítulo 5. Su invarianza a escala en eventos le permite clasificar correctamente imágenes generadas a partir de un menor número de eventos, comparado con la imagen vectorizada. Además, es muy rápida gracias a su implementación mediante la FFT (*Fast Fourier Transform*) y a que en imágenes binarias de eventos se calcula únicamente para imágenes de unos y ceros. También permite reducir el número de componentes que componen el descriptor, mediante PCA, obteniendo los mejores resultados; y es mejor opción que vectorizar la imagen para clasificar imágenes generadas en intervalos de tiempo fijo. En definitiva, las ventajas de emplear descriptores basados en frecuencia (en este caso la magnitud de la DFT) son muchas, han sido demostradas en este trabajo y, por todo lo anterior, se considera la elección definitiva en la etapa de descripción de nuestro método.

Ventajas de reducir los componentes del descriptor mediante PCA

La reducción de componentes del descriptor puede ser un requisito en algunas aplicaciones. En este trabajo se han demostrado las ventajas de emplear PCA para reducir la dimensión de los datos, ya que, además de proporcionar descriptores de menor dimensión, en algunos casos incluso mejora la clasificación mediante SVM, como se observa experimentalmente en el Capítulo 5.

Ventajas de usar clasificadores SVM

La ventaja principal por la cual se han empleado clasificadores SVM es porque son eficaces en espacios de grandes dimensiones, incluso en casos donde la dimensión de características es mayor que el número de muestras. Este caso ocurre en este proyecto con la DFT y la imagen vectorizada si no se reduce la dimensión del descriptor. Además, es eficiente en memoria, ya que para la función de decisión solo se emplea un subconjunto de patrones. Finalmente, es un método versátil, pues permite ajustar la función kernel en función de los datos a clasificar. Para otras aplicaciones en las que se emplee este método, hay que tener en cuenta que si el número de características es mayor que el número de muestras hay que prestar atención en no incurrir en un sobreajuste al elegir la función kernel. En nuestra aplicación, para los distintos escenarios de vuelo del robot ornitóptero, se ha comprobado que la clasificación mantiene su efectividad en vuelos no vistos en la fase de entrenamiento.

6.2 Aplicación en tiempo real

Otra característica que presenta nuestro método es su gran velocidad. Como se ha presentado en el Capítulo 5, el tiempo desde que se forma la imagen hasta que se indica su clase es del orden del milisegundo. Esto puede ser interesante en numerosas aplicaciones de robótica en las que la velocidad juegue un papel importante. Por ejemplo, podría implementarse un procesamiento paralelo, en tiempo real, en el que por un lado se predice la clase a la que pertenece una imagen y por otro se va creando la imagen siguiente. A modo de ejemplo, si se considerasen imágenes generadas en intervalos de tiempo de 1 milisegundo, el clasificador SVM con kernel lineal, empleando la DFT como descriptor de la imagen, predeciría la clase a la que pertenece antes de recibir la siguiente imagen.

6.3 Desarrollo futuro

En esta sección se presentan algunas vías de desarrollo futuro para este proyecto:

- Sería interesante comparar los resultados del método de reducción de datos ICA, el cual se comenta en el Capítulo 2, frente a los obtenidos en este proyecto mediante PCA.
- Probar el método con más *datasets* de eventos, generando imágenes binarias, permitiría defender con mayor firmeza y generalidad las ventajas que ofrecen los descriptores globales basados en frecuencia para realizar una rápida y correcta clasificación.
- Comparar los resultados de otros clasificadores frente a los obtenidos en este proyecto mediante SVM. Existen aplicaciones en las que podría no conocerse a priori la clase a la que pertenecen los datos a clasificar. Para ello, se necesitaría emplear un método de aprendizaje no supervisado. Adaptar el método para que sea útil en estas situaciones, sustituyendo el clasificador SVM por un algoritmo de clustering, es una opción interesante para considerar en el futuro.
- Reducir la magnitud de la transformada de Fourier descartando altas frecuencias, estableciendo los límites experimentales para cada *dataset* particular, podría servir para eliminar ruido en imágenes de eventos y conseguir un descriptor global de menor dimensión. Además, para casos en los que el clasificador necesitase un PCI muy bajo, podría considerarse el uso de filtros a coste de un mayor tiempo de cómputo.
- En robótica, existen numerosas aplicaciones en las que la información sobre el movimiento de la escena es muy importante. Considerar la polaridad y/o el instante de tiempo en el que se generan los eventos, aporta información sobre el movimiento relativo entre la escena y el robot. En este caso, sería necesario considerar otro tipo de representaciones de eventos, como las superficies temporales explicadas en el Capítulo 2. En robótica aérea no tripulada, por ejemplo, identificar situaciones de riesgo u obstáculos rápidamente puede ser crucial. La información sobre la velocidad a la que se aproxima un determinado objeto puede ser determinante a la hora de considerarlo o no un riesgo. El método presentado en este proyecto es rápido, por lo que sería interesante considerar variantes de este que tengan en cuenta la información sobre el movimiento que aportan las cámaras de eventos, y poder así ser aplicado en este contexto. Evaluar su rendimiento clasificando diferentes superficies temporales, diferenciando si se corresponden o no con diferentes situaciones de riesgo, es una interesante vía de desarrollo futuro.
- En este trabajo se ha presentado la aplicación de un único clasificador capaz de identificar el escenario al que pertenecen las diferentes imágenes de eventos para las que ha sido entrenado. Sin embargo, el ornitóptero, en este caso, una vez ha reconocido el escenario en el que se encuentra, podría, dentro de ese mismo escenario, querer identificar en qué parte de él se sitúa. Para ello, el robot podría migrar a un nuevo "subclasificador" SVM. Las máquinas SVM son eficientes en memoria, por lo que investigar la viabilidad de implementar SVMs anidados, su coordinación y coste computacional, es una investigación futura que considerar.

REFERENCIAS

[1] GALLEGO, Guillermo, et al. Event-based vision: A survey. arXiv preprint arXiv:1904.08405, 2019.

[2] LICHTSTEINER, Patrick; POSCH, Christoph; DELBRUCK, Tobi. A 128×128 120 dB 15 μs latency asynchronous temporal contrast vision sensor. IEEE journal of solid-state circuits, 2008, vol. 43, no 2, p. 566-576.

[3] POSCH, Christoph; MATOLIN, Daniel; WOHLGENANNT, Rainer. A QVGA 143 dB dynamic range frame-free PWM image sensor with lossless pixel-level video compression and time-domain CDS. *IEEE Journal of Solid-State Circuits*, 2010, vol. 46, no 1, p. 259-275.

[4] BRANDLI, Christian, et al. A 240×180 130 dB 3 μs latency global shutter spatiotemporal vision sensor. *IEEE Journal of Solid-State Circuits*, 2014, vol. 49, no 10, p. 2333-2341.

[5] DELBRUCK, Tobi. Frame-free dynamic digital vision. En *Proceedings of Intl. Symp. on Secure-Life Electronics, Advanced Electronics for Quality Life and Society*. 2008. p. 21-26.

[6] GEHRIG, Daniel, et al. Asynchronous, photometric feature tracking using events and frames. En *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018. p. 750-765.

[7] HARRIS, Christopher G., et al. A combined corner and edge detector. En *Alvey vision conference*. 1988. p. 10-5244.

[8] ROSTEN, Edward; DRUMMOND, Tom. Machine learning for high-speed corner detection. En *European conference on computer vision*. Springer, Berlin, Heidelberg, 2006. p. 430-443.

[9] J. A. Sánchez, "Descripción de imágenes de eventos: HOG, GIST y DFT," Sevilla, 2021.

[10] PAYÁ, Luis, et al. Using omnidirectional vision to create a model of the environment: A comparative evaluation of global-appearance descriptors. *Journal of Sensors*, 2016, vol. 2016.

[11] OLIVA, Aude; TORRALBA, Antonio. Building the gist of a scene: The role of global image features in recognition. *Progress in brain research*, 2006, vol. 155, p. 23-36.

[12] PAYÁ, Luis, et al. Performance of global-appearance descriptors in map building and localization using omnidirectional vision. Sensors, 2014, vol. 14, no 2, p. 3033-3064.

[13] A. Jeffares, "K-means: A Complete Introduction. K-means is an unsupervised clustering… | by Alan Jeffares | Towards Data Science," *Towar. Data Sci.*, Nov. 2019. [Online]. Available: https://towardsdatascience.com/k-means-a-complete-introduction-1702af9cd8c.

[14] BOSER, Bernhard E.; GUYON, Isabelle M.; VAPNIK, Vladimir N. A training algorithm for optimal

margin classifiers. En *Proceedings of the fifth annual workshop on Computational learning theory*. 1992. p. 144-152.

[15] CORTES, Corinna; VAPNIK, Vladimir. Support-vector networks. *Machine learning*, 1995, vol. 20, no 3, p. 273-297.

[16] G. Liu, S. Mao, and J. H. Kim, "A mature-tomato detection algorithm using machine learning and color analysis," *Sensors (Switzerland)*, vol. 19, no. 9, May 2019, doi: 10.3390/s19092023.

[17] JAIN, Anil K.; MURTY, M. Narasimha; FLYNN, Patrick J. Data clustering: a review. *ACM computing surveys (CSUR)*, 1999, vol. 31, no 3, p. 264-323.

[18] ABBAS, Osama Abu. Comparisons between data clustering algorithms. *International Arab Journal of Information Technology (IAJIT)*, 2008, vol. 5, no 3.

[19] SINGH, Amanpreet; THAKUR, Narina; SHARMA, Aakanksha. A review of supervised machine learning algorithms. En *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*. Ieee, 2016. p. 1310-1315.

[20] PERNKOPF, Franz. Bayesian network classifiers versus selective k-NN classifier. Pattern recognition, 2005, vol. 38, no 1, p. 1-10.

[21] MARTIRIGGIANO, Tommaso, et al. Face recognition by kernel independent component analysis. En *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*. Springer, Berlin, Heidelberg, 2005. p. 55-58.

[22] MOGHADDAM, Baback. Principal manifolds and probabilistic subspaces for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2002, vol. 24, no 6, p. 780-788.

[23] VICENTE, M. Asunción, et al. Equivalencia entre ICA y PCA como métodos de extracción de características en reconocimiento visual basado en apariencia. *Dpto. de Ingeniería de Sistemas Industriales, Universidad Miguel Hernández. Alicante. Espana*, 2007.

[24] HYVARINEN, Aapo; KARHUNEN, J.; OJA, E. Independent component analysis and blind source separation. 2001.

[25] "The GRIFFIN Perception Dataset: Bridging the Gap Between Flapping-Wing Flight and Robotic Perception | GRVC." https://grvc.us.es/eye-bird-dataset/.

[26] RODRÍGUEZ-GÓMEZ, Juan Pablo, et al. The GRIFFIN Perception Dataset: Bridging the Gap Between Flapping-Wing Flight and Robotic Perception. IEEE Robotics and Automation Letters, 2021, vol. 6, no 2, p. 1066-1073.

[27] OpenCV. 2021. Home - OpenCV. [Online] Available at: <https://opencv.org/>.

[28] Scikit-learn.org. 2021. scikit-learn: machine learning in Python — scikit-learn 0.24.2 documentation. [Online] Available at: <https://scikit-learn.org/stable/>.

[29] Es.mathworks.com. 2021. *MathWorks - Fabricantes de MATLAB y Simulink*. [Online] Available at: <https://es.mathworks.com/products.html>.

GLOSARIO

AT&T American Telephone & Telegraph 12

ATIS Asynchronous Time Based Image Sensor 6

BPNN Back Propagation Neural Network 13

CPU Central Processing Unit 8

DAG Directed Acyclic Graph 12, 44

DAGSVM Directed Acyclic Graph SVM 44

DAVIS Dynamic and Active-pixel Vision Sensor 6

DFT Discrete Fourier Transform 10, …

DVS Dynamic Vision Sensor 6

FAST Features from Accelerated Segment Test 9

FFT Fast Fourier Transform 64

GPU Graphics Processing Unit 8

GRVC Grupo de Robótica, Visión y Control 9, 17

HOG Histogram of Oriented Gradients 9, 10

ICA Independent Components Analysis 13

k-NN k-nearest neighbors 13

PCA Principal Component Analysis 10, …

PCI Porcentaje de Clasificaciones Incorrectas 27, 31

PNN Probabilistic Neural Network 13

RBF Radial Basis Function 32, 38

RBNN Radial Basis Function Neural Network 13

SLAM Simultaneous Localization and Mapping 8

SNNs Spiking Neural Networks 7

SVM Support Vector Machines 12, …

V-ceils Variance Ceils 20, 35