Universidad de Córdoba

Departamento de informática y análisis numérico

Contribuciones a la estimación de pose de cámara

Autor:
Víctor Manuel Mondéjar Guerra

Directores:
Rafael Muñoz Salinas
Rafael Medina Carnicer

Programa de Doctorado: Computación Avanzada, Energía y Plasmas

Línea del Programa: Sistemas inteligentes en Visión
Departamento de Informática y Análisis numérico

31 de mayo de 2016

TITULO: *CONTRIBUCIONES A LA ESTIMACIÓN DE POSE DE CÁMARA*

AUTOR: *Víctor Manuel Mondéjar Guerra*

Campus de Rabanales
Ctra. Nacional IV, Km. 396 A
14071 Córdoba

www.uco.es/publicaciones
publicaciones@uco.es

Abstract

Camera pose estimation is the problem of finding the orientation and localization of a camera with respect to an arbitrary coordinate system.

Image-based solutions for this problem are an interesting option because its reduced cost. However, their main drawback is that the accuracy of the results is affected by the presence of noise in the images.

The use of images for the *camera pose estimation* task is strongly related to the *Perspective-n-Point* (PnP) and *Bundle Adjustment* problem. Given a set of n correspondences between 3D points and its 2D projections on the image, PnP methods provide estimations of the camera pose. In addition, when the information about the 3D positions is unknow but a set of 2D projections taken from different viewpoints of the same 3D point are known, Bundle Adjustment methods are capable of finding simultaneously the 3D position of the points and the camera pose.

Then the task of finding correspondences between 3D points and its 2D projections, and between 2D projections of different images is a fundamental step for the above mentioned problems. This PhD Thesis proposes two novel approaches to solve the problem of finding correspondeces using both natural and artificial features.

In our first contribution, based on natural features, we propose a novel approach to find 2D correspondeces between images by a novel fusion approach combining information provided by several descriptors using the Dempster-Shafer Theory. The proposed method is able to fuse different sources of information considering their relative confidence in order to provide a better solution.

Our second contribution focuses on the problem of finding the 2D projections of 3D points. We propose a novel approach for identification of artificial landmarks, which are a very popular method when robustness and speed are required. In particular, we propose to tackle the marker identification problem as a classification one. As a consequence, we develop methods able to detect such markers in complex real situations such as blurring and non-uniform lightning.

The two contributions made in this Thesis have been compared with the state-of-art methods showing statistically significant improvements.

Resumen

El problema cuya resolución tiene como objetivo determinar la orientación y localización de una cámara respecto a un sistema de coordenadas se denomina *Estimación de la pose de la cámara*.

Las soluciones basadas en imágenes para la resolución de este problema son una opción interesante debido a su bajo coste. El inconveniente fundamental de esta opción es que su precisión puede verse afectada debido a la presencia de ruido en la imagen.

Trabajar con imágenes para estimar la pose de cámara está muy relacionado con dos problemas denominados *Perspective-n-Point* (PnP) y *Bundle Adjustment* (ajuste del haz). Dado un conjunto de n correspondencias entre puntos del espacio 3D y sus proyecciones 2D en una imagen, los métodos PnP tratan de obtener la pose de la cámara. Cuando la información acerca de la posición 3D de los puntos es desconocida, pero sí se tiene conocimiento de una serie de proyecciones 2D tomadas desde diferentes puntos de vista del mismo punto 3D, el *ajuste del haz* trata de estimar simultáneamente la posición tridimensional de los puntos y la pose de la cámara.

Debido a esto la tarea de buscar correspondencias, ya sea entre puntos de la escena 3D y su proyección 2D en la imagen, o entre varias proyecciones 2D de imágenes diferentes no es trivial y resulta fundamental para la resolución de los problemas mencionados anteriormente. En esta Tesis Doctoral se han propuesto dos métodos novedosos para el problema de búsqueda de correspondencias usando marcas naturales y artificiales.

En nuestra primera contribución, basada en el uso de marcas naturales, proponemos un método para encontrar correspondencias entre puntos 2D de diferentes imágenes, utilizando un nuevo enfoque de fusión que combina la información proporcionada por varios descriptores haciendo uso de la Teoría de Dempster-Shafer. El método propuesto es capaz de fusionar diferentes fuentes de información teniendo en cuenta además su confianza relativa con el fin de obtener una mejor solución.

La segunda contribución se centra en el problema de búsqueda de proyecciones 2D de puntos 3D conocidos. Proponemos un enfoque novedoso para identificar marcadores artificiales, que son una alternativa muy popular cuando se requiere robustez y velocidad. En concreto, proponemos abordar el problema de identificación de marcadores artificiales como un problema de clasificación. Como consecuencia, hemos entrenado métodos capaces de detectar marcadores en imágenes afectadas por situaciones complejas como el desenfoque o la luz no uniforme.

Ambas propuestas realizadas en esta Tesis han sido comparadas con métodos del estado del arte mostrando mejoras que son estadísticamente significativas.

A mi madre y mi padre
a mi hermano y mi hermana

Agradecimientos

A Rafael Muñoz Salinas y Rafael Medina Carnicer por su gran ayuda y guía prestada durante la dirección de la tesis. A Manuel Jesús Marín Jiménez quien también me ayudó en numerosas ocasiones.

En general, a todos los miembros del grupo Aplicaciones de la Visión Artificial.

Y en especial quiero agradecer a mis compañeros de laboratorio Eusebio Jesús Aguilera Aguilera, Sergio Garrido Jurado, David López Fernández y Manuel Ignacio López Quintero, con los que he pasado tantísimos buenos ratos durante estos últimos tres años y de los que también he recibido ayuda.

DECLARACIÓN

Rafael Muñoz Salinas y Rafael Medina Carnicer, profesores del Departamento de Informática y Análisis Numérico de la Universidad de Córdoba, INFORMAN que la Tesis Doctoral realizada por D. Víctor Manuel Mondéjar Guerra con título *Contribuciones a la estimación de pose de cámara* ha sido desarrollada bajo su dirección y que la misma presenta contribuciones originales que permiten que la misma pueda ser defendida.

Director:

Rafael Muñoz Salinas

Director:

Rafael Medina Carnicer

Córdoba 31 de mayo de 2016

TÍTULO DE LA TESIS: Contribuciones a la estimación de la pose de cámara

DOCTORANDO/A: Víctor Manuel Mondéjar Guerra

INFORME RAZONADO DEL/DE LOS DIRECTOR/ES DE LA TESIS

El alumno ha realizado un excelente trabajo durante estos años en su labor investigadora. Su esfuerzo ha permitido realizar importantes contribuciones en el ámbito de la Visión por Ordenador, y en particular en el ámbito de la estimación de la pose de la cámara utilizando tanto técnicas de puntos característicos, como técnicas de marcadores fiduciales. Los trabajos realizados han permitido la publicación del trabajo:

VM Mondéjar-Guerra, Rafael Muñoz-Salinas, Manuel J Marín-Jiménez, A Carmona-Poyato, R Medina-Carnicer "Keypoint descriptor fusion with Dempster-Shafer theory",International Journal of Approximate Reasoning, 2015,Volume 60, 57-70.

Además, como aportación adicional a la tesis, se ha realizado el trabajo

VM Mondéjar-Guerra, Sergio Garrido-Jurado Rafael Muñoz-Salinas, Manuel J Marín-Jiménez,, Rafael Medina-Carnicer, "Robust detection of fiducial markers in challenging conditions" Journal of Artificial Intelligence Research

que se encuentra actualmente en proceso de revisión.

Por estas razones, entendemos que la Tesis doctoral presentada por Víctor Manuel cumple con los requisitos necesarios de excelencia para otorgarle el grado de Doctor en Informática.

Por todo ello, se autoriza la presentación de la tesis doctoral.

Córdoba, 6 de Junio de 2016

Firma del/de los director/es

Fdo.:Rafael Muñoz Salinas

Fdo.: Rafael Medina Carnicer

Índice general

Índice de figuras

Índice de tablas

Acrónimos

BBA	Basic Belief Assignment
BRIEF	Binary Robust Independent Elementary Features
BRISK	Binary Robust Invariant Scalable Keypoints
CNN	Convolutional Neural Network
DLT	Direct Linear Transformation
DoG	Difference-of-Gaussian
FAST	Features from Accelerated Segment Test
FREAK	Fast Retina Keypoint
NN	Neural Network
NNDR	Nearest Neighbor Distance Ratio matching
MLP	Multilayer Perceptron
PCA	Principal Component Analysis
ORB	Oriented FAST and Rotated BRIEF
RBF	Radial Basis Function
SIFT	Scale-Invariant Feature Transform
SfM	Structure from Motion
SLAM	Simultaneous Localization And Mapping
SURF	Speeded Up Robust Features
SVM	Support Vector Machine
TBM	Transferable Belief Model
TDS	Teoría de Dempster-Shafer

Capítulo 1

Introducción y Objetivo de la Tesis Doctoral

En el ámbito de la visión artificial se denomina *estimación de la pose de la cámara* al problema cuya resolución tiene como objetivo determinar la orientación y localización de la misma respecto a un sistema de coordenadas. La localización y orientación de la cámara se define con los *seis grados de libertad* (6-DoF) de un movimiento en un espacio tridimensional, es decir mediante los tres valores de traslación (X, Y, Z) y los tres ángulos de navegación, uno por cada eje: cabeceo, alabeo y guiñada (ver Figura 1.1).

La estimación de pose de cámara es un paso imprescindible en multitud de aplicaciones de la visión artificial entre las que podemos destacar: realidad aumentada [1-3], localización y mapeado simultáneos [4-6] o reconstrucción tridimensional [7, 8].

Estimar *la pose de la cámara* puede llevarse a cabo de dos formas diferentes según la fuente de la información utilizada [9]: a partir de mecanismos hardware o a partir de imágenes. Dentro de la primera opción podemos mencionar los sistemas de *Inertial Navigation Systems (INS)* y de *Global Positioning Systems (GPS)*. Las soluciones

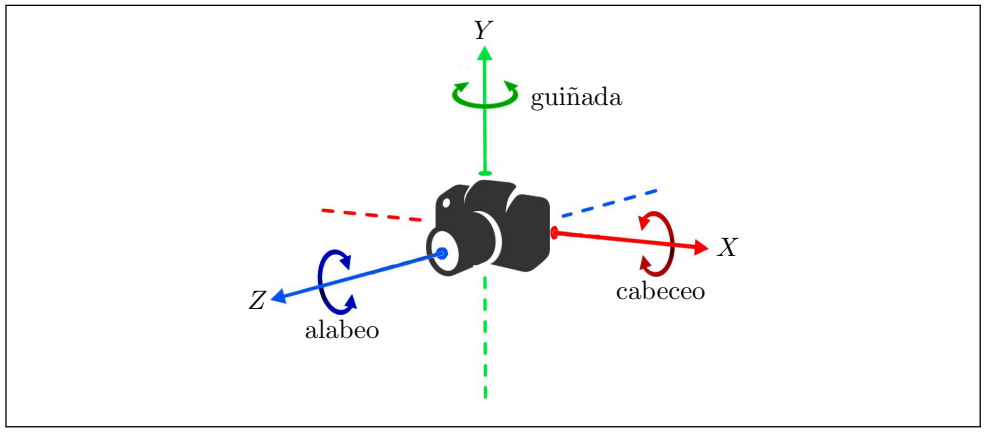

Figura 1.1: *Los seis grados de libertad: adelante/atrás, arriba/abajo, izquierda/derecha y los tres ejes de rotación: cabeceo, alabeo y guiñada.*

basadas en *hardware* permiten obtener resultados muy precisos, aunque tienen dos inconvenientes: su elevado precio y sus limitaciones de uso en base al entorno.

Por el contrario, las soluciones basadas en imágenes presentan como ventaja su bajo coste y su facilidad de implementación en cualquier tipo de entorno. El inconveniente fundamental de esta opción es que su precisión puede verse afectada debido a la presencia de ruido en la imagen.

En general, trabajar con imágenes para estimar la pose de cámara se asocia con dos problemas denominados *Perspective-n-Point* (PnP), y *Bundle Adjustment* (ajuste del haz), dependiendo de si la información 3D necesaria es conocida o no.

Dado un conjunto de n correspondencias entre puntos 3D y sus proyecciones 2D en una imagen, el problema PnP trata de obtener la pose de la cámara. Aunque existen técnicas para el caso $n = 3$, denominado $P3P$ [10, 11] el caso general se corresponde cuando $n \geq 4$ para el que se han propuesto multitud de trabajos [12-14].

Sin embargo la solución anterior requiere que el conjunto de puntos 3D sea conocido *a priori*, y esto no siempre es posible. Para aquellos casos en los la posición tridimensional de los puntos es desconocida se puede aplicar el *ajuste del haz*. Dadas una serie de imágenes tomadas desde distintos puntos de vista de una escena, de las que además se conoce un conjunto de proyecciones 2D correspondientes, es decir, que hacen referencia al mismo punto del espacio, *el ajuste del haz* trata de optimizar la búsqueda de la información tridimensional de los puntos a la vez que estima la pose de la cámara en cada una de las vistas. Esta técnica es un paso fundamental tanto en el problema denominado Simultaneous Localization And Mapping (SLAM) [15] como en los problemas de reconstrucción de la escena, también conocidos como Structure from Motion (SfM) [7, 8].

Para establecer las correspondencias entre puntos existen dos alternativas principales en la literatura científica: *usar marcas naturales o artificiales*.

Usar marcas naturales requiere detectar la localización en píxeles de la imagen de aquellos puntos de la escena que presenten una alta probabilidad de ser visibles bajo una amplia variedad de situaciones como pueden ser distintos puntos de vista, tamaños de escala, desenfoque, entre otras. Dentro de este tópico se conoce como puntos de interés, o *keypoints*, a los píxeles detectados en cada una de las imágenes.

Una de las estrategias más usadas en esta alternativa consiste en usar un descriptor local que identifique la región centrada sobre cada *keypoint*. Si el descriptor usado es *bueno*, caracterizará de forma correcta (con un valor muy parecido) la región en ambas imágenes, por lo que al usar una medida de similitud *correcta* se podrá establecer la correspondencia deseada entre los *keypoints* sin ambigüedades. En general, el uso de *keypoints* permite encontrar puntos 2D en dos imágenes distintas que se corresponden con el mismo punto del espacio 3D (ver Figura 1.2). Por tanto, es frecuente el uso de esta alternativa en técnicas de *ajuste del haz* para estimar la pose de la cámara.

La alternativa anterior ha sido ampliamente estudiada en la última década y, como consecuencia de ello, han surgido multitud de propuestas de descriptores. Podemos citar, entre los descriptores más conocidos, a Scale-Invariant Feature Transform (SIFT)

Figura 1.2: *Ejemplo de emparejamiento de keypoints. Se muestran dos imágenes tomadas desde diferentes puntos de vista de la misma escena. Los keypoints se representan con círculos. Las líneas verdes representan los emparejamientos correctos mientras que las líneas rosas corresponden con emparejamientos erróneos.*

[16] o Speeded Up Robust Features (SURF) [17]. Existen diferentes estudios comparativos entre varios descriptores locales que no llegan a establecer un claro ganador [18-21] ante cualquier situación. Por tanto, a fecha actual, no podemos afirmar cuál de los descriptores propuestos es el mejor.

Por otro lado, si bien las técnicas basadas en búsqueda de correspondencias entre *keypoints* usando marcas naturales tienen la ventaja de no requerir la modificación del entorno (no es necesario introducir elementos artificiales), éstas tienen un serio inconveniente: en entornos reales de interior y exterior es habitual encontrar zonas con poca textura en las que éstas técnicas no son capaces de funcionar correctamente (por ejemplo pasillos con largas paredes blancas).

Como consecuencia de lo anterior, existe otra alternativa para resolver este problema: introducir marcadores artificiales en el entorno, asumiendo que el entorno de trabajo será modificado para colocar estos marcadores. Este inconveniente se contrapone con la ventaja de que el uso de este tipo de marcadores, con forma cuadrada, es una de las opciones preferidas cuando lo que se desea es sencillez, robustez, precisión y velocidad. Los marcadores artificiales son elementos fácilmente distinguibles, generalmente con un borde externo negro, y que, en los casos más populares, codifican en su interior un identificador único mediante un código binario que suele incluir bits de detección y de corrección de errores. Para localizar un marcador artificial con forma de cuadrado solamente hay que emplear en la imagen un detector de bordes de formas cuadradas. De esta forma se obtienen un conjunto de candidatos compuestos por los marcadores y, eventualmente, elementos del fondo de la escena (ver Figura 1.3). Posteriormente, la extracción del código binario de cada candidato permitirá decidir si se trata de uno de los marcadores artificiales o no. Por tanto, la selección de los códigos binarios internos de los marcadores artificiales es de gran importancia para reducir la posibilidad de errores en su detección. Existen trabajos en este sentido y, en general, cada autor propone su propio conjunto predefinido de marcadores (denominado *diccionario*) y diferentes métodos para reducir la tasa de error en la

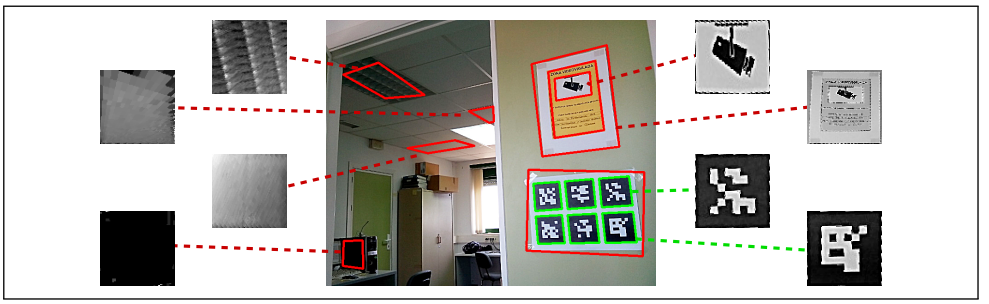

Figura 1.3: *Ejemplo de detección de marcadores candidatos cuadrados. Los candidatos en verde hacen referencia a marcadores mientras que los rojos pertenecen al fondo.*

identificación [2, 22]. Nótese que cada marcador puede proporcionar cuatro puntos prominentes, las esquinas. Ya que el marcador es previamente conocido por el sistema, es decir se conoce la información 3D del mismo, solamente es necesario detectar las proyección 2D de las esquinas del marcador en la imagen para poder realizar la estimación de la pose por medio de técnicas de PnP.

A pesar de ser una alternativa precisa y robusta, actualmente, aún persisten dificultades en muchas aplicaciones de la vida real debido a que las imágenes están sometidas a circunstancias en las que no es posible una correcta identificación de estos marcadores. Entre las circunstancias mencionadas citamos:

- Desenfoque debido al movimiento, muy común en cámaras con *rolling-shutter* como las que usan los móviles, o drones.
- Iluminación no uniforme. Muy común en entornos de exterior.
- Entornos con visibilidad reducida. Cuando el marcador debe ser visto desde lejos, y el área visible es muy pequeña.

Objetivo de la Tesis Doctoral

El objetivo central de esta Tesis ha sido proponer nuevos métodos que permitan aumentar la precisión durante la etapa de búsqueda de puntos correspondientes para poder estimar *la pose de la cámara*. Se proponen mejoras para dos alternativas distintas: (a) con marcadores naturales (basadas en el uso de *keypoints* y descriptores) y (b) con marcadores artificiales. El presente documento se organiza de la siguiente manera.

En el Capítulo 2 se comentan cuestiones generales relacionadas con la estimación de la pose de cámara, además de hacer una revisión del estado del arte de las técnicas basadas en descriptores locales, sistemas de marcadores artificiales, y otras técnicas de las que hemos hecho uso en nuestras propuestas: cómo fusionar información procedente de distintas fuentes y cómo usar conceptos de aprendizaje automático.

El Capítulo 3 incluye la primera propuesta de esta Tesis Doctoral: *Estimando la pose de cámara mediante fusión de información de descriptores para emparejamiento de Keypoints*. El Capítulo 4 desarrolla la segunda propuesta: *Estimando la pose de cámara mediante detección de marcadores artificiales en situaciones complejas*.

Finalmente, las Conclusiones obtenidas en esta Tesis Doctoral se detallan en el Capítulo 5 resaltando los aspectos fundamentales que muestran el rendimiento de las dos propuestas que se hacen.

Capítulo 2

Trabajos relacionados

En este capítulo se comentan cuestiones relacionadas con el objetivo principal de la Tesis Doctoral sobre las que se sustentan las contribuciones originales realizadas: modelo y calibración de la cámara, emparejamiento de correspondencias entre dos imágenes de una misma escena y uso de descriptores para caracterizar puntos de una imagen, técnicas básicas para estimar la pose de la cámara cuando la información 3D es desconocida, sistemas de marcadores artificiales, técnicas básicas para estimar la pose de cámara cuando la información 3D es conocida, teoría para fusión de información y técnicas de aprendizaje automático.

2.1. Modelo de la cámara y calibración

La luz proveniente de un emisor viaja por el espacio hasta alcanzar un objeto. Cuando la luz impacta un objeto, gran parte de esta luz es absorbida, el resto la percibimos como color. La luz reflejada que llega a nuestros ojos (nuestra cámara) se recoge por nuestra retina. La modelización matemática que describe el viaje del rayo de luz desde el objeto, a través de la lente de nuestros ojos (o de la cámara), hacia la retina es de gran interés para la visión artificial. El modelo pinhole de la cámara es un modelo simple y útil que tiene como objetivo la representación de este proceso.

Para relacionar las medidas entre un punto 3D de la realidad y su proyección 2D es fundamental haber realizado previamente una correcta calibración de la cámara. La calibración de la cámara se refiere al proceso de determinar el conjunto de características geométricas y ópticas de la misma, (parámetros intrínsecos), y la pose de la cámara con respecto a un sistema de coordenadas global (parámetros extrínsecos).

2.1.1. Modelo pinhole

El modelo pinhole describe la relación matemática entre las coordenadas de un punto 3D y su proyección en el plano imagen de una cámara ideal, donde la apertura de la cámara se corresponde con un punto sin necesidad de hacer uso de lentes, ver Figura 2.1. Este modelo es muy utilizado por la comunidad científica para escenificar

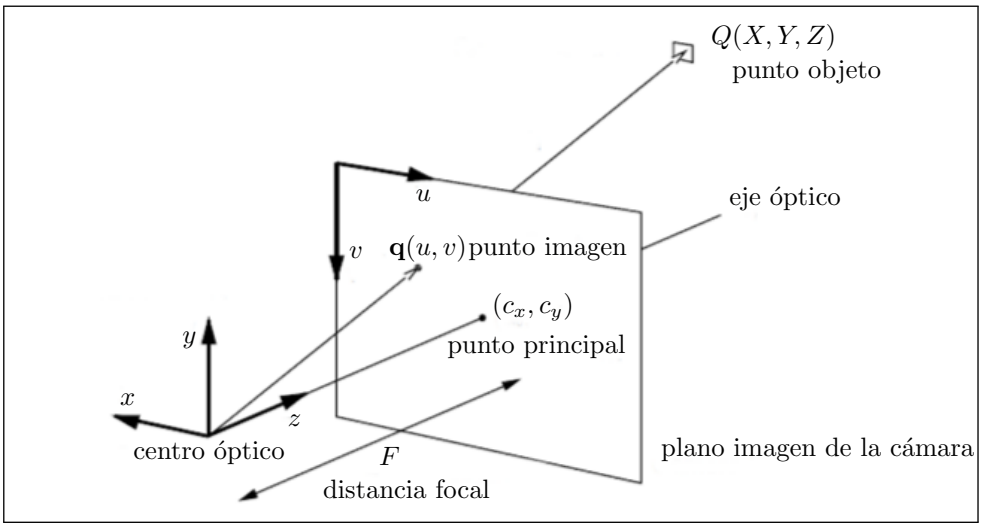

Figura 2.1: *Modelo pinhole de la cámara.*

este problema, a pesar de asumir unas condiciones ideales que no permiten representar correctamente la realidad.

El modelo pinhole se basa en el principio de puntos colineales, en el que cada punto del espacio es proyectado por una línea recta que pasa por el centro de proyección del plano imagen. El origen del sistema de coordenadas de la cámara se encuentra en el centro de proyección en la localización (X_0, Y_0, Z_0) con respecto al sistema de coordenadas del objeto, y el eje-z del fotograma de la cámara es perpendicular al plano de imagen.

Siendo Q un punto objeto en la localización (X, Y, Z), para expresarlo en las coordenadas de la imagen (u, v), primero se debe transformar a las coordenadas de la cámara (x, y, z). Esta transformación consiste en aplicar una rotación seguida de una traslación, que se puede realizar utilizando la siguiente ecuación:

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & t_x \\ m_{21} & m_{22} & m_{23} & t_y \\ m_{31} & m_{32} & m_{32} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} + \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix}, \tag{2.1}$$

donde

$$\begin{array}{lll} m_{11} = \cos\varphi\cos\kappa & m_{12} = \sin\omega\sin\varphi\cos\kappa - \cos\omega\sin\kappa & m_{13} = \cos\omega\sin\varphi\cos\kappa + \sin\omega\sin\kappa \\ m_{21} = \cos\varphi\sin\kappa & m_{22} = \sin\omega\sin\varphi\sin\kappa + \cos\omega\cos\kappa & m_{23} = \cos\omega\sin\varphi\sin\kappa + \sin\omega\cos\kappa \\ m_{31} = -\sin\varphi & m_{32} = \sin\omega\sin\varphi & m_{33} = \cos\omega\cos\varphi \end{array}$$

siendo los ángulos ω, φ y κ las rotaciones sobre los ejes x, y, z, y t_x,t_y y t_z los componentes de translación.

Los parámetros intrínsecos de la cámara incluyen la distancia focal F y el centro de la imagen o punto principal (c_x, c_y). Haciendo uso del modelo pinhole, la proyección

2D de (x, y, z) al plano imagen se expresa de la siguiente manera:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \frac{1}{z} \begin{bmatrix} f_x x \\ f_y y \end{bmatrix} + \begin{bmatrix} c_x \\ c_y \end{bmatrix}, \tag{2.2}$$

donde f_x y f_y se corresponden con el producto de la distancia focal por los factores s_x, s_y expresados en unidad de píxel. Téngase en cuenta que F se encuentra en milímetros mientras que los valores de s_x, s_y en píxel por milímetro, $f_x = Fs_x, f_y = Fs_y$.

El modelo pinhole es solo una aproximación de la proyección real de la cámara, pero es útil para formular matemáticamente la relación entre los objetos y las coordenadas de la imagen. Sin embargo, este modelo no es válido cuando se requiere una alta precisión. Esto es debido a que las lentes de las cámaras no son perfectas y por tanto, provocan distorsión en la imagen adquirida.

Es común utilizar el modelo pinhole como modelo base sobre el que añadir distintos factores que permitan corregir la distorsión producida en las coordenadas de imagen. En general se suelen modelar dos tipos de distorsión: radial y tangencial. La distorsión radial se localiza en los bordes de la imagen, de forma que este tipo de distorsión es nula en el centro óptico y aumenta a medida que se va alejando de él. Para aproximar la distorsión radial generalmente es suficiente con utilizar los dos coeficientes iniciales:

$$\begin{bmatrix} u^{(r)} \\ v^{(r)} \end{bmatrix} = \begin{bmatrix} u^{'}(k_1 r^2 + k_2 r^4 + ...) \\ v^{'}(k_1 r^2 + k_2 r^4 + ...) \end{bmatrix}, \tag{2.3}$$

donde $k_1, k_2, ...$ son los coeficientes para la distorsión radial, $u^{'}, v^{'}$ representan las coordenadas en metros, es decir $u^{'} = (u - c_x)/s_x$, $v^{'} = (v - c_y)/s_y$, y $r = \sqrt{u'^2 + v'^2}$.

La distorsión tangencial se produce debido a que la lente no se encuentra de forma totalmente paralela al plano imagen. Se puede expresar de la siguiente manera:

$$\begin{bmatrix} u^{(t)} \\ v^{(t)} \end{bmatrix} = \begin{bmatrix} 2p_1 u^{'} v^{'} + p_2(r^2 + 2u^{'^2}) \\ p_1(r^2 + 2v^{'^2}) + 2p_2 u^{'} v^{'} \end{bmatrix}, \tag{2.4}$$

donde los coeficientes de la distorsión tangencial se corresponden con p_1 y p_2. Existen más tipos de distorsión en la cámara pero no son tan agresivos como la distorsión radial y tangencial. El modelo considerando los dos tipos de distorsión quedaría como:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} s_x(u^{'} + u^{(r)} + u^{(t)} \\ s_y(v^{'} + v^{(r)} + v^{(t)} \end{bmatrix} + \begin{bmatrix} c_x \\ c_y \end{bmatrix}. \tag{2.5}$$

2.1.2. Calibración de la cámara

El objetivo del proceso de calibración de la cámara es determinar los valores óptimos de los parámetros intrínsecos de la cámara (f_x, f_y, c_x, c_y) y los coeficientes de distorsión $(k_1, ..., k_n,\ p_1, p_2)$. La manera más común de determinar estos parámetros es mediante la observación e identificación de un objeto 3D conocido en un conjunto de

imágenes tomadas desde diferentes vistas. Un algoritmo muy utilizado para resolver este proceso es el algoritmo Direct Linear Transformation (DLT) [23] que formula el problema en su forma cerrada. Se han realizado varias extensiones a este trabajo [24, 25]. Aunque se puede utilizar cualquier objeto para la calibración, en la práctica es muy común el uso de un patrón regular como, por ejemplo, un tablero de ajedrez [26], que presente una serie de puntos característicos cuya identificación sea un proceso sencillo. En una primera etapa el proceso de calibración ignora la distorsión existente en la lente. Se deben resolver por tanto, diez parámetros (cuatro parámetros intrínsecos y seis parámetros extrínsecos) por cada vista. No obstante se debe tener en cuenta que los parámetros intrínsecos son iguales para todas las vistas. Además, como se explica a continuación, al hacer uso de un objeto plano como es el tablero de ajedrez se pueden reducir el número de parámetros extrínsecos a cuatro.

En visión artificial se denomina homografía planar a la transformación proyectiva de un plano a otro. Es posible expresar esta transformación en términos de multiplicación de matrices si hacemos uso de las coordenadas homogéneas para expresar el cambio de un punto $\tilde{Q} = [X, Y, Z, 1]^T$ a un punto $\tilde{\mathbf{q}} = [u, v, 1]^T$ como $\tilde{\mathbf{q}} = s\mathbf{H}\tilde{Q}$, donde s representa el factor de escala y $\mathbf{H}$ es la matriz de homografía.

La matriz $\mathbf{H}$ se puede descomponer en dos partes: la transformación física, o punto de vista desde donde vemos el objeto, y la proyección que introduce los parámetros intrínsecos de la cámara. La transformación física incluye una matriz de rotación $\mathbf{R}$ con dimensión 3×3 y un vector traslación $\mathbf{t}$ de 3 dimensiones. Trabajando con coordenadas homogéneas esto se puede representar en una única matriz $\mathbf{W} = [\mathbf{R}\mathbf{t}]$.

Los parámetros intrínsecos también pueden modelarse usando coordenadas homogéneas como una matriz:

$$\mathbf{M} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.6}$$

y por tanto la ecuación $\tilde{\mathbf{q}} = s\mathbf{H}\tilde{Q}$ puede expresarse como $\tilde{\mathbf{q}} = s\mathbf{M}\mathbf{W}\tilde{Q}$.

En la práctica, se puede asumir que el plano observado del patrón se encuentra en $Z = 0$ del sistema de coordenadas global. Considerando además las 3 columnas de la matriz de rotación ($\mathbf{R} = [\mathbf{r}_1\mathbf{r}_2\mathbf{r}_3]$), se puede establecer la siguiente simplificación que elimine una de estas columnas. Concretamente:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = s\mathbf{M}[\mathbf{r}_1\mathbf{r}_2\mathbf{r}_3\mathbf{t}] \begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix} = s\mathbf{M}[\mathbf{r}_1\mathbf{r}_2\mathbf{t}] \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}. \tag{2.7}$$

Ahora la matriz de homografía que transforma los puntos de un objeto plano en la imagen queda descrita completamente mediante $\mathbf{H} = s\mathbf{M}[\mathbf{r}_1\mathbf{r}_2\mathbf{t}]$ y es una matriz cuadrada de dimensión de 3.

Para obtener el valor de $\mathbf{H}$ se requiere un mínimo de dos imágenes correspondientes a dos vistas diferentes de un tablero de ajedrez de dimensión 3×3 (4 puntos internos). En la práctica se aconseja un mínimo de 10 imágenes con un tablero de 7×8 para

obtener unos resultados buenos que hagan frente al posible ruido presente en la imagen [26].

Como ya hemos mencionado, el cálculo inicial de los parámetros intrínsecos se lleva a cabo asumiendo que la cámara no presenta distorsión. Para cada vista del tablero se obtiene la siguiente igualdad:

$$\mathbf{H} = [\mathbf{h}_1\mathbf{h}_2\mathbf{h}_3] = s\mathbf{M}[\mathbf{r}_1\mathbf{r}_2\mathbf{t}], \tag{2.8}$$

de donde:

$$\mathbf{h}_1 = s\mathbf{M}\mathbf{r}_1 \quad \text{o} \quad \mathbf{r}_1 = \frac{1}{s}\mathbf{M}^{-1}\mathbf{h}_1,$$

$$\mathbf{h}_2 = s\mathbf{M}\mathbf{r}_2 \quad \text{o} \quad \mathbf{r}_2 = \frac{1}{s}\mathbf{M}^{-1}\mathbf{h}_2,$$

$$\mathbf{h}_3 = s\mathbf{M}\mathbf{t} \quad \text{o} \quad \mathbf{t} = \frac{1}{s}\mathbf{M}^{-1}\mathbf{h}_3.$$

Los vectores de rotación $\mathbf{r}_1, \mathbf{r}_2$ son ortonormales entre sí, lo que implica: tienen la misma magnitud y su producto escalar es 0. De acuerdo con esto se tiene que:

$$\mathbf{h}_1^T\mathbf{M}^{-T}\mathbf{M}^{-1}\mathbf{h}_2 = 0, \tag{2.9}$$

$$\mathbf{h}_1^T\mathbf{M}^{-T}\mathbf{M}^{-1}\mathbf{h}_1 = \mathbf{h}_2^T\mathbf{M}^{-T}\mathbf{M}^{-1}\mathbf{h}_2. \tag{2.10}$$

Denotando $\mathbf{B}$ como $\mathbf{M}^T\mathbf{M}^{-1}$ para hacer la representación más sencilla

$$\mathbf{B} = \mathbf{M}^T\mathbf{M}^{-1} = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{21} & B_{22} & B_{23} \\ B_{31} & B_{32} & B_{33} \end{bmatrix}, \tag{2.11}$$

la matriz $\mathbf{B}$ tiene una solución general de forma cerrada:

$$\mathbf{B} = \begin{bmatrix} \frac{1}{f_x^2} & 0 & \frac{-c_x}{f_x^2} \\ 0 & \frac{1}{f_y^2} & \frac{-c_y}{f_y^2} \\ \frac{-c_x}{f_x^2} & \frac{-c_y}{f_y^2} & \frac{c_x^2}{f_x^2}\frac{c_y^2}{f_y^2} + 1 \end{bmatrix}. \tag{2.12}$$

Usando esta matriz $\mathbf{B}$, las restricciones (2.9)-(2.10) tienen la forma general $\mathbf{h}_i^T\mathbf{B}\mathbf{h}_j$ y al ser $\mathbf{B}$ simétrica, se puede representar como un producto escalar de vectores de 6 dimensiones:

$$\mathbf{h}_i^T\mathbf{B}\mathbf{h}_j = \mathbf{v}_{ij}^T\mathbf{b} = \begin{bmatrix} h_{i1}h_{j1} \\ h_{i1}h_{j2} + h_{i2}h_{j1} \\ h_{i2}h_{j2} \\ h_{i3}h_{j1} + h_{i1}h_{j3} \\ h_{i3}h_{j2} + h_{i2}h_{j3} \\ h_{i3}h_{j3} \end{bmatrix}^T \begin{bmatrix} B_{11} \\ B_{12} \\ B_{22} \\ B_{13} \\ B_{23} \\ B_{33} \end{bmatrix}^T, \tag{2.13}$$

Por otro lado, usando la definición de $\mathbf{v}_{ij}^T$, las dos restricciones (2.9)-(2.10) se pueden escribir de la siguiente manera:

$$\begin{bmatrix} \mathbf{v}_{12}^T \\ (\mathbf{v}_{11} - \mathbf{v}_{22})^T \end{bmatrix} \mathbf{b} = \mathbf{0}. \tag{2.14}$$

Nótese que si se dispone de K imágenes del tablero de ajedrez, se tienen K ecuaciones del tipo $\mathbf{Vb} = \mathbf{0}$, donde $\mathbf{V}$ es una matriz de dimensión $2K \times 6$. De manera que cumpliendo la condición $K \geq 2$ se puede obtener el valor de $\mathbf{b} = [B_{11}, B_{12}, B_{22}, B_{13}B_{32}, B_{33}]^T$.

Despejando los parámetros intrínsecos de la matriz $\mathbf{B}$ mediante la solución de forma cerrada se tiene:

$$f_x = \sqrt{\frac{\lambda}{B_{11}}} \quad f_y = \sqrt{\frac{\lambda B_{11}}{(B_{11}B_{22} - B_{12}^2)}} \quad c_x = \frac{-B_{13}f_x^2}{\lambda} \quad c_y = \frac{(B_{12}B_{13} - B_{11}B_{23})}{B_{11}B_{22} - B_{12}^2}, \tag{2.15}$$

siendo $\lambda = B_{33} - \frac{(B_{13}^2 + c_y(B_{12}B_{13} - B_{11}B_{23}))}{B_{11}}$.

Los parámetros extrínsecos se pueden calcular entonces como:

$$\mathbf{r}_1 = \lambda \mathbf{M}^{-1}\mathbf{h}_1 \quad \mathbf{r}_2 = \lambda \mathbf{M}^{-1}\mathbf{h}_2 \quad \mathbf{r}_3 = \mathbf{r}_1 \times \mathbf{r}_2 \quad \mathbf{t} = \lambda \mathbf{M}^{-1}\mathbf{h}_3. \tag{2.16}$$

Resolviendo este sistema de ecuaciones no siempre se obtiene una matriz de rotación exacta en la que se cumpla $\mathbf{R}^T\mathbf{R} = \mathbf{R}\mathbf{R}^T = \mathbf{I}$. Una manera de resolver esto consiste en realizar la descomposición en valores singulares de $\mathbf{R}$. La descomposición en valores singulares es un método que factoriza una matriz en dos matrices ortonormales, $\mathbf{U}, \mathbf{V}$ y una matriz $\mathbf{D}$ situada en medio que contiene valores de escala en su diagonal. Esto permite transformar $\mathbf{R}$ en $\mathbf{R} = \mathbf{U}\mathbf{D}\mathbf{V}^T$. Para que $\mathbf{R}$ sea ortonormal la matriz $\mathbf{D}$ debe ser la matriz identidad de forma que $\mathbf{R} = \mathbf{U}\mathbf{I}\mathbf{V}^T$. De esta forma, igualando la matriz $\mathbf{D}$ con la matriz identidad, y multiplicando de nuevo por la descomposición de valores singulares, se obtiene la matriz de rotación $\mathbf{R}'$.

Hasta este punto se ha considerado el proceso de calibración de la cámara asumiendo que no existe distorsión en la lente de la cámara. Sin embargo, para realizar este proceso de manera más precisa se debe considerar como mínimo la distorsión radial y tangencial. Una estrategia común consiste en estimar los valores de distorsión (2.5) mediante la resolución de un sistema de mínimos cuadrados. Este sistema dependerá del número de coeficientes de distorsión utilizados para ajustar el problema [24, 25].

Por último para ajustar el conjunto completo de parámetros (incluyendo los de distorsión) se suele optar por tratar minimizar la función:

$$\sum_{i=1}^{n} \sum_{j=1}^{m} \left\| \mathbf{q}_{ij} - \hat{\mathbf{q}}(\mathbf{M}, k_1, k_2, p_1, p_2, \mathbf{R}_i, \mathbf{t}_i, Q_j) \right\|^2, \tag{2.17}$$

donde $\hat{\mathbf{q}}(\mathbf{M}, k_1, k_2, p_1, p_2, \mathbf{R}_i, \mathbf{t}_i, Q_j)$ es la proyección del punto Q_j en la imagen i, acorde a $s\tilde{\mathbf{q}} = \mathbf{H}\tilde{Q}$ seguido de la corrección de la distorsión de la cámara (2.5). Este problema de minimización no lineal puede ser resuelto con el algoritmo de Levenberg-Marquard [27]. Téngase en cuenta que los valores de los parámetros intrínsecos y extrínsecos pueden inicializarse en base a los resultados obtenidos en (2.11)-(2.16).

En resumen el proceso de calibración consta de los siguientes pasos:

1. Usar un tablero de ajedrez como patrón y colocarlo sobre una superficie plana.
2. Tomar varias imágenes del patrón bajo distintas vistas (moviendo el plano o la cámara).
3. Detectar los puntos situados en las esquinas internas del patrón
4. Estimar los parámetros intrínsecos y extrínsecos usando la solución de forma cerrada (2.11)-(2.16).
5. Estimar los coeficientes de la distorsión radial y tangencial mediante un sistema de mínimos cuadrados.
6. Refinar todos los parámetros minimizando la función en (2.17).

2.2. Emparejamiento con descriptores para correspondencias entre dos imágenes

El objetivo final del procedimiento denominado *correspondencia entre dos imágenes* es emparejar un mismo punto real de la escena en dos imágenes distintas. Para la consecución de este objetivo, el uso de descriptores locales de *keypoints* se ha probado con éxito en multitud de aplicaciones de la visión artificial. Este proceso puede dividirse en tres pasos:

- En una primera aproximación deben detectarse puntos de interés o *keypoints* en ambas imágenes. Un detector ideal selecciona como *keypoint* aquellos píxeles de la imagen que representan los puntos de la escena con mayor probabilidad de volver a ser detectados desde diferentes puntos de vista. La alternativa más clásica en esta etapa consiste en buscar píxeles ubicados en esquinas en cada una de las dos imágenes.
- En un segundo paso, los *keypoints* deben ser caracterizados con alguna propiedad más a efectos de que su posterior identificación biunívoca no plantee ningún género de duda. A estos efectos se usa un descriptor, un vector de características, que represente la región local de cada *keypoint* identificado en el paso anterior.
- Finalmente, el análisis de la similitud entre los descriptores de las dos imágenes debe conducir al emparejamiento de *keypoints* de dos en dos y aquellos para los que ello sea posible determinarán las correspondencias de la escena.

El análisis de la similitud entre descriptores obliga a hacer uso de algún tipo de distancia. Dependiendo del tipo de descriptor se suele emplear la distancia de Hamming [28] (para descriptores binarios) o la distancia euclídea, entre otras posibles. El uso de una distancia implica emplear alguna estrategia final [29] (por ejemplo umbralización, vecino más cercano, ratio de distancia del vecino más cercano, etc...) para producir el emparejamiento de *keypoints* buscado.

La Figura 1.2 muestra un ejemplo de emparejamiento de *keypoints* de dos imágenes que han sido capturadas desde diferentes puntos de vista de la misma escena.

Como se ha comentado anteriormente un descriptor incluye dos pasos: (a) detección de *keypoints* y (b) obtención para cada *keypoint* de uno o varios valores que caractericen la región del mismo.

2.2.1. Etapa de detección de un Keypoint

La *repetibilidad* es la medida que indica la calidad del detector de *keypoints*. La tasa de *repetibilidad* indica el número de *keypoints* en común que presentan dos imágenes. Así cuanto mayor sea la *repetibilidad* de un detector para un par de imágenes, mayor será el número de posibles correspondencias.

Aunque el objetivo final de un detector es proporcionar los *keypoints* de la escena, existen también otras cualidades del detector que son deseables: seleccionar el tamaño de la región alrededor del píxel (escala) y detectar el ángulo predominante u orientación de la región. Ambas cualidades aportan la posibilidad de en una etapa posterior, disponer de descriptores invariantes a la escala y a la rotación. Con respecto al valor de escala comentar que, si es inapropiado, puede hacer que los descriptores creados sean inestables a causa de su variabilidad.

Lindeberg [30] fue el precursor al proponer un método de detección de *keypoints* con selección automática de escala. Su método usa el determinante de la matriz Hessiana y el Laplaciano.

Más tarde Lowe [16], en la etapa de detección de su método SIFT, modifica la idea anterior aproximando la escala normalizada del Laplaciano de Gauss mediante el uso de filtros Difference-of-Gaussian (DoG). El objetivo es detectar los puntos de interés en los extremos del espacio de escala. La implementación de Lowe exige realizar la convolución de varios filtros Gaussianos con la imagen. Una octava se define por un número fijo de convoluciones con distintos filtros Gaussianos que van incrementando su valor σ. Cuando se completa una octava, el tamaño de la imagen se reduce a la mitad y la imagen resultante es procesada por la siguiente octava (ver Figura 2.2(a)), reiterando este procedimiento se construye una pirámide. El filtro DoG se calcula a partir de la resta de convoluciones adyacentes pertenecientes a una misma octava, (ver Figura 2.2(b)). Finalmente, los puntos de interés del espacio de escala se corresponden con los mínimos y máximos locales de las DoG. Para que un punto sea seleccionado como candidato debe ser máximo o mínimo, no solo con respecto a los ocho vecinos de la DoG a la que pertenece, sino también con respecto a los nueve vecinos de la DoG superior y los nueve de la inferior (ver Figura 2.2(c)).

De entre los candidatos seleccionados deben ser descartados todos aquellos que presenten un bajo contraste o que se encuentren a lo largo de un borde, debido a que dichas localizaciones son más inestables ante pequeñas cantidades de ruido.

Para calcular la orientación principal de un punto, se suaviza la imagen con un filtro Gaussiano (utilizando el valor de σ previamente estimado) y se calcula la magnitud y orientación del gradiente sobre la imagen suavizada, fijando una ventana alrededor

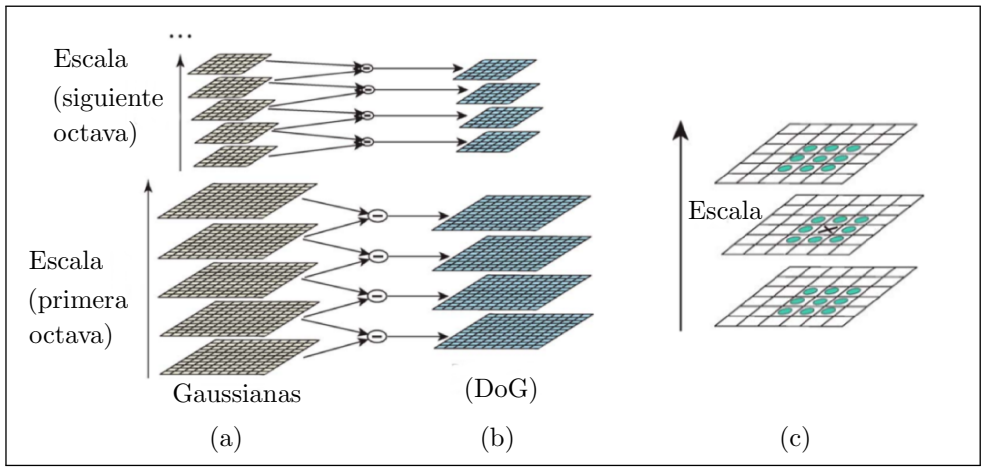

Figura 2.2: *Esquema etapa de detección SIFT (a) Octavas de convolución filtro de Gauss (b) DoG (c) Máximo/mínimo local espacio-escala.*

del punto. El tamaño de la ventana depende del valor de la escala. Finalmente se calcula un histograma de orientación compuesto por 36 intervalos, cubriendo los 360º, y la orientación del punto se corresponde con el pico más alto del histograma.

Propuestas como las de Bay *et al* [17] en SURF siguen un esquema similar al usado por SIFT en su etapa de detección, pero cambiando ciertas funciones con el objetivo de reducir el tiempo de cómputo necesario. Para la detección de los candidatos y la selección automática de escala, SURF se basa en el uso de la matriz Hessiana de la convolución Gaussiana. Para aproximar esta función utiliza filtros de caja, que pueden calcularse de manera muy eficiente mediante el uso de imágenes integrales. Para analizar el espacio de escala, en lugar de reducir el tamaño de la imagen por cada octava como ocurría en SIFT, este método aumenta progresivamente el tamaño del filtro de caja. Para establecer una localización como *keypoint*, esta debe corresponderse con un máximo en una región de vecindad $3 \times 3 \times 3$. El cálculo de la orientación implica el uso de una ventana que depende del valor de escala seleccionado, y en este caso calcula el valor de los Wavelet de Haar [31] en las direcciones x e y. Las respuestas de Wavelet se representan en un espacio vectorial, donde la respuesta horizontal se corresponde con el eje de abscisas y la vertical con el eje de ordenadas. El espacio se divide en intervalos de ángulo de tamaño $\pi/3$ y se suman las respuestas de Wavelet. Finalmente, el intervalo con mayor longitud de vector es seleccionado como la orientación del punto.

2.2.2. Etapa de descripción de un Keypoint

En la actualidad existen múltiples propuestas, en la literatura científica, de descriptores que enfatizan diferentes propiedades de la imagen como pueden ser la intensidad

de los píxeles, el color, la textura o los bordes. Comentaremos algunos de los descriptores más conocidos basados en el nivel de gris de la imagen, en función de los buenos resultados que pueden ofrecer.

El objetivo de un descriptor es representar una región local en forma de vector de características de manera que la misma región pueda ser identificada en otras imágenes y a la vez, se pueda diferenciar del resto de regiones de interés de la misma escena. El descriptor más simple estaría formado por el valor de intensidad de los píxeles que pertenecen a la región. Sin embargo esto generaría vectores de una alta dimensión, lo que conllevaría una alta complejidad durante el cálculo posterior de distancias entre los valores de los descriptores. Además un descriptor de este tipo sería muy susceptible ante pequeños cambios, como por ejemplo el punto de vista o la rotación. Por esta razón, los descriptores basados en el nivel de intensidad del píxel suelen hacer uso de técnicas que hagan uso de histogramas o comparaciones entre los valores de intensidad de un conjunto de puntos de la región, de manera que se obtenga una descripción robusta ante distintos cambios y a la vez compacta.

El descriptor SIFT [16] ha sido inspiración para multitud de trabajos y es considerado actualmente como uno de los mejores descriptores locales. SIFT representa las regiones por medio de histogramas de la orientación del gradiente. En primer lugar, divide la región alrededor del punto en regiones más pequeñas. Sobre cada una de estas regiones calcula un histograma de orientación del gradiente, en el que cada píxel añade un peso al intervalo del histograma correspondiente en base a su magnitud y orientación. La concatenación de los histogramas de cada región componen el vector de características. En su versión estándar los descriptores SIFT dividen la región en 4×4 y usan 8 intervalos de orientación, generando vectores de dimensión 128. A pesar de haber sido muy utilizado, SIFT ha recibido críticas por la elevada dimensión de sus vectores y la complejidad tanto en su creación como en la posterior etapa de cálculo de distancias. Con el objetivo de mejorar estos aspectos se han propuesto numerosas alternativas, de entre las que destaca la propuesta de Herbert *et al.* [17] denominada SURF.

En SURF también se divide la región que rodea al punto en varias regiones más pequeñas. Este método acumula, por cada región, la suma de las respuestas de Wavelet de Haar en dirección horizontal, vertical y los valores absolutos de ambas direcciones. La concatenación de estos cuatro valores por cada región conforman el descriptor SURF. En su versión estándar utiliza regiones de 4×4, lo que produce vectores de características de 64 dimensiones.

Los descriptores binarios surgen como una alternativa aún más eficiente en el uso de la memoria, haciendo así más eficiente también el posterior cálculo de las distancias entre los descriptores.

Calonder *et al.* [32] desarrollaron el primer descriptor binario para *keypoints* denominado Binary Robust Independent Elementary Features (BRIEF). Este método realiza una serie de test de comparación τ sobre una ventana $\mathbf{p}$ de tamaño $S \times S$ centrada en el *keypoint*

$$\tau(\mathbf{p}; \mathbf{x}, \mathbf{y}) = \begin{cases} 1 & si\ I(\mathbf{p}, \mathbf{x}) < I(\mathbf{p}, \mathbf{y}) \\ 0 & \text{otro caso} \end{cases}, \tag{2.18}$$

donde $I(\mathbf{p}, \mathbf{x})$ se corresponde con el valor de intensidad del píxel $\mathbf{x} = (u, v)^T$ perteneciente a la ventana $\mathbf{p}$.

La localización de los pares de puntos $(\mathbf{x}, \mathbf{y})$ se realiza de manera aleatoria siguiendo una distribución Gaussiana $(0, \frac{1}{25}S^2)$ (ver Figura 2.3(a)). Con objetivo de lograr robustez ante pequeños ruidos se aplica un suavizado Gaussiano sobre los valores de intensidad de la imagen.

El descriptor está formado por la concatenación de ceros y unos resultantes de la realización de los test. Al estar formado por cadenas binarias, la dimensión estándar de este método es de 256 *bits*, o lo que es lo mismo, 32 *bytes*.

Este descriptor, sin embargo, no es suficientemente robusto a cambios en la rotación o escala en comparación con los métodos SIFT y SURF.

Con el objetivo de solucionar esa problemática, conservando las ventajas del uso de descriptores binarios, han surgido propuestas como la de Rublee *et al.* [33], denominada Oriented FAST and Rotated BRIEF (ORB), Leutenegger *et al.* [34], denominada Binary Robust Invariant Scalable Keypoints (BRISK), o la más reciente, de Alahi *et al.* [35], denominada Fast Retina Keypoint (FREAK). Estas propuestas siguen un esquema muy similar al de BRIEF. Se basan en la realización de una serie test binarios sobre el valor de intensidad de un conjunto de pares de puntos, diferenciándose fundamentalmente en el criterio de selección usado.

El proceso de selección usado por ORB también se basa en una distribución Gaussiana, pero este método propone además un algoritmo con objetivo de reducir la correlación entre los puntos seleccionados. ORB realiza la estimación del ángulo predominante mediante el método *centroide de intensidad* [36] que se basa en el uso de momentos. Con la orientación estimada ORB rota la ventana antes de realizar los test de intensidad. ORB se compone de 256 test o 32 *bytes*.

La selección de pares de BRISK se basa en el uso de un patrón circular muy similar al utilizado por el descriptor DAISY [37]. Este patrón está formado por un conjunto

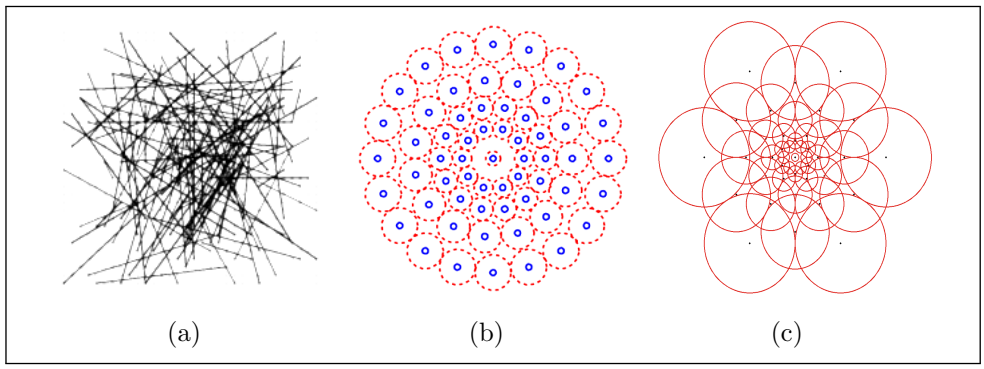

Figura 2.3: *Patrones de selección de pares de puntos en descriptores binarios. (a) Distribución Gaussiana (BRIEF)(ORB). (b) Patrón de muestreo BRISK. (c) Patrón de muestro de retina de FREAK.*

de puntos equidistantes situados en círculos concéntricos (ver Figura 2.3(b)). Dependiendo de la distancia al centro, se varía el tamaño del *kernel* con el que se suaviza la imagen, creando dos subconjuntos en base a la distancia entre los pares de puntos seleccionados: de larga distancia y de corta distancia. Los primeros contribuyen al cálculo del ángulo de orientación del *keypoint*, ángulo que se usará para rotar los puntos pertenecientes al segundo subconjunto, que a su vez son usados para realizar las comparaciones del valor de intensidad. El descriptor BRISK está compuesto por un total de 512 test, y por tanto tienen una dimensión de 64 *bytes*.

Finalmente, FREAK utiliza lo que denomina patrón circular de muestreo de retina (ver Figura 2.3(c)), en el que cada círculo se corresponde con un campo receptivo que suaviza la imagen con su correspondiente *kernel* Gaussiano. A diferencia del patrón usado por BRISK, en este existe una mayor densidad de puntos cerca del centro y presenta solapamiento entre campos receptivos. Para seleccionar los pares de puntos utilizados para realizar los test utiliza un algoritmo similar al usado en ORB, evitando así la alta correlación entre los pares. FREAK calcula su propio ángulo de orientación mediante el uso de un conjunto de pares de puntos como hace BRISK. Aunque en el caso de FREAK, los pares de puntos que contribuyen al cálculo de la orientación se encuentran en campos receptivos simétricos con respecto al centro. La dimensión de FREAK es de 512 test y por tanto 64 *bytes*.

2.3. Bundle Adjustment

En general, cuando trabajamos con marcas naturales (*keypoints*), buscamos píxeles pertenecientes a distintas imágenes que se correspondan con el mismo punto del espacio, del que su posición 3D nos es desconocida. Por tanto, al no conocer la información 3D no podemos aplicar técnicas de PnP. Este problema se denomina *bundle adjustment* (ajuste del haz) [38] y suele ser un paso fundamental en problemas de SfM [8] o de SLAM [15].

Dada la localización de los píxeles de un conjunto de correspondencias establecidas entre imágenes tomadas desde puntos de vista diferentes, el objetivo del *ajuste del haz* es encontrar la posición tridimensional de dichas correspondencias a la vez que se estiman los parámetros de la cámara minimizando el error de retroproyección [38].

La Figura 2.4 extraída del trabajo [39] ejemplifica de manera muy clara el problema descrito. Considerando tres imágenes de un cubo (Fig. 2.4(a)) de las que se desconoce el punto de vista desde el que fueron tomadas, y conocido un conjunto de correspondencias entre varios puntos de las imágenes que hacen referencia a los mismos puntos del espacio 3D (los píxeles 2D del mismo color se corresponden con un mismo punto 3D). Las correspondencias citadas aportan un conjunto de restricciones sobre la geometría 3D de las cámaras y los puntos. Teniendo en cuenta estas restricciones y dada la geometría de la escena formada por los puntos 3D y la pose de cada cámara, se puede predecir la localización en la imagen de las proyecciones 2D de cada punto sobre las cámaras mediante el uso de las ecuaciones de proyección de

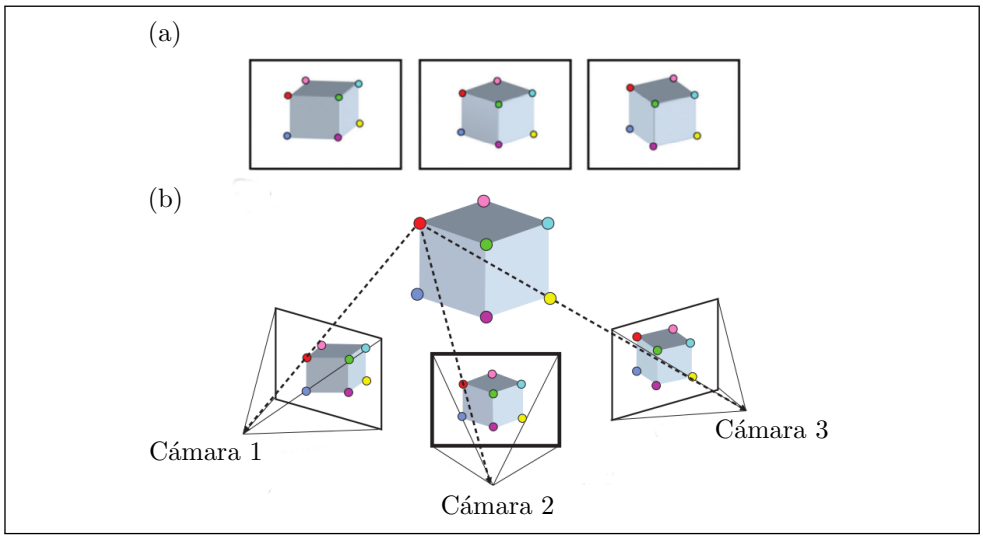

Figura 2.4: *Ejemplo del error de retroproyección producido en el ajuste del haz. (a) Tres imágenes de un cubo tomadas desde un punto de vista desconocido. Los puntos coloreados de las esquinas muestran las correspondencias conocidas entre estas imágenes, cada conjunto de puntos del mismo color son proyecciones del mismo punto 3D. (b) Reconstrucción 3D de los puntos. Cada imagen presenta una pose estimada. La línea punteada representa la proyección del punto 3D estimado en las imágenes. En el caso de la cámara 3 la linea de proyección no coincide con el punto. El residuo resultante se conoce como el error de retroproyección que se desea minimizar.*

perspectiva (Fig. 2.4(b)). De esta manera, es posible comparar dichas proyecciones con las observaciones originales y obtener un error de retroproyección.

De acuerdo con la Figura 2.4, sean $X_i, i = 1, ..., 8$ las posiciones 3D de las esquinas del cubo y $\mathbf{R}_j$, $\mathbf{t}_j$ y $f_j, j = 1, 2, 3$ la orientación, posición y distancia focal de las tres cámaras. Entonces si $\mathbf{x}_{ij}$ identifica al punto X_i en la imagen j es posible formular la ecuación de la imagen como:

$$\mathbf{x}_{ij} = f_j \prod(\mathbf{R}_j(X_i - \mathbf{t}_j)), \tag{2.19}$$

donde $\prod$ es la función de proyección

$$\prod(x, y, z) = (x/z, y/z). \tag{2.20}$$

En este problema se debe inferir X_i, $\mathbf{R}_j$, $\mathbf{t}_j$ y f_j a partir de las observaciones $\mathbf{x}_{ij}$. La forma más común de hacer esta inferencia es mediante la formulación del problema como un problema de optimización de mínimos cuadrados que trate de minimizar el error de retroproyección:

$$\arg \min_{X_i, \mathbf{R}_j, \mathbf{t}_j, f_j} \sum_{i \sim j} \left\| \mathbf{x}_{ij} - f_j \prod(\mathbf{R}_j(X_i - \mathbf{t}_j) \right\|^2 \tag{2.21}$$

donde, $i \sim j$ indica que el punto X_i es visible en la imagen j.

De manera similar a lo realizado en el apartado de calibración de cámaras, la resolución de este problema de mínimos cuadrados puede ser resuelta por medio del algoritmo Levenberg-Marquardt [27].

2.4. Sistemas de marcadores artificiales

Un sistema de marcadores artificiales está compuesto por un conjunto de marcadores y por un algoritmo que sea capaz de detectarlos, identificarlos, y corregir los errores en la identificación de los mismos en la medida de lo posible. En la literatura científica existen numerosos sistemas de marcadores artificiales. Se muestran algunos ejemplos en la Figura 2.5.

Las propuestas más simples utilizan puntos como marcadores, ya sean LEDs, esferas reflectantes o puntos planos [40, 41]. Para la detección de este tipo de marcadores suelen utilizarse técnicas de segmentación en situaciones controladas. Un inconveniente de este tipo de sistemas es que su identificación se obtiene a partir de la posición relativa entre los marcadores, lo que frecuentemente involucra un proceso complejo.

Una evolución de las propuestas anteriores son los marcadores circulares. Estos marcadores codifican su identificación en sectores circulares o anillos concéntricos [42] (Fig. 2.5(a)). En general presentan una única correspondencia, el centro. Esto supone una ventaja respecto a los puntos simples, ya que los marcadores circulares se pueden identificar de manera individual. Su inconveniente es que para poder estimar

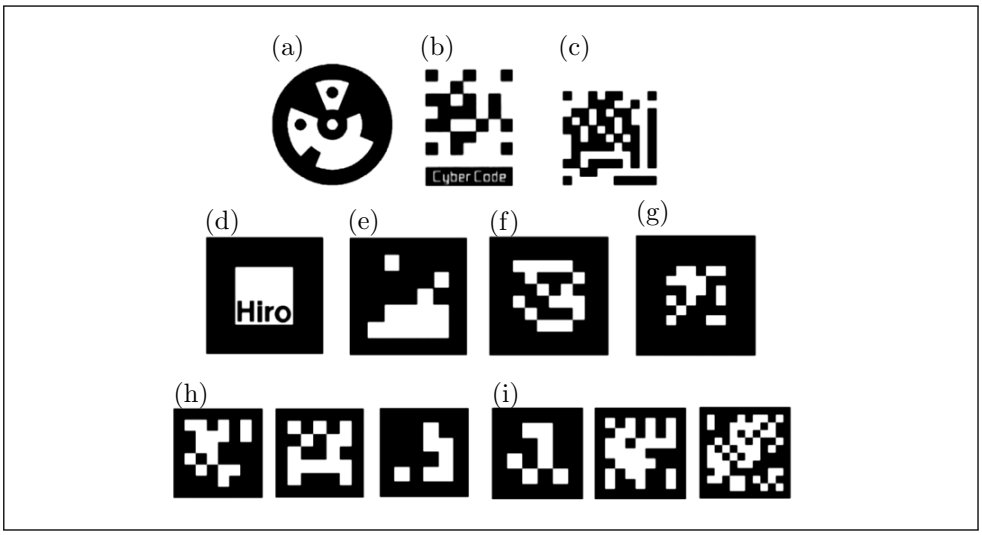

Figura 2.5: *Ejemplos de sistemas de marcadores existentes.*

la posición de la cámara sigue siendo necesaria la identificación simultánea de varios marcadores de este tipo.

Otro tipo de marcadores artificiales se basan en la detección de manchas, como por ejemplo Cybercode[43] o VisualCode[44] (Fig. 2.5(b) y Fig. 2.5(c)) que derivan de técnicas basadas en los códigos de barras como MaxiCode o QR. Este tipo de sistemas presentan la ventaja de proporcionar de manera precisa varios puntos de correspondencia.

Los sistemas de marcadores cuadrados constituyen la opción más popular. Su principal ventaja reside en la presencia de cuatro puntos salientes que se pueden usar para obtener la pose usando un único marcador, mientras que la región interna se usa para la identificación. El hecho de poder estimar la pose de la cámara utilizando un único marcador hace que este tipo de sistemas sean especialmente interesantes en esta Tesis Doctoral.

Uno de los sistemas más conocidos es ARToolKit [45], un proyecto de código libre que ha sido muy usado en la última década especialmente por la comunidad académica. Los marcadores de ARToolkit están compuestos por un borde negro que encierra una imagen en el interior. La imagen utilizada para la identificación del marcador se encuentra almacenada en una base de datos de patrones válidos (Fig. 2.5(d)). A pesar de su popularidad, este sistema presenta algunos inconvenientes. En primer lugar, usa un método de *template matching* para la identificación de marcadores lo que produce un elevado número de falsos positivos y una alta tasa de confusión entre marcadores [46]. En segundo lugar, el sistema utiliza un método de umbral global fijo para detectar los cuadrados, lo que lo hace especialmente sensible ante cambios en las condiciones lumínicas.

La mayoría de los sistemas de marcadores cuadrados utilizan códigos binarios en lugar de imágenes. Esto hace posible la corrección de errores durante la etapa de identificación.

Una de las primeras propuestas de este tipo es Matrix [47], que además es también de las más simples. Utiliza un código binario con redundancia de bits para la detección de errores (Fig. 2.5(e)). El sistema ARTag [48] se basa en los mismos principios, pero mejora la detección de los cuadrados utilizado un método basado en bordes (Fig. 2.5(f)) en lugar de usar un umbral fijo como ARToolKit. Esto lo hace más robusto ante cambios lumínicos y de oclusión parcial. Adicionalmente, ARTag utiliza marcadores de tamaño fijo de 6×6 bits y usa un diccionario que trata de maximizar la distancia entre marcadores, de los que puede corregir un bit.

BinARyID [49] propone un método de generación del diccionario imponiendo como restricción, para la adición nuevos marcadores, que estos presenten una distancia de Hamming igual o mayor a uno con el resto de marcadores seleccionados, de manera que se puedan evitar ambigüedades en la rotación. Sin embargo, este sistema no permite la corrección de bits al mantener solo en uno la distancia Hamming entre sus marcadores.

ARToolKit Plus [50] (Fig. 2.5(g)) mejora algunas de las características de su predecesor ARToolKit. En primer lugar, utiliza un método dinámico para actualizar el valor

del umbral global que depende del valor de los píxeles pertenecientes a los marcadores que han sido detectados previamente. En segundo lugar, proporciona un código binario para la identificación de los marcadores. La primera versión de ARToolKit Plus incluía 512 marcadores cuya codificación logra una distancia mínima de cuatro bits entre cualquier par de marcadores. Su última versión utiliza un diccionario de 4096 marcadores que presenta una distancia mínima de dos bits entre marcadores, de forma que esta versión no permite la corrección de errores.

Existen dos sistemas que permiten una buena corrección de errores: AprilTags[3] (Fig. 2.5(h)) y ArUco [22] (Fig. 2.5(i)).

AprilTags utiliza un método de generación de marcadores similar al de BinARyID, pero la distancia Hamming no es fija sino que puede ser introducida como un parámetro. De esta manera, se permite la creación de diccionarios con una gran distancia entre marcadores, permitiendo así la detección y corrección de errores. AprilTags utiliza una heurística para encontrar marcadores que presenten una alta distancia con el resto, pero esta técnica no es suficientemente rápida cuando se usan marcadores de elevado tamaño.

ArUco trata de maximizar tanto la distancia entre marcadores como el número de transiciones de bits en su proceso de generación de marcadores. Su propuesta original [2] presentaba dos inconvenientes: (a) utiliza una estrategia de búsqueda estocástica y (b) la memoria necesaria aumenta de manera exponencial al aumentar el tamaño del marcador.

En [22] se propone el uso de programación entera mixta para el proceso de generación de ArUco. Los métodos de programación entera mixta pueden obtener resultados óptimos a modelos matemáticos que estén representados por relaciones lineales y donde las incógnitas se limiten a números enteros. Así, esta propuesta genera el diccionario óptimo en término de la distancia entre marcadores para cualquier número de marcadores y bits. Sin embargo, los tiempos de computación son demasiado largos cuando aumenta el tamaño del diccionario o el número de bits. Por ello, proponen además una segunda aproximación que obtiene el diccionario subóptimo en un tiempo restringido.

El procedimiento utilizado en ArUco [2] para la detección de los marcadores en una imagen (ver Figura 2.6(a)) es el siguiente:

- Segmentación de la imagen: se destacan los contornos de la imagen en escala de gris mediante un método de umbralización adaptativa (Fig. 2.6(b)).
- Extracción del contorno y filtrado: se utiliza el algoritmo de Suzuki y Abe [51] para extraer los contornos de la imagen (Fig. 2.6(c)) y a continuación se usa el algoritmo de Douglas-Peucker [52] para realizar una aproximación poligonal. De esta manera, se descartan todos aquellos contornos que no están formados por 4 vértices (Fig. 2.6(d)), téngase en cuenta que se está trabajando con marcadores cuadrados.
- Extracción del código del marcador: este paso analiza la región interior del marcador para extraer su código interno. Para ello en primer lugar se calcula la matriz de homografía para poder eliminar la proyección perspectiva

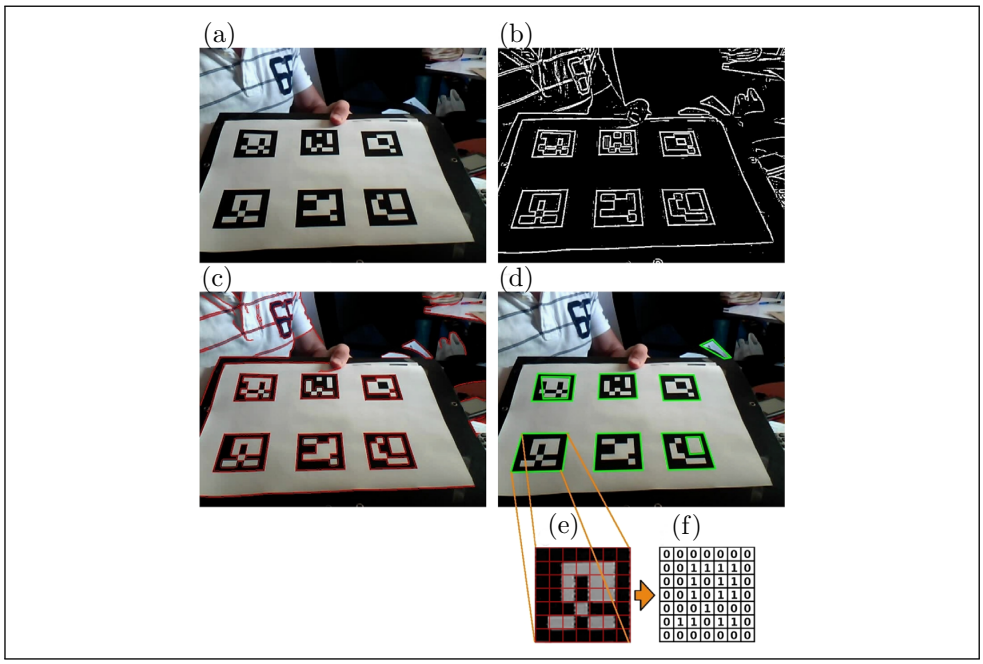

Figura 2.6: *Proceso de detección automática de marcador. (a) Imagen original. (b) Resultado tras aplicar la umbralización adaptativa. (c) Detección de contornos. (d) Aproximación poligonal y eliminación de contornos irrelevantes. (e) Marcador tras la transformación perspectiva. (f) Asignación de bit para cada celda.*

(Fig. 2.6(e)). La imagen resultante se umbraliza con el método de Otsu [53]. La imagen binaria se divide en una cuadrícula regular y se asigna un valor a cada elemento 0 ó 1, dependiendo del mayor número de píxeles en su interior (Fig. 2.6(e,f)).

- Identificación del marcador y corrección del error: una vez extraído el código en el paso anterior, se obtienen cuatro identificadores diferentes (uno por cada posibles rotación). Si uno de ellos se encuentra en el diccionario, el candidato se considera como un marcador válido. En caso contrario el método trata de corregir el error dentro de sus límites.

- Refinamiento de las esquinas y estimación de la pose: una vez que se ha detectado el marcador con éxito, se procede a estimar la pose de la cámara a partir de las proyecciones 2D de las cuatro esquinas. Téngase en cuenta, que el marcador es conocido previamente y por tanto se conoce su información 3D. De esta manera, identificadas cuatro correspondencias entre puntos 3D y sus respectivas proyecciones 2D en la imagen es posible aplicar técnicas PnP para obtener la pose de la cámara.

2.5. Perspective-n-Point

Tras el primer paso de calibración de la cámara, para obtener sus parámetros intrínsecos y de distorsión, y tras obtener una serie de correspondencias entre puntos 3D y sus proyecciones 2D en la imagen, el paso final para estimar la pose de la cámara puede llevarse a cabo mediante la técnica denominada *Perspective-n-Point* (PnP). Como ya se ha explicado previamente, durante el proceso de calibración de la cámara mediante DLT ya se realizaba una estimación de la pose de la cámara. Sin embargo, esta estimación se realizaba con la restricción del uso de un patrón fijo (tablero de ajedrez), lo que conlleva la ventaja de tener todos sus puntos coplanarios además de utilizar varias imágenes para poder despejar las incógnitas del sistema. Sin embargo, estas restricciones no permiten generalizar el problema de la estimación de la pose. Esa generalización se puede resolver haciendo uso de técnicas que se engloban bajo la denominación PnP.

Una de las técnicas más populares se conoce como EPnP (*Efficient PnP*) [12]. Este método obtiene la solución para $n \geq 4$, presenta una complejidad $O(n)$ y permite trabajar tanto con puntos que estén situados o no en un mismo plano. Para estimar la pose de la cámara este método trata de obtener las coordenadas de los n puntos respecto al sistema de coordenadas de la cámara. Una vez realizada esa tarea es sencillo obtener los valores de orientación y traslación que definen la pose de la cámara [54]. La idea principal consiste en expresar los n puntos tridimensionales como una suma ponderada de cuatro puntos de control virtuales. Así el problema se reduce a estimar las coordenadas de los puntos de control en el sistema de referencia de la cámara.

Sean $\mathbf{p}_i, \quad i = 1, ..., n$ los n puntos de referencia 3D conocidos y sean $\mathbf{c}_j, \quad j = 1, ..., 4$ los puntos de control:

$$\mathbf{p}_i^w = \sum_{j=1}^{4} \alpha_{ij} \mathbf{c}_j^w \quad \mathbf{p}_i^c = \sum_{j=1}^{4} \alpha_{ij} \mathbf{c}_j^c \quad \text{con} \quad \sum_{j=1}^{4} \alpha_{ij} = 1, \tag{2.22}$$

donde los superíndices w y c hacen referencia respectivamente a las coordenadas globales y de la cámara.

Haciendo uso de las proyecciones 2D de los puntos de referencia, de las que su posición (u_i, v_i) es conocida, y de la matriz $\mathbf{M}$ que contiene los parámetros intrínsecos de la cámara se tiene:

$$w_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \mathbf{M}\mathbf{p}_i^c = \mathbf{M} \sum_{j=1}^{4} \alpha_{ij} \mathbf{c}_j^c, \tag{2.23}$$

donde w_i es un factor de proyección. Considerando las coordenadas 3D (x_j^c, y_j^c, z_j^c) de cada punto de control c_j^c la ecuación anterior se corresponde con:

$$w_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \sum_{j=1}^{4} \alpha_{ij} \begin{bmatrix} x_j^c \\ y_j^c \\ z_j^c \end{bmatrix}, \tag{2.24}$$

En este caso se asume que la cámara está calibrada y por tanto los parámetros intrínsecos son conocidos. Las variables desconocidas de este sistema lineal son las 12 coordenadas de los puntos de control (x_j^c, y_j^c, z_j^c) $j = 1, ..., 4$ y los n parámetros de proyección w_i $i = 1, ..., n$. Dado que la última fila de (2.24) implica $w_i = \sum_{j=1}^{4} \alpha_{ij} z_j^c$, es posible reformular esta expresión para obtener las siguientes ecuaciones lineales para cada punto de referencia:

$$\sum_{j=1}^{4} \alpha_{ij} f_x x_j^c + \alpha_{ij}(c_x - u_i) z_j^c = 0, \tag{2.25}$$

$$\sum_{j=1}^{4} \alpha_{ij} f_y y_j^c + \alpha_{ij}(c_y - v_i) z_j^c = 0. \tag{2.26}$$

Mediante la concatenación de las ecuaciones (2.25) y (2.26) para los n puntos de referencia se genera un sistema lineal de la forma $\mathbf{Ax} = \mathbf{0}$, donde $\mathbf{x} = [c_1^{cT}, c_2^{cT}, c_3^{cT}, c_4^{cT}]^T$ es un vector de dimensión 12 que contiene las variables desconocidas y A es una matriz $2n \times 12$ compuesta por los coeficientes de (2.25) y (2.26) para cada punto de referencia. La solución pertenece al espacio nulo, o núcleo, de $\mathbf{A}$ y puede expresarse como:

$$\mathbf{x} = \sum_{i=1}^{N} \beta_i \mathbf{v}_i, \tag{2.27}$$

donde $\mathbf{v}_i$ son las columnas de los vectores singulares de $\mathbf{A}$, que se corresponden con los N valores singulares de $\mathbf{A}$. Se pueden resolver como los autovectores nulos de la matriz $\mathbf{A}^T\mathbf{A}$ de tamaño constante 12×12. Una vez resuelto esto, se pueden obtener los valores de β_i $i = 1, ..., N$ de (2.27). La dimensión de N del espacio nulo de $\mathbf{A}^T\mathbf{A}$ puede variar entre 1 y 4 dependiendo de la configuración de los puntos de referencia, la distancia focal y el ruido de la cámara. En la práctica se calculan las soluciones para los cuatro valores de N y se selecciona aquella que produzca el menor error de retroproyección.

$$res = \sum_i \text{dist}^2(\mathbf{M}[\mathbf{R}|\mathbf{t}] \begin{bmatrix} \mathbf{p}_i^w \\ 1 \end{bmatrix}, [u_i v_i 1]^T), \tag{2.28}$$

donde $[\mathbf{R}|\mathbf{t}]$ son los parámetros de la pose de la cámara y dist() es la distancia 2D entre los dos puntos. Dependiendo del valor de $N = 1, 2, 3, 4$ las restricciones cuadráticas variarán. En cualquiera de los casos se podrá obtener el valor de β. Una vez obtenido β y $\mathbf{x}$ los valores de orientación y traslación de la cámara se pueden calcular de manera sencilla mediante la búsqueda de mínimos cuadrados de la descomposición de valores singulares de una matriz 3×3, similar a lo ya explicado en la sección de calibración de la cámara.

Un inconveniente que tiene la solución anterior, y que es bastante frecuente en la mayoría de técnicas propuestas para resolver la estimación de la pose, es la ambigüedad que se produce en la orientación estimada. Gerald Schweighofer y Axel Pinz [14] demuestran que la función de minimización del error de los algoritmos de estimación de la pose (similares a (2.28)) pueden producir dos mínimos locales para diferentes

valores de orientación. Para resolver este problema proponen un algoritmo que a partir de una primera pose inicial $(\mathbf{R}_1, \mathbf{t}_1)$ calcule una segunda pose que garantice una única orientación.

2.6. Teoría de la evidencia de Dempster-Shafer

La Teoría de Dempster-Shafer (TDS) [55], también conocida como la teoría de la evidencia, es una generalización de la teoría de la probabilidad de Bayes. Es un modelo especialmente atractivo para representar la incertidumbre ante una determinada evidencia. A diferencia del modelo de probabilidad, TDS permite asignar un valor de confianza a un conjunto de hipótesis sin tener que repartir dicho valor entre las hipótesis individuales que lo componen.

La TDS se ha aplicado a varias disciplinas como el seguimiento de personas [56], detección de fraude [57], clasificación [58], análisis de riesgo [59], clustering [60, 61], procesamiento de imagen [62-65], mapeado autónomo de robot [66], reconstrucción de la silueta [67], interacción persona-ordenador [68], detección de minas terrestres [69] y asistente de conducción [70], entre otras.

Especialmente interesante es el trabajo de Denoeux *et. al.* [71] con relación a esta Tesis. En el mismo se presenta una solución a un problema con múltiples objetivos. Para describir cada objetivo utilizan tres atributos, cada atributo se considera una fuente de evidencia independiente que se fusiona utilizando TDS. Como se describe más adelante en el Capítulo 3, este problema es similar al nuestro ya que se trata de un problema con múltiples objetivos de asignación, en el que cada objetivo es descrito por varios atributos o características.

2.6.1. Fundamento teórico de la TDS

Dado un conjunto de hipótesis $\Omega = \{w_1, ..., w_K\}$, también conocido como marco de discernimiento, una asignación básica de evidencia (Basic Belief Assignment (BBA)) es una función $m : \Omega \rightarrow [0, 1]$, verificando:

$$\sum_{A \in \Omega} m(A) = 1, \tag{2.29}$$

donde A es un subconjunto de hipótesis de Ω. Se denominan elementos focales de m a los subconjuntos A de Ω que cumplen la condición $m(A) > 0$. El modelo original de Dempster-Shafer considera imposible que se produzca una hipótesis no incluida $(\emptyset)$ en Ω. Esto es, impone la condición $m(\emptyset) = 0$.

Mientras que el valor asignado a la ocurrencia de un suceso en un enfoque bayesiano debe ser el valor de una función de distribución de probabilidad, el valor $m(A)$ puede ser cualquier función subjetiva que exprese cuanta evidencia soporta el hecho A. El valor de evidencia de una o varias de las hipótesis que componen el subconjunto A puede ser desconocido. Este hecho supone una gran diferencia con respecto al punto

de vista de la teoría probabilística donde el valor de $m(A)$ se repartiría entre todos los elementos de A, lo cual supone igual probabilidad para todas las hipótesis y nunca la falta de conocimiento para discriminar entre unas y otras.

En nuestro trabajo usaremos una modificación del modelo de Dempster-Shafer: el modelo de evidencia transferible propuesto por Smets (Transferable Belief Model (TBM)) [72], en el que la función de evidencia BBA, cumple los criterios de TDS y además acepta la cláusula de mundo abierto: $m(\emptyset) > 0$. Es decir permite representar la posibilidad de que ocurra un hecho no incluido en Ω. Esto nos permitirá representar en nuestro problema el desconocimiento de un hecho por parte de un descriptor.

Usando una re-normalización siempre se puede transformar una evidencia m de Smet en una evidencia de Demspter m^*:

$$\begin{array}{l} m^*(\emptyset) = 0, \\ m^*(A) = \frac{m(A)}{1-m(\emptyset)} \quad si\ A \neq \emptyset. \end{array} \tag{2.30}$$

A partir de la función de asignación de evidencia, se definen otras como: la credibilidad (*bel*), la plausibilidad (*pl*) y la *comunidad* (*q*). La credibilidad indica la cantidad de credibilidad total depositada en el conjunto A, dado que la relación de inclusión es, en términos de afirmaciones, una relación de implicación. Es por tanto la suma de todas las probabilidades asignadas a hipótesis individuales contenidas en A, es decir, que hacen necesario A.

$$bel(A) = \sum_{\emptyset \neq B \subseteq A} m(B). \tag{2.31}$$

La plausibilidad incorpora además aquellos subconjuntos que tienen alguna hipótesis individual en común con A y que, por tanto, hacen posible que se verifique la hipótesis A.

$$pl(A) = \sum_{B \cap A \neq \emptyset} m(B). \tag{2.32}$$

Para nuestra propuesta destacaremos la *comunidad* como función de representación de m:

$$q(A) = \sum_{B \supseteq A} m(B),\ \forall A, B \subseteq \Omega. \tag{2.33}$$

Más adelante se aclara la importancia de esta representación.

2.6.2. Combinación de evidencias

Si dos fuentes de información proporcionan evidencias favoreciendo o negando hipótesis correspondientes a un mismo marco de discernimiento, aunque no sean las mismas hipótesis concretas, cada una de las evidencias representada por su BBA m_1, y m_2, pueden combinarse mediante las siguientes reglas de combinación.

Regla conjuntiva TBM y la regla de Dempster

La regla conjuntiva TBM y de Dempster se representan como $\textcircled{\cap}$ y $\bigoplus$, respectivamente. Sean m_1 y m_2 dos BBAs, el resultado de su combinación se define:

$$m_1 \textcircled{\cap} m_2(A) = \sum_{B \cap C = A} m_1(B) m_2(C) \quad \forall A \subseteq \Omega. \tag{2.34}$$

y, asumiendo que $m_{1\textcircled{\cap}2}(\emptyset) \neq 1$:

$$m_1 \oplus m_2(A) = \begin{cases} 0 & \text{si } A = \emptyset, \\ \frac{m_{1\textcircled{\cap}2}(A)}{1 - m_{1\textcircled{\cap}2}(\emptyset)} & \text{en otro caso.} \end{cases} \tag{2.35}$$

La regla de Dempster es equivalente a la conjuntiva TBM seguida de la normalización (2.30). Ambas reglas son conmutativas, asociativas y admiten un único elemento neutro: la BBA *vacía*, es decir cuando Ω es el único elemento focal $A \subset \Omega$. La regla conjuntiva TBM puede expresarse de manera simple en términos de funciones de *comunidad*:

$$q_{1\textcircled{\cap}2} = q_1 \cdot q_2.$$

Regla conjuntiva cauta

Antes de explicar la regla conjuntiva cauta debe aclararse el concepto de descomposición canónica. Se denomina BBA simple (SBBA) m y se representa como A^w cuando:

$$\begin{aligned} &m(A) = 1 - w \textit{ para cualquier } A \neq \Omega \\ &m(\Omega) = w, \end{aligned} \tag{2.36}$$

donde $w \in [0, 1]$.

Shafer [73] definió una BBA separable como el resultado de la $\oplus$ combinación de varios SBBAs. Para cada BBA separable se tiene:

$$m = \bigoplus_{\emptyset \neq A \subset \Omega} A^{w(A)}, \tag{2.37}$$

con $w(A) \in [0, 1]$ para todo $A \subset \Omega, A \neq \emptyset$. Dicha representación es única si todos los elementos focales son diferentes y m es no dogmática. Shafer denominó a esta representación la descomposición canónica de m.

La función separable de cualquier BBA no dogmático, es decir con $m(\Omega) > 0$, fue propuesto por Smets [74] como una extensión de la función separable original propuesta por Dempster (2.37). Para realizar dicha descomposición se hace uso de la generalización del SBBA (GSBBA), denotada también como A^w que se define como una función $\mu : 2^\Omega \to (-\infty, \infty)$, con $w \in (-\infty, \infty)$ y un elemento focal $A \subseteq \Omega$, tal que:

$$\begin{aligned} &\mu(A) = 1 - w, \\ &\mu(\Omega) = w, \\ &\mu(B) = 0 \quad \forall B \in 2^\Omega \setminus \{A, \Omega\}. \end{aligned} \tag{2.38}$$

Cuando $w \leq 1$, μ es una SBBA. Cuando $w > 1$, μ no es una BBA, ya que deja de pertenecer al espacio 2^Ω a $[0,1]$. Cualquier BBA no dogmática puede representarse como la combinación conjuntiva de GSBBAs:

$$m = \bigcirc\!\!\!\!\!\!\bigcap_{\emptyset \neq A \subset \Omega} A^{w(A)}, \tag{2.39}$$

con $w(A) \in (0, +\infty) \ \ \forall A \subset \Omega$. Los pesos $w(A)$ para cualquier $A \subset \Omega$ se pueden obtener a partir de las *comunidades* usando la siguiente fórmula:

$$w(A) = \prod_{B \supseteq A} q(B)^{(-1)^{|B|-|A|+1}}. \tag{2.40}$$

Sean m_1 y m_2 dos BBAs no dogmáticos. Su combinación usando la regla conjuntiva cauta se representa como $m_{1\textcircled{\wedge}2} = m_1 \textcircled{\wedge} m_2$. Se define como la BBA con la siguiente función de peso:

$$w_{1\textcircled{\wedge}2}(A) = w_1(A) \wedge w_2(A), \quad \forall A \subset \Omega,$$

teniendo así

$$m_1 \textcircled{\wedge} m_2 = \bigcirc\!\!\!\!\!\!\bigcap_{A \subset \Omega} A^{w_1(A) \wedge w_2(A)}. \tag{2.41}$$

Esta regla tiene las siguientes propiedades:

Conmutativa: *para todo* m_1 *y* m_2, $m_1 \textcircled{\wedge} m_2 = m_2 \textcircled{\wedge} m_1$;

Asociativa: *para todo* m_1, m_2, *y* m_3, $m_1 \textcircled{\wedge} (m_2 \textcircled{\wedge} m_3) = (m_1 \textcircled{\wedge} m_2) \textcircled{\wedge} m_3$;

Idempotencia: *para todo* m, $m \textcircled{\wedge} m = m$;

Distributiva de $\bigcirc\!\!\!\!\bigcap$ con respecto a $\textcircled{\wedge}$: *para todo* m_1, m_2, *y* m_3,

$$m_1 \bigcirc\!\!\!\!\bigcap (m_2 \textcircled{\wedge} m_3) = (m_1 \bigcirc\!\!\!\!\bigcap m_2) \textcircled{\wedge} (m_1 \bigcirc\!\!\!\!\bigcap m_3)$$

Combinación de reglas basadas en normas triangulares

Como se ha mostrado previamente, las reglas conjuntivas cauta y TBM se basan en combinaciones puntuales de pesos conjuntivos, usando respectivamente, el mínimo y el producto. En el intervalo $[0,1]$, el producto y el mínimo se pueden representar como normas triangulares *t-norms*. Por lo que estas reglas de combinación se pueden generalizar, permitiendo de esta manera además, la definición de nuevas reglas de combinación con propiedades interesantes. Concretamente, haciendo uso de la familia de *t-norms*:

$$x \top_s y = \begin{cases} x \wedge y & \text{si } s = 0, \\ xy & \text{si } s = 1, \\ log_s \left(1 + \frac{(s^x - 1)(s^y - 1)}{s - 1}\right) & \text{otro caso,} \end{cases} \tag{2.42}$$

para todo $x, y \in [0,1]$, donde s es un parámetro positivo. Cada valor de s define una *t-norm*. Cuando $s \to 0$, (2.42) tiende al mínimo entre x e y, y al producto cuando $s = 1$.

Todos las reglas de combinación resultantes de seguir la expresión anterior son conmutativas y asociativas, pero solo es idempotente la regla cauta, $s = 0$, $\textcircled{\wedge}$. A pesar de que estas reglas no están totalmente justificadas desde un punto de vista teórico, pueden ser útiles en ciertos problemas de fusión o clasificación.

La regla conjuntiva cauta a diferencia de la regla TBM $\textcircled{\cap}$, no requiere que las fuentes de información de las que derivan las BBAs sean distintas e independientes.

Por último, en ciertas aplicaciones es necesario tomar una decisión y elegir además la hipótesis ω con mayor confianza. Para hacer esto, Smets [75] propone el uso de la transformación pignística que se define para una BBA normal como:

$$BetP(\omega) = \sum_{A \subseteq \Omega, \omega \in A} \frac{m(A)}{|A|}. \tag{2.43}$$

2.7. Aprendizaje automático

El objetivo del aprendizaje automático es diseñar procedimientos que conviertan datos en información relevante para toma de decisiones. Según la naturaleza del problema y de la información aportada por los datos se puede dividir el aprendizaje automático en aprendizaje supervisado y no supervisado.

Dado un conjunto de datos de entrenamiento, de los que se conoce el par vector de entrada y resultado deseado, las técnicas de aprendizaje supervisado entrenan modelos para obtener una función capaz de predecir el resultado aún cuando se le presentan datos de entrada que nunca antes ha visto. Este tipo de aprendizaje se usa para resolver problemas de clasificación.

Cuando los datos de entrenamiento no aportan el resultado deseado, es el propio método el que los agrupa. En este caso, a diferencia del aprendizaje supervisado, no existe un conocimiento *a priori*. Este tipo de aprendizaje se usa en problemas de agrupación *clustering*.

Dado que el problema descrito en el Capítulo 4 lo abordamos como un problema de clasificación en el que disponemos de los resultados deseados, la revisión de los métodos de aprendizaje se realiza desde la perspectiva del aprendizaje supervisado. Se puede profundizar más sobre este tipo de técnicas en [76, 77].

Planteamiento general de un problema de clasificación

El objetivo en clasificación es: dado un vector de entrada $\mathbf{x}$ asignar el mismo a una clase k_i, del conjunto discreto de clases $K = k_1, ..., k_n$. Generalmente en los problemas de clasificación las clases son disjuntas, es decir cada entrada solo puede ser asignada a una única clase.

Supongamos además que por cada entrada $\mathbf{x}_i, i = 1, .., l$, conocemos el resultado esperado y_i (el que nos permitiría asignar la entrada a una determinada clase). Al

conjunto de pares $\{\mathbf{x}_i, y_i\}$ se le denomina conjunto de entrenamiento, y el objetivo es encontrar una función $f(\mathbf{x}, \alpha)$, siendo α un conjunto de parámetros ajustables, que permita usar la transformación $\mathbf{x}_i \rightarrow f(\mathbf{x}_i, \alpha)$ cumpliendo $f(\mathbf{x}_i, \alpha) = y_i$ para todo el conjunto de entrenamiento. Al cumplir este objetivo, encontrar la función $f(\mathbf{x}, \alpha)$ y los valores de α, hablamos de *máquina entrenada.*

La figura Figura 2.7 muestra un ejemplo del caso más simple en una tarea de clasificación: un problema de clasificación binaria, que notamos por $f(x, \alpha) \in \{1, -1\} \forall x, \alpha$, y en el que el conjunto de entrenamiento es linealmente separable. En este caso la función $f(x, \alpha)$ se corresponde con una línea recta orientada, de manera que todos los puntos a un lado de la línea se asignen a la clase 1, y todos los puntos del otro lado a la clase -1. Habitualmente los problemas de clasificación en la vida real no suelen ser linealmente separables e incluso pueden no ser resolubles mediante una función del tipo $f(x, \alpha)$. Por esta razón se han propuesto alternativas diferentes para conseguir los objetivos de aprendizaje

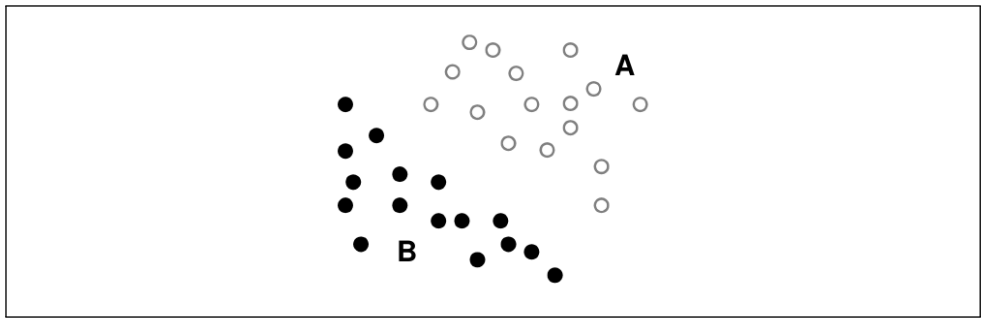

Figura 2.7: *Conjunto de datos separables. Cada color representa la clase a la que pertenece el punto.*

2.7.1. Support Vector Machine

Las máquinas de soporte vectorial (Support Vector Machine (SVM)) son clasificadores que utilizan como espacio de hipótesis funciones lineales en espacios de características de elevada dimensión sobre las que se realizan técnicas de optimización que separen lo máximo posible los datos de dos clases. Nacen como consecuencia de la investigación en *minimización de riesgo estructural* [78] y permiten resolver problemas que son no linealmente separables usando funciones lineales. Para ello, utilizan funciones *kernel* que permiten proyectar los datos de entrada a espacios de características de mayor dimensión donde se construye un hiperplano que separe los datos.

Las SVM han sido utilizadas en muchos trabajos de reconocimiento de patrones, reconocimiento de dígitos [78, 79], reconocimiento de objetos [80], detección de caras [81] y categorización de texto [82] entre otros.

Datos separables

Primero se considera el caso más simple: máquinas lineales entrenadas con datos separables hasta alcanzar el caso general (máquinas no lineales entrenadas sobre datos no separables).

Suponiendo que se dispone de algún hiperplano que separa los ejemplos positivos de los negativos, los puntos $\mathbf{x}$ que se encuentran en el hiperplano cumplen $\mathbf{w} \cdot \mathbf{x} + b = 0$, con vector normal $\mathbf{w}$ al hiperplano. Entonces $|b| / \|\mathbf{w}\|$ es la distancia perpendicular del hiperplano al origen, y $\|\mathbf{w}\|$ es la norma euclídea de $\mathbf{w}$.

Suponiendo que todos los datos de entrenamiento cumplen las siguientes restricciones:

$$\mathbf{x}_i \cdot \mathbf{w} + b \geq +1 \text{ para } y_i = +1 \tag{2.44}$$

$$\mathbf{x}_i \cdot \mathbf{w} + b \leq -1 \text{ para } y_i = -1 \tag{2.45}$$

que se pueden combinar en:

$$y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1 \geq 0 \forall i. \tag{2.46}$$

De esta forma, los puntos cuyo $y_i = +1$ se encuentran en el hiperplano $H_1 : \mathbf{x}_i \cdot \mathbf{w} + b = 1$ con normal $\mathbf{w}$ y con una distancia perpendicular al origen $|1 - b| / \|\mathbf{w}\|$. De manera similar, los puntos con $y_i = -1$ se encuentran en el hiperplano $H_2 : \mathbf{x}_i \cdot \mathbf{w} + b = -1$ con igual normal pero en esta caso con distancia al origen $|-1 - b| / \|\mathbf{w}\|$. La distancia entre hiperplanos o margen es igual a $2/\|\mathbf{w}\|$. Téngase en cuenta que H_1 y H_2 son paralelos (tienen la misma normal) y ningún punto de entrenamiento se encuentra entre ellos. El objetivo es encontrar la pareja de hiperplanos con margen máximo, es decir, se quiere minimizar $\|\mathbf{w}\|^2$, sujeto a (2.46).

La formulación lagrangiana de este problema aporta dos ventajas principales. En primer lugar, se pueden reemplazar las restricciones de (2.46) por los multiplicadores de Lagrange, que son más fáciles de tratar. En segundo lugar, en esta reformulación los datos de entrada solo aparecen en forma de productos escalares entre vectores. Esta ventaja permite generalizar el procedimiento, como veremos más adelante para el caso no lineal.

Se introducen los multiplicadores positivos de Lagrange $\alpha_i, i = 1, ..., l$, uno para cada una de las inecuaciones (2.46). Esto da:

$$L_P \equiv \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^{l} \alpha_i y_i (\mathbf{x}_i \cdot \mathbf{w} + b) + \sum_{i=1}^{l} \alpha_i. \tag{2.47}$$

Esto se puede resolver mediante el problema "dual":

- (C_1) minimizar L_P con respecto a $\mathbf{w}, b$ y que a la vez las derivadas de L_P para todos los α_i desaparezcan, $\alpha_i \geq 0$.
- (C_2) maximizar L_P con la restricción de que el gradiente de L_P sea cero con respecto a $\mathbf{w}, b$ y que $\alpha_i \geq 0$.

El problema dual tiene la particularidad de que el máximo de L_P sujeto a C_2 ocurre para los mismos valores de $\mathbf{w}, b, \alpha$ que el mínimo de L_P sujeto a C_1. Hacer que el gradiente de L_P con respecto a $\mathbf{w}, b$ se anule da las siguientes condiciones:

$$w = \sum_i \alpha_i y_i \mathbf{x}_i, \tag{2.48}$$

$$\sum_i \alpha_i y_i = 0, \tag{2.49}$$

estas dos condiciones se pueden sustituir en (2.47) generando la siguiente ecuación para la formulación dual:

$$L_D = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j \tag{2.50}$$

Téngase en cuenta que L_P y L_D hacen referencia al problema *primal* (P) o *dual* (D).

El entrenamiento de los vectores de soporte (para el caso lineal, separable) trata de maximizar L_D con respecto a α_i sujeto a la restricción (2.49) y a la vez α_i debe ser positivo, con solución dada por (2.48). En la solución, se denominan vectores de soporte a aquellos puntos para los que $\alpha_i > 0$ y que se encuentran en uno de los hiperplanos H_1, H_2. Estos elementos son críticos para las SVM ya que se encuentran en la frontera de decisión. De modo que si el resto de puntos son eliminados o desplazados de manera que no crucen H_1 o H_2, y se repitiera el entrenamiento, se encontraría el mismo hiperplano de separación.

Haciendo uso de las condiciones de Karush-Kuhn-Tucker (KKT) [77] se puede determinar el valor de b y $\mathbf{w}$.

Datos no separables

Si se aplica el algoritmo anterior sobre datos no separables, no se encontrará una solución porque la función objetivo (el lagrangiano dual) crecería de manera arbitraria. Sin embargo, no es difícil ampliar las ideas generales del caso separable al caso no separable introduciendo una variable ξ de holgura en las restricciones y planteando un nuevo conjunto de restricciones:

$$\mathbf{x}_i \cdot \mathbf{w} + b \geq +1 - \xi_i \text{ para } y_i = +1 \tag{2.51}$$

$$\mathbf{x}_i \cdot \mathbf{w} + b \leq -1 + \xi_i \text{ para } y_i = -1 \tag{2.52}$$

$$\xi \geq 0 \forall i. \tag{2.53}$$

Para que se produzca un error en la clasificación para un patrón de entrada utilizando estas restricciones, es necesario que el valor correspondiente a ξ_i sea superior a la unidad. Para asignar el coste extra del error se debe cambiar la función objetivo a:

$$\frac{1}{2} \|\mathbf{w}\|^2 + C(\sum_{i=1} \xi_i)^k, \tag{2.54}$$

con $k \geq 1$, y donde C es un parámetro elegido por el usuario. Un valor grande de C asigna una alta penalización a los errores cometidos. Utilizando $k = 1$, ningún valor ξ_i ni sus correspondientes multiplicadores de Lagrange aparecen en la formulación del problema dual. En este caso de nuevo se debe maximizar (2.50) sujeto a (2.49), con la única diferencia de que en este caso α_i tiene un límite superior igual a C:

Nuevamente se usan las condiciones KKT para obtener el valor de b.

Máquina de soporte vectorial no lineal

En este apartado se trata la búsqueda del hiperplano de separación óptimo cuando los datos son no separables. Teniendo en cuenta que en la formulación anterior (2.50) los datos de entrada solo aparecen en forma de producto escalar $\mathbf{x}_i \cdot \mathbf{x}_j$, suponiendo ahora que podemos transformar los datos a otro espacio Euclídeo $\mathcal{H}$ (un espacio de una dimensión mucho mayor) utilizando una función Φ:

$$\Phi : \mathbf{R}^d \longrightarrow \mathcal{H}. \tag{2.55}$$

De esta manera el algoritmo de entrenamiento solo dependerá de los datos en forma de producto escalar en $\mathcal{H}$, es decir funciones con la siguiente forma $\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$. Asumiendo que se dispone de una *función kernel* K de modo que $K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$, solo se necesitaría usar K en el algoritmo de entrenamiento, de manera que no se requeriría el conocimiento explícito de qué es Φ. Existen varias funciones *kernel* conocidas que se suelen probar en la práctica. Un ejemplo de K podría ser la denominada "Radial Basis Function (RBF)":

$$K(\mathbf{x}_i, \mathbf{x}_j) = e^{-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / 2\sigma^2}. \tag{2.56}$$

Entonces, el vector solución quedaría como:

$$\mathbf{w} = \sum_{i=1} \alpha_i y_i \Phi(\mathbf{x}_i). \tag{2.57}$$

Por último, durante la fase de evaluación, lo que se quiere obtener es:

$$f(x) = sgn(\sum_{i=1}^{N_S} \alpha_i y_i K(\mathbf{s}_i, \mathbf{x}) + b), \tag{2.58}$$

donde *sgn* (*signum*) es la función que obtiene el signo, N_S es el número de vectores de soporte y s_i son los vectores de soporte.

En resumen, la idea consiste en plantear el mismo problema en un espacio de mayor dimensión donde se encuentra una función lineal, que puede no ser lineal en el espacio original.

Aunque las SVM son clasificadores binarios, existen alternativas que permiten aplicar los SVM a problemas multiclase:

- *1 vs rest*: en este tipo cada clasificador enfrenta los vectores de una clase contra el resto de clases.
- *1 vs 1*: se construye un clasificador por cada pareja de clases.

Al utilizar estas alternativas es necesario el uso de una etapa final en la que se implemente algún esquema de votación que permita adoptar una decisión final.

2.7.2. Multilayer Perceptron

El concepto de red neuronal nace con los intentos de encontrar representaciones matemáticas del procesamiento de la información en los sistemas biológicos [83]. Han sido utilizadas en sistemas de reconocimiento del habla [84], reconocimiento de expresiones faciales [85], previsión de series temporales [86] o como filtro de contenido Web [87] entre otros trabajos. El modelo de red neuronal alimentada hacia adelante (feed-forward), también conocida como Multilayer Perceptron (MLP), es el modelo con más éxito de red neuronal en el contexto de reconocimiento de patrones.

Para muchas aplicaciones, el modelo resultante puede ser considerablemente más compacto, y por lo tanto más rápido de evaluar, que un SVM teniendo la misma capacidad de generalización. El precio a pagar por esta compactibilidad, es que la función de verosimilitud, que forma la base del entrenamiento de la red, ya no es una función convexa de los parámetros del modelo. En la práctica, sin embargo, a menudo merece la pena invertir recursos computacionales durante la fase de entrenamiento con objetivo de lograr un modelo más compacto que procese los nuevos datos rápidamente.

El MLP está formado por múltiples capas de nodos (o neuronas), y cada capa está totalmente conectada a la siguiente, es decir cada nodo presenta una conexión a cada uno de los nodos de la capa posterior. La conexión entre dos nodos se denomina peso w, y su valor será ajustado durante la fase de entrenamiento. La primera capa es conocida como la capa de entrada, que alimenta al resto de la red con los datos de entrada. A continuación, se sitúan como mínimo una capa oculta. Por último, se sitúa la capa de salida que devuelve la decisión final de la red. El número de nodos en cada una de las capas de la red puede ser diferente.

Consideremos la red de dos capas de la Figura 2.8, con datos de entrada de dimensión D, $x_1, ..., x_D$, una única capa oculta con M nodos y una salida de dimensión K, $y_1^*, ..., y_K^*$.

En cada nodo de la capa oculta se construye una combinación lineal:

$$a_j = \sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}, \tag{2.59}$$

donde $j = 1, ..., M$, el superíndice (1) indica que los parámetros correspondientes se encuentran en la primera capa de la red. Los parámetros $w_{ji}^{(1)}$ son los pesos entre la

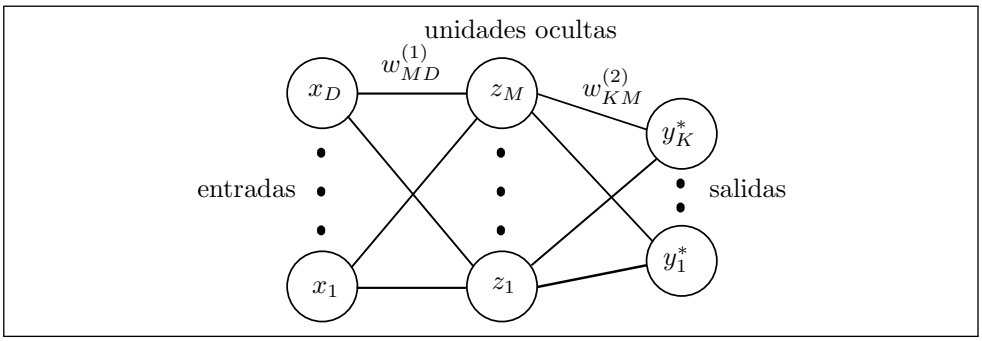

Figura 2.8: *Diagrama de red para la red neuronal de dos capas.*

capa de entrada y la primera capa oculta y $w^{(1)}_{j0}$ se conoce como "bias". Las variables a_j se corresponden con las activaciones de cada nodo, que serán transformadas posteriormente usando la función de activación no lineal $f(\cdot)$:

$$z_j = f(a_j). \tag{2.60}$$

Las variables z_j se corresponden con las *unidades ocultas*. La función denominada *Rectified Linear Unit* (ReLU) es un ejemplo de función de activación no lineal, que se define como $f(x) = \text{máx}(0, x)$. Para los problemas de regresión estándar, la función de activación se corresponde con la función *identidad* $f(x) = x$. A continuación, los valores de z_j se vuelven a combinar linealmente en las siguientes capas generando las *unidades de activación de salida*:

$$a_k = \sum_{j=1}^{M} w^{(2)}_{kj} z_j + w^{(2)}_{k0}, \tag{2.61}$$

donde $k = 1, ..., K$, y K es el número total de salidas. Esta transformación se corresponde con la segunda capa de la red. Por último, se transforman las unidades de activación de salida de la última capa usando una función de pérdida, σ, para dar un conjunto de salidas de la red y_k. Estas etapas previas (2.59)-(2.61) pueden combinarse en una única función general de la red:

$$y^*_k(\mathbf{x}, \mathbf{w}) = \sigma(\sum_{j=1}^{M} w^{(2)}_{kj} f(\sum_{i=1}^{D} w^{(1)}_{ji} x_i + w^{(1)}_{j0}) + w^{(2)}_{k0}). \tag{2.62}$$

En definitiva, el modelo de red neuronal es una función no lineal que transforma un conjunto de variables de entrada x_i en un conjunto de variables de salida y^*_k controladas por un vector w de parámetros ajustables.

Para problemas multiclase, se puede usar la función de pérdida denominada *softmax*. Esta función es una generalización de la función logística para considerar múltiples

clases. La función toma un vector K-dimensional, siendo K el número de clases, y devuelve un vector K-dimensional de valores reales $0 \leq y_k^* \leq 1$ cumpliendo $\sum_k y_k^* = 1$. Se representa por:

$$y_k^*(\mathbf{x}, \mathbf{w}) = \frac{e^{a_k(\mathbf{x},\mathbf{w})}}{\sum_j e^{a_j(\mathbf{x},\mathbf{w})}}. \tag{2.63}$$

La manera más común de entrenar una red neuronal es mediante el algoritmo de retropropagación (*backpropagation*)[88]. El algoritmo de retropropagación es un algoritmo de aprendizaje supervisado que se divide en dos pasos. Primero, en el paso hacia adelante (*forward*), cada neurona genera un impulso que se va propagando por toda la red hasta obtener una salida. Dado un conjunto de entrenamiento compuesto por los vectores de entrada $\mathbf{x}_n$, donde $n = 1, ..., N$, y un conjunto de salidas etiquetadas como y_n, el algoritmo de retropropagación intenta minimizar la función de error obtenida por la salida de la red y_k^* y la salida deseada y_k. Considerando un problema multiclase la función error presenta la siguiente forma:

$$E(\mathbf{w}) = -\sum_{n=1}^{N}\sum_{k=1}^{K} y_{kn} \ln y_k^*(\mathbf{x}_n, \mathbf{w}) \tag{2.64}$$

Una vez calculado el error cometido por la red, se usa este error para actualizar los pesos de cada neurona en el paso conocido como *backward*,

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{\tau} + \mathbf{\Delta w}^{\tau}, \tag{2.65}$$

donde τ indica el paso de iteración. Diferentes algoritmos implican diferentes elecciones de la función de actualización de pesos $\mathbf{\Delta w}(\tau)$.

El algoritmo de retropropagación alterna de manera iterativa entre los dos pasos hasta que se cumpla el criterio de parada (error obtenido menor que un umbral determinado, alcanzado número máximo de iteraciones, ...).

2.7.3. Convolutional Neural Network

Las redes neuronales convolucionales (Convolutional Neural Network (CNN)) son variantes del MLP diseñadas para usar las mínimas unidades de procesamiento. A pesar de que el origen de las CNNs se remonta varias décadas atrás [89], han mostrado recientemente un aumento de popularidad debido principalmente al uso de GPUs más modernas que han permitido el procesamiento de enormes cantidades de datos dentro de tiempos razonables. Las CNNs han sido probadas con éxito en tareas de clasificación de imágenes [90], detección de objetos [91], reconocimiento facial [92] y super-resolución [93] entre otras. En general, las CNNs se han aplicado extensamente utilizando imágenes como datos de entrada, esto se debe principalmente al hecho de que las CNNs explotan la fuerte correlación espacio-local presente en las imágenes.

Consideremos la arquitectura de ejemplo mostrada en la Figura 2.9 para esta explicación.

Podemos destacar tres ideas clave en las CNN:

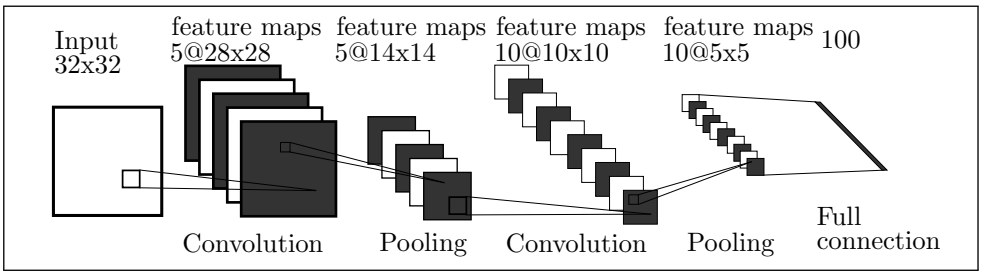

Figura 2.9: *Arquitectura de ejemplo de una red neuronal convolucional.*

- Campos receptivos locales: frente al resto de clasificadores que suelen tratar las imágenes como un único vector, las CNN permiten trabajar con la imagen en forma de matriz bidimensional. Además, frente al caso del MLP en el que por cada píxel se crea una conexión, las CNN utilizan pequeñas regiones (ventanas) denominadas campos receptivos locales que se van desplazando por todas las posiciones de la imagen. Por cada campo receptivo se crea un nodo oculto en la primera capa. Nótese que la primera capa reduce su tamaño en cuatro píxeles por dimensión, esto se debe al hecho de que la ventana no pueda centrarse en los bordes y usamos un tamaño 5×5 para la misma. Cada conexión de la ventana aprende un peso y el nodo oculto aprende además un bias.

- Pesos compartidos: las conexiones y bias mencionadas en el punto anterior se utilizan para todos los nodos ocultos de la primera capa (28×28). Así la salida del nodo oculto $j, k -$ ésimo, de la primera capa sería:

$$a_{j,k} = w_0 \sum_{l=0}^{4} \sum_{m=0}^{4} w_{l,m} x_{j+l,k+m}, \tag{2.66}$$

  donde w_0 se corresponde con el bias, w con los distintos pesos y x con los valores de la capa de entrada. Estos pesos compartidos, junto con el bias, son conocidos en las CNN como *kernel* o filtro. De manera que en cada capa convolucional se realiza una convolución 2D entre la entrada y el *kernel* y el resultado de esta convolución se denomina mapa de características. Además, por cada capa de convolución se suelen utilizar varios filtros (cada uno genera un mapa de características). Así, en el caso de nuestro ejemplo, la primera capa de convolución utiliza cinco filtros mientras que en la segunda capa de convolución se utilizan diez. Los pesos de los filtros serán ajustados durante el entrenamiento de manera similar a como se ajustaban los pesos de las conexiones en el MLP. Gracias al uso de los pesos compartidos las CNNs requieren de un menor número de parámetros a entrenar frente a otros modelos de redes neuronales. Esto se hace más notable cuanto mayor sean las dimensiones de la imagen de entrada.

- Inmediatamente después de las capas de convolución se suele incluir una capa denominada *pooling*. El objetivo de esta capa es resumir la información condensando los valores de cada región de los mapas de características en un único valor. Así, en nuestro ejemplo, se utiliza una capa de *pooling* con tamaño de

ventana 2×2, para reducir el tamaño de cada mapa de características a la mitad. Existen dos tipos de *pooling*: *max-pooling* y *avg-poling*. Si desplazando la ventana por cada mapa de características se selecciona el valor máximo de cada región hablamos de *max-pooling* y si se toma el valor medio de todos los píxeles decimos *avg-pooling*.

En la práctica, generalmente se suelen situar varios pares de capas de convolución y *pooling*. En cada etapa existe un alto grado de invariancia a las transformaciones de entrada comparadas con la capa anterior. La reducción gradual provocada por las capas de *pooling* se suele compensar incrementando el número de mapas de características en las siguientes capas de convolución. En nuestro ejemplo aumentan de cinco a diez mapas de características en la segunda capa de convolución. Por último, antes de la capa de pérdida, suele ponerse al menos una capa totalmente conectada.

La red completa puede ser entrenada utilizando el algoritmo de retropropagación. El entrenamiento es similar al caso del MLP aunque requiere de unas pequeñas modificaciones para ajustar el entrenamiento a las capas convolucionales y de pooling.

Capítulo 3

Estimando la pose de cámara mediante fusión de información de descriptores para emparejamiento de *Keypoints*

3.1. Introducción

En la Sección 2.2 hemos comentado algunos de los descriptores locales de *keypoints* más conocidos. SIFT sigue siendo uno de los descriptores que mejores resultados obtiene, a pesar de la aparición de numerosas alternativas como SURF o los descriptores binarios. En la fecha actual no es posible afirmar que exista un descriptor que sea el mejor frente al resto. Numerosos trabajos han realizado estudios comparativos entre varios descriptores locales sin llegar a obtener un claro ganador [18-21].

En función de lo anterior, nuestra estrategia se ha basado en efectuar una propuesta para mejorar el proceso de emparejamiento de *keypoints*, mediante la fusión de información de varios descriptores usando la Teoría de Dempster-Shafer (en adelante TDS) para este propósito. Los detalles básicos de la TDS pueden consultarse en la Sección 2.6.

En términos generales, la TDS utiliza el concepto de grados de evidencia (una versión más débil de probabilidad), haciendo la manipulación de la incertidumbre especialmente atractiva debido a su simplicidad y a que no necesita la especificación de ninguna probabilidad *a priori* o condicional cuya obtención pudiera no ser factible en ciertos problemas.

En este capítulo se analizan en primer lugar las estrategias empleadas en trabajos directamente relacionados con la fusión de información de descriptores. En segundo lugar se desarrolla la propuesta original que se plantea en este trabajo y la evaluación del rendimiento de la misma frente a otras alternativas existentes.

3.2. Estrategias de fusión de información de descriptores

A pesar del esfuerzo dedicado al desarrollo de nuevos descriptores de *keypoints* se ha prestado poca atención a la idea de fusionar la información proporcionada por varios descriptores. Además, los trabajos existentes que abordan esta idea, en general, analizan los resultados obtenidos sobre tareas de más alto nivel como la detección de caras o clasificación de imágenes. Esta metodología impide evaluar el rendimiento específico de la fusión de información en una tarea de más bajo nivel. Por esta razón, en el desarrollo de este trabajo se pretende evaluar los beneficios producidos por la fusión de varios descriptores en una tarea de bajo nivel: búsqueda de parejas de *keypoints* que se correspondan con el mismo punto de la escena.

La fusión de información puede realizarse de dos maneras diferentes dependiendo del momento en el que se aplique el proceso de fusión. De acuerdo con ello hablamos de fusión temprana o fusión tardía. Un ejemplo del primer caso es la concatenación de dos o más vectores de características en un único vector que más tarde será clasificado. Un ejemplo del segundo caso sería combinar las decisiones de clasificación obtenidas por cada descriptor de manera individual en un nuevo valor del que extraer la decisión final.

Como ejemplo de fusión temprana relacionado con los objetivos de este capítulo podemos citar la propuesta de Weng *et. al.*[94]. En este trabajo proponen fusionar los descriptores SIFT y SURF en el problema del reconocimiento de caras con presencia de oclusión y cambios lumínicos. Esta propuesta utiliza como descriptor la concatenación de ambos vectores de características. Para calcular la distancia, al usar descriptores compuestos por características de diferentes modalidades, no se recomienda utilizar el espacio euclídeo ya que no permite representar correctamente la información. Por este motivo, optan por un esquema con aprendizaje de métrica (*metric learning*) [95], que permite aprovechar el poder discriminatorio transformando la distancia. Este esquema requiere una etapa de aprendizaje, de forma que el nuevo conjunto de distancia sea lo más pequeña posible entre descriptores de la misma clase y lo más grande posible entre descriptores pertenecientes a distintas clases.

Dentro de la categoría de fusión tardía existen varios trabajos relacionados. He *et al.* [96], proponen utilizar varios descriptores de manera conjunta para la clasificación de imágenes. En primer lugar detectan un conjunto de *keypoints* sobre los que construyen varios descriptores de características. Por cada descriptor construyen una "bolsa de palabras visuales" independiente y calculan los vecinos más cercanos para cada clase. Para clasificar una nueva imagen, calculan una distancia que depende de la diferencia entre su histograma y el de los vecinos más cercanos de cada descriptor. Para tomar una decisión utilizan un algoritmo que tiene en cuenta las distancias al vecino más cercano por cada descriptor. Como inconveniente de esta propuesta podemos mencionar que no se propone una manera efectiva de combinar los descriptores, obviando el hecho de que algunos descriptores se comportan mejor que otros en ciertos casos.

Mountney *et al.* [97], proponen un modelo bayesiano para fusionar las decisiones dadas por un conjunto de descriptores en la búsqueda de *keypoints* correspondentes orientado a aplicaciones de cirugía de mínima invasión. En una etapa inicial, seleccionan el conjunto de descriptores óptimo usando un algoritmo de aprendizaje automático que descarta los descriptores más irrelevantes intentando seleccionar del resto, aquellos descriptores que presenten una correlación baja entre sí. Posteriormente, para cada pareja de *keypoints*, calculan una medida de similitud sobre el conjunto de descriptores seleccionados. Finalmente, una red bayesiana, que ha sido previamente entrenada con un conjunto de datos supervisado, se utiliza para fusionar las medidas de similitud tomando una decisión final sobre si los *keypoints* son o no emparejados. El principal inconveniente de esta propuesta es que usa un esquema de entrenamiento que depende de la base de datos usada. Puede darse el caso de que un descriptor que no sea muy apropiado para una base de datos concreta lo sea en una situación desconocida, pero al ser descartado en un primer paso no se podrá evaluar su contribución.

El trabajo de Perakis *et al* [98] propone la fusión de información de descriptores 2D y 3D en el problema de detección de puntos de referencia faciales. La estrategia se basa en el cálculo inicial de un conjunto de descriptores cuyas distancias entre los mismos son transformadas en valores de similitud. Estos valores de similitud se combinan y se usan como puntuación para el emparejamiento final. En su esquema de fusión evalúan diferentes funciones de transformación de distancia-similitud y diferentes reglas de combinación como la regla de la suma, del producto, del máximo, del mínimo, de la mediana y el voto por mayoría. Se hablará de estas propuestas en la Sección 3.3.1 de este mismo capítulo, debido a que haremos uso de algunas de estas reglas en la sección de experimentación.

En general, los trabajos existentes para fusionar la información de varios descriptores requieren una etapa de entrenamiento en la que se asignen los pesos a cada descriptor. La propuesta que se presenta en este trabajo intenta abordar el problema de búsqueda de la mejor correspondencia para un *keypoint* origen entre un conjunto de *keypoints* objetivo sin hacer uso de ningún conocimiento previo y a la vez utilizando un factor de confianza para cada descriptor, para ello haremos uso de la TDS.

3.3. Nuevo método para emparejamiento de Keypoints

En esta sección, se explica nuestra propuesta para la búsqueda de correspondencias entre puntos clave mediante la fusión de información de múltiples descriptores utilizando la TDS.

Sea p un *keypoint* determinado por un detector en una imagen origen. Nuestro objetivo es encontrar los *keypoints* q del conjunto $\mathcal{Q} = \{q^1, \ldots, q^N\}$ detectados en una segunda imagen que hagan referencia al mismo punto del espacio que p. Debe tenerse en cuenta que el *keypoint* correspondiente puede no estar presente en $\mathcal{Q}$, ya sea por oclusión en la imagen o debido a un error del detector.

Sea

$$\Psi(p) = \{\psi_k(p) \mid k = 1 \dots K\}, \tag{3.1}$$

el conjunto de K descriptores usados para describir el *keypoint* p, donde

$$\psi_k(p) = (f_{1p}^k, f_{2p}^k, f_{3p}^k, \dots f_{n_k p}^k,) \tag{3.2}$$

es el vector de n_k características del *keypoint* p correspondiente al descriptor k-ésimo. Téngase en cuenta que estamos usando un solo detector de *keypoints* y calculamos K descriptores sobre cada uno de ellos. En un primer paso se comparan los vectores de características $\psi_k(p)$ con aquellos $\psi_k(q)$ pertenecientes a los elementos del conjunto $\mathcal{Q}$.

Usamos

$$d_k(p, q) = |\psi_k(p) - \psi_k(q)| \tag{3.3}$$

como una medida de distancia entre los *keypoints* p y q para el descriptor ψ_k.

Para cada descriptor ψ_k, los elementos del conjunto $\mathcal{Q}$ se ordenan de menor a mayor distancia utilizando la medida d_k:

$$\phi_k(p, \mathcal{Q}) = \left\{(\phi_k^1, \dots, \phi_k^N) \ \in \mathcal{Q} \mid d_k(p, \phi_k^i) \leq d_k(p, \phi_k^j) \ \forall i > j\right\}. \tag{3.4}$$

Cada vector $\phi_k(p, \mathcal{Q})$ incluye los puntos de $\mathcal{Q}$ ordenados de manera ascendente en base a d_k y es por tanto una permutación de los elementos del conjunto $\mathcal{Q}$.

Finalmente sea $\Phi(p, \mathcal{Q})$ el conjunto asociado al *keypoint* p y al conjunto $\mathcal{Q}$:

$$\Phi(p, \mathcal{Q}) = \{\phi_k(p, \mathcal{Q}) \mid k = 1 \dots K\}, \tag{3.5}$$

Obsérvese que:

- Para cada valor de k, los primeros elementos del vector $\phi_k(p, \mathcal{Q})$ se corresponden con aquellos *keypoints* con mayor probabilidad de ser la pareja correcta del *keypoint* p. Así, en un caso ideal, el primer elemento de $\phi_k(p, \mathcal{Q})$ debería ser la pareja perfecta de p para todos los descriptores, y, en general, todos los conjuntos $\phi_k(p, \mathcal{Q})$ deberían producir el mismo vector ordenado. Sin embargo, en la práctica, esto no suele ocurrir, ya que cada descriptor se suele comportar mejor en determinadas circunstancias. Por esta razón cada vector $\phi_k(p, \mathcal{Q})$ se corresponde con una permutación diferente de los elementos del conjunto $\mathcal{Q}$.
- En el momento de analizar las correspondencias entre el *keypoint* p y los elementos de $\phi_k(p, \mathcal{Q})$, para determinar la pareja perfecta de p, el uso de todos los candidatos del vector $\phi_k(p, \mathcal{Q})$ implicaría un gasto innecesario de memoria y tiempo de computación ya que los candidatos con mayor probabilidad se corresponden con los elementos iniciales del vector. Por este motivo, proponemos reducir el tamaño de los vectores conservando tan solo los primeros n elementos, es decir, $|\phi_k(p, \mathcal{Q})| = n$. La Figura 3.1 muestra un ejemplo para $K = 3$ descriptores, usando sólo los 3 primeros elementos de cada vector ($n = 3$).
- En $\Phi(p, \mathcal{Q})$ se encuentra la información final, para el *keypoint* p y el conjunto $\mathcal{Q}$, proporcionada por los K descriptores usados.

Figura 3.1: *Ejemplo de los resultados de emparejamiento para $K = 3$. (Izquierda) Imagen origen con el keypoint p. (Derecha) Imagen objetivo con los 3 mejores candidatos ϕ_k^i de cada descriptor con $i = 1,2,3$. Aunque los tres descriptores de forma individual se equivocan y seleccionan al candidato correcto en segunda posición, nuestra propuesta al fusionar la información podría asignar la primera posición para dicho punto.*

3.3.1. Propuesta de fusión de descriptores con TDS para emparejamiento de Keypoints

Usando reglas ya propuestas

Las reglas de fusión de información empleadas en los trabajos de Perakis y Jain [98, 99] son muy interesantes. Explicamos en primer lugar cómo adaptar estas propuestas al problema planteado.

Siendo $\psi_k(p)$ y $\psi_k(q)$, los vectores de características para los *keypoints* p y q, respectivamente, obtenidos con el descriptor k-ésimo, la distancia normalizada entre ambos $\hat{d}_k(p,q)$ se obtiene mediante:

$$\hat{d}_k(p,q) = \frac{d_k(p,q)}{d_k(p,\phi_k^n)}, \tag{3.6}$$

donde ϕ_k^n se corresponde con el último elemento de $\phi_k(p,\mathcal{Q})$ del descriptor k, es decir aquel con mayor distancia respecto al *keypoint* p y por tanto con menor probabilidad para ser su par óptimo.

A partir de esta distancia se define como medida de similitud normalizada para el descriptor k la función:

$$S_k(p,q) = 1 - \hat{d}_k(p,q)^e, \tag{3.7}$$

donde e regula el tipo de transformación. Cuando $e = 1$ obtenemos una transformación lineal, mientras que con $e = 2$ obtenemos una transformación cuadrática que favorece a los elementos más cercanos al objetivo [98]. Obsérvese que el valor de la medida de similitud $S_k(p,q)$ es 1 si los dos vectores de características son iguales, y menor si ambos vectores son diferentes.

Los valores de similitud $S_k(p,q)$ obtenidos para los K descriptores se pueden fusionar usando las siguientes reglas:

- Regla de la suma:

$$S_A(p,q) = \frac{1}{N}\sum_{k=1}^{K} S_k(p,q), \tag{3.8}$$

  que coincide con la media aritmética o distancia Manhattan (L1).

- Regla de la media cuadrática:

$$S_E(p,q) = \frac{1}{\sqrt{K}}\left(\sum_{k=1}^{K} S_k(p,q)^2\right)^{1/2}, \tag{3.9}$$

  que coincide con la distancia Euclídea (L2).

- Regla del producto:

$$S_G(p,q) = \left(\prod_{k=1}^{K} S_k(p,q)\right)^{1/K}, \tag{3.10}$$

  que se corresponde con la media geométrica.

- Regla del máximo:

$$S_{max}(p,q) = max_{k=1}^{K}\left(S_k(p,q)\right). \tag{3.11}$$

Cada una de estas reglas provoca un comportamiento diferente en la fusión. Por ejemplo, el uso de la regla del producto es bastante severa, ya que basta con que una única medida de similitud para un descriptor tenga un valor cercano a cero para excluir a esa posible pareja q. En la sección de experimentación, evaluamos distintas combinaciones de las reglas de fusión y las medidas de similitud normalizadas ($e = \{1,2\}$), con objetivo de encontrar la mejor configuración para nuestro problema.

Nueva regla para fusión

Siguiendo la notación de la Sección 2.6, nuestra propuesta sobre cómo utilizar la TDS para el problema de fusionar la información de los diferentes descriptores es la siguiente:

Usando como entrada los vectores $\phi_k(p,\mathcal{Q})$ del conjunto $\Phi(p,\mathcal{Q})$ construimos el conjunto de distribuciones de evidencia:

$$\Pi(p) = \{\pi_k(p) \mid k = 1 \dots K\}, \tag{3.12}$$

siendo

$$\pi_k(p) = \left(\delta_k(p,\phi_k^1), \dots, \delta_k(p,\phi_k^n)\right), \tag{3.13}$$

y

$$\delta_k(p,q) = \begin{cases} \varphi \left(\frac{d_k(p,\phi_k^1)}{d_k(p,q)} \right)^{\beta} & si\ q \in \phi_k(p,\mathcal{Q}) \\ 0 & \text{en otro caso.} \end{cases} \tag{3.14}$$

Obsérvese que:

- la variable ϕ_k^1 se corresponde con el primer elemento de $\phi_k(p,\mathcal{Q})$, φ es un factor constante para que la integral de la distribución sea uno, y β es un parámetro exponencial (evaluado en la Sección 3.4) que modifica la forma de la distribución.
- Las distribuciones de evidencia creadas con las ecuaciones (3.13) presentan las siguientes propiedades: (a) el mejor candidato (aquel con la distancia más pequeña) recibe el valor de evidencia más alto, (b) es una distribución normalizada que se puede usar para comparar los resultados de diferentes descriptores (debe recordarse que cada descriptor trabaja en su propio espacio de características, con su propia dimensión, y en consecuencia, las distancias entre descriptores diferentes no son directamente comparables) y (c) los valores del parámetro β mayores a uno tienden a incrementar la importancia de los primeros términos, mientras que los valores menores a uno tienden a suavizar la distribución.
- Un descriptor ideal produciría una distancia muy pequeña para el candidato correcto y una distancia muy grande para el resto. Por tanto, un descriptor ideal produciría una distribución de evidencia con un valor muy alto para el primer elemento y valores muy bajos para el resto. Sin embargo, si el descriptor no es bueno, puede producir valores de distancia grandes (o pequeños) para todos los elementos, incluyendo al que debería ser la pareja correcta, generando así una distribución uniforme. Por otro lado, si el *keypoint* p de la imagen origen forma parte de un patrón repetitivo (esto ocurriría, por ejemplo, si la imagen fuera un tablero de ajedrez), incluso un buen descriptor podría considerar muchas parejas con una distancia pequeña, produciendo nuevamente una distribución uniforme.

Basándonos en estos comentarios, se propone usar la entropía de Shannon como un factor de confianza que se calcula como:

$$c_k(p) = 1 - \sum_{i=1}^{n} -\delta_k(p,\phi_k^i) \log\left(\delta_k(p,\phi_k^i)\right), \tag{3.15}$$

porque cuando la entropía de la distribución es cero (es decir, solo hay un único elemento con un valor de evidencia uno y el resto es cero) entonces, $c_k(p)$ tiene valor 1, mientras que si la distribución de evidencia $\pi_k(p)$ tienda a ser uniforme, $c_k(p)$ tiende al valor 0.

Nuestra propuesta de uso de la entropía se sustenta sobre la base de que esta idea ha sido previamente utilizada como medida de confianza en la fusión de varios clasificadores de distintos ámbitos [100-102].

Las distribuciones de evidencia $\pi_k(p)$ y los valores de confianza $c_k(p)$ se utilizan para definir la BBA para cada *keypoint*. En nuestro problema, el conjunto de hechos $\mathcal{S}$ está compuesto por todos los elementos de $\mathcal{Q}$ seleccionados para crear las distribuciones de evidencia. Por tanto, podemos definir $\mathcal{S}$ como el conjunto de elementos de $\mathcal{Q}$ contenidos en $\Phi(p, \mathcal{Q})$:

$$\mathcal{S} = \{\mathrm{w} \in \mathcal{Q} \mid \exists k,\ \mathrm{w} \in \phi_k(p, \mathcal{Q})\}. \tag{3.16}$$

Téngase en cuenta que $\mathcal{S}$ es un subconjunto de $\mathcal{Q}$. Considerando cada descriptor k como un sensor independiente, la BBA del k-ésimo sensor se define a partir de las ecuaciones (3.14) y (3.15) como:

$$\begin{array}{l} m_k(\Omega) = c_k(p), \\ m_k(\mathrm{w}) = \delta_k(p, \mathrm{w}). \end{array} \tag{3.17}$$

Usando la regla de combinación basada en *t-norms* explicada previamente (2.42), se obtienen los valores de creencia fusionada sobre el conjunto exhaustivo de hipótesis $\mathcal{S}$. La transformación *pignistic* (2.43) se aplica sobre los resultados fusionados ordenando los *keypoints* $\mathcal{S}$ en orden descendente:

$$\mathcal{S}' = \left(\mathrm{w}^1, \mathrm{w}^2, \ldots, \mathrm{w}^{|\mathcal{S}|} \mid \mathrm{w}^i \in \mathcal{S} \ \wedge\ BetP(\mathrm{w}^i) > BetP(\mathrm{w}^j)\ \forall i < j\right). \tag{3.18}$$

Finalmente, el *keypoint* w^1 es el elemento más probable de $\mathcal{Q}$ para ser la pareja correcta del *keypoint* p, w^2 es el segundo más probable y así sucesivamente.

Por último, en la siguiente Sección 3.4 se explica la medida utilizada para prevenir la aparición de falsos positivos en el problema.

3.4. Experimentación, resultados y discusión

En esta sección se presenta la metodología de evaluación empleada, los experimentos realizados para validar el método propuesto, y la discusión de los resultados.

3.4.1. Metodología de evaluación

Para la experimentación se ha utilizado la popular base de datos de Oxford, Schmid y Mikolajczyk [18].

Esta base de datos está formada por siete escenas. Cada escena contiene un total de cinco imágenes. La primera imagen toma siempre el papel de imagen origen y el resto forman las imágenes objetivo, que contienen cambios con respecto a la primera. Los cambios producidos en cada escena son los siguientes: desenfoque (*bikes*, y *trees*), cambio de punto de vista (*graffiti* y *wall*), cambio lumínico (*leuven*), compresión JPEG (*ubc*), zoom y rotación (*boat*) (ver Figura 3.2). En la citada base de datos se incluye también la matriz de homografía, que establece la relación entre cada par de

imágenes origen-objetivo, lo que permite obtener el *ground truth* necesario para la evaluación.

Para evaluar la propuesta que se hace en el marco de esta Tesis Doctoral, sobre fusión de información de descriptores para emparejamiento óptimo de *keypoints* correspondientes, se compara el rendimiento de la misma con respecto al uso de los citados descriptores de forma individual para el mismo problema y también con respecto a métodos previos para fusionar información de descriptores (ver Sección 3.3.1). Se ha aplicado la siguiente metodología:

Usando cada descriptor de forma individual

(a) Para cada par de imágenes (origen y objetivo), se extraen dos conjuntos de *keypoints*, uno en cada imagen, usando el mismo detector. Los detectores ensayados han sido dos, SURF y SIFT, y por tanto se han llevado a cabo dos

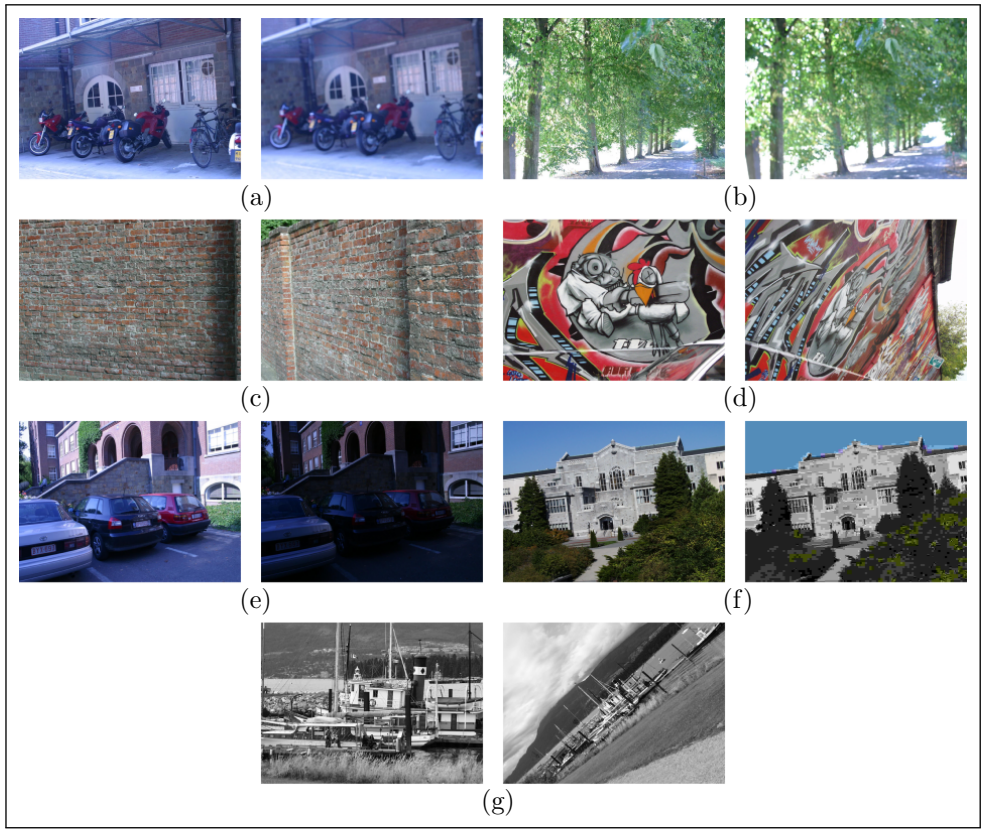

Figura 3.2: *Imágenes de la base de datos de Oxford: desenfoque (a)(b), cambio punto de vista (c)(d), cambio lumínico (e), compresión JPEG (f), zoom y rotación (g)*

experimentos distintos, uno para cada detector, para cada par imagen origen-imagen objetivo.

(b) Para cada *keypoint* de los dos conjuntos extraídos se ha calculado el valor de cinco descriptores. Los descriptores ensayados son SIFT, SURF, ORB, BRISK y FREAK. Obsérvese que el hecho de usar un único detector, en cada experimento, garantiza que los resultados entre los distintos descriptores usados sean comparables.

(c) Como estrategia final de correspondencia de pares de *keypoints* (entre imagen origen y objetivo) se ha utilizado el ratio de la distancia al vecino más cercano (Nearest Neighbor Distance Ratio matching (NNDR) [29]) para todos los métodos evaluados. Según esta estrategia un candidato ϕ_k^1 es seleccionado como pareja del *keypoint* p sólo si cumple la condición $d(p, \phi_k^1)/d(p, \phi_k^2) < \alpha$, donde $\alpha \in [0, 1]$. Esta medida previene la aparición de falsos positivos, rechazando aquellos casos en que un descriptor obtiene distancias muy similares para los dos mejores candidatos. Para los descriptores SURF y SIFT se han usado las distancias L1 y L2. Para el resto de descriptores, al ser binarios, se usa la distancia Hamming. Para evaluar si una pareja seleccionada es correcta, se usa el ratio de la intersección sobre la unión [29].

Sea ϵ el error de solapamiento definido como:

$$\epsilon = 1 - \frac{P \cap \mathbf{H}^T \cdot Q \cdot \mathbf{H}}{P \cup \mathbf{H}^T \cdot Q \cdot \mathbf{H}}, \tag{3.19}$$

donde P y Q son las regiones centradas respectivamente en el *keypoint* de la imagen origen p y el *keypoint* de la imagen objetivo q, y $\mathbf{H}$ es la matriz de homografía que relaciona ambas imágenes. Una pareja se considera correcta si el área de la intersección entre las dos regiones es inferior al 50 por ciento de la unión de ambas regiones, $\epsilon < 50\,\%$.

(d) Para cada descriptor se han calculado las medidas de precisión y exhaustividad y la medida-F, que permiten evaluar el rendimiento individual de cada uno de ellos. Estas tres medidas:

$$\text{precisión} = \frac{tp}{tp + fp}; \quad \text{exhaustividad} = \frac{tp}{tp + fn}; \tag{3.20}$$

$$F = 2\frac{\text{precisión} \cdot \text{exhaustividad}}{\text{precisión} + \text{exhaustividad}}; \tag{3.21}$$

se calculan a partir de los valores de la tabla de confusión.

Debe tenerse en cuenta que los fp son aquellos errores derivados del mal emparejamiento entre un par de *keypoints* que no se corresponden con el mismo punto de la escena. Por otro lado los fn son aquellos fallos producidos por omisión, es decir el caso de un *keypoint* en la imagen origen para el que existe un *keypoint* correspondiente en la imagen objetivo, pero que debido a la estrategia de emparejamiento ha quedado sin emparejar.

La precisión indica lo bueno que es un método rechazando parejas incorrectas, mientras que la exhaustividad mide cómo de bueno es un método encontrando parejas correctas. Estas dos medidas, por tanto, se complementan entre sí. La medida-F resume las dos medidas anteriores en un único valor mediante la media armónica entre la precisión y la exhaustividad. Por tanto, nuestro análisis de resultados se basa en la medida-F, ya que permite determinar el rendimiento de un método usando un único valor.

Fusionando información de varios descriptores

Con respecto al método de fusión de información que se propone se ha seguido una metodología similar a la descrita anteriormente:

- Se procede de forma idéntica en lo que se refiere a los apartados (a) y (d).
- Con respecto a lo indicado en el apartado (b) se contemplan cada una las posibles combinaciones de los 5 descriptores mencionados para fusionar la información.
- Con respecto a lo indicado en el apartado (c) se usa el ratio entre los dos candidatos más probables, $(\mathbf{w}^2/\mathbf{w}^1) < \alpha$.

Finalmente resaltar que, con el objetivo de evaluar el rendimiento del método de fusión que se propone, se comparan sus resultados con otros métodos de fusión ya propuestos para el mismo objetivo.

En todos los casos se contemplan todas las posibles combinaciones de los 5 descriptores para fusionar información.

Para comparar rendimientos se ha aplicado el test no paramétrico de rango con signo de Wilcoxon [103] sobre la medida-F, ya que con esta prueba es posible comparar dos métodos mediante el uso de la hipótesis nula H_0, que asume que ambos métodos proporcionan resultados similares, y la hipótesis alternativa H_1, que indica la existencia de diferencias estadísticamente significativas entre los dos métodos. Para todas nuestras pruebas realizadas se ha usado un nivel de significación igual a 0.05, lo que indica que los resultados de las pruebas son confiables en un 95.5 %. Por tanto, para poder rechazar la hipótesis nula se debe cumplir la condición $p \leq 0.05$.

3.4.2. Resultados y discusión

Usando de forma individual cada descriptor

Los valores presentados en la Tabla 3.1 se corresponden con el mejor resultado obtenido por cada descriptor sobre todas las imágenes de la base de datos. Siguiendo la metodología explicada, cada descriptor ha sido evaluado sobre todas las escenas variando los valores de α en el intervalo $[0, 1]$ con un incremento de 0.01. En la tabla aparece la solución obtenida con la medida-F más alta.

Las principales conclusiones son:

- Con respecto al tipo de distancia utilizada se puede comprobar que el uso de la distancia $L2$ empeora el rendimiento en todos los casos.

Como consecuencia de lo anterior, para el resto de experimentos se utilizan las configuraciones resaltadas en negrita de la Tabla 3.1.

Usando propuestas previas en fusión de descriptores

Se evalúa el rendimiento de los métodos de fusión de características estándar presentados en la Sección 3.3.1. En total hemos probado dos medidas de similitud normalizadas ($e = \{1, 2\}$ (3.7)) y cuatro reglas de fusión (suma, media cuadrática, producto y máximo (3.8)-(3.11)). En total, se han probado ocho propuestas de fusión distintas, y para cada una de ellas se han evaluado todas las posibles combinaciones entre los cinco descriptores. Los resultados se muestran en la Tabla 3.2, donde los asteriscos indican qué descriptores han intervenido en cada fusión. En todos los casos se muestra el mejor resultado obtenido sobre toda la base de datos variando el valor de α en el rango $[0, 1]$ con incrementos de 0.01.

Las principales conclusiones son:

- Obsérvese cómo, por ejemplo, la línea 3, columna *($e = 1$,max)* presenta el resultado de la fusión de todos los descriptores salvo ORB, utilizando una función lineal ($e = 1$) y la regla del máximo, $(+0.30\mathbf{E}^{-2}, \mathbf{H}_0)$. En este caso, el método de fusión obtiene un incremento de $+0.30\mathbf{E}^{-2}$ sobre la medida-F con respecto al mejor descriptor de los cuatro utilizado de forma individual, que en este caso resulta ser FREAK. Sin embargo, pese al incremento obtenido, el valor $\mathbf{H}_0$ indica que este resultado no es estadísticamente significativo siguiendo el test de Wilcoxon. Es decir, ambos métodos se pueden considerar iguales en rendimiento.
- La última línea (26) de la Tabla 3.2, resume los resultados de todas las pruebas realizadas para un mismo método. Así pues, para el caso *($e = 1$,max)*, el resultado $(-1.92\mathbf{E}^{-2}, \mathbf{H}_1)$ indica que, considerando todas las posibles combinaciones, el método de fusión, en media, provoca una reducción del rendimiento

SURF detector	medida-F	SIFT detector	medida-F
SIFT-L1	**0.721**	SIFT-L1	0.451
SIFT-L2	0.726	SIFT-L2	0.437
SURF-L1	**0.629**	SURF-L1	0.188
SURF-L2	0.559	SURF-L2	0.167
ORB	**0.570**	ORB	0.369
BRISK	**0.688**	BRISK	0.377
FREAK	**0.755**	FREAK	0.354

Tabla 3.1: *Rendimiento de los descriptores con los keypoints detectados por SURF y SIFT. En negrita se indican las configuraciones seleccionadas (descriptor y medida) para las comparaciones posteriores.*

	SIFT	SURF	ORB	BRISK	FREAK	($e=1$,max)	($e=1$, product)	($e=1$,rms)	($e=1$,sum)	($e=2$,max)	($e=2$,product)	($e=2$,rms)	($e=2$,sum)
0	*	*	*	*	*	$(-0.76\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-15.71\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+0.42\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+1.41\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.21\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-13.68\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-0.04\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+1.07\mathbf{E}^{-2}, \mathbf{H}_0)$
1	*	*	*	*		$(+0.20\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-8.50\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+1.45\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.72\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.83\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-7.93\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+0.16\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.19\mathbf{E}^{-2}, \mathbf{H}_0)$
2	*	*	*		*	$(-1.63\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-14.80\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-0.34\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.28\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-2.04\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-13.24\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.10\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.17\mathbf{E}^{-2}, \mathbf{H}_0)$
3	*	*		*	*	$(+0.30\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-9.18\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+0.94\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+1.86\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.70\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-8.23\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+0.45\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+1.60\mathbf{E}^{-2}, \mathbf{H}_1)$
4	*		*	*	*	$(-1.46\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-12.08\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.00\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.60\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.15\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-10.52\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.48\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.45\mathbf{E}^{-2}, \mathbf{H}_0)$
5		*	*	*	*	$(-1.84\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-11.90\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.01\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.94\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-2.05\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-11.19\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.15\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.61\mathbf{E}^{-2}, \mathbf{H}_0)$
6	*	*	*			$(-1.60\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-7.71\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-0.89\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.24\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-2.45\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-7.74\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.64\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.33\mathbf{E}^{-2}, \mathbf{H}_0)$
7	*	*		*		$(+1.01\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-3.14\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+1.12\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+1.28\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.15\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-3.02\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+0.50\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.58\mathbf{E}^{-2}, \mathbf{H}_0)$
8	*	*			*	$(-1.12\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-7.94\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-0.90\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.14\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.48\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-7.05\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.28\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.04\mathbf{E}^{-2}, \mathbf{H}_0)$
9	*		*	*		$(-1.00\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-5.82\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-0.43\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.41\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-2.11\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-6.00\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.35\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.53\mathbf{E}^{-2}, \mathbf{H}_1)$
10	*		*		*	$(-3.06\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-10.42\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.57\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.85\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-3.53\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-9.73\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.06\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.65\mathbf{E}^{-2}, \mathbf{H}_0)$
11	*			*	*	$(-0.70\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-5.96\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.13\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.65\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-2.00\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-5.67\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.55\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.05\mathbf{E}^{-2}, \mathbf{H}_0)$
12		*	*	*		$(-2.92\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-9.36\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.25\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.82\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.17\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-8.50\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.84\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.23\mathbf{E}^{-2}, \mathbf{H}_1)$
13		*	*		*	$(-3.43\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-10.43\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.65\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.44\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-3.87\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-10.14\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.90\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.09\mathbf{E}^{-2}, \mathbf{H}_1)$
14		*		*	*	$(-1.27\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-6.50\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.24\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.49\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-2.24\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-6.33\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.93\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.37\mathbf{E}^{-2}, \mathbf{H}_0)$
15			*	*	*	$(-3.04\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-9.04\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.34\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.97\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-3.57\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-8.70\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.62\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.37\mathbf{E}^{-2}, \mathbf{H}_1)$
16	*	*				$(-1.01\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-2.38\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-0.97\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.67\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.25\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-2.66\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.35\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.25\mathbf{E}^{-2}, \mathbf{H}_0)$
17	*		*			$(-4.03\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-5.96\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.68\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.49\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.84\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-6.56\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.12\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.84\mathbf{E}^{-2}, \mathbf{H}_1)$
18	*			*		$(-0.63\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.21\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.70\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.06\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.52\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.91\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.35\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.33\mathbf{E}^{-2}, \mathbf{H}_0)$
19	*				*	$(-0.90\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.86\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-4.09\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-0.90\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.99\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.88\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-4.05\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-0.99\mathbf{E}^{-2}, \mathbf{H}_0)$
20		*	*			$(-0.60\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-2.65\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.25\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.60\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.19\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-3.07\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.56\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.19\mathbf{E}^{-2}, \mathbf{H}_0)$
21		*		*		$(-2.42\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.47\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.67\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.42\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.37\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.41\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.72\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.37\mathbf{E}^{-2}, \mathbf{H}_1)$
22		*			*	$(-4.07\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-5.59\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.44\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.07\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.69\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-6.13\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.87\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.69\mathbf{E}^{-2}, \mathbf{H}_1)$
23			*	*		$(-4.96\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-7.21\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.17\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.33\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-5.25\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-7.42\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-5.41\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-5.25\mathbf{E}^{-2}, \mathbf{H}_1)$
24			*		*	$(-5.75\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-7.79\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.86\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.36\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-6.32\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-8.22\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-5.95\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-5.67\mathbf{E}^{-2}, \mathbf{H}_1)$
25				*	*	$(-3.27\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.37\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.97\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.27\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.84\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.78\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.51\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.84\mathbf{E}^{-2}, \mathbf{H}_1)$
26	-	-	-	-	-	$(-1.92\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-7.31\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.68\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.01\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.52\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-7.03\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.37\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.69\mathbf{E}^{-2}, \mathbf{H}_1)$

Tabla 3.2: *Tests de Wilcoxon comparando el rendimiento de ocho propuestas de fusión estándar que aparecen en [98].*

	SIFT	SURF	ORB	BRISK	FREAK	t-norm $s = 0$	t-norm $s = 0.5$	t-norm $s = 1$
0	*	*	*	*	*	$(+3.79\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-0.27\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+5.78\mathbf{E}^{-2}, \mathbf{H}_1)$
1	*	*	*	*		$(+5.70\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+0.13\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+5.27\mathbf{E}^{-2}, \mathbf{H}_1)$
2	*	*	*		*	$(+3.58\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+1.51\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+5.73\mathbf{E}^{-2}, \mathbf{H}_1)$
3	*	*		*	*	$(+5.10\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+2.14\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+7.17\mathbf{E}^{-2}, \mathbf{H}_1)$
4	*		*	*	*	$(+3.78\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+1.22\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+4.66\mathbf{E}^{-2}, \mathbf{H}_1)$
5		*	*	*	*	$(+3.11\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.62\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+3.83\mathbf{E}^{-2}, \mathbf{H}_1)$
6	*	*	*			$(+4.42\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+2.12\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+5.12\mathbf{E}^{-2}, \mathbf{H}_1)$
7	*	*		*		$(+7.16\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+3.28\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+7.11\mathbf{E}^{-2}, \mathbf{H}_1)$
8	*	*			*	$(+4.79\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+5.04\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+7.00\mathbf{E}^{-2}, \mathbf{H}_1)$
9	*		*	*		$(+5.15\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+1.38\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+5.08\mathbf{E}^{-2}, \mathbf{H}_1)$
10	*		*		*	$(+2.47\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+2.10\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+4.37\mathbf{E}^{-2}, \mathbf{H}_1)$
11	*			*	*	$(+4.88\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+4.03\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+6.10\mathbf{E}^{-2}, \mathbf{H}_1)$
12		*	*	*		$(+1.15\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.97\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+1.70\mathbf{E}^{-2}, \mathbf{H}_1)$
13		*	*		*	$(+2.41\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+0.25\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+3.47\mathbf{E}^{-2}, \mathbf{H}_1)$
14		*		*	*	$(+4.42\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-0.16\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+4.92\mathbf{E}^{-2}, \mathbf{H}_1)$
15			*	*	*	$(+2.67\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.95\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+2.67\mathbf{E}^{-2}, \mathbf{H}_1)$
16	*	*				$(+6.14\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+5.92\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+6.54\mathbf{E}^{-2}, \mathbf{H}_1)$
17	*		*			$(+2.00\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+2.66\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+2.61\mathbf{E}^{-2}, \mathbf{H}_1)$
18	*			*		$(+6.67\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+7.07\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+7.19\mathbf{E}^{-2}, \mathbf{H}_1)$
19	*				*	$(+2.75\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+3.64\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+3.70\mathbf{E}^{-2}, \mathbf{H}_1)$
20		*	*			$(+3.32\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+3.29\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+3.69\mathbf{E}^{-2}, \mathbf{H}_1)$
21		*		*		$(+2.18\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+1.25\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+2.51\mathbf{E}^{-2}, \mathbf{H}_1)$
22		*			*	$(+3.36\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+3.50\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+4.09\mathbf{E}^{-2}, \mathbf{H}_1)$
23			*	*		$(+0.29\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.89\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.67\mathbf{E}^{-2}, \mathbf{H}_0)$
24			*		*	$(+1.89\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+0.47\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+1.90\mathbf{E}^{-2}, \mathbf{H}_1)$
25				*	*	$(+3.64\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+1.99\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+3.25\mathbf{E}^{-2}, \mathbf{H}_1)$
26	-	-	-	-	-	$(+3.72\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+1.81\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+4.47\mathbf{E}^{-2}, \mathbf{H}_1)$

Tabla 3.3: *Tests de Wilcoxon comparando el desempeño de nuestra propuesta de fusión para los distintos esquemas de combinación.*

y que además existen diferencias estadísticamente significativas que lo afirman. Entonces, podemos afirmar que este método de fusión es peor que utilizar el mejor descriptor de forma individual.

- Si analizamos los resultados para el resto de métodos de fusión, puede observarse que ninguno obtiene mejores resultados que el mejor descriptor de forma individual.

Usando nuestra propuesta sobre método de fusión

Evaluamos el rendimiento del método propuesto basado en la TDS. Como en el caso anterior, nuestro método ha sido evaluado seleccionando todas las posibles combina-

ciones de los cinco descriptores. Nuestra propuesta contiene tres parámetros: n (el cardinal de $\phi_k(p, \mathcal{Q})$ en (3.4)), β de (3.14) y el valor de s para la *t-norm* utilizada en (2.42).

El primer parámetro se corresponde con el número de *keypoints* que se usan para crear la distribución de evidencia. Para nuestras pruebas se han utilizado los valores $n = (4, 3, 2)$. El segundo parámetro β es el factor que modula la forma de la distribución de evidencia. Se han probado los valores en el rango $\beta = [1, 15]$ usando un incremento de 0.25. Para modificar la *t-norm*, se han usado los valores $s = \{0, 0.5, 1\}$. Por último, el valor de α ha sido variado en el rango $[0, 1]$ con un incremento de 0.1 como en los casos anteriores. Los resultados obtenidos se muestran en la Tabla 3.3.

Las principales conclusiones son:

- En general, el método propuesto obtiene mejores resultados que los métodos probados anteriormente.
- Cuando se usan las *t-norms* $s = \{0, 1\}$ (normas cauta y conjuntiva del TBM), nuestro método obtiene siempre mejores resultados con respecto al mejor descriptor de manera individual.
- En general, la norma conjuntiva obtiene unos resultados ligeramente mejores que la cauta, probablemente esto sea debido a que la información tratada por los descriptores no pueda considerarse como información de alta correlación.
- Obsérvese la línea 3 y la columna $s = 1$, donde nuestro método fusiona todos los descriptores excepto ORB. En este caso, el incremento del rendimiento es de $(+7.17\mathbf{E}^{-2})$ con respecto a FREAK, que obtuvo un valor de medida-F=0.775. Esto supone, en términos relativos, una mejora cercana a un 10 % al usar nuestra propuesta. En general, los mejores resultados se obtienen al no usar el descriptor ORB que coincide con el descriptor que peores resultados ha obtenido de manera individual en la base de datos probada.

Influencias de los parámetros en el rendimiento del método propuesto

Los resultados presentados muestran el rendimiento del método que se propone para la mejor configuración de los parámetros n y β. Analizamos cómo varían estos resultados al variar los valores de los parámetros.

- La influencia del valor de n se analiza ejecutado un test de Wilcoxon para comparar los resultados con la mejor configuración de β para los valores de $n = (2, 3, 4)$. Los resultados se muestran en la Tabla. 3.4. Por filas se muestran las distintas combinaciones de los descriptores fusionados, mientras que las tres últimas columnas muestran el resultado de los test de Wilcoxon.

  Por ejemplo, la columna "2 vs 3" de la línea 1 compara el mejor resultado encontrado para $n = 2$ con el mejor resultado usando $n = 3$ cuando nuestro método fusiona todos los descriptores salvo FREAK. El test de Wilcoxon indica que aunque se obtenga un incremento en los resultados de 0.12 % en la medida-F

	SIFT	SURF	ORB	BRISK	FREAK	2 vs 3	2 vs 4	3 vs 4
0	*	*	*	*	*	$(-0.38\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.49\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.12\mathbf{E}^{-2}, \mathbf{H}_0)$
1	*	*	*	*		$(+0.12\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.13\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.01\mathbf{E}^{-2}, \mathbf{H}_0)$
2	*	*	*		*	$(-0.46\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.29\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.16\mathbf{E}^{-2}, \mathbf{H}_0)$
3	*	*		*	*	$(-0.44\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.12\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.32\mathbf{E}^{-2}, \mathbf{H}_0)$
4	*		*	*	*	$(-0.04\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.37\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.33\mathbf{E}^{-2}, \mathbf{H}_0)$
5		*	*	*	*	$(-0.08\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.19\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.26\mathbf{E}^{-2}, \mathbf{H}_0)$
6	*	*	*			$(+0.01\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.03\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.04\mathbf{E}^{-2}, \mathbf{H}_0)$
7	*	*		*		$(-0.61\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.60\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.01\mathbf{E}^{-2}, \mathbf{H}_0)$
8	*	*			*	$(+0.76\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.87\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.11\mathbf{E}^{-2}, \mathbf{H}_0)$
9	*		*	*		$(+0.16\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.04\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.21\mathbf{E}^{-2}, \mathbf{H}_0)$
10	*		*		*	$(+0.15\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.17\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.01\mathbf{E}^{-2}, \mathbf{H}_0)$
11	*			*	*	$(-0.07\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.77\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.70\mathbf{E}^{-2}, \mathbf{H}_0)$
12		*	*	*		$(+0.14\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.05\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.09\mathbf{E}^{-2}, \mathbf{H}_0)$
13		*	*		*	$(-0.20\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.16\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.03\mathbf{E}^{-2}, \mathbf{H}_0)$
14		*		*	*	$(-0.63\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.34\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.29\mathbf{E}^{-2}, \mathbf{H}_0)$
15			*	*	*	$(+0.00\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.10\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.10\mathbf{E}^{-2}, \mathbf{H}_0)$
16	*	*				$(-0.59\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.04\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.64\mathbf{E}^{-2}, \mathbf{H}_0)$
17	*		*			$(+0.32\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+1.06\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+0.75\mathbf{E}^{-2}, \mathbf{H}_0)$
18	*			*		$(+0.81\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.89\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.07\mathbf{E}^{-2}, \mathbf{H}_0)$
19	*				*	$(+0.39\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.64\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.25\mathbf{E}^{-2}, \mathbf{H}_0)$
20		*	*			$(-0.01\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.23\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.22\mathbf{E}^{-2}, \mathbf{H}_0)$
21		*		*		$(-0.44\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.79\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.34\mathbf{E}^{-2}, \mathbf{H}_0)$
22		*			*	$(-0.14\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.02\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.16\mathbf{E}^{-2}, \mathbf{H}_0)$
23			*	*		$(-0.17\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.46\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.28\mathbf{E}^{-2}, \mathbf{H}_0)$
24			*		*	$(+0.01\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.4\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.42\mathbf{E}^{-2}, \mathbf{H}_0)$
25				*	*	$(-0.20\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.21\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.02\mathbf{E}^{-2}, \mathbf{H}_0)$
26	-	-	-	-	-	$(-0.06\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.05\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.01\mathbf{E}^{-2}, \mathbf{H}_0)$

Tabla 3.4: *Tests de Wilcoxon comparando el desempeño de nuestra propuesta de fusión variando los valores del parámetro n. Los resultados indican que los cambios en el valor de n no producen diferencias estadísticamente significativas (la hipótesis nula H_0 no se rechaza).*

al comparar $n = 2$ vs $n = 3$, las diferencias observadas no son estadísticamente significativas. Si se analizan el resto de las pruebas, se observa que no hay un claro ganador. Por tanto, podemos llegar a la conclusión de que el impacto del parámetro n no es crítico en el rendimiento del método. Creemos que esto se justifica porque si la pareja correcta no se encuentra entre los dos primeros candidatos, entonces, el descriptor no es fiable para ese *keypoint* concreto. Es decir, es improbable que un descriptor clasifique correctamente una pareja si el candidato se encuentra en la posición tercera o cuarta.

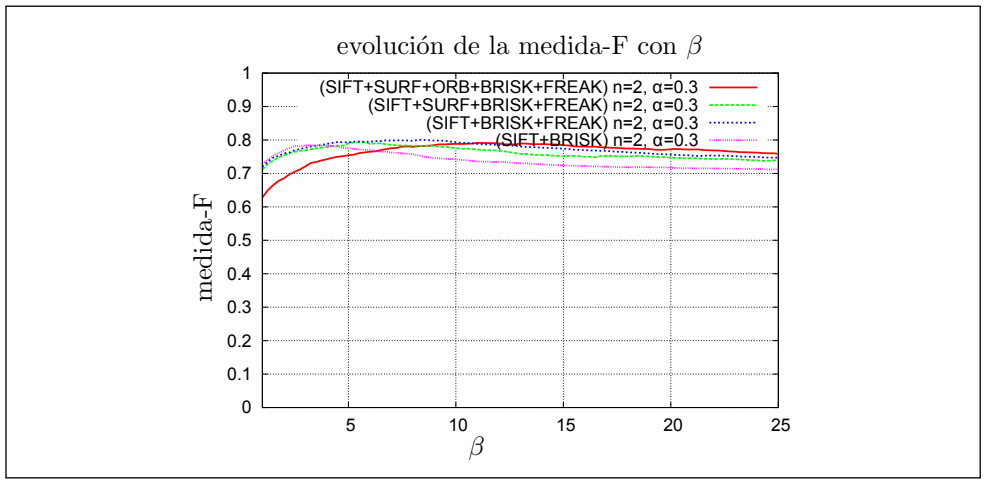

Figura 3.3: *Evolución de la medida-F ante la variación del parámetro β (3.13). Como se puede observar, el rendimiento de nuestro método es muy estable para un amplio rango de valores de β.*

- Para analizar el comportamiento de nuestro método al variar el valor de β mostramos la evolución de la medida-F para varias combinaciones de fusión y para un valor constante de n y α (ver Figura 3.3). Solo se han seleccionado un pequeño grupo de combinaciones debido a que para el resto de los casos el comportamiento es muy parecido. Como se puede observar, la variación en β se visualiza con una curva suave. Los valores de la medida-F se incrementan lentamente hasta que alcanzan un máximo y desde ahí la función empieza a decrecer suavemente.

La principal conclusión, a la luz de los resultados obtenidos, es que el método propuesto es robusto, es decir, los valores de los parámetros no son críticos o determinantes.

Capítulo 4

Estimando la pose de cámara mediante detección de marcadores artificiales en situaciones complejas

4.1. Introducción

En la introducción de este documento se comentó cómo el uso de los denominados sistemas de marcadores artificiales son una alternativa para estimar la pose de cámara. En la Sección 2.4 se hace una revisión del estado actual del arte con relación a los sistemas de marcadores artificiales, resaltando que los marcadores artificiales cuadrados son los más populares dado que un único marcador aporta cuatro correspondencias (sus cuatro esquinas), suficiente para realizar la estimación de la pose de la cámara.

Para estimar la pose de cámara mediante marcadores artificiales hay que cubrir dos pasos: (a) detectar este tipo de marcadores, lo habitual es detectar en la imagen la presencia de bordes cuadrados (esta operación produce un conjunto de candidatos entre los que estarán los marcadores y también elementos de la imagen con forma cuadrada pero que no son marcadores (ver Figura 1.3)) y (b) analizar para aceptar o rechazar cada candidato, dependiendo de si es realmente un marcador o simplemente era parte del escenario, y en caso de aceptación extraer el código binario en donde se encuentra la información del mismo.

Asumiendo que los marcadores han sido detectados correctamente, entre otros posibles objetos de la escena con similar forma, para extraer el código binario de los mismos existen dos propuestas en la literatura específica. La primera (ver Figura 4.1(a)) consiste en obtener la vista canónica del marcador y a continuación umbralizar [2, 104] (ver Sección 2.4 para conocer más detalles del proceso de detección e identificación). La segunda (ver Figura 4.1(b)) utiliza la homografía inversa para determinar la localización sobre la propia imagen donde realizar el análisis [3, 48].

Finalmente, obsérvese que, si el código binario extraído de un objeto que pensamos que se corresponde con un marcador no pertenece al conjunto de marcadores válidos

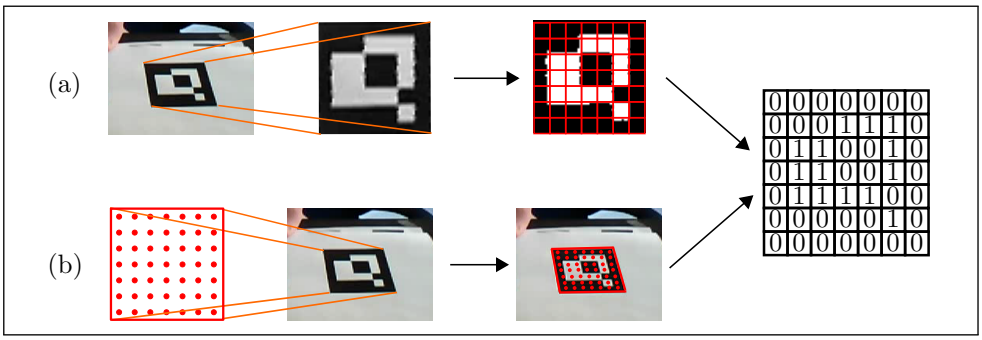

Figura 4.1: *Métodos de extracción del código de identificación de un marcador. (a) Primero, se obtiene la vista canónica del marcador. La imagen resultante es binarizada y dividida en celdas. Se contabilizan el número de píxeles blancos y negros en cada celda para determinar el valor binario de cada bit. (b) Se calcula la transformación de la perspectiva inversa del marcador y se utiliza para obtener la posición de los centros de las celdas sobre la imagen de entrada. Los valores de los píxeles de la imagen original determinan el valor de cada bit.*

del diccionario, el candidato puede ser rechazado. Este hecho implica que el procedimiento comentado también consigue separar los marcadores válidos de aquellos que realmente no lo son.

De acuerdo con lo anterior, es obvio que seleccionar un correcto código de marcadores binarios para una aplicación tiene una gran importancia para reducir la aparición de errores. Algunos autores han utilizado técnicas de codificación de señales [48, 104, 105], otros han usado aproximaciones heurísticas [2, 3] e incluso se ha utilizado Programación Entera Mixta (Mixed Integer Linear Programming (MILP)) [22] para obtener soluciones óptimas. No obstante, en muchas aplicaciones en la vida real, las imágenes sufren transformaciones en las que la identificación de los marcadores utilizando los métodos anteriores no son fiables. El desenfoque debido al movimiento, la iluminación no uniforme o la visibilidad reducida (ver Figura 4.2) son ejemplos de las circunstancias por las que los métodos citados pueden no dar buenos resultados. A efectos de paliar la problemática citada, nuestra propuesta para afrontar el problema de identificación de marcadores se ha orientado como la resolución de un problema de clasificación. Así en lugar de binarizar la imagen y aplicar técnicas de detección/corrección, nosotros proponemos entrenar un clasificador para el proceso de identificación. Además, nuestra propuesta incluye la creación de una base de datos sintética de entrenamiento que contenga marcadores bajo las diferentes transformaciones, ya que obtener una base de datos extensa de marcadores extraídos de imágenes reales en una amplia variedad de situaciones no es práctico. Adicionalmente, como desconocemos *a priori* el mejor método de aprendizaje automático para este problema particular, estudiamos el uso de MLP, CNN, y SVM y realizamos una serie de test estadísticos para determinar el mejor. Los clasificadores entrenados se han validado sobre un conjunto de escenas reales y se han comparado con algunos sistemas de marcadores del estado del arte actual, ArUco [2] y AprilTags [3], que garantizan las distancias más largas en el diccionario, lo que permite una mejor capacidad en la corrección de errores frente a alternativas como ArToolKitPlus [104] y ARTag [48].

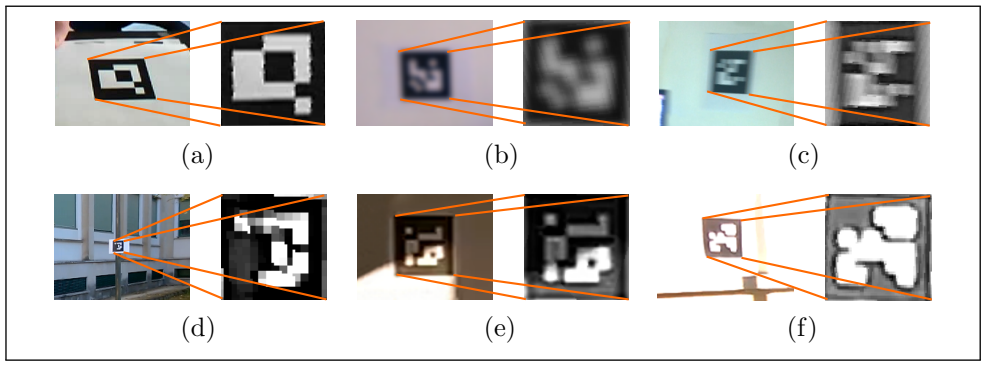

Figura 4.2: *Imágenes canónicas de marcadores bajo condiciones complejas. (a) Condición ideal. (b) Desenfoque. (c) Desenfoque de movimiento. (d) Escala pequeña. (e) Luz no uniforme. (f) Sobreexposición. La baja calidad de las imágenes en (b-f) provocan la extracción errónea de los códigos binarios, produciendo errores durante el proceso de identificación de marcadores.*

Hasta donde alcanza nuestro conocimiento, Gukhwan Kim y E.M. Petri [106] son los únicos que han intentado seguir una estrategia similar: abordar la identificación de marcadores como un problema de clasificación. Sin embargo, las diferencias entre la estrategia seguida y nuestra idea son relevantes. En primer lugar ellos asumen que todos los candidatos detectados son marcadores válidos, es decir dan por hecho que en la etapa de detección de bordes cuadrados no se incluyen elementos del fondo. Esto limita la propuesta ante escenarios reales como el que puede verse en la Figura 1.3. Además utilizan letras como identificadores, y entrenan una red neuronal para reconocerlos bajo diferentes transformaciones de la imagen. Sin embargo, en lugar de alimentar al clasificador con los datos en bruto, primero calculan doce características extraídas de las letras que se corresponden con los datos de entrada. Por el contrario, la estrategia que hemos seguido en este trabajo no realiza ninguna presunción sobre los patrones del marcador, que pueden ser códigos binarios, letras o imágenes, además hemos probado el funcionamiento con varios clasificadores para seleccionar la mejor alternativa.

En este capítulo se propone un nuevo método que, usando marcadores artificiales, permitiría estimar la pose de cámara. El mismo se fundamenta en el uso de técnicas de aprendizaje cuyos principios básicos pueden verse en la Sección 2.7. El capítulo incluye la evaluación del rendimiento de la propuesta que se hace, frente a propuestas anteriores, finalizando con las principales conclusiones.

4.2. Usando marcadores artificiales para estimar la pose de cámara: método propuesto

Abordamos en esta sección los detalles de la propuesta original para determinar la pose de cámara usando marcadores artificiales.

Sea $\mathbb{D}$ un diccionario compuesto por los n marcadores utilizados en una aplicación particular. Asumiendo que disponemos de un método que sea capaz de detectar los marcadores candidatos, es decir las regiones de la imagen con posibilidad de ser un marcador de la escena. Para cada candidato calculamos la homografía del cuadrado proyectado obteniendo su vista canónica como se muestra en la Figura 4.1(a). De esta manera, en lugar de utilizar un método que extraiga e identifique los bits, utilizaremos la imagen canónica como entrada de un clasificador que nos indique el identificador apropiado al marcador.

Téngase en cuenta que existen cuatro vistas canónicas posibles para cada marcador que se corresponden con las rotaciones de 0, 90, 180 y 270 grados. Para una correcta estimación de la pose de la cámara es necesario conocer la rotación del marcador para poder identificar cada esquina del mismo correctamente y, por tanto, para cada candidato se debe determinar si pertenece a uno de los marcadores de $\mathbb{D}$ y a la vez determinar la rotación de la vista canónica.

Formulado como un problema de clasificación, nuestro conjunto de entrada consiste en el nivel de gris de los píxeles que forman cada imagen canónica, y se definen un total de $4 \times n + 1$ clases: una por cada posible rotación del marcador, y una clase adicional que represente el caso negativo, es decir cuando la entrada no se corresponde con ningún marcador válido sino con un elemento del fondo de la escena.

Propuesta para generación de un conjunto de datos de entrenamiento

Se requiere un conjunto de datos etiquetados que incluya muestras de todas las clases consideradas con objetivo de aplicar las técnicas de clasificación.

En nuestro problema, este conjunto de datos consiste en una colección de imágenes canónicas que incluyan las cuatro vistas de todos los marcadores del diccionario bajo la aparición de diferentes situaciones adversas, y también imágenes que no sean marcadores para la clase negativa. Para incluir en el conjunto de datos elementos que no sean marcadores reales, se han considerado imágenes del conjunto de datos de imágenes MIRFLICKR-25000[107], extrayendo de cada una de ellas una o varias ventanas.

Dado que obtener un extenso conjunto de marcadores que representen una amplia variedad de condiciones a partir de imágenes reales no es práctico, y con el fin de limitar el conjunto de marcadores, se propone la creación de un conjunto de datos sintéticos generando distintas transformaciones sobre los marcadores que se vayan a usar. Dichas transformaciones tienen como objetivo emular las situaciones típicas que se producen en la realidad, incluyendo condiciones complejas como las que se muestran en la Figura 4.2.

Se han considerado cinco transformaciones. Dada una imagen de entrada, primero seleccionamos de manera aleatoria qué transformaciones se van a aplicar y en qué orden, ya que no son conmutativas. La probabilidad de seleccionar cada transformación varía conforme la posibilidad de que ocurra le situación emulada en la realidad,

como explicaremos posteriormente. A continuación describimos las transformaciones utilizadas.

Siendo I una imagen original de un marcador de dimensiones $s \times s$, $I(p)$ el valor de intensidad del píxel p en un rango entre 0 y 255 e I' la imagen sintética que será generada, contemplamos las siguientes situaciones:

- Desenfoque: con el objetivo de simular el desenfoque del marcador con respecto la cámara y el desenfoque de movimiento. Se utiliza un kernel de desenfoque con ancho y alto seleccionado de manera aleatoria siguiendo una distribución bidimensional uniforme $\mathbf{U_2}((1, \eta_1 s)^2)$.
- Compresión de rango dinámico: debido a que las imágenes originales de los marcadores son binarias, únicamente presentan dos valores 0 ó 255. Dado que este hecho no se corresponde con lo que sucede en imágenes reales, realizamos una transformación aleatoria de este rango aplicando la siguiente transformación sobre la intensidad de los píxeles:

  $$I'(p) = min(255, aI(p) + b), \tag{4.1}$$

  donde a y b se obtienen de las siguientes distribuciones de probabilidad unidimensionales uniformes $a \backsim \mathbf{U}_1(\eta_2, 1)$ y $b \backsim \mathbf{U}_1(0, \eta_3)$. La función min asegura que los valores resultantes no sean superiores a 255.
- Transformación afín: se aplica una transformación proyectiva que simule el error cometido durante la estimación de las esquinas de los candidatos. La transformación traslada las esquinas de I aleatoriamente añadiendo un desplazamiento que sigue una distribución uniforme $\mathbf{U_2}((-\eta_4 s, \eta_4 s)^2)$.
- Luz no uniforme: se añade un degradado que se modela como una distribución bidimensional Gaussiana $\mathbf{N}_2(c, \Sigma)$ siendo c las coordenadas del punto origen de la fuente de luz que se encuentra fuera de la imagen. La transformación se define como:

  $$I'(p) = min\left(255, I(p) + 50 \cdot e^{-\frac{1}{2}(p-c)^t \Sigma^{-1}(p-c)}\right). \tag{4.2}$$

  Se asume que la distribución es simétrica, es decir que $\Sigma = \sigma^2 \mathbf{I}$, donde $\mathbf{I}$ es la matriz de identidad, y que $\sigma \backsim \mathbf{U}_1(s/4, s/2)$ y $c \backsim \mathbf{U_2}(\Omega^2)$, donde:

  $$\Omega = [-s, 0] \cup [s, 2s],$$

- Dilatación: simula un efecto derivado de la sobreexposición que tiene como consecuencia incrementar los bordes blancos internos de los marcadores (ver Figura 4.2(f)). El tamaño del kernel utilizado para la dilatación se selecciona de manera aleatoria siguiendo la distribución $\mathbf{U_2}((1, \eta_5 s)^2)$.

En nuestro modelo hemos fijado para cada transformación una probabilidad de ocurrencia. El suceso correspondiente a aplicar una determinada transformación es independiente del suceso similar correspondiente a otra. Además existe aleatoriedad en el orden secuencial seguido para aplicar o no cada transformación. Para cada una de las

Figura 4.3: *Transformaciones aleatorias generadas para un marcador. El marcador original (arriba-izquierda), el resto son las transformaciones sintéticas generadas.*

tres primeras transformaciones citadas asumimos una probabilidad de 0.75 para ser aplicada, mientras que para la cuarta y quinta transformación se fijan en 0.25 y 0.15 respectivamente. La menor probabilidad asignada a las dos últimas se debe, a que cuando son aplicadas, producen efectos de transformación más intensos. Además, se corresponden con casos que no se observan con tanta frecuencia en las escenas reales como el resto de las transformaciones.

Los valores de los parámetros utilizados en nuestro modelo para las distintas distribuciones han sido determinados de manera empírica tras la realización de una serie de pruebas preliminares. Los citados valores son: $\eta_1 = 0.2$, $\eta_2 = 0.4$. $\eta_3 = 25$, $\eta_4 = 0.025$ y $\eta_5 = 0.08$.

La Figura 4.3 muestra una imagen de marcador original y algunas transformaciones sintéticas generadas automáticamente de acuerdo con el modelo propuesto.

4.3. Experimentación, resultados y discusión

En esta sección se describe la experimentación llevada a cabo para validar nuestra propuesta. Se han realizado diferentes análisis estadísticos con objetivo de hacer una rigurosa evaluación de los resultados. En primer lugar, se explica cómo se han elaborado los conjuntos de datos de entrenamiento, validación y test. A continuación, se describe la metodología utilizada para comparar los distintos métodos. Después, se evalúan las diferentes configuraciones de SVM, MLP y CNN. Por último, se comparan las mejores opciones de cada método junto a dos sistemas de marcadores del estado del arte ArUco[2] y AprilTags[3]. Adicionalmente, hemos analizado los tiempos de computación de cada método para determinar su viabilidad en aplicaciones de tiempo real.

4.3.1. Creación de los conjuntos de datos

Para realizar nuestras pruebas, se ha generado un diccionario $\mathbb{D}$ compuesto por cincuenta marcadores de tamaño 6×6 bits usando para ello el método propuesto en [22], el cual garantiza la mejor capacidad de corrección para las propuestas de ArUco y AprilTags. Hemos considerado cincuenta marcadores un número suficientemente grande para cubrir las necesidades de diversas aplicaciones reales. En total, hay $4 \times 50 + 1 = 201$ clases diferentes.

Se han obtenido tres conjuntos de datos para entrenamiento, validación y test, respectivamente. Los dos primeros han sido generados sintéticamente siguiendo los pasos indicados en la Sección 4.2. El primero se utiliza durante el entrenamiento para la obtención de los parámetros óptimos de los métodos de clasificación, mientras que el segundo tiene la finalidad de evitar el sobreentrenamiento.

Los conjuntos de entrenamiento y validación están formados por 250 imágenes sintéticas por cada clase, de las que 200 son para entrenamiento y el resto para validación. Adicionalmente se han incluido unas 210 000 imágenes para la clase negativa, que han sido extraídas de la colección de imágenes MIRFLICKR-25000 [107]. De ellas 200 000 son utilizadas para entrenamiento y el resto para validación.

El conjunto de datos para test consta de siete escenas grabadas tanto en escenarios al aire libre como de interior e incluyen "diversas situaciones desafiantes para aplicaciones de la vida real". La primera escena (Fig. 4.4(a)) contiene desenfoque de movimiento debido al rápido desplazamiento de la cámara. En la segunda (Fig. 4.4(b)), se ha provocado que los marcadores se vean desenfocados por la cámara. La tercera (Fig. 4.4(c)) se centra en la detección de marcadores a una distancia lejana o lo que es lo mismo, identificar marcadores con escala pequeña. En la cuarta secuencia (Fig. 4.4(d)), algunos marcadores están iluminados por una luz no uniforme. En la quinta (Fig. 4.4(e)), el escenario presenta una elevada iluminación, lo que provoca una frecuente sobreexposición de la cámara. La sexta secuencia (Fig. 4.4(f)) muestra un escenario grande al aire libre con un número elevado de marcadores. De forma similar se procede en la séptima secuencia (Fig. 4.4(g)), pero en un escenario de interior. Los candidatos de las secuencias de test se han extraído usando el método propuesto en [2]. El tamaño original de los candidatos es de 40×40 píxeles, y han sido etiquetados manualmente. En total, hay 3924 fotogramas de los que se han extraído 102 829 candidatos y 84 030 pertenecen al fondo.

Téngase en cuenta que el entrenamiento se realiza exclusivamente con imágenes sintéticas, mientras que el test está formado por imágenes reales.

4.3.2. Metodología de comparación

Para obtener una comparación formal de los métodos de clasificación considerados, se han utilizado pruebas no paramétricas [108]. Primero, se aplica la prueba estadística de Friedman [109] como test "omnibus", para averiguar si los resultados de los distintos métodos presentan diferencias estadísticamente significativas o no. En el caso de encontrar diferencias estadísticamente significativas (se rechaza la hipótesis nula)

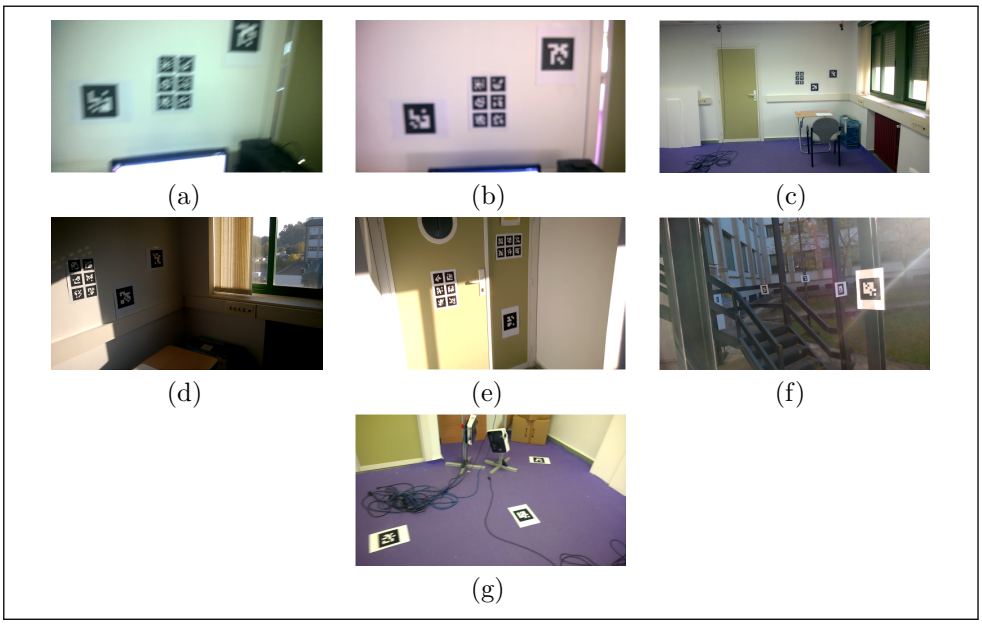

Figura 4.4: *Escenas de evaluación: (a) desenfoque de movimiento, (b) desenfoque, (c) escala, (d) luz no uniforme, (e) sobreexposición, (f) escenario de exterior, (g) escenario de interior.*

se utiliza la prueba "post-hoc" de Finner [110] para comparar al mejor método clasificado (método de control) frente a los demás. Esta última comparación se realiza por pares, y para cada pareja se obtiene el valor p_{Finner} ajustado del test de Finner. Se ha fijado el nivel de significación en 0.05.

De nuevo hemos utilizado la medida-F para analizar el rendimiento de los métodos probados.

$$F = 2\frac{precision \cdot exhaustividad}{precision + exhaustividad}, \tag{4.3}$$

$$precision = \frac{tp}{tp + fp}; \quad exhaustividad = \frac{tp}{tp + fn}, \tag{4.4}$$

En este caso tp hace referencia a los marcadores candidatos identificados correctamente, fp se corresponde con los errores, ya sean producidos por la identificación de un elemento de fondo como un marcador o por una confusión en el identificador de un marcador, y fn indica aquellos marcadores que pertenecen a nuestro diccionario pero han sido asignados a la clase negativa.

Determinando la configuración del clasificador SVM

Para determinar la mejor configuración hemos usado los datos del conjunto de test con el tamaño original de las imágenes (40×40 píxeles) y también con versiones reducidas de los mismos (20×20 y 10×10 píxeles). Se han probado también los *kernels* lineales

y RBF. Adicionalmente, se ha usado la técnica Principal Component Analysis (PCA) [111] para reducir la dimensión de los datos manteniendo las componentes principales en 25, 50 o 100 cuando sea posible.

Se ha entrenado un clasificador SVM para cada combinación de parámetros.

Configuración	Ranking	p_{Finner}
10-RBF	4.28	-
20-LIN	5.07	$8.20E^{-1}$
20-LIN-100	5.50	$7.43E^{-1}$
10-LIN-50	6.21	$6.15E^{-1}$
20-RBF	7.64	$3.77E^{-1}$
20-RBF-100	8.07	$3.28E^{-1}$
10-LIN	8.28	$3.20E^{-1}$
40-LIN	8.35	$3.20E^{-1}$
10-RBF-50	9.78	$1.64E^{-1}$
40-LIN-100	11.42	$6.31E^{-2}$
20-LIN-50	11.85	$5.04E^{-2}$
40-RBF	12.00	$\mathbf{4.95}E^{-2}$
40-LIN-50	13.00	$\mathbf{2.51}E^{-2}$
20-RBF-50	13.92	$\mathbf{1.27}E^{-2}$
20-LIN-25	14.35	$\mathbf{9.71}E^{-3}$

Tabla 4.1: *Clasificación media de las configuraciones SVM realizada por el test de Friedman. En negrita, las configuraciones rechazadas por el procedimiento de Finner. La tabla es obtenida para un nivel de significación de 0.05.*

La Tabla 4.1 resume los quince mejores clasificadores ordenados por la clasificación de Friedman. La convención de nomenclatura utilizada en la primera columna es $XX - KER(-PCA)$, donde XX indica el tamaño de la imagen, KER hace referencia al *kernel* del clasificador (Lineal o RBF) y, opcionalmente, PCA indica el numero de componentes principales usadas. Por ejemplo, la primera línea (10-RBF) se corresponde con un clasificador SVM que usa como entrada imágenes de tamaño 10×10 píxeles, un *kernel* RBF y no hace uso de PCA. La segunda columna indica el valor de clasificación de cada combinación, un indicador de cómo de bien se comporta el mismo SVM en comparación al resto. El resultado debe interpretarse de forma que un valor igual a 1 indicaría que la configuración ha obtenido el mejor valor de medida-F en las siete secuencias que forman el conjunto de test. Por último, la tercera columna muestra el valor de p ajustado utilizado durante el test de Finner. Aquellas configuraciones con un valor $p \leq 0.05$ son rechazadas por el procedimiento de Finner, aparecen resaltadas en negrita en la tabla para facilitar su localización. Estas configuraciones se consideran estadísticamente diferentes al método de control, por tanto podemos afirmar que dichas configuraciones son peores respecto a la mejor. Para el caso de las configuraciones que no aparecen en negrita, cuyo $p > 0.05$, no existe una diferencia estadísticamente demostrada.

A la luz de estos resultados, podemos afirmar que:

- Las configuraciones que utilizan PCA reduciendo a solo veinticinco componentes obtienen resultados significativamente peores.
- La elección del *kernel* no parece afectar en especial a los resultados.
- Las imágenes con tamaño de entrada 10×10 píxeles han demostrado ser de un tamaño suficiente para obtener buenos resultados en la clasificación.

En la Sección 4.3.3 se explica con mayor nivel de detalle el rendimiento de la mejor configuración (10-RBF) y en la Tabla 4.4 se compara el mismo junto con las mejores configuraciones del resto de métodos de clasificación probados. Por último indicar que la mejor configuración de SVM encontrada está formada por 8546 vectores de soporte.

Determinando la configuración del clasificador MLP

Para el caso del MLP, se ha probado con las imágenes de tamaño de entrada 10×10, 20×20 y 40×40 píxeles y las funciones de activación denominadas Identidad y ReLU.

El número de capas ocultas varía entre cero y dos, y siempre están formadas por 201 neuronas. Por último, en todos los casos, la capa de salida consiste en 201 neuronas (número de clases de nuestro problema) conectadas a una función *softmax*. El número máximo de épocas de entrenamiento se ha fijado en cincuenta.

La clasificación de Friedman así como los valores de los tests de Finner se muestran en la Tabla 4.2 para los quince mejores clasificadores. La convención de nomenclatura para este método es $XX - AF(-H*)$, donde XX indica el tamaño de las imágenes, AF hace referencia a la función de activación (Identidad o ReLU), y, por último, el número de repeticiones de H indica el número de capas ocultas que contiene la red.

De los resultados obtenidos podemos afirmar que:

- El uso de la función de activación ReLU produce peores resultados.
- En general, las imágenes de mayor tamaño se comportan mejor.

La comparación de la mejor MLP configuración (40-Identity-H) con el resto de clasificadores determinados aparece más adelante en la Tabla 4.4.

Determinando la configuración del clasificador CNN

En este caso se ha probado con el tamaño de entrada de las imágenes originales, 40×40 píxeles, y con la versión reducida de 20×20 píxeles. El uso de imágenes con un tamaño menor no era posible debido al tamaño de los filtros convolucionales empleados.

Como funciones de activación se ha probado con la función Identidad y ReLU. En todos los casos, el tamaño de los filtros convolucionales utilizados ha sido de 5×5. Hemos probado configuraciones usando uno o dos etapas de convolución y variando

Configuración	Ranking	p_{Finner}
40-Identity-H	1.57	-
20-Identity-H	2.85	$5.91E^{-1}$
40-Identity	3.00	$5.76E^{-1}$
20-Identity	3.42	$4.88E^{-1}$
10-Identity-H	5.14	$1.68E^{-1}$
40-Identity-HH	5.57	$1.29E^{-1}$
20-Identity-HH	6.99	$\mathbf{3.57}E^{-2}$
10-Identity	7.57	$\mathbf{2.13}E^{-2}$
10-Identity-HH	8.85	$\mathbf{4.61}E^{-3}$
20-ReLU	10.28	$\mathbf{6.22}E^{-4}$
40-ReLU-H	11.42	$\mathbf{1.04}E^{-4}$
10-ReLU	12.28	$\mathbf{2.58}E^{-5}$
40-ReLU	12.57	$\mathbf{1.95}E^{-5}$
10-ReLU-H	14.14	$\mathbf{1.46}E^{-6}$
20-ReLU-H	14.28	$\mathbf{1.46}E^{-6}$

Tabla 4.2: *Clasificación media de las configuraciones MLP realizada por el test de Friedman. En negrita, las configuraciones rechazadas por el procedimiento de Finner. La tabla es obtenida para un nivel de significación de 0.05.*

el número de filtros: $\{10, 25, 50\}$. Se ha insertado una capa de max-pooling con *kernel* de tamaño 2×2 después de cada etapa de convolución. Después, opcionalmente, se ha insertado una capa oculta de 201 neuronas antes de la red totalmente conectada de 201 neuronas que está conectada a la capa *softmax* de salida. Como en el caso anterior de la MLP, todas las configuraciones se entrenaron con un número máximo de cincuenta épocas.

La Tabla 4.3 muestra las quince mejores configuraciones de CNN de acuerdo a los tests de Friedman. La convención de nomenclatura usada para CNN es $XX - AF - YY(-ZZ)(-H?)$, donde XX se refiere al tamaño de entrada de las imágenes, AF indica la función de activación utilizada (Identidad o ReLU), YY indica el número de filtros utilizados para la primera capa de convolución, ZZ indica el número de filtros de la segunda capa de convolución (si se usa), y, por último, la adición de capas ocultas en la red se indica con H.

De los resultados de la Tabla 4.3 se puede deducir que:

- El uso de la función ReLU produce peores resultados.
- La adición de una segunda capa de convolución produce peores resultados.
- Cuando el tamaño de la imagen de entrada es igual a 20×20 la inclusión de una capa oculta totalmente conectada empeora los resultados. Para el caso de imágenes de dimensión 40×40 ocurre lo contrario.

En la Tabla 4.4 se puede observar una comparativa de la mejor configuración (40-Identity-10-H) con las mejores configuraciones de los otros clasificadores seleccionados.

Configuración	Ranking	p_{Finner}
40-Identity-10-H	3.28	-
40-Identity-25-H	3.42	$9.52E^{-1}$
40-Identity-50-H	3.85	$8.33E^{-1}$
20-Identity-25	5.28	$4.51E^{-1}$
20-Identity-50	5.42	$4.44E^{-1}$
40-Identity-50	7.35	$1.21E^{-1}$
20-Identity-10	7.50	$1.18E^{-1}$
40-Identity-25	7.78	$1.02E^{-1}$
40-Identity-50-25-H	9.00	$\mathbf{3.37}E^{-2}$
40-Identity-25-25-H	9.57	$\mathbf{1.98}E^{-2}$
20-Identity-10-H	10.00	$\mathbf{1.38}E^{-2}$
40-Identity-10	10.21	$\mathbf{1.31}E^{-2}$
20-Identity-25-H	10.57	$\mathbf{1.07}E^{-2}$
20-Identity-50-H	11.71	$\mathbf{2.95}E^{-3}$
40-ReLU-10-H	14.99	$\mathbf{1.33}E^{-5}$

Tabla 4.3: *Clasificación media de las configuraciones CNN realizada por la test de Friedman. En negrita, las configuraciones rechazadas por el procedimiento de Finner. La tabla es obtenida para un nivel de significación de 0.05.*

4.3.3. Comparando las mejores configuraciones de SVM, MLP y CNN con propuestas existentes

Se evalúan las mejores configuraciones de los clasificadores comentados previamente con los sistemas de marcadores artificiales AuUco [2] y ApriTags [3].

Los valores obtenidos para la medida-F de cada método, en las siete escenas, se muestran en la Tabla 4.4.

	SVM	MLP	CNN	ArUco	AprilTags
desenfoque de movimiento	**0.999**	0.996	0.996	0.894	0.890
desenfoque	0.942	0.910	**0.961**	0.306	0.149
sobreexposición	0.988	0.994	**0.996**	0.946	0.807
luz no uniforme	0.922	**0.993**	0.988	0.900	0.914
escala	**0.871**	0.825	0.741	0.492	0.029
interior	**0.997**	0.990	0.990	0.816	0.774
exterior	**0.952**	0.933	0.936	0.856	0.691

Tabla 4.4: *Medida-F para cada método sobre el conjunto de datos de test. En negrita se remarca el método que mejores resultados obtiene en cada secuencia.*

La principal conclusión es que los resultados de los tres clasificadores entrenados son mejores que los obtenidos por ArUco y AprilTags. Además, SVM obtiene la mejor puntuación en cuatro de las siete escenas.

Adicionalmente, mostramos en la Tabla 4.5 los valores medios para la precisión, exhaustividad y medida-F sobre las siete secuencias. Se observa que ArUco y AprilTags

	SVM	MLP	CNN	ArUco	AprilTags
precisión	0.986	0.951	0.972	0.996	**0.997**
exhaustividad	0.924	**0.949**	0.922	0.639	0.511
Medida-F	**0.953**	0.948	0.944	0.744	0.608

Tabla 4.5: *Precisión, exhaustividad y medida-F media obtenida por los distintos métodos sobre las escenas de test.*

presentan una alta restricción en el proceso de identificación de marcadores, conduciéndolos a lograr un alto valor en precisión a costa de un bajo valor de exhaustividad. Esto es especialmente destacable en el caso de AprilTags, que solo puede corregir hasta tres bits erróneos. Sin embargo, cuando analizamos la medida-F, vemos que los tres métodos de clasificación entrenados se comportan mejor que ArUco y AprilTags.

Una comprensión más exhaustiva de estos resultados es posible realizando un análisis estadístico. En primer lugar se ha realizado el test de Friedman para clasificar los métodos y, a continuación, se ha usado el test de rango con signo de Wilcoxon [103] para determinar si existen diferencias significativas entre cualquier par de métodos.

	Ranking	$p_{Wilcoxon}$			
		SVM	CNN	MLP	ArUco
SVM	1.71	-	-	-	-
CNN	1.99	1.0	-	-	-
MLP	2.28	$3.98E^{-1}$	$8.65E^{-1}$	-	-
ArUco	4.14	$\mathbf{1.79}E^{-2}$	$\mathbf{1.79}E^{-2}$	$\mathbf{1.79}E^{-2}$	-
AprilTags	4.85	$\mathbf{1.79}E^{-2}$	$\mathbf{1.79}E^{-2}$	$\mathbf{1.79}E^{-2}$	$\mathbf{4.25}E^{-2}$

Tabla 4.6: *Clasificación media de todos los métodos comparados con el test de Friedman. En negrita, se destacan los métodos que son significativamente diferentes respecto al test de Wilcoxon. La tabla es obtenida para un nivel de significación de 0.05.*

La Tabla 4.6 muestra la clasificación de Friedman y el valor $p_{Wilcoxon}$ de todas las comparaciones por pares. Los métodos que han resultado ser significativamente peores que algún otro método, son resaltados en negrita en su columna correspondiente.

Analizando los resultados de la tabla, se puede afirmar que:

- Los tres métodos de aprendizaje automático han obtenido mejores resultados que ArUco y AprilTags que son estadística significativos.
- SVM ha sido clasificado por las pruebas estadísticas como el mejor método, aunque no se puede afirmar que existan diferencias significativas entre este y MLP o CNN.

La Figura 4.5 intenta ilustrar el impacto de usar nuestra propuesta en una aplicación real. Usando la escena *escala* en función al tamaño del marcador en píxeles de la imagen, las gráficas muestran el valor de la exhaustividad obtenida por los distintos métodos para esa escena. Es fácil apreciar que los resultados de los tres métodos de aprendizaje superan claramente a los obtenidos por ArUco y AprilTags, especialmente

en los tamaños más pequeños. En este prueba concreta, el método MLP obtiene un valor de exhaustividad incluso mejor que el obtenido por SVM y CNN, a pesar de que analizando el valor de la medida-F como se muestra en la anterior Tabla 4.4, el MLP no se correspondía con el mejor método.

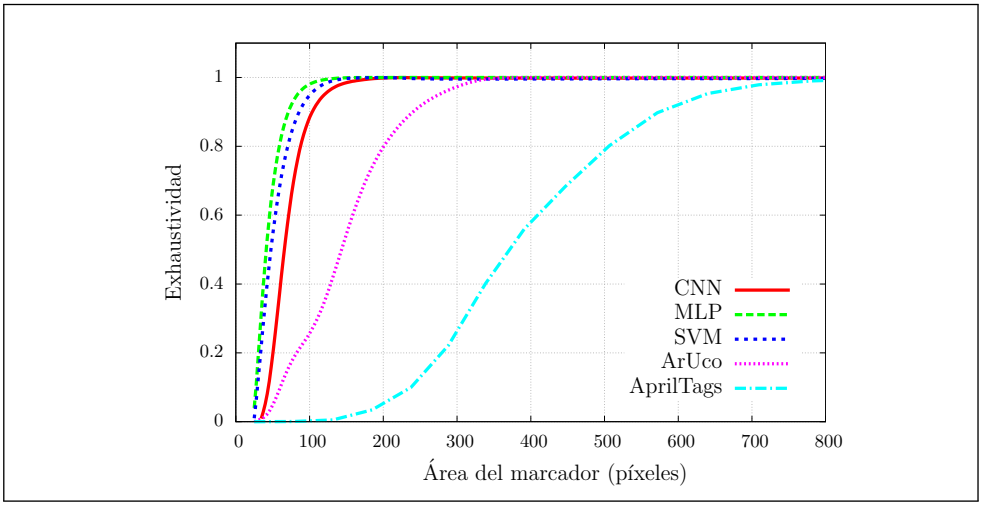

Figura 4.5: *Evolución de la exhaustividad como función del área del marcador en la escena* escala.

Finalmente, la Tabla 4.7 muestra el tiempo de computación requerido para procesar un único candidato por cada método. Esta prueba ha sido realizada en una máquina con un procesador Intel i7-3930K usando un único hilo de ejecución. En aquellos casos en las que las implementaciones de los métodos nos han permitido hacer uso de la GPU se ha usado el modelo Nvidia GeForce GTX 770. Los resultados muestran que ArUco es más rápido que los otros métodos en órdenes de magnitud. Sin embargo, los tiempos de computación logrados para las propuestas son totalmente válidos para aplicaciones en tiempo real.

Method	SVM	CNN	MLP	ArUco	AprilTags
CPU	1.47	3.95	$6.07E^{-1}$	$9.56E^{-2}$	4.52
GPU		$7.99E^{-1}$	$5.27E^{-1}$		

Tabla 4.7: *Tiempo de computación por cada método para clasificar una imagen (ms).*

Capítulo 5

Conclusiones

En esta Tesis Doctoral se ha abordado el problema de la *estimación de la pose de cámara* usando imágenes.

Se han estudiado en profundidad los mecanismos propuestos en la literatura sobre la base de dos alternativas para resolver el problema: usando marcadores naturales, mediante la detección de correspondencias de *keypoints* usando descriptores, y usando sistemas de marcadores artificiales.

Se realizan dos aportaciones originales. Una para cada una de las alternativas indicadas.

Con respecto a la primera aportación:

- Nuestra propuesta mejora el proceso de emparejamiento de correspondencias, usando fusión de información de varios descriptores mediante la Teoría de Dempster-Shafer [55]. En nuestra propuesta, cada descriptor se utiliza para generar una lista ordenada de candidatos que será transformada en una distribución de evidencia, de la que se calcula un factor de confianza mediante su entropía. Por tanto, la TDS se utiliza para fusionar las distribuciones de evidencia usando el factor de confianza como peso de ponderación en la contribución de cada descriptor.
- Para evaluar la propuesta hemos utilizando todas las combinaciones posibles de fusión con los descriptores SIFT [16], SURF [17], ORB [33], BRISK [34] y FREAK [35]. Todos los experimentos realizados incluyen análisis estadísticos que justifican la mejora real de la propuesta. Además nuestro método se ha comparado favorablemente con otras propuestas estándar de fusión presentadas en [98].
- Dada una serie de descriptores nuestro método es capaz de mejorar el rendimiento, en la búsqueda de correspondencias, obtenido por cada descriptor de forma individual empleando la fusión de información de los mismos. Los experimentos realizados en la base de datos Oxford [18] muestran que el método propuesto mejora los resultados del mejor descriptor involucrado en la fusión, hasta un 10 % en el mejor caso.

- El método propuesto ha sido publicado en la revista International Journal of Approximate Reasoning [112]

Con respecto a la segunda aportación:

- Nuestra propuesta aborda el problema de identificación de marcadores artificiales como un problema de clasificación. En lugar de umbralizar la imagen y luego aplicar técnicas de detección/corrección de bits, como hacen la mayoría de propuestas existentes, proponemos entrenar clasificadores para el proceso de identificación.

- El aspecto básico de la propuesta es usar un conjunto de datos de entrenamiento sintético, amplio, que refleje las situaciones complejas reales en las que se observarán los marcadores, entre las que destacamos el desenfoque producido por el movimiento o simplemente por el incorrecto enfoque de la cámara, la sobreexposición lumínica, la iluminación no uniforme y la visibilidad reducida.

- El conjunto de datos de entrenamiento se ha construido a partir de la generación automática de imágenes sobre las que se han realizado una serie de transformaciones que simulen las situaciones complejas frecuentes en la práctica. Entre las transformaciones se incluyen un desenfoque, compresión del rango dinámico, transformación afín, luz no uniforme y dilatación. Nuestro modelo asigna una probabilidad de ocurrencia a cada una de las transformaciones indicadas.

- Se ha analizado el rendimiento de tres métodos de aprendizaje, SVM [78] , MLP [113] y CNN [114], realizando un estudio estadístico comparativo entre ellos así como respecto de las técnicas más recientes de identificación de marcadores.

- Los resultados obtenidos muestran que nuestra propuesta mejora el proceso de identificación comparado con los sistemas de marcadores ArUco[2], AprilTags[3], habitualmente empleados.

- La propuesta que se realiza se encuentra actualmente en proceso de revisión, bajo el título “Robust detection of fiducial markers in challenging conditions”.

Trabajos futuros que pueden derivarse del desarrollo de la Tesis Doctoral

- La idea de fusión de información mediante la teoría de Dempster-Shafer, utilizando la entropía como medida de confianza, puede ser utilizada no solo en el ámbito de la búsqueda de correspondencias entre *keypoints*, sino también de forma más general en otros problemas de clasificación como el reconocimiento de gestos, identificación de personas o estimación de la pose. Por tanto, uno de los posibles trabajos futuros es determinar la validez de esta técnica en otros ámbitos.

- El sistema propuesto para detección de marcadores tiene aún margen de mejora respecto del tiempo de cómputo requerido. En la actualidad los sistemas altamente paralelizables están teniendo un gran auge debido al desarrollo de tarjetas gráficas (GPU) de alto rendimiento. Poder realizar todo el proceso de detección e identificación de marcadores en GPU tendría un considerable impacto en las aplicaciones que hacen uso de ellos.

Bibliografía

[1] Ronald T Azuma. "A survey of augmented reality". In: *Presence: Teleoperators and virtual environments* 6.4 (1997), pp. 355–385.

[2] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez. "Automatic Generation and Detection of Highly Reliable Fiducial Markers Under Occlusion". In: *Pattern Recogn.* 47.6 (2014), pp. 2280–2292. ISSN: 0031-3203. DOI: 10.1016/j.patcog.2014.01.005.

[3] E. Olson. "AprilTag: A robust and flexible visual fiducial system". In: *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. May 2011, pp. 3400–3407. DOI: 10.1109/ICRA.2011.5979561.

[4] H. Durrant-Whyte and T. Bailey. "Simultaneous localization and mapping: part I". In: *IEEE Robotics Automation Magazine* 13.2 (June 2006), pp. 99–110. ISSN: 1070-9932. DOI: 10.1109/MRA.2006.1638022.

[5] Michael Montemerlo, Sebastian Thrun, Daphne Koller, and Ben Wegbreit. "FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem". In: *Eighteenth National Conference on Artificial Intelligence*. Edmonton, Alberta, Canada: American Association for Artificial Intelligence, 2002, pp. 593–598. ISBN: 0-262-51129-0.

[6] R. Mur Artal, J. M. M. Montiel, and Juan D. Tardos. "ORB-SLAM: a Versatile and Accurate Monocular SLAM System". In: *IEEE Transactions on Robotics* 31.5 (2015), pp. 1147–1163. DOI: 10.1109/TRO.2015.2463671.

[7] A. Irschara, C. Zach, and H. Bischof. "Towards Wiki-based Dense City Modeling". In: *2007 IEEE 11th International Conference on Computer Vision*. Oct. 2007, pp. 1–8. DOI: 10.1109/ICCV.2007.4409216.

[8] Changchang Wu. "Towards Linear-Time Incremental Structure from Motion". In: *Proceedings of the 2013 International Conference on 3D Vision*. 3DV '13. Washington, DC, USA: IEEE Computer Society, 2013, pp. 127–134. ISBN: 978-0-7695-5067-1. DOI: 10.1109/3DV.2013.25.

[9] Brad Grinstead, Andreas Koschan, and Mongi A. Abidi. "Abidi, "A Comparison of Pose Estimation Techniques: Hardware vs. Video". In: *in Proc. of SPIE Unmanned Vehicle Technology VII*. 2005, pp. 166–173.

[10] Xiao-Shan Gao, Xiao-Rong Hou, Jianliang Tang, and Hang-Fei Cheng. "Complete Solution Classification for the Perspective-Three-Point Problem". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 25.8 (Aug. 2003), pp. 930–943. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2003.1217599.

[11] L. Kneip, D. Scaramuzza, and R. Siegwart. "A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation". In: *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. June 2011, pp. 2969–2976. DOI: 10.1109/CVPR.2011.5995464.

[12] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. "EPnP: An Accurate O(n) Solution to the PnP Problem". In: *International Journal of Computer Vision* 81.2 (2008), pp. 155–166. ISSN: 1573-1405. DOI: 10.1007/s11263-008-0152-6.

[13] F. Moreno-Noguer, V. Lepetit, and P. Fua. "Accurate Non-Iterative O(n) Solution to the PnP Problem". In: *2007 IEEE 11th International Conference on Computer Vision*. Oct. 2007, pp. 1–8. DOI: 10.1109/ICCV.2007.4409116.

[14] G. Schweighofer and A. Pinz. "Robust Pose Estimation from a Planar Target". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.12 (Dec. 2006), pp. 2024–2030. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2006.252.

[15] K. Konolige and M. Agrawal. "FrameSLAM: From Bundle Adjustment to Real-Time Visual Mapping". In: *Trans. Rob.* 24.5 (Oct. 2008), pp. 1066–1077. ISSN: 1552-3098. DOI: 10.1109/TRO.2008.2004832.

[16] David G Lowe. "Distinctive image features from Scale-Invariant Keypoints". In: *International Journal of Computer Vision* 60.2 (2004), pp. 91–110. DOI: 10.1023/B:VISI.0000029664.99615.94.

[17] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. "SURF: Speeded Up Robust Features". In: *Computer Vision – ECCV 2006*. Vol. 3951. Lecture Notes in Computer Science. Graz, Austria: Springer Berlin Heidelberg, 2006, pp. 404–417. ISBN: 978-3-540-33832-1. DOI: 10.1007/11744023_32.

[18] C Schmid and K Mikolajczyk. "A performance evaluation of local descriptors". In: *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*. Madison, Wisconsin, 2003, pp. 257–263. DOI: 10.1109/CVPR.2003.1211478.

[19] O. Miksik and K. Mikolajczyk. "Evaluation of local detectors and descriptors for fast feature matching". In: *Pattern Recognition (ICPR), 2012 21st International Conference on*. Nov. 2012, pp. 2681–2684. DOI: 10.1.1.301.6783.

[20] Man Hee Lee and In Kyu Park. "Computer Vision - ACCV 2014 Workshops: Singapore, Singapore, November 1-2, 2014, Revised Selected Papers, Part I". In: ed. by C.V. Jawahar and Shiguang Shan. Cham: Springer International Publishing, 2015. Chap. Performance Evaluation of Local Descriptors for Affine Invariant Region Detector, pp. 630–643. ISBN: 978-3-319-16628-5. DOI: 10.1007/978-3-319-16628-5_45.

[21] A. L. Dahl, H. Aanæs, and K. S. Pedersen. "Finding the Best Feature Detector-Descriptor Combination". In: *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2011 International Conference on*. May 2011, pp. 318–325. DOI: 10.1109/3DIMPVT.2011.47.

[22] S. Garrido-Jurado, R. Muñoz-Salinas, F.J Madrid-Cuevas, and R. Medina-Carnicer. "Generation of fiducial marker dictionaries using mixed integer linear programming". In: *Pattern Recognition* 51 (2016), pp. 481–491. ISSN: 0031-3203. DOI: 10.1016/j.patcog.2015.09.023.

[23] Abdel Y. I. Aziz and H. M. Karara. "Direct linear transformation into object space coordinates in close-range photogrammetry". In: *Proc. of the Symposium on Close-Range Photogrammetry*. Urbana, Illinois, 1971, pp. 1–18.

[24] Z. Zhang. “A flexible new technique for camera calibration”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.11 (Nov. 2000), pp. 1330–1334. ISSN: 0162-8828. DOI: 10.1109/34.888718.

[25] J. Heikkila and O. Silven. “A four-step camera calibration procedure with implicit image correction”. In: *Computer Vision and Pattern Recognition. Proceedings., 1997 IEEE Computer Society Conference on*. 1997, pp. 1106–1112. DOI: 10.1109/CVPR.1997.609468.

[26] Dr. Gary Rost Bradski and Adrian Kaehler. *Learning Opencv, 1st Edition*. First. O’Reilly Media, Inc., 2008. ISBN: 9780596516130.

[27] Jorge J. Moré. “Numerical Analysis: Proceedings of the Biennial Conference Held at Dundee, June 28–July 1, 1977”. In: ed. by G. A. Watson. Berlin, Heidelberg: Springer Berlin Heidelberg, 1978. Chap. The Levenberg-Marquardt algorithm: Implementation and theory, pp. 105–116. ISBN: 978-3-540-35972-2. DOI: 10.1007/BFb0067700.

[28] R. W. Hamming. “Error Detecting and Error Correcting Codes”. In: *Bell System Technical Journal* 29.2 (1950), pp. 147–160. ISSN: 1538-7305. DOI: 10.1002/j.1538-7305.1950.tb00463.x.

[29] Krystian Mikolajczyk and Cordelia Schmid. “A performance evaluation of local descriptors”. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 27.10 (2005), pp. 1615–1630. DOI: 10.1109/TPAMI.2005.188.

[30] Tony Lindeberg. “Feature Detection with Automatic Scale Selection”. In: *Int. J. Comput. Vision* 30.2 (Nov. 1998), pp. 79–116. ISSN: 0920-5691. DOI: 10.1023/A:1008045108935.

[31] Charles K. Chui. *An Introduction to Wavelets*. San Diego, CA, USA: Academic Press Professional, Inc., 1992. ISBN: 0-12-174584-8.

[32] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. “BRIEF: binary robust independent elementary features”. In: *Computer Vision – ECCV 2010*. Vol. 6314. Lecture Notes in Computer Science. Crete, Greece: Springer, 2010, pp. 778–792. DOI: 10.1007/978-3-642-15561-1_56.

[33] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. “ORB: An efficient alternative to SIFT or SURF”. In: *Computer Vision (ICCV), 2011 IEEE International Conference on*. Barcelona, Spain, Nov. 2011, pp. 2564–2571. DOI: 10.1109/ICCV.2011.6126544.

[34] Stefan Leutenegger, Margarita Chli, and Roland Y Siegwart. “BRISK: Binary Robust Invariant Scalable Keypoints”. In: *Computer Vision (ICCV), 2011 IEEE International Conference on*. Barcelona, Spain, 2011, pp. 2548–2555. DOI: 10.1109/ICCV.2011.6126542.

[35] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. “FREAK: Fast retina keypoint”. In: *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. Rhode Island, Providence, USA, 2012, pp. 510–517. DOI: 10.1109/CVPR.2012.6247715.

[36] Paul L. Rosin. “Measuring Corner Properties”. In: *Comput. Vis. Image Underst.* 73.2 (Feb. 1999), pp. 291–307. ISSN: 1077-3142. DOI: 10.1006/cviu.1998.0719.

[37] Engin Tola, Vincent Lepetit, and Pascal Fua. "DAISY: An Efficient Dense Descriptor Applied to Wide-Baseline Stereo". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.5 (2010), pp. 815–830. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2009.77.

[38] Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. "Bundle Adjustment - A Modern Synthesis". In: *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*. ICCV '99. London, UK, UK: Springer-Verlag, 2000, pp. 298–372. ISBN: 978-3-540-44480-0. DOI: 10.1007/3-540-44480-7_21.

[39] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M. Seitz, and Richard Szeliski. "Building Rome in a Day". In: *Commun. ACM* 54.10 (Oct. 2011), pp. 105–112. ISSN: 0001-0782. DOI: 10.1145/2001269.2001293.

[40] Klaus Dorfmüller and Hanno Wirth. "Modelling and Motion Capture Techniques for Virtual Environments: International Workshop, CAPTECH'98 Geneva, Switzerland, November 26–27, 1998 Proceedings". In: ed. by Nadia Magnenat-Thalmann and Daniel Thalmann. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998. Chap. Real-Time Hand and Head Tracking for Virtual Environments Using Infrared Beacons, pp. 113–127. ISBN: 978-3-540-49384-6. DOI: 10.1007/3-540-49384-0_9.

[41] M. Ribo, A. Pinz, and A. L. Fuhrmann. "A new optical tracking system for virtual and augmented reality applications". In: *Instrumentation and Measurement Technology Conference, 2001. IMTC 2001. Proceedings of the 18th IEEE*. Vol. 3. 2001, 1932–1936 vol.3. DOI: 10.1109/IMTC.2001.929537.

[42] L. Naimark and E. Foxlin. "Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker". In: *Mixed and Augmented Reality, 2002. ISMAR 2002. Proceedings. International Symposium on*. IEEE Computer Society, 2002, pp. 27–36. DOI: 10.1109/ISMAR.2002.1115065.

[43] Jun Rekimoto and Yuji Ayatsuka. "CyberCode: Designing Augmented Reality Environments with Visual Tags". In: *Proceedings of DARE 2000 on Designing Augmented Reality Environments*. DARE '00. Elsinore, Denmark: ACM, 2000, pp. 1–10. DOI: 10.1145/354666.354667.

[44] Michael Rohs and Beat Gfeller. "Using Camera-Equipped Mobile Phones for Interacting with Real-World Objects". In: *Advances in Pervasive Computing*. 2004, pp. 265–271. DOI: 10.1.1.2.7195.

[45] H. Kato and M. Billinghurst. "Marker tracking and HMD calibration for a video-based augmented reality conferencing system". In: *Augmented Reality, 1999. (IWAR '99) Proceedings. 2nd IEEE and ACM International Workshop on*. 1999, pp. 85–94. DOI: 10.1109/IWAR.1999.803809.

[46] M. Fiala. "Comparing ARTag and ARToolkit Plus fiducial marker systems". In: *Haptic Audio Visual Environments and their Applications, 2005. IEEE International Workshop on*. Oct. 2005, pp. 6–. DOI: 10.1109/HAVE.2005.1545669.

[47] J. Rekimoto. “Matrix: a realtime object identification and registration method for augmented reality”. In: *Computer Human Interaction, 1998. Proceedings. 3rd Asia Pacific*. July 1998, pp. 63–68. DOI: 10.1109/APCHI.1998.704151.

[48] M. Fiala. “Designing Highly Reliable Fiducial Markers”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.7 (2010), pp. 1317–1324. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2009.146.

[49] D. Flohr and J. Fischer. “A Lightweight ID-Based Extension for Marker Tracking Systems”. In: *Eurographics Symposium on Virtual Environments (EGVE) Short Paper Proceedings*. Weimar, 2007, pp. 59–64.

[50] Daniel Wagner and Dieter Schmalstieg. *ARToolKitPlus for Pose Tracking on Mobile Devices*. 2007.

[51] Satoshi Suzuki and KeiichiA be. “Topological structural analysis of digitized binary images by border following”. In: *Computer Vision, Graphics, and Image Processing* 30.1 (1985), pp. 32–46. ISSN: 0734-189X. DOI: 10.1016/0734-189X(85)90016-7.

[52] D. H Douglas and T. K Peucker. “Algorithms for the reduction of the number of points required to represent a digitized line or its caricature”. In: *Cartographica: The International Journal for Geographic Information and Geovisualization* 10 (2 1973), pp. 112–122. DOI: 10.3138/FM57-6770-U75U-7727.

[53] Nobuyuki Otsu. “A Threshold Selection Method from Gray-Level Histograms”. In: *IEEE Transactions on Systems, Man, and Cybernetics* 9.1 (Jan. 1979), pp. 62–66. ISSN: 0018-9472. DOI: 10.1109/TSMC.1979.4310076.

[54] K. S. Arun, T. S. Huang, and S. D. Blostein. “Least-Squares Fitting of Two 3-D Point Sets”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-9.5 (Sept. 1987), pp. 698–700. ISSN: 0162-8828. DOI: 10.1109/TPAMI.1987.4767965.

[55] A. P. Dempster. “Upper and Lower Probabilities Induced by a Multivalued Mapping”. In: *Ann. Math. Statist.* 38.2 (Apr. 1967), pp. 325–339. DOI: 10.1214/aoms/1177698950.

[56] R. Muñoz-Salinas, R. Medina-Carnicer, F.J. Madrid-Cuevas, and A. Carmona-Poyato. “Multi-camera people tracking using evidential filters”. In: *International Journal of Approximate Reasoning* 50 (2009), pp. 732–749. DOI: 10.1016/j.ijar.2009.02.001.

[57] Suvasini Panigrahi, Amlan Kundu, Shamik Sural, and Arun K. Majumdar. “Use of Dempster-Shafer Theory and Bayesian Inferencing for Fraud Detection in Mobile Communication Networks”. In: *Information Security and Privacy*. Vol. 4586. Lecture Notes in Computer Science. Townsville, Australia: Springer Berlin Heidelberg, 2007, pp. 446–460. DOI: 10.1007/978-3-540-73458-1_32.

[58] T. Denoeux and P. Smets. “Classification using Belief Functions: the Relationship between the Case-based and Model-based Approaches”. In: *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* 36 (6 2006), pp. 1395–1406. DOI: 10.1109/TSMCB.2006.877795.

[59] S. Demotier, W. Schon, and T. Denoeux. "Risk Assessment Based on Weak Information using Belief Functions: A Case Study in Water Treatment". In: *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* 36 (3 2006), pp. 382–396. DOI: 10.1109/TSMCC.2004.840057.

[60] T. Denoeux and M.-H. Masson. "EVCLUS: Evidential Clustering of Proximity Data". In: *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* 34 (1 2004), pp. 95–109. DOI: 10.1109/TSMCB.2002.806496.

[61] M.-H. Masson and T. Denoeux. "ECM: An evidential version of the fuzzy c-means algorithm". In: *Pattern Recognition* 41.4 (2008), pp. 1384–1397. DOI: 10.1016/j.patcog.2007.08.014.

[62] A-O. Boudraa, A. Bentabet, and F. Salzenstein. "Dempster-Shafer's Basic Probability Assignment Based on Fuzzy Membership Functions". In: *Electronic Letters on Computer Vision and Image Analysis* 4 (1 2004), pp. 1–10. DOI: 10.5565/rev/elcvia.68.

[63] I. Bloch. "Defining belief functions using mathematical morphology - Application to image fusion under imprecision". In: *International Journal of Approximate Reasoning* 48 (2 2008), pp. 437–465. DOI: 10.1016/j.ijar.2007.07.008.

[64] Z. Hammal, L. Couvreur, A. Caplier, and M. Rombaut. "Facial expression classification: An approach based on the fusion of facial deformations using the transferable belief model". In: *International Journal of Approximate Reasoning* 46 (3 2007), pp. 542–567. DOI: 10.1016/j.ijar.2007.02.003.

[65] W. Pieczynski. "Multisensor triplet Markov chains and theory of evidence". In: *International Journal of Approximate Reasoning* 45 (1 2007), pp. 1–16. DOI: 10.1016/j.ijar.2006.05.001.

[66] Z. Yi, H.Y. Khing, C.C. Seng, and Z.X. Wei. "Multi-ultrasonic sensor fusion for autonomous mobile robots". In: *SPIE proceedings series: Architectures, Algorithms and Applications IV*. Orlando, Florida, 2000, pp. 314–321. DOI: 10.1117/12.381644.

[67] L. Díaz-Más, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and R. Medina-Carnicer. "Shape from Silhouette Using Dempster-Shafer Theory". In: *Pattern Recogn.* 43.6 (June 2010), pp. 2119–2131. ISSN: 0031-3203. DOI: 10.1016/j.patcog.2010.01.001.

[68] H. Wu, M. Siegel, R. Stiefelhagen, and J. Yang. "Sensor Fusion Using Dempster-Shafer Theory". In: *Instrumentation and Measurement Technology Conference, 2002. IMTC/2002. Proceedings of the 19th IEEE*. Anchorage, AK, USA, 2002, 7–12 vol.1. DOI: 10.1109/IMTC.2002.1006807.

[69] N. Milisavljevic, I. Bloch, S. Broek, and M. Acheroy. "Improving mine recognition through processing and Dempster-Shafer fusion of ground-penetrating radar data". In: *Pattern Recognition* 36 (2003), pp. 1233–1250. DOI: 10.1016/S0031-3203(02)00251-0.

[70] Philippe Xu, Franck Davoine, Jean-Baptiste Bordes, Huijing Zhao, and Thierry Denoeux. "Multimodal information fusion for urban scene understanding". In: *Machine Vision and Applications* (2014), pp. 1–19. DOI: 10.1007/s00138-014-0649-7.

[71] T Denoeux, N El Zoghby, V Cherfaoui, and A Jouglet. "Optimal Object Association in the Dempster–Shafer Framework". In: *Cybernetics, IEEE Transactions on* 44.12 (2014), pp. 2521–2531. DOI: 10.1109/TCYB.2014.2309632.

[72] P. Smets. "The combination of evidence in the transferable belief model". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 12 (5 1990), pp. 447–458. DOI: 10.1109/34.55104.

[73] G. Shafer. *A mathematical theory of evidence*. Princeton university press, 1976.

[74] Philippe Smets. "The Canonical Decomposition of a Weighted Belief". In: *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*. IJCAI'95. Montreal, Quebec, Canada: Morgan Kaufmann Publishers Inc., 1995, pp. 1896–1901. ISBN: 1-55860-363-8.

[75] P. Smets and R. Kennes. "The Transferable Belief Model". In: *Artificial Intelligence* 66.2 (1994), pp. 191–243. DOI: 10.1016/0004-3702(94)90026-4.

[76] C.M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, 2006. ISBN: 9780387310732.

[77] Christopher J. C. Burges. "A tutorial on support vector machines for pattern recognition". In: *Data Mining and Knowledge Discovery* 2 (1998), pp. 121–167.

[78] Corinna Cortes and Vladimir Vapnik. "Support-Vector Networks". In: *Machine Learning* 20.3 (1995), pp. 273–297. ISSN: 1573-0565. DOI: 10.1023/A:1022627411411.

[79] Bernhard Schölkopf, Chris Burges, and Vladimir Vapnik. "Artificial Neural Networks — ICANN 96: 1996 International Conference Bochum, Germany, July 16–19, 1996 Proceedings". In: ed. by Christoph Malsburg, Werner Seelen, Jan C. Vorbruggen, and Bernhard Sendhoff. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996. Chap. Incorporating invariances in support vector learning machines, pp. 47–52. ISBN: 978-3-540-68684-2. DOI: 10.1007/3-540-61510-5_12.

[80] V. Blanz, B. Schölkopf, H. Bülthoff, C. Burges, V. Vapnik, and T. Vetter. "Artificial Neural Networks — ICANN 96: 1996 International Conference Bochum, Germany, July 16–19, 1996 Proceedings". In: ed. by Christoph Malsburg, Werner Seelen, Jan C. Vorbruggen, and Bernhard Sendhoff. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996. Chap. Comparison of view-based object recognition algorithms using realistic 3D models, pp. 251–256. ISBN: 978-3-540-68684-2. DOI: 10.1007/3-540-61510-5_45.

[81] E. Osuna, R. Freund, and F. Girosit. "Training support vector machines: an application to face detection". In: *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*. June 1997, pp. 130–136. DOI: 10.1109/CVPR.1997.609310.

[82] Thorsten Joachims. “Text Categorization with Suport Vector Machines: Learning with Many Relevant Features”. In: *Proceedings of the 10th European Conference on Machine Learning*. ECML ’98. London, UK, UK: Springer-Verlag, 1998, pp. 137–142. ISBN: 3-540-64417-2.

[83] WarrenS McCulloch and Walter Pitts. “A logical calculus of the ideas immanent in nervous activity”. In: *Bulletin of Mathematical Biology* 5.4 (Dec. 1943), pp. 115–133. ISSN: 0007-4985. DOI: 10.1007/bf02478259.

[84] N. Morgan and H. Bourlard. “Continuous speech recognition using multilayer perceptrons with hidden Markov models”. In: *Acoustics, Speech, and Signal Processing, 1990. ICASSP-90., 1990 International Conference on*. Apr. 1990, 413–416 vol.1. DOI: 10.1109/ICASSP.1990.115720.

[85] Zhengyou Zhang. “Feature-based facial expression recognition: Sensitivity analysis and experiments with a multilayer perceptron”. In: *International journal of pattern recognition and Artificial Intelligence* 13.06 (1999), pp. 893–911.

[86] Tim Hill, Leorey Marquez, Marcus O’Connor, and William Remus. “Artificial neural network models for forecasting and decision making”. In: *International Journal of Forecasting* 10.1 (1994), pp. 5–15. ISSN: 0169-2070. DOI: 10.1016/0169-2070(94)90045-0.

[87] P. Y. Lee, S. C. Hui, and A. C. M. Fong. “Neural networks for web content filtering”. In: *IEEE Intelligent Systems* 17.5 (Sept. 2002), pp. 48–57. ISSN: 1541-1672. DOI: 10.1109/MIS.2002.1039832.

[88] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. “Learning Internal Representations by Error Propagation”. In: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*. Ed. by David E. Rumelhart and James L. Mcclelland. Cambridge, MA: MIT Press, 1986, pp. 318–362.

[89] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. “Backpropagation Applied to Handwritten Zip Code Recognition”. In: *Neural Comput.* 1.4 (Dec. 1989), pp. 541–551. ISSN: 0899-7667.

[90] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in Neural Information Processing Systems*. 2012.

[91] W. Ouyang et al. “Deepid-net: multi-stage and deformable deep convolutional neural networks for object detection”. In: *arXiv preprint arXiv:1409.3505*. 2014.

[92] Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. “Deep Learning Face Representation by Joint Identification-Verification”. In: *Advances in Neural Information Processing Systems 27*. Ed. by Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger. Curran Associates, Inc., 2014, pp. 1988–1996.

[93] C. Dong, C. C. Loy, K. He, and X. Tang. “Image Super-Resolution Using Deep Convolutional Networks”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38.2 (Feb. 2016), pp. 295–307. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2015.2439281.

[94] Renliang Weng, Jiwen Lu, Junlin Hu, Gao Yang, and Yap-Peng Tan. “Robust Feature Set Matching for Partial Face Recognition”. In: *Computer Vision (ICCV), 2013 IEEE International Conference on*. Sydney, Australia, 2013, pp. 601–608. DOI: 10.1109/ICCV.2013.80.

[95] Eric P Xing, Andrew Y Ng, Michael I Jordan, and Stuart Russell. “Distance metric learning with application to clustering with side-information”. In: *Advances in neural information processing systems* (2002), pp. 505–512.

[96] Dongjian He, Shangsong Liang, and Yong Fang. “A Multi-Descriptor, Multi-Nearest Neighbor Approach for Image Classification”. In: *Advanced Intelligent Computing Theories and Applications*. Vol. 6215. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2010, pp. 515–523. DOI: 10.1007/978-3-642-14922-1_64.

[97] Peter Mountney, Benny Lo, Surapa Thiemjarus, Danail Stoyanov, and Guang Zhong-Yang. “A Probabilistic Framework for Tracking Deformable Soft Tissue in Minimally Invasive Surgery”. In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2007*. Vol. 4792. Lecture Notes in Computer Science. Brisbane, Australia: Springer Berlin Heidelberg, 2007, pp. 34–41. DOI: 10.1007/978-3-540-75759-7_5.

[98] Panagiotis Perakis, Theoharis Theoharis, and Ioannis A. Kakadiaris. “Feature fusion for facial landmark detection”. In: *Pattern Recognition* 47.9 (2014), pp. 2783–2793. ISSN: 0031-3203. DOI: 10.1016/j.patcog.2014.03.007.

[99] Anil Jain, Karthik Nandakumar, and Arun Ross. “Score Normalization in Multimodal Biometric Systems”. In: *Pattern Recogn.* 38.12 (2005), pp. 2270–2285. ISSN: 0031-3203. DOI: 10.1016/j.patcog.2005.01.012.

[100] Van-Nam Huynh, Tri Thanh Nguyen, and Cuong Anh Le. “Adaptively entropy-based weighting classifiers in combination using Dempster–Shafer theory for word sense disambiguation”. In: *Computer Speech & Language* 24.3 (2010), pp. 461–473. DOI: 10.1016/j.csl.2009.06.003.

[101] S. Ranoeliarivao, F. de Morsier, D. Tuia, S. Rakotoniaina, M. Borgeaud, J.-P. Thiran, and S. Rakotondraompiana. “Multisource clustering of remote sensing images with Entropy-based Dempster-Shafer fusion”. In: *Signal Processing Conference (EUSIPCO), 2013 Proceedings of the 21st European*. Marrakech, Morocco, Sept. 2013, pp. 1–5.

[102] D. Pan, X. Lu, J. Liu, and Y. Deng. “A Ranking Procedure by Incomplete Pairwise Comparisons Using Information Entropy and Dempster-Shafer Evidence Theory”. In: *The Scientific World Journal* 2014 (2014). DOI: 10.1155/2014/904596.

[103] Frank Wilcoxon. “Individual Comparisons by Ranking Methods”. In: *Biometrics Bulletin* 1.6 (Dec. 1945), pp. 80–83. ISSN: 00994987. DOI: 10.2307/3001968.

[104] D. Wagner and D. Schmalstieg. “ARToolKitPlus for Pose Tracking on Mobile Devices”. In: *Computer Vision Winter Workshop*. 2007, pp. 139–146.

[105] J. Rekimoto. “Matrix: a realtime object identification and registration method for augmented reality”. In: *Computer Human Interaction, 1998. Proceedings. 3rd Asia Pacific*. July 1998, pp. 63–68. DOI: 10.1109/APCHI.1998.704151.

[106] Gukhwan Kim and E.M. Petriu. “Fiducial marker indoor localization with Artificial Neural Network”. In: *Advanced Intelligent Mechatronics (AIM), 2010 IEEE/ASME International Conference on*. July 2010, pp. 961–966. DOI: 10.1109/AIM.2010.5695801.

[107] Mark J. Huiskes and Michael S. Lew. “The MIR Flickr Retrieval Evaluation”. In: *Proceedings of the 1st ACM International Conference on Multimedia Information Retrieval*. MIR ’08. Vancouver, British Columbia, Canada: ACM, 2008, pp. 39–43. ISBN: 978-1-60558-312-9. DOI: 10.1145/1460096.1460104.

[108] Salvador García, Alberto Fernández, Julián Luengo, and Francisco Herrera. “Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power”. In: *Information Sciences* 180.10 (2010). Special Issue on Intelligent Distributed Information Systems, pp. 2044–2064. ISSN: 0020-0255. DOI: 10.1016/j.ins.2009.12.010.

[109] Milton Friedman. “The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance”. In: *Journal of the American Statistical Association* 32.200 (Dec. 1937), pp. 675–701. ISSN: 01621459. DOI: 10.2307/2279372.

[110] H. Finner. “On a Monotonicity Problem in Step-Down Multiple Test Procedures”. In: *Journal of the American Statistical Association* 88.423 (1993), pp. 920–923. DOI: 10.1080/01621459.1993.10476358.

[111] K. Pearson. “On lines and planes of closest fit to systems of points in space”. In: *Philosophical Magazine* 2.6 (1901), pp. 559–572.

[112] V.M. Mondéjar-Guerra, R. Muñoz-Salinas, M.J. Marín-Jiménez, A. Carmona-Poyato, and R. Medina-Carnicer. “Keypoint descriptor fusion with Dempster-Shafer theory”. In: *Int. J. Approx. Reasoning* 60.C (2015), pp. 57–70. ISSN: 0888-613X. DOI: 10.1016/j.ijar.2015.03.001.

[113] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. “Neurocomputing: Foundations of Research”. In: ed. by James A. Anderson and Edward Rosenfeld. Cambridge, MA, USA: MIT Press, 1988. Chap. Learning Representations by Back-propagating Errors, pp. 696–699. ISBN: 0-262-01097-6.

[114] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (Nov. 1998), pp. 2278–2324. ISSN: 0018-9219. DOI: 10.1109/5.726791.

Capítulo A

Keypoint descriptor fusion with Dempster–Shafer theory

En esta sección se incluye el artículo correspondiente a la primera contribución realizada durante la presente Tesis Doctoral y que se corresponde con la propuesta descrita en el Capítulo 3.

El artículo titulado "*Keypoint descriptor fusion with Dempster–Shafer theory*" se encuentra publicado en la revista *International Journal of Approximate Reasoning*:

V.M. Mondéjar-Guerra, R. Muñoz-Salinas, M.J. Marín-Jiménez, A. Carmona-Poyato, R. Medina-Carnicer, Keypoint descriptor fusion with Dempster–Shafer theory, International Journal of Approximate Reasoning, Volume 60, May 2015, Pages 57-70, ISSN 0888-613X, dx.doi.org/10.1016/j.ijar.2015.03.001.

International Journal of Approximate Reasoning 60 (2015) 57–70

Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

www.elsevier.com/locate/ijar

Keypoint descriptor fusion with Dempster–Shafer theory ☆

V.M. Mondéjar-Guerra, R. Muñoz-Salinas *, M.J. Marín-Jiménez, A. Carmona-Poyato, R. Medina-Carnicer

Maimonides Institute for Biomedical Research (IMIBIC), Department of Computing and Numerical Analysis, Cordoba University, 14071 Cordoba, Spain

ARTICLE INFO

Article history:
Received 13 June 2014
Received in revised form 4 March 2015
Accepted 4 March 2015
Available online 6 March 2015

Keywords:
Keypoint matching
Local descriptor
Dempster–Shafer

ABSTRACT

Keypoint matching is the task of accurately finding the location of a scene point in two images. Many keypoint descriptors have been proposed in the literature aiming at providing robustness against scale, translation and rotation transformations, each having advantages and disadvantages. This paper proposes a novel approach to fuse the information from multiple keypoint descriptors using Dempster–Shafer theory of evidence [1], which has proven particularly efficient in combining sources of information providing incomplete, imprecise, biased, and conflictive knowledge. The matching results of each descriptor are transformed into an evidence distribution on which a confidence factor is computed making use of its entropy. Then, the evidence distributions are fused using Dempster–Shafer Theory (DST), considering its confidence. As a result of the fusion, a new evidence distribution that improves the result of the best descriptor is obtained. Our method has been tested with SIFT, SURF, ORB, BRISK and FREAK descriptors using all possible their combinations. Results on the Oxford keypoint dataset [2] show that the proposed approach obtains an improvement of up to 10% compared to the best one (FREAK).

1. Introduction

Keypoint matching is an important task with many applications in areas such as image retrieval [3], image stitching [4], object recognition [5] and stereo matching [6,7], among others. The goal of keypoint matching is to find pixel correspondences representing the same real point in two images. Many local 2D descriptors have been proposed in the last few years for this task, each having its weaknesses and strengths. The most popular ones are SIFT [5], SURF [8], ORB [9], BRISK [10] and FREAK [11]. However, little attention has been paid to the idea of fusing multiple descriptors as to improve their matching capability.

The process of finding and matching the correspondences between two images with 2D descriptors is usually composed by the next three steps. First, keypoints are detected at the most distinctive locations in the images. An ideal detector would detect the same keypoint locations under different image transformations (i.e. rotation, scale, viewpoint or illumination changes). Second, for each detected keypoint a descriptor is built to describe the local region. Finally, keypoints are matched

☆ This work has been developed with the support of the Research Projects BROCA and TIN2012-32952, both financed by the Ministry of Science and Technology of Spain and FEDER.

* Corresponding author. Tel.: +34 957 21 22 89; fax: +34 957 21 86 30.
E-mail address: rmsalinas@uco.es (R. Muñoz-Salinas).

http://dx.doi.org/10.1016/j.ijar.2015.03.001
0888-613X/

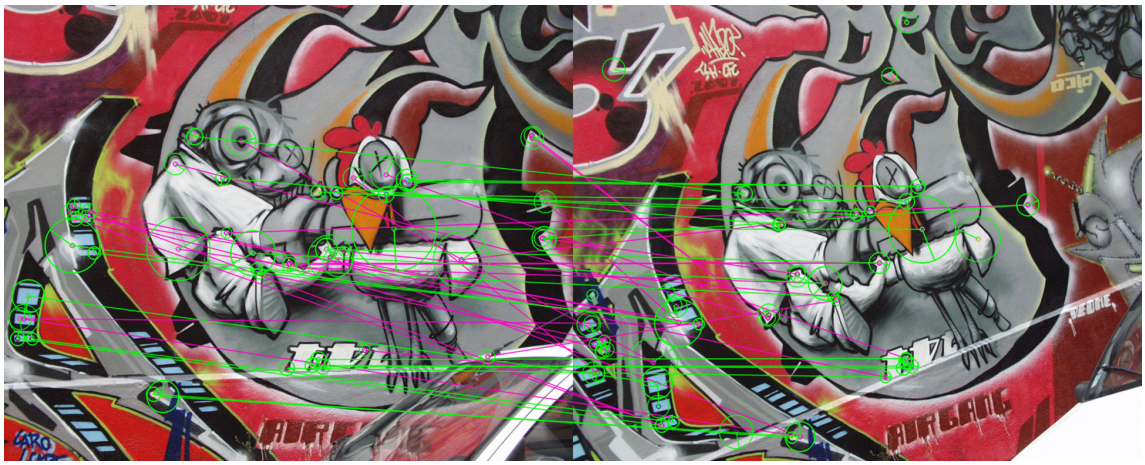

Fig. 1. Keypoint matching example from the Oxford dataset [2]. Two images of the same scene taken from different angles. Keypoints in both images are represented as circles. Green lines represent correct matches while pink lines are erroneous matches. (Best viewed in color; see the web version of this article.)

by analyzing the similarity between descriptors in both images. Fig. 1 shows a keypoint matching example with two images of the same scene taken from different viewpoints.

This work proposes a novel approach to improve the matching process by fusing the information provided by multiple descriptors using Dempster–Shafer Theory (DST) [1]. DST employs degrees of evidence (a weaker version of probabilities), making the manipulation of uncertainty especially attractive because of its simplicity and because it does not require specifying priors or conditionals that might be unfeasible to obtain in certain problems. In our proposal, each descriptor is employed to generate a sorted list of candidates, which is then transformed into an evidence distribution, from which a global confidence factor is obtained using its entropy. Then, DST is employed to fuse the evidence distributions using the confidence factor to weight the contribution of each descriptor. Finally, the best candidate is selected using the pignistic probability on the fused evidence distribution. Compared to other approaches, our method does not require a training step determining the performance of each descriptor in order to assign a confidence factor. Instead, the confidence assigned to each descriptor is calculated independently for each keypoint using the entropy of the evidence distribution generated. The experiments conducted on the Oxford dataset [2] show that the proposed method improves the results of the individual components fused, up to 10% in the best case. In addition, our method compares favorably with the standard fusion approaches presented in [12].

The rest of this paper is organized as follows. After presenting in Section 2 the related works, Section 3 provides a general formulation of the keypoint descriptor fusion problem. Section 4 presents the standard methods for fusion employed in [12] which are compared with our method. Then, Section 5 introduces DST and our proposal. Finally, Section 6 presents the experiments conducted and Section 7 draws some conclusions.

2. Related works

2.1. Keypoint descriptors

In the last years, several local keypoint descriptors have been proposed. The most well-known descriptor is SIFT (Scale-Invariant Feature Transform) by Lowe [5]. It convolves the image with Difference of Gaussians (DoG) functions at multiple scales to detect keypoint locations that are scale-space extremes. The descriptor is built computing several histograms of gradient orientations resulting in a feature vector that is invariant to scale and rotation. It is shown in [13] that SIFT is one of the most effective descriptors but its low speed has been criticized. To deal with this problem Bay et al. [8] propose SURF (Speeded Up Robust Features) as a faster descriptor that also maintains a similar matching performance. For the extraction step SURF uses the Fast-Hessian detector, which is based on the determinant of the Hessian matrix. SURF descriptors rely on sums of 2D Haar wavelet responses in (x, y) directions. Many real-time applications demand fast keypoint extractors and low-memory descriptors. As a result, the feature extractor FAST [14] (Features from Accelerated Segment Test) and various binary descriptors have been proposed. FAST analyzes a circular region around each keypoint candidate, if there are a minimum number of connected pixels on the region which are brighter or darker than the central pixel the keypoint candidate is selected. Besides the low memory requirements one advantage of binary descriptors is their matching speed: since binary features are employed, the Hamming distance (much faster to compute than the Euclidean one) can be used. Calonder et al. [15] introduced the first binary keypoint descriptor called BRIEF (Binary Robust Independent Elementary Features). It computes some binary tests over an image patch and concatenates them into a bit string that compose the descriptor. Each

test is a simple binary comparison between the intensity value of two pixel locations belonging to the patch. Later, Rublee et al. [9] proposed ORB (Oriented Fast and Rotate BRIEF) as an enhancement of the FAST detector and BRIEF descriptor. It adds a scale pyramid image to create FAST features at different levels and uses the intensity centroid, computed by standard moments, to obtain the orientation. Then, ORB steers pixels accordingly to build a descriptor that is invariant to rotation. Leutenegger et al. [10] proposed BRISK (Binary Robust Invariant Scalable Keypoints) which relies on a circular sampling pattern from which it computes several brightness comparisons creating a binary string invariant to scale and rotation. The last remarkable descriptor published recently is FREAK (Fast Retina Keypoint) by Alahi et al. [11] which relies on a retinal sampling pattern, inspired by the human retina, which uses an algorithm to select the set of binary tests that maximizes the diversity between them in order to construct the feature descriptors.

2.2. Keypoint fusion

Despite the many efforts devoted to the development of new keypoint descriptors, little attention has been paid to the idea of fusing the existing keypoint descriptors in order to improve their results. In [16], He et al. propose a Multi-Descriptor Multi-Nearest Neighbor, which classifies images by employing its nearest neighbors coming from all of the categories and different kinds of feature descriptors from images. Compared to other NN-based image classification methods, their approach combines different kinds of descriptors with the nearest neighbors coming from all classes instead of using only the K nearest neighbors with only one feature descriptor.

In the work [17], Bakshi et al. propose a method for IRIS people recognition that fuses the matches obtained by SURF and SIFT keypoint descriptors. Their approach does the matching with both descriptors independently, and then, analyzes the number of matches obtained by the two descriptors, which is used as a matching score itself.

In the work of Mountney et al. [18], the authors propose a Bayesian framework for fusing keypoint descriptors. In their approach, a first training step selects among a set of keypoint descriptors those that are most discriminative. Then, a Naive Bayesian Network trained on a subset of data with ground truth is employed to fuse the selected keypoint descriptors. The main drawback of this approach is that it uses a trained scheme which depends on the dataset employed. While a particular keypoint descriptor might not be very informative for a given dataset, it might be useful in an unknown situation, but if it has already been discarded, its contribution cannot be evaluated.

The work of Perakis et al. [12] evaluates the fusion of 2D and 3D features in the context of facial landmark detection. In a first step, landmarks are learned from an annotated dataset, and later, a template matching approach is applied for recognition. They evaluate several fusion approaches for template matching showing their advantages and disadvantages. This method is compared with ours in the experimental section.

The work of Weng et al. [19] proposes a feature set matching approach focused on the face recognition problem when there are occlusion or illumination changes. For each keypoint detected, they compute SIFT and SURF descriptors and combine them by simple concatenation. However, as they state, this simple combination cannot be effectively represented in Euclidean space. In order to exploit the discriminative potential of the two concatenated descriptors the authors use Metric Learning [20]. Their experiments show that the combination of SIFT and SURF descriptors can improve the invariance of local features to illumination, viewpoint and pose variations.

In general, the related works require a training step in order to assign weights to the different features. In contrast, we deal with the problem of determining the best match for a source point between a set of target points with no previous knowledge. To that end, our approach creates a set of evidence distributions that are fused using DST.

DST, which is also known as the evidence theory, is a generalization of the Bayes theory of subjective probability. It includes several models of reasoning under uncertainty such as the Smets' Transferable Belief Model (TBM) [21]. It has been applied to several disciplines such as people tracking [22], fraud detection [23], classification [24], risk analysis [25], clustering [26,27], image processing [28–31], autonomous robot mapping [32], human–computer interaction [33], land mine detection [34] and driver assistance [35], among others.

In [36] Denoeux et al. present a solution to the multi-target tracking problem. Three attributes (position, velocity, and class) are employed to describe the targets. Then, each attribute is considered a piece of evidence and they are fused using DST. This problem is similar to ours since it is in essence a multi-target assignment problem, where targets are described by multiple attributes.

In the context of classification with DST, entropy has been previously employed as a measure of confidence. In [37], the authors fuse the output of several classifiers for the problem of *word sense disambiguation*. They define the weights of the classifiers adaptively based on ambiguity measures associated with their decisions with respect to each particular pattern under classification. In their approach, the ambiguity is defined by Shannon's entropy. The work of Ranoeliarivao et al. [38] tackles the problem of multi-source clustering of remote sensing images based on an unsupervised assignment of evidences. Instead of using a user-defined ambiguity threshold to decide if a sample is highly ambiguous, they employ an ambiguity factor based on the local entropy among the cluster memberships. In [39], a method to predict ranking preferences is proposed using a discount rate derived from the information entropy. The goodness of the decision-making between different groups of experts is evaluated by analyzing the entropy of their decisions. In our work, the entropy of the evidence distributions is employed as a measure of confidence in the keypoint descriptor.

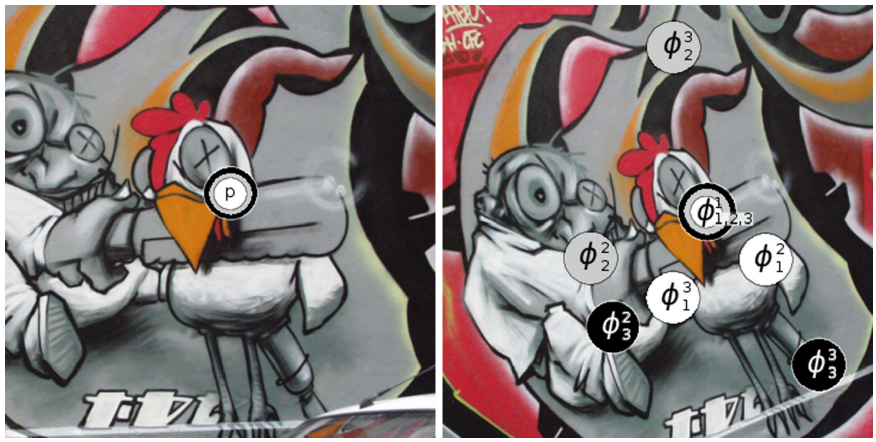

Fig. 2. Example of the problem for $K = 3$, where $k = 1$ SIFT, $k = 2$ SURF, $k = 3$ FREAK. (Left) Source image with the keypoint p to be matched. (Right) Target image with the n best matches ϕ_k^i for each descriptor when $n = 3$.

3. Problem formulation

Let us denote by p a point detected by a keypoint detector in a source image. Our aim is to find the best correspondence q in a target image from the set of keypoints $\mathcal{Q} = \{q^1, \ldots, q^N\}$. It must be noticed that the correspondence might not be in $\mathcal{Q}$ because of occlusion or error of the keypoint detector. Let us denote by

$$\Psi(p) = \{\psi_k(p) \mid k = 1 \ldots K\}, \tag{1}$$

the set of keypoint descriptors used to describe the point p, where

$$\psi_k(p) = (f_{1p}^k, f_{2p}^k, f_{3p}^k, \ldots f_{n_k p}^k,) \tag{2}$$

is the feature vector provided by the k-th descriptor. Please notice that we are employing a single keypoint detector to extract the keypoints, and then, different descriptors are used on each keypoint.

The first step is to compare the feature vector $\psi_k(p)$ with those of the points in the set $\mathcal{Q}$. Let us define then

$$d_k(p, q) = |\psi_k(p) - \psi_k(q)| \tag{3}$$

as a distance measure between two keypoint descriptors. Using d_k, the points of $\mathcal{Q}$ are sorted in ascending order independently for each descriptor creating the sets

$$\Phi(p, \mathcal{Q}) = \{\phi_k(p, \mathcal{Q}) \mid k = 1 \ldots K\}, \tag{4}$$

$$\phi_k(p, \mathcal{Q}) = \left\{(\phi_k^1, \ldots, \phi_k^N) \in \mathcal{Q} \mid d_k(p, \phi_k^i) \leq d_k(p, \phi_k^j)\ \forall i > j\right\}. \tag{5}$$

The vector $\phi_k(p, \mathcal{Q})$ includes the points from $\mathcal{Q}$ sorted in ascending order according to d_k, i.e., it is a permutation of the elements in $\mathcal{Q}$.

One might expect the first element of the vector $\phi_k(p, \mathcal{Q})$ to be the one with higher likelihood of being the correct match. And ideally, all keypoint descriptors should produce the same ordered vector. However, in practice, some descriptors are better suited for detecting points in particular circumstances than others. Therefore, each vector $\phi_k(p, \mathcal{Q})$ corresponds to a different permutation of the elements in the set $\mathcal{Q}$. Our goal then is to combine the information in $\Phi(p, \mathcal{Q})$ to find the correct match from the conflicting evidences provided by the different descriptors.

Since the most probable candidates are the first terms of the vector $\phi_k(p, \mathcal{Q})$, using all of them implies an unnecessary increase of memory and computing time. Consequently, we reduce the size of the vectors by retaining only the first n elements, i.e., $|\phi_k(p, \mathcal{Q})| = n$. Fig. 2 shows an example with $K = 3$ descriptors keeping the best $n = 3$ matches.

4. Standard feature fusion approaches

This section explains the proposal of Perakis et al. [12] for keypoint descriptor fusion, adapted to our work. Although their work is based on facial landmark detection, their approach is a general method valid for any application requiring fusion of descriptors.

In a first step, a descriptor similarity mapping function is employed providing a normalized value indicating how similar two descriptors are. Given the feature vector $\psi_k(p)$, and a target feature vector $\psi_k(q)$, the *normalized distance* $\hat{d}_k(p,q)$ is given by:

$$\hat{d}_k(p,q) = \frac{d_k(p,q)}{d_k(p,\phi_k^n)}. \tag{6}$$

The idea is to divide the distance between p and q by the maximum distance in the set $\phi_k(p,\mathcal{Q})$. Then, a *normalized similarity measure* can be defined by using the function:

$$S_k(p,q) = 1 - \hat{d}_k(p,q)^e, \tag{7}$$

where e favors the nearest elements to the target. When $e=1$ we obtain a linear mapping, while using $e=2$ we obtain the quadratic mapping (as expressed in [12]). The similarity measure $S_k(p,q)$ is one if both feature vectors are equal, and is 0 for the farthest element.

The similarity measures obtained for the K keypoint descriptors can be fused by using any of the following rules:

- Sum rule:

$$S_A(p,q) = \frac{1}{N}\sum_{k=1}^{K} S_k(p,q). \tag{8}$$

- Root-mean-square rule:

$$S_E(p,q) = \frac{1}{\sqrt{K}}\left(\sum_{k=1}^{K} S_k(p,q)^2\right)^{1/2}. \tag{9}$$

- Product rule:

$$S_G(p,q) = \left(\prod_{k=1}^{K} S_k(p,q)\right)^{1/K}. \tag{10}$$

- Max rule:

$$S_{max}(p,q) = max_{k=1}^{K}\left(S_k(p,q)\right). \tag{11}$$

Each of these rules causes a different fusion behavior. In this work, we are testing different combinations of normalized similarity measures and fusion rules so as to find the best one for our problem.

5. Dempster–Shafer proposal for keypoint fusion

This section explains our proposal for keypoint descriptor fusion using DST. First, we provide a brief introduction to DST, and then, we detail our fusion approach.

5.1. Dempster–Shafer theory of evidence

Let us consider a variable ω taking values in the *frame of discernment* Ω, and let us denote *the power set* by 2^{Ω}. A basic belief assignment (bba)

$$m : 2^{\Omega} \rightarrow [0,1],$$

is a function that assigns masses of belief to the subsets A of the power set, verifying:

$$\sum_{A\in\Omega} m(A) = 1. \tag{12}$$

While the evidence assigned to an event in the Bayesian approach must be a probability distribution function, the mass $m(A)$ of a power set element can be a subjective function expressing how much evidence supports the fact A. Furthermore, complete ignorance about the problem can be represented by $m(\Omega)=1$.

The original Shafer's model imposes the condition $m(\emptyset)=0$ in addition to that expressed in Eq. (12), i.e., the empty subset should not have mass of belief. However, Smets' TBM model relaxes that condition so that $m(\emptyset)>0$ stands for the possibility of incompleteness and conflict [21]. In the first case, $m(\emptyset)$ is interpreted as the belief that something out of Ω happens, i.e. accepting the *open-world assumption*. In the second case, the mass of the empty set can be seen as a

measure of conflict arising when merging information from sources pointing towards different directions. Nonetheless, a renormalization can transform a Smets' bba m into a Dempster's bba m^* as:

$$\begin{aligned} &m^*(\emptyset) = 0, \\ &m^*(A) = \tfrac{m(A)}{1-m(\emptyset)} \quad \text{if } A \neq \emptyset. \end{aligned} \tag{13}$$

A mass function m may be represented in several equivalent ways, of which *commonality* is of importance in this work:

$$q(A) = \sum_{B \supseteq A} m(B), \ \forall A, B \subseteq \Omega. \tag{14}$$

Using the above concepts, it is possible to define the generalized simple bba [40] as a function μ from 2^Ω to $\mathbb{R}$ verifying

$$\begin{aligned} &\mu(A) = 1 - w(A) \\ &\mu(\Omega) = w(A) \\ &\mu(B) = 0 \ \forall B \in 2^\Omega \setminus \{A, \Omega\} \end{aligned} \tag{15}$$

for some $A \neq \Omega$ and some $w \in [0, +\infty)$. Let us denote such a mass function as A^w. The weights $w(A)$ can be obtained from the commonalities using the following formula:

$$w(A) = \prod_{B \supseteq A} q(B)^{(-1)^{|B|-|A|+1}}. \tag{16}$$

This is another alternative representation of a bba which is very useful for fusing different pieces of evidences. Obtaining the masses back from the weights is possible using the Fast Möbius Transform [41].

In this work, we employ the Frank's family of t-norms [42]:

$$x \top_s y = log_s \left(1 + \frac{(s^x - 1)(s^y - 1)}{s - 1}\right), \tag{17}$$

where log_s represents the logarithm function with base $s > 0$. Each value of s defines a t-norm. As $s \to 0$, Eq. (17) tends to the minimum between x and y, and to the product as $s = 1$. Using this terminology, the combination of two bbas is defined as:

$$m_1 \textcircled{\scriptsize T}_s m_2 = \bigcirc\!\!\!\!\!\!\bigcirc_{A \in \Omega} A^{w1(A) \top_s w_2(A)}, \tag{18}$$

which is a flexible way to modify with the parameter s the properties of the combination rule. When $s = 0$, the Denoeux [43] cautious rule is obtained, which implements the Least Commitment Principle (LCP). It postulates that, given two belief functions compatible with a set of constraints, the most appropriate is the least informative. This is a rule which assumes that the sources manage dependent pieces of information. On the other hand, when $s = 1$, we obtain Dempster's conjunctive sum operation [44] which can be equivalently defined as:

$$(m_1 \textcircled{\cap} m_2)(A) = \sum_{B \cap C = A} m_1(B) m_2(C) \ \ \forall A \subseteq \Omega. \tag{19}$$

This rule assumes that sources are uncorrelated.

In some applications it is necessary to make a decision and choose the most reliable single hypotheses ω. To do so, Smets [45] proposed the use of the pignistic transformation that is defined for a normal bba as:

$$BetP(\omega) = \sum_{A \subseteq \Omega, \omega \in A} \frac{m(A)}{|A|}. \tag{20}$$

Other approaches for that end are the intersection probability [46], the orthogonal projection and the relative plausibility [47].

5.2. Proposed approach: keypoint descriptor fusion with DST

Using as input the vectors of $\Phi(p, \mathcal{Q})$ (i.e., each $\phi_k(p, \mathcal{Q})$), we construct the evidence distribution set

$$\Pi(p) = \{\pi_k(p) \mid k = 1 \dots K\}, \tag{21}$$

such that

$$\pi_k(p) = \left(\delta_k(p, \phi_k^1), \dots, \delta_k(p, \phi_k^n)\right), \tag{22}$$

where

$$\delta_k(p, q) = \begin{cases} \varphi\left(\frac{d_k(p, \phi_k^1)}{d_k(p, q)}\right)^\beta & \text{if } q \in \phi_k(p, \mathcal{Q}) \\ 0 & \text{otherwise.} \end{cases} \tag{23}$$

The value ϕ_k^1 is the first element of $\phi_k(p, \mathcal{Q})$, φ is a constant factor so that the integral of the distribution is one, and β is an exponential parameter (experimentally evaluated in Section 6) that modifies the shape of the distribution.

The evidence distributions created with Eq. (22) have the following properties. First, the best candidate (the one with smallest distance) receives the highest evidence. Second, it is a normalized distribution that can be used to compare the results of different keypoint descriptors. It must be reminded that each descriptor works in its own feature space (with its own number of features), thus, distances between different descriptors are not directly comparable. Finally, the parameter β is employed to modify the distribution. Values of β greater than one tend to increase the influence of the first terms, while values smaller than one tend to smooth the distribution.

An ideal descriptor would produce a very small distance for the correct candidate and a very high distance for the rest. So, an ideal descriptor would produce an evidence distribution with a very high value for the first element and very low values for the rest. However, if the descriptor is not good, it might provide high (or low) distance values for all keypoints, including for the correct match, thus producing a uniform distribution. Finally, if the source keypoint is part of a repetitive pattern (such as a chessboard), even a good descriptor would have many possible correct matches, again producing a uniform distribution.

Based on these ideas, the Shannon's entropy of the evidence distribution is proposed as confidence factor, which is calculated as:

$$c_k(p) = 1 - \sum_{i=1}^{n} -\delta_k(p, \phi_k^i) \log\left(\delta_k(p, \phi_k^i)\right). \tag{24}$$

When the entropy of the distribution is 0 (i.e., there is only one element with evidence 1 and the rest is zero) then, $c_k(p)$ is one. As the evidence distribution $\pi_k(p)$ tends to a uniform one, $c_k(p)$ tends to zero.

The evidence distributions $\pi_k(p)$ and the confidence values $c_k(p)$ are employed to define the bba for each keypoint descriptor. In our problem, the set of facts $\mathcal{S}$ is comprised by all the points from $\mathcal{Q}$ selected to create the evidence distributions. Thus, we can define $\mathcal{S}$ as the set of points from $\mathcal{Q}$ that are in $\Phi(p, \mathcal{Q})$:

$$\mathcal{S} = \{\mathrm{w} \in \mathcal{Q} \mid \exists k,\ \ \mathrm{w} \in \phi_k(p, \mathcal{Q})\}. \tag{25}$$

Please notice that $\mathcal{S}$ is a subset of $\mathcal{Q}$. Considering each keypoint descriptor as an independent sensor, the bba of the k-th sensor is defined from Eqs. (23) and (24) as:

$$\begin{aligned} m_k(\Omega) &= c_k(p), \\ m_k(\mathrm{w}) &= \delta_k(p, \mathrm{w}). \end{aligned} \tag{26}$$

Using the t-norm combination rules explained previously (Eq. (17)), the fused masses of belief over the power set $\mathcal{S}$ are obtained. Finally, the pignistic transformation (Eq. (20)) is applied on the fused results sorting the keypoints of $\mathcal{S}$ in descending order:

$$\mathcal{S}' = \left(\mathrm{w}^1, \mathrm{w}^2, \ldots, \mathrm{w}^{|\mathcal{S}|} \mid \mathrm{w}^i \in \mathcal{S} \ \wedge\ BetP(\mathrm{w}^i) > BetP(\mathrm{w}^j)\ \forall i < j\right). \tag{27}$$

In other words, the keypoint w^1 is the most likely keypoint in $\mathcal{Q}$ to be the correct match, w^2 the second most likely and so on. It must be noted that conflict between the descriptors has not been managed at this level. Instead, the ratio between the two best descriptors ($\mathbf{w}^2/\mathbf{w}^1$) is later analyzed to see if the match can be reliably considered, by ensuring it is below a threshold α, thus preventing false positives. This is better explained in the next section.

6. Experiments and results

This section presents the experiments conducted to test the proposed method. The well-known Oxford dataset[1] of Schmid and Mikolajczyk [2] has been employed in this work for evaluation purposes. It consists of seven scenes that include different image transformations: zoom and rotation (Boat), image blur (Bikes and Trees), view-point change (Graffiti and Wall), light change (Leuven) and JPEG compression (UBC), see Fig. 3. Each scene contains a sequence of five images sorted by the amount of transformation. The homography H with respect to the first image that is used to determine the ground truth is also provided by the dataset. The matching comparatives are performed against the first sequence image in all cases, i.e., the first image plays the role of source image, and the rest are the target images.

For all tested methods, the nearest neighbor distance ratio matching (NNDR) [13] strategy has been applied. It means that a possible match provided by a descriptor is only selected if it satisfies the condition $d(p, \phi_k^1)/d(p, \phi_k^2) < \alpha$, where $\alpha \in [0, 1]$. This approach prevents false positives when a descriptor provides similar responses to the two best matching candidates. In our method, the same result is obtained as the ratio between the two most likely candidates, i.e., $(\mathbf{w}^2/\mathbf{w}^1) < \alpha$.

[1] The dataset is available at http://www.robots.ox.ac.uk/~vgg/research/affine/.

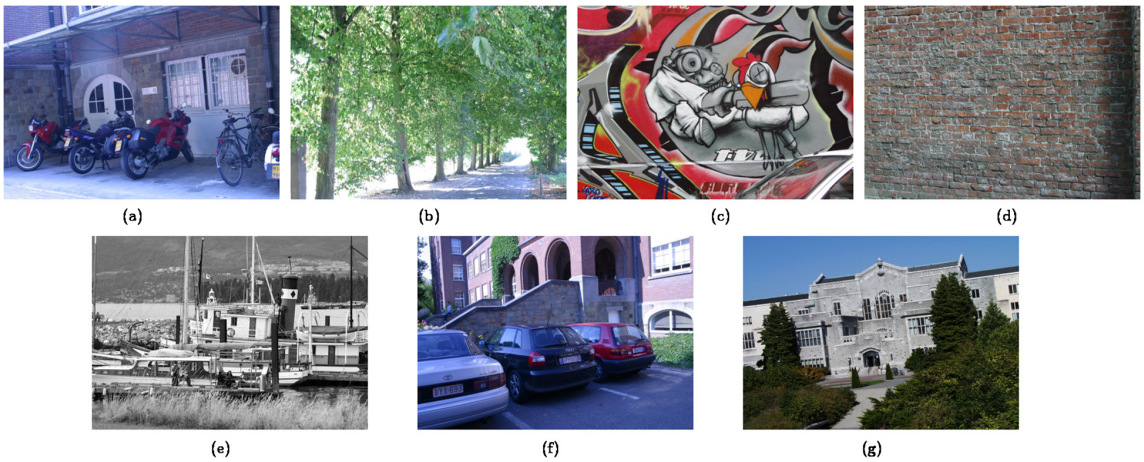

Fig. 3. Images from Oxford dataset: blur (a, b), viewpoint change (c, d), zoom and rotation (e), lighting change (f) and JPEG compression (g).

In order to evaluate if a match is correct, the ratio of the intersection over union [13] is applied. In short, a match is considered correct if the overlap error in the image area covered by two corresponding regions ϵ is less than 50% of the region union. For that purpose, the parameter ϵ is calculated as:

$$\epsilon = 1 - \frac{P \cap H^T \cdot Q \cdot H}{P \cup H^T \cdot Q \cdot H}, \tag{28}$$

where P and Q are the regions centered on the source point p and the target point matched q, and H is the homography that relates source and target images.

6.1. Evaluation methodology

In order to test our proposal, the following methodology has been applied. First, keypoints have been selected using a keypoint detector (both SURF and SIFT have been tested). Afterwards, the five keypoint descriptors SIFT, SURF, ORB, BRISK and FREAK have been applied to the detected keypoints. Using the same keypoint detector for all descriptors ensures that the results are comparable. The list of keypoints detected, their descriptors as well as the source code employed in this work are available as supplementary material of this paper for evaluation purposes.[2]

The precision, recall and F measure of each descriptor have been evaluated in the dataset, providing the individual baseline performance. These three measures are defined over the true positives (tp), the false positives (fp), and the false negatives (fn) matches as:

$$precision = \frac{tp}{tp + fp};\ recall = \frac{tp}{tp + fn};\ F = 2\frac{precision \cdot recall}{precision + recall}. \tag{29}$$

Precision is a measure indicating how good a method is at not selecting incorrect matches, while recall measures how many good matches a method is capable of finding. These two measures complement to each other. Finally, the F measure is the harmonic mean of both providing a unique score about the performance of a method. We will focus our analysis on the F measure since it is a unique indicator to determine the performance of a method.

In order to evaluate the performance of the different fusion methods, the non-parametric Wilcoxon signed rank test [48] has been applied on the F measure. The test evaluates two methods by considering two hypotheses: the null hypothesis H_0, which assumes that both methods present similar results, and the alternative hypothesis H_1, indicating that there exists statistically significant differences between the observed results. In all our tests, the value $p = 0.05$ has been employed, indicating that results of the test are reliable at 95.5%.

6.2. Baseline performance

At this point, each descriptor is evaluated individually so as to later analyze the improvements produced by fusion. The values presented in Tables 1 and 2, are the results of each descriptor obtained for the best value of α over the entire dataset using values in the range $[0, 1]$ at regular intervals of 0.01. In other words, each descriptor has been applied to all

[2] The supplementary material is available at http://www.uco.es/investiga/grupos/ava/node/48/.

Table 1
Baseline performance of individual keypoint descriptors (Kp) in the database when using the SURF keypoint detector. Lines in bold font are selected for comparison.

Kp-detector	Kp-descriptor	F measure
SURF	**SIFT-L1**	**0.721**
SURF	SIFT-L2	0.726
SURF	**SURF-L1**	**0.629**
SURF	SURF-L2	0.559
SURF	**ORB**	**0.570**
SURF	**BRISK**	**0.688**
SURF	**FREAK**	**0.755**

Table 2
Baseline performance of individual keypoint descriptors in the database when using the SIFT keypoint detector.

Kp-detector	Kp-descriptor	F measure
SIFT	SIFT-L1	0.451
SIFT	SIFT-L2	0.437
SIFT	SURF-L1	0.188
SIFT	SURF-L2	0.167
SIFT	ORB	0.369
SIFT	BRISK	0.377
SIFT	FREAK	0.354

the sequences for varying values of α, and the solution with the highest F measure is the one shown in the table. The first and second columns indicate the keypoint detector and descriptor employed, respectively. For SURF and SIFT descriptors, both the L1 and L2 norms have been employed. Since the other descriptors are binary, the Hamming distance has been employed.

First, it can be seen that the SURF keypoint detector provides better performance in all cases. Thus, this will be the one selected for the rest of the experiments. It can also be observed that, when using the SURF keypoint detector, the FREAK descriptor is the best one, while ORB is the worst one. Regarding the distance employed, it can be seen that *L2* causes a performance reduction in most cases. The options presented in bold font in Table 1 are the descriptors employed for the rest of our comparisons.

6.3. Standard feature fusion performance

The following section evaluates the performance of the standard feature fusion methods presented in Section 4. In total, we have tested two normalized similarity measures ($e = \{1, 2\}$ of Eq. (7)) and four fusing rules (sum, root-mean-square, product and max in Eqs. (8)–(11)). In total, the eight different fusion approaches have been tested, and for each one, all possible combinations of the five descriptors have been tested. The results are presented in Table 3, where the asterisks indicate the descriptors fused. Again, the results reported are those obtained for the best value of α over the entire dataset using values in the range $[0, 1]$ at regular intervals of 0.01.

For instance, line 3, column $(e = 1, max)$ presents the result of fusing all the descriptors but ORB using a linear mapping ($e = 1$) and the max fusion rule which obtain $(+0.30\mathbf{E}^{-2}, \mathbf{H}_0)$. In this case, the fusion method results in an increase of the F measure of $+0.30\mathbf{E}^{-2}$ compared to the best individual descriptor of the four employed, which in this case is FREAK. However, the value $\mathbf{H}_0$ means that the Wilcoxon test [48] indicates that the result is not statistically significant. In other words, the two methods can be considered equivalent in performance. The last line (26), summarizes the results of all the tests performed for a method. So, for the case $(e = 1, max)$, the result $(-1.92\mathbf{E}^{-2}, \mathbf{H}_1)$ indicates that, considering all possible combinations, the fusion method, in average, obtains a performance reduction and that it is a statistically significant result. So, it can be said that, overall, this fusion method is worst than using the best individual descriptor.

If we analyze the results for the rest of the fusion methods, it can be seen that no one obtains better results that the use of individual classifiers.

6.4. Evidential feature fusion performance

This section evaluates the performance of the proposed approach. As in the previous case, our method has been evaluated by selecting all possible combinations of the five descriptors. Our method has three parameters: n (the cardinal of $\phi_k(p, \mathcal{Q})$ in Eq. (5)), β from Eq. (23) and the value of s for the t-norm employed (Eq. (17)). The first parameter is the number of keypoints that are employed to create the evidence distribution. The values $n = (4, 3, 2)$ have been employed in our tests. The second parameter β is the factor that modulates the shape of the evidence distribution. Values in the range $\beta = [1, 15]$ at regular intervals of 0.25 have been tested. For the t-norm, the values $s = \{0, 0.5, 1\}$ have been employed. Finally, we have employed values of α in the range $[0, 1]$ at regular intervals of 0.1. The results obtained are presented in Table 4.

As can be observed, the proposed approach obtains better results than the methods tested previously. When using the t-norms $s = \{0, 1\}$ (Cautious and Conjunctive rules), our method always improves the results of the best individual classifier. In general, the conjunctive rule obtains slightly better results than the cautious one, presumably because information managed by the descriptors cannot be considered highly correlated. We would like to draw attention to line 3 and column $s = 1$, where our methods fuses all descriptors but ORB. In that case, we increase performance $(+7.17\mathbf{E}^{-2})$ with respect to FREAK, that scored 0.775. So, in relative terms, it is an improvement of nearly 10% using our approach. In general, the best results are obtained when the ORB keypoint descriptor is not employed.

6.4.1. Influence of the parameters in the method performance

The results reported previously have shown the performance of our method for the best configuration of the parameters n and β. Below, we analyze how the results of our method vary as these parameters change. First, the influence of n has

Table 3
Wilcoxon tests comparing the performance of eight standard fusion approaches presented in [12].

	SIFT	SURF	ORB	BRISK	FREAK	($e=1$, max)	($e=1$, product)	($e=1$, rms)	($e=1$, sum)	($e=2$, max)	($e=2$, product)	($e=2$, rms)	($e=2$, sum)
0	*	*	*	*	*	$(-0.76\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-15.71\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+0.42\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+1.41\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.21\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-13.68\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-0.04\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+1.07\mathbf{E}^{-2}, \mathbf{H}_0)$
1	*	*	*	*		$(+0.20\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-8.50\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+1.45\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.72\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.83\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-7.93\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+0.16\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.19\mathbf{E}^{-2}, \mathbf{H}_0)$
2	*	*	*		*	$(-1.63\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-14.80\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-0.34\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.28\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-2.04\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-13.24\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.10\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.17\mathbf{E}^{-2}, \mathbf{H}_0)$
3	*	*		*	*	$(+0.30\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-9.18\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+0.94\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+1.86\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.70\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-8.23\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+0.45\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+1.60\mathbf{E}^{-2}, \mathbf{H}_1)$
4	*		*	*	*	$(-1.46\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-12.08\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.00\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.60\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.15\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-10.52\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.48\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.45\mathbf{E}^{-2}, \mathbf{H}_0)$
5		*	*	*	*	$(-1.84\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-11.90\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.01\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.94\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-2.05\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-11.19\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.15\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.61\mathbf{E}^{-2}, \mathbf{H}_0)$
6	*	*	*			$(-1.60\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-7.71\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-0.89\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.24\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-2.45\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-7.74\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.64\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.33\mathbf{E}^{-2}, \mathbf{H}_0)$
7	*	*		*		$(+1.01\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-3.14\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+1.12\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+1.28\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.15\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-3.02\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+0.50\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.58\mathbf{E}^{-2}, \mathbf{H}_0)$
8	*	*			*	$(-1.12\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-7.94\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-0.90\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.14\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.48\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-7.05\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.28\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.04\mathbf{E}^{-2}, \mathbf{H}_0)$
9	*		*	*		$(-1.00\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-5.82\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-0.43\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.41\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-2.11\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-6.00\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.35\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.53\mathbf{E}^{-2}, \mathbf{H}_1)$
10	*		*		*	$(-3.06\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-10.42\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.57\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.85\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-3.53\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-9.73\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.06\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.65\mathbf{E}^{-2}, \mathbf{H}_0)$
11	*			*	*	$(-0.70\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-5.96\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.13\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.65\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-2.00\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-5.67\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.55\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.05\mathbf{E}^{-2}, \mathbf{H}_0)$
12		*	*	*		$(-2.92\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-9.36\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.25\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.82\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.17\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-8.50\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.84\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.23\mathbf{E}^{-2}, \mathbf{H}_1)$
13		*	*		*	$(-3.43\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-10.43\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.65\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.44\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-3.87\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-10.14\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.90\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.09\mathbf{E}^{-2}, \mathbf{H}_1)$
14		*		*	*	$(-1.27\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-6.50\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.24\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.49\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-2.24\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-6.33\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.93\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.37\mathbf{E}^{-2}, \mathbf{H}_0)$
15			*	*	*	$(-3.04\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-9.04\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.34\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.97\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-3.57\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-8.70\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.62\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.37\mathbf{E}^{-2}, \mathbf{H}_1)$
16	*	*				$(-1.01\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-2.38\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-0.97\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.67\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.25\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-2.66\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.35\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.25\mathbf{E}^{-2}, \mathbf{H}_0)$
17	*		*			$(-4.03\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-5.96\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.68\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.49\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.84\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-6.56\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.12\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.84\mathbf{E}^{-2}, \mathbf{H}_1)$
18	*			*		$(-0.63\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.21\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.70\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.06\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.52\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.91\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.35\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.33\mathbf{E}^{-2}, \mathbf{H}_0)$
19	*				*	$(-0.90\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.86\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-4.09\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-0.90\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.99\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.88\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-4.05\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-0.99\mathbf{E}^{-2}, \mathbf{H}_0)$
20		*	*			$(-0.60\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-2.65\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.25\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.60\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.19\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-3.07\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.56\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-1.19\mathbf{E}^{-2}, \mathbf{H}_0)$
21		*		*		$(-2.42\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.47\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.67\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.42\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.37\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.41\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.72\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.37\mathbf{E}^{-2}, \mathbf{H}_1)$
22		*			*	$(-4.07\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-5.59\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.44\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.07\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.69\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-6.13\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.87\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.69\mathbf{E}^{-2}, \mathbf{H}_1)$
23			*	*		$(-4.96\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-7.21\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.17\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.33\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-5.25\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-7.42\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-5.41\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-5.25\mathbf{E}^{-2}, \mathbf{H}_1)$
24			*		*	$(-5.75\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-7.79\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.86\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.36\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-6.32\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-8.22\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-5.95\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-5.67\mathbf{E}^{-2}, \mathbf{H}_1)$
25				*	*	$(-3.27\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.37\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.97\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.27\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.84\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.78\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-4.51\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-3.84\mathbf{E}^{-2}, \mathbf{H}_1)$
26	–	–	–	–	–	$(-1.92\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-7.31\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.68\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.01\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.52\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-7.03\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-2.37\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.69\mathbf{E}^{-2}, \mathbf{H}_1)$

Table 4
Wilcoxon tests comparing the performance of our fusion approach for different combination schemes.

	SIFT	SURF	ORB	BRISK	FREAK	t-norm, $s=0$	t-norm, $s=0.5$	t-norm, $s=1$
0	*	*	*	*	*	$(+3.79\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-0.27\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+5.78\mathbf{E}^{-2}, \mathbf{H}_1)$
1	*	*	*	*		$(+5.70\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+0.13\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+5.27\mathbf{E}^{-2}, \mathbf{H}_1)$
2	*	*	*		*	$(+3.58\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+1.51\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+5.73\mathbf{E}^{-2}, \mathbf{H}_1)$
3	*	*		*	*	$(+5.10\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+2.14\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+7.17\mathbf{E}^{-2}, \mathbf{H}_1)$
4	*		*	*	*	$(+3.78\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+1.22\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+4.66\mathbf{E}^{-2}, \mathbf{H}_1)$
5		*	*	*	*	$(+3.11\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.62\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+3.83\mathbf{E}^{-2}, \mathbf{H}_1)$
6	*	*	*			$(+4.42\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+2.12\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+5.12\mathbf{E}^{-2}, \mathbf{H}_1)$
7	*	*		*		$(+7.16\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+3.28\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+7.11\mathbf{E}^{-2}, \mathbf{H}_1)$
8	*	*			*	$(+4.79\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+5.04\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+7.00\mathbf{E}^{-2}, \mathbf{H}_1)$
9	*		*	*		$(+5.15\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+1.38\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+5.08\mathbf{E}^{-2}, \mathbf{H}_1)$
10	*		*		*	$(+2.47\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+2.10\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+4.37\mathbf{E}^{-2}, \mathbf{H}_1)$
11	*			*	*	$(+4.88\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+4.03\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+6.10\mathbf{E}^{-2}, \mathbf{H}_1)$
12		*	*	*		$(+1.15\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.97\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+1.70\mathbf{E}^{-2}, \mathbf{H}_1)$
13		*	*		*	$(+2.41\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+0.25\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+3.47\mathbf{E}^{-2}, \mathbf{H}_1)$
14		*		*	*	$(+4.42\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-0.16\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+4.92\mathbf{E}^{-2}, \mathbf{H}_1)$
15			*	*	*	$(+2.67\mathbf{E}^{-2}, \mathbf{H}_1)$	$(-1.95\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+2.67\mathbf{E}^{-2}, \mathbf{H}_1)$
16	*	*				$(+6.14\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+5.92\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+6.54\mathbf{E}^{-2}, \mathbf{H}_1)$
17	*		*			$(+2.00\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+2.66\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+2.61\mathbf{E}^{-2}, \mathbf{H}_1)$
18	*			*		$(+6.67\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+7.07\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+7.19\mathbf{E}^{-2}, \mathbf{H}_1)$
19	*				*	$(+2.75\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+3.64\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+3.70\mathbf{E}^{-2}, \mathbf{H}_1)$
20		*	*			$(+3.32\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+3.29\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+3.69\mathbf{E}^{-2}, \mathbf{H}_1)$
21		*		*		$(+2.18\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+1.25\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+2.51\mathbf{E}^{-2}, \mathbf{H}_1)$
22		*			*	$(+3.36\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+3.50\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+4.09\mathbf{E}^{-2}, \mathbf{H}_1)$
23			*	*		$(+0.29\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.89\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.67\mathbf{E}^{-2}, \mathbf{H}_0)$
24			*		*	$(+1.89\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+0.47\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+1.90\mathbf{E}^{-2}, \mathbf{H}_1)$
25				*	*	$(+3.64\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+1.99\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+3.25\mathbf{E}^{-2}, \mathbf{H}_1)$
26	–	–	–	–	–	$(+3.72\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+1.81\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+4.47\mathbf{E}^{-2}, \mathbf{H}_1)$

been evaluated in each case. For that purpose, we have run a Wilcoxon test comparing the results of the best configuration of β for the values of $n = (2, 3, 4)$. The results are in Table 5. While each row of the table shows one fusion combination, the last three columns show the result of the Wilcoxon test comparing the best configuration in terms of the F measure. For instance, column "2 vs 3" of line 1 compares the best matcher found using $n = 2$ with the best matcher using $n = 3$ when our method is applied fusing all descriptors but FREAK. The Wilcoxon test indicates that while the approach using $n = 2$ vs $n = 3$ obtains an increment of 0.12% in terms of the F measure, the differences observed are not statistically significant. Analyzing the rest of the tests, it can be observed that there is no clear winner. So, we might conclude that the impact of the parameter n is not critical in the performance of the method. We believe it is because if the correct match is not among the first two, then, the descriptor is not reliable for that particular keypoint. In other words, it is unlikely for a descriptor to classify the correct match in the third or fourth position.

In order to analyze the behavior of our method as a function of β, we show the evolution of the F measure for different fusion combinations in Fig. 4 for a constant value of n and α. We have selected only a set of combinations since for the rest of cases the behavior is very similar. As can be observed, a smooth function is obtained. The values of the F measure increase slowly until reaching its maximum and then the function decreases smoothly.

At the light of the results presented, it can be seen that the results of the proposed method are robust, i.e., they do not strongly depend on the parameter selection.

7. Conclusions

This work has proposed a new keypoint matching approach. Our method fuses two of more keypoint descriptors creating an evidence distribution. Then, an analysis of the entropy of the distribution is employed to calculate a confidence factor. The confidence factor is an indicator of how good a descriptor is at estimating the correct match of a keypoint. DST is employed to fuse the evidence distributions, and finally the pignistic probability is applied to select the most likely match.

The proposed method has been tested on the Oxford keypoint dataset [2] running statistical tests on the results. The tests conducted show that the proposed method obtains a performance improvement (which is statistically relevant) in the majority of the cases. Also, we have observed that the better the descriptors employed for fusion are, the higher the improvement obtained. In the best case, our method has obtained an improvement of 10% with respect to the best keypoint descriptor FREAK. Finally, our experiments have also shown that the performance of the proposed method is robust to a wide range of values of its parameters.

Table 5
Wilcoxon tests comparing the performance of our fusion approach using different values of the parameter n. Results indicate that the values of n produce differences in performance that are not statistically significant (null hypothesis H_0 holds).

	SIFT	SURF	ORB	BRISK	FREAK	2 vs 3	2 vs 4	3 vs 4
0	*	*	*	*	*	$(-0.38\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.49\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.12\mathbf{E}^{-2}, \mathbf{H}_0)$
1	*	*	*	*		$(+0.12\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.13\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.01\mathbf{E}^{-2}, \mathbf{H}_0)$
2	*	*	*		*	$(-0.46\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.29\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.16\mathbf{E}^{-2}, \mathbf{H}_0)$
3	*	*		*	*	$(-0.44\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.12\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.32\mathbf{E}^{-2}, \mathbf{H}_0)$
4	*		*	*	*	$(-0.04\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.37\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.33\mathbf{E}^{-2}, \mathbf{H}_0)$
5		*	*	*	*	$(-0.08\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.19\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.26\mathbf{E}^{-2}, \mathbf{H}_0)$
6	*	*	*			$(+0.01\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.03\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.04\mathbf{E}^{-2}, \mathbf{H}_0)$
7	*	*		*		$(-0.61\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.60\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.01\mathbf{E}^{-2}, \mathbf{H}_0)$
8	*	*			*	$(+0.76\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.87\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.11\mathbf{E}^{-2}, \mathbf{H}_0)$
9	*		*	*		$(+0.16\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.04\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.21\mathbf{E}^{-2}, \mathbf{H}_0)$
10	*		*		*	$(+0.15\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.17\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.01\mathbf{E}^{-2}, \mathbf{H}_0)$
11	*			*	*	$(-0.07\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.77\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.70\mathbf{E}^{-2}, \mathbf{H}_0)$
12		*	*	*		$(+0.14\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.05\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.09\mathbf{E}^{-2}, \mathbf{H}_0)$
13		*	*		*	$(-0.20\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.16\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.03\mathbf{E}^{-2}, \mathbf{H}_0)$
14		*		*	*	$(-0.63\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.34\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.29\mathbf{E}^{-2}, \mathbf{H}_0)$
15			*	*	*	$(+0.00\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.10\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.10\mathbf{E}^{-2}, \mathbf{H}_0)$
16	*	*				$(-0.59\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.04\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.64\mathbf{E}^{-2}, \mathbf{H}_0)$
17	*		*			$(+0.32\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+1.06\mathbf{E}^{-2}, \mathbf{H}_1)$	$(+0.75\mathbf{E}^{-2}, \mathbf{H}_0)$
18	*			*		$(+0.81\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.89\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.07\mathbf{E}^{-2}, \mathbf{H}_0)$
19	*				*	$(+0.39\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.64\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.25\mathbf{E}^{-2}, \mathbf{H}_0)$
20		*	*			$(-0.01\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.23\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.22\mathbf{E}^{-2}, \mathbf{H}_0)$
21		*		*		$(-0.44\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.79\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.34\mathbf{E}^{-2}, \mathbf{H}_0)$
22		*			*	$(-0.14\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.02\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.16\mathbf{E}^{-2}, \mathbf{H}_0)$
23			*	*		$(-0.17\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.46\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.28\mathbf{E}^{-2}, \mathbf{H}_0)$
24			*		*	$(+0.01\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.4\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.42\mathbf{E}^{-2}, \mathbf{H}_0)$
25				*	*	$(-0.20\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.21\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.02\mathbf{E}^{-2}, \mathbf{H}_0)$
26	–	–	–	–	–	$(-0.06\mathbf{E}^{-2}, \mathbf{H}_0)$	$(-0.05\mathbf{E}^{-2}, \mathbf{H}_0)$	$(+0.01\mathbf{E}^{-2}, \mathbf{H}_0)$

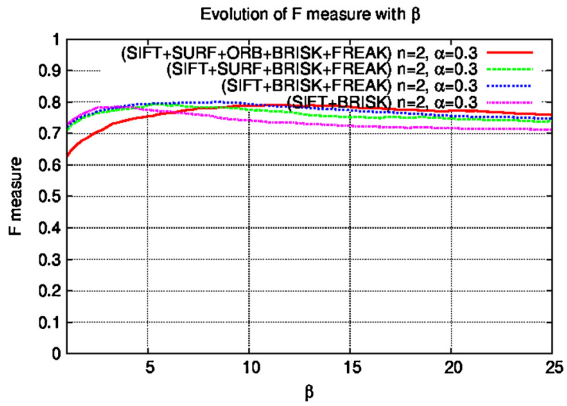

Fig. 4. Evolution of the F measure as the smoothing parameter β changes (Eq. (22)). As can be observed, the performance of our method is very regular for a wide range of values of β.

Acknowledgements

This work was partially supported by the Research Projects TIN2012-32952 and BROCA, both financed by the Spanish Ministry of Science and Technology and FEDER.

References

[1] G. Shafer, A Mathematical Theory of Evidence, Princeton University Press, 1976.
[2] C. Schmid, K. Mikolajczyk, A performance evaluation of local descriptors, in: 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Proceedings, Madison, Wisconsin, 2003, pp. 257–263.

[3] C. Schmid, R. Mohr, Local grayvalue invariants for image retrieval, IEEE Trans. Pattern Anal. Mach. Intell. 19 (5) (1997) 530–535, http://dx.doi.org/10.1109/34.589215.
[4] M. Brown, D.G. Lowe, Automatic panoramic image stitching using invariant features, Int. J. Comput. Vis. 74 (1) (2007) 59–73, http://dx.doi.org/10.1007/s11263-006-0002-3.
[5] D.G. Lowe, Distinctive image features from scale-invariant keypoints, Int. J. Comput. Vis. 60 (2) (2004) 91–110, http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94.
[6] D. Scharstein, R. Szeliski, A taxonomy and evaluation of dense two-frame stereo correspondence algorithms, Int. J. Comput. Vis. 47 (1–3) (2002) 7–42, http://dx.doi.org/10.1023/A:1014573219977.
[7] J. Sun, N.-N. Zheng, H.-Y. Shum, Stereo matching using belief propagation, IEEE Trans. Pattern Anal. Mach. Intell. 25 (7) (2003) 787–800, http://dx.doi.org/10.1109/TPAMI.2003.1206509.
[8] H. Bay, T. Tuytelaars, L. Van Gool, SURF: speeded up robust features, in: Computer Vision – ECCV 2006, Graz, Austria, in: Lect. Notes Comput. Sci., vol. 3951, Springer, Berlin, Heidelberg, 2006, pp. 404–417.
[9] E. Rublee, V. Rabaud, K. Konolige, G. Bradski, ORB: an efficient alternative to SIFT or SURF, in: 2011 IEEE International Conference on Computer Vision (ICCV), Barcelona, Spain, 2011, pp. 2564–2571.
[10] S. Leutenegger, M. Chli, R.Y. Siegwart, BRISK: binary robust invariant scalable keypoints, in: 2011 IEEE International Conference on Computer Vision (ICCV), Barcelona, Spain, 2011, pp. 2548–2555.
[11] A. Alahi, R. Ortiz, P. Vandergheynst, FREAK: fast retina keypoint, in: 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Rhode Island, Providence, USA, 2012, pp. 510–517.
[12] P. Perakis, T. Theoharis, I.A. Kakadiaris, Feature fusion for facial landmark detection, Pattern Recognit. 47 (9) (2014) 2783–2793, http://dx.doi.org/10.1016/j.patcog.2014.03.007.
[13] K. Mikolajczyk, C. Schmid, A performance evaluation of local descriptors, IEEE Trans. Pattern Anal. Mach. Intell. 27 (10) (2005) 1615–1630, http://dx.doi.org/10.1109/TPAMI.2005.188.
[14] E. Rosten, T. Drummond, Machine learning for high-speed corner detection, in: Computer Vision – ECCV 2006, Graz, Austria, in: Lect. Notes Comput. Sci., vol. 3951, Springer, Berlin, Heidelberg, 2006, pp. 430–443.
[15] M. Calonder, V. Lepetit, C. Strecha, P. Fua, BRIEF: binary robust independent elementary features, in: Computer Vision – ECCV 2010, Crete, Greece, in: Lect. Notes Comput. Sci., vol. 6314, Springer, 2010, pp. 778–792.
[16] D. He, S. Liang, Y. Fang, A multi-descriptor, multi-nearest neighbor approach for image classification, in: Advanced Intelligent Computing Theories and Applications, International Conference on Intelligent Computing, ICIC 2010, Changsha, China, in: Lect. Notes Comput. Sci., vol. 6215, Springer, Berlin, Heidelberg, 2010, pp. 515–523.
[17] S. Bakshi, H. Mehrotra, R. Raman, P.K. Sa, Score level fusion of SIFT and SURF for IRIS, in: 2012 International Conference on Devices, Circuits and Systems (ICDCS), Coimbatore, India, IEEE, 2012, pp. 527–531.
[18] P. Mountney, B. Lo, S. Thiemjarus, D. Stoyanov, G. Zhong-Yang, A probabilistic framework for tracking deformable soft tissue in minimally invasive surgery, in: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2007, Brisbane, Australia, in: Lect. Notes Comput. Sci., vol. 4792, Springer, Berlin, Heidelberg, 2007, pp. 34–41.
[19] R. Weng, J. Lu, J. Hu, G. Yang, Y.-P. Tan, Robust feature set matching for partial face recognition, in: 2013 IEEE International Conference on Computer Vision (ICCV), Sydney, Australia, 2013, pp. 601–608.
[20] E.P. Xing, A.Y. Ng, M.I. Jordan, S. Russell, Distance metric learning with application to clustering with side-information, Adv. Neural Inf. Process. Syst. (2002) 505–512.
[21] P. Smets, The combination of evidence in the transferable belief model, IEEE Trans. Pattern Anal. Mach. Intell. 12 (1990) 447–458, http://dx.doi.org/10.1109/34.55104.
[22] R. Munoz-Salinas, R. Medina-Carnicer, F. Madrid-Cuevas, A. Carmona-Poyato, Multi-camera people tracking using evidential filters, Int. J. Approx. Reason. 50 (2009) 732–749, http://dx.doi.org/10.1016/j.ijar.2009.02.001.
[23] S. Panigrahi, A. Kundu, S. Sural, A.K. Majumdar, Use of Dempster–Shafer theory and Bayesian inferencing for fraud detection in mobile communication networks, in: Information Security and Privacy, Townsville, Australia, in: Lect. Notes Comput. Sci., vol. 4586, Springer, Berlin, Heidelberg, 2007, pp. 446–460.
[24] T. Denoeux, P. Smets, Classification using belief functions: the relationship between the case-based and model-based approaches, IEEE Trans. Syst. Man Cybern., Part B, Cybern. 36 (2006) 1395–1406, http://dx.doi.org/10.1109/TSMCB.2006.877795.
[25] S. Demotier, W. Schon, T. Denoeux, Risk assessment based on weak information using belief functions: a case study in water treatment, IEEE Trans. Syst. Man Cybern., Part C, Appl. Rev. 36 (2006) 382–396, http://dx.doi.org/10.1109/TSMCC.2004.840057.
[26] T. Denoeux, M.-H. Masson, EVCLUS: evidential clustering of proximity data, IEEE Trans. Syst. Man Cybern., Part B, Cybern. 34 (2004) 95–109, http://dx.doi.org/10.1109/TSMCB.2002.806496.
[27] M.-H. Masson, T. Denoeux, ECM: an evidential version of the fuzzy c-means algorithm, Pattern Recognit. 41 (4) (2008) 1384–1397, http://dx.doi.org/10.1016/j.patcog.2007.08.014.
[28] A.-O. Boudraa, A. Bentabet, F. Salzenstein, Dempster–Shafer's basic probability assignment based on fuzzy membership functions, Electron. Lett. Comput. Vis. Image Anal. 4 (2004) 1–10, http://dx.doi.org/10.5565/rev/elcvia.68.
[29] I. Bloch, Defining belief functions using mathematical morphology – application to image fusion under imprecision, Int. J. Approx. Reason. 48 (2008) 437–465, http://dx.doi.org/10.1016/j.ijar.2007.07.008.
[30] Z. Hammal, L. Couvreur, A. Caplier, M. Rombaut, Facial expression classification: an approach based on the fusion of facial deformations using the transferable belief model, Int. J. Approx. Reason. 46 (2007) 542–567, http://dx.doi.org/10.1016/j.ijar.2007.02.003.
[31] W. Pieczynski, Multisensor triplet Markov chains and theory of evidence, Int. J. Approx. Reason. 45 (2007) 1–16, http://dx.doi.org/10.1016/j.ijar.2006.05.001.
[32] Z. Yi, H. Khing, C. Seng, Z. Wei, Multi-ultrasonic sensor fusion for autonomous mobile robots, in: Architectures, Algorithms and Applications IV, Orlando, Florida, in: SPIE Proceedings Series, 2000, pp. 314–321.
[33] H. Wu, M. Siegel, R. Stiefelhagen, J. Yang, Sensor fusion using Dempster–Shafer theory, in: Proceedings of the 19th IEEE Instrumentation and Measurement Technology Conference, IMTC/2002, vol. 1, Anchorage, AK, USA, 2002, pp. 7–12.
[34] N. Milisavljevic, I. Bloch, S. Broek, M. Acheroy, Improving mine recognition through processing and Dempster–Shafer fusion of ground-penetrating radar data, Pattern Recognit. 36 (2003) 1233–1250, http://dx.doi.org/10.1016/S0031-3203(02)00251-0.
[35] P. Xu, F. Davoine, J.-B. Bordes, H. Zhao, T. Denoeux, Multimodal information fusion for urban scene understanding, Mach. Vis. Appl. (2014) 1–19, http://dx.doi.org/10.1007/s00138-014-0649-7.
[36] T. Denoeux, N. El Zoghby, V. Cherfaoui, A. Jouglet, Optimal object association in the Dempster–Shafer framework, IEEE Trans. Cybern. 44 (12) (2014) 2521–2531, http://dx.doi.org/10.1109/TCYB.2014.2309632.
[37] V.-N. Huynh, T.T. Nguyen, C.A. Le, Adaptively entropy-based weighting classifiers in combination using Dempster–Shafer theory for word sense disambiguation, Comput. Speech Lang. 24 (3) (2010) 461–473, http://dx.doi.org/10.1016/j.csl.2009.06.003.

[38] S. Ranoeliarivao, F. de Morsier, D. Tuia, S. Rakotoniaina, M. Borgeaud, J.-P. Thiran, S. Rakotondraompiana, Multisource clustering of remote sensing images with entropy-based Dempster–Shafer fusion, in: 2013 Proceedings of the 21st European Signal Processing Conference (EUSIPCO), Marrakech, Morocco, 2013, pp. 1–5.
[39] D. Pan, X. Lu, J. Liu, Y. Deng, A ranking procedure by incomplete pairwise comparisons using information entropy and Dempster–Shafer evidence theory, The Scientific World Journal (2014), http://dx.doi.org/10.1155/2014/904596.
[40] T. Denoeux, Conjunctive and disjunctive combination of belief functions induced by nondistinct bodies of evidence, Artif. Intell. 172 (2–3) (2008) 234–264, http://dx.doi.org/10.1016/j.artint.2007.05.008.
[41] R. Kennes, P. Smets, Computational aspects of the Mobius transformation, in: Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence, UAI '90, Elsevier Science Inc., New York, NY, USA, 1991, pp. 401–416.
[42] B. Quost, M.-H. Masson, T. Denoeux, Classifier fusion in the Dempster–Shafer framework using optimized t-norm based combination rules, Int. J. Approx. Reason. 52 (3) (2011) 353–374, http://dx.doi.org/10.1016/j.ijar.2010.11.008.
[43] T. Denoeux, The cautious rule of combination for belief functions and some extensions, in: 9th International Conference on Information Fusion, Florence, Italy, 2006, pp. 1–8.
[44] P. Smets, Belief functions: the disjunctive rule of combination and the generalized Bayesian theorem, Int. J. Approx. Reason. 9 (1993) 1–35, http://dx.doi.org/10.1016/0888-613X(93)90005-X.
[45] P. Smets, R. Kennes, The transferable belief model, Artif. Intell. 66 (2) (1994) 191–243, http://dx.doi.org/10.1016/0004-3702(94)90026-4.
[46] F. Cuzzolin, The intersection probability and its properties, in: Symbolic and Quantitative Approaches to Reasoning with Uncertainty, Verona, Italy, in: Lect. Notes Comput. Sci., vol. 5590, Springer, Berlin, Heidelberg, 2009, pp. 287–298.
[47] F. Cuzzolin, Geometry of relative plausibility and relative belief of singletons, Ann. Math. Artif. Intell. 59 (2010) 47–79, http://dx.doi.org/10.1007/s10472-010-9186-x.
[48] F. Wilcoxon, Individual comparisons by ranking methods, Biom. Bull. 1 (6) (1945) 80–83.

Capítulo B

Robust detection of fiducial markers in challenging conditions

En esta sección se incluye el artículo correspondiente a la segunda contribución realizada durante la presente Tesis Doctoral y que se corresponde con la propuesta descrita en el Capítulo 4.

El trabajo titulado "*Robust detection of fiducial markers in challenging conditions*" se encuentra en proceso de revisión.

Robust detection of fiducial markers in challenging conditions

Víctor Manuel Mondéjar-Guerra i72moguv@uco.es
Sergio Garrido-Jurado i52gajus@uco.es
Rafael Muñoz-Salinas in1musar@uco.es
Manuel Jesús Marín-Jiménez mjmarin@uco.es
Computing and Numerical Analysis Department,
Edificio C3. Campus de Rabanales, Córdoba University, 14071, Córdoba, Spain

Abstract

Square fiducial markers are one of the most popular tools for camera pose estimation because of their high robustness and performance. However, the state-of-the-art methods perform poorly under difficult image conditions, such as camera defocus, motion blur, small scale or non-uniform lighting. This work tackles the marker identification problem as a classification one, and proposes a methodology to train such classifiers by creating a synthetic dataset of markers affected by several transformations. Different types of classifiers have been tested to prove the validity of our proposal (namely, Multilayer Perceptron (MLP), Convolutional Neural Networks (CNN) and Support Vector Machines (SVM)), and statistical analyses have been performed in order to determine the best approach for our problem. Finally, the obtained classifiers have been compared to the ArUco and AprilTags fiducial marker systems in challenging video sequences. The results obtained show that the proposed method performs significantly better.

1. Introduction

Camera pose estimation is an important requirement in many computer vision applications. Despite the growing interest in natural features (such as keypoints or textures) for that purpose, fiducial markers are still one of the preferred alternatives since their detection is faster and more robust, providing camera pose at the only expense of placing some markers in the scene. They are actively used in many applications such as augmented reality (Khattak et al., 2014; Cubillo et al., 2014), robot navigation (Sanchez-Lopez et al., 2015; Olivares-Mendez et al., 2015) or structure from motion (Rumpler et al., 2014; Klopschitz & Schmalstieg, 2007).

Square fiducial markers, composed by an external wide black border and an inner code (most ofter binary), are the most popular ones. Their main advantage is that a single marker provides four correspondence points (its four corners), which are enough to do camera pose estimation. Detection of such markers is normally composed by two steps. The first one consists in looking for square borders, which produces a set of candidates comprised by markers and background elements (see Fig. 1). In the second step, each candidate is analysed to extract its binary code and deciding whether it is a marker or part of the background. Two main approaches have been applied for extracting the binary code. The first one (see Fig. 2(a)) consists in obtaining the canonical view of the candidate which is then thresholded (Garrido-Jurado et al., 2014; Wagner & Schmalstieg, 2007). The

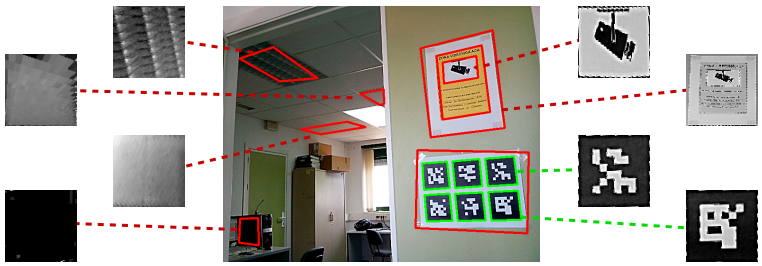

Figure 1: Example of square candidates detection in a real scene. Green candidates refer to real markers while red ones are elements of the background. (Best seen in color)

second one (see Fig. 2(b)) uses the inverse homography to determine image locations where to perform the analysis (Olson, 2011; Fiala, 2010). Finally, if the binary code does no belong to the set of valid markers (marker dictionary), the candidate is rejected.

Selecting the binary marker codes for an application is of great relevance to reduce the chance of errors. Some authors have employed classic signal coding techniques (Rekimoto, 1998; Fiala, 2010; Wagner & Schmalstieg, 2007), others heuristic approaches (Garrido-Jurado et al., 2014; Olson, 2011) and even Mixed Integer Linear Programming (MILP) (Garrido-Jurado et al., 2016) have been used to obtain optimal solutions.

Nonetheless, in many real-life applications, images undergo circumstances where a correct identification using the previously explained methods is not reliable, such as motion blur, non-uniform lighting conditions or reduced visibility (see Fig. 5).

This work proposes to handle the marker identification problem as a classification one. Instead of binarizing the image and then applying bit detection/correction techniques, we propose to train a classifier for the identification process. Since obtaining a large dataset of markers from real images representing a wide variety of conditions is not practical, we propose the creation of a synthetic training dataset by applying different transformations that emulate those happening in real situations. Additionally, since it is not known a priori the best machine learning approach for this particular problem, this paper investigates the use of MLP, CNN, and SVM, and statistical tests are run to determine the best one. The trained classifiers are validated in real scenes and compared to the state-of-the-art fiducial marker systems ArUco (Garrido-Jurado et al., 2014) and AprilTags (Olson, 2011), which guarantee the largest dictionary distances, allowing better error correction capabilities than alternatives like ArToolKitPlus (Wagner & Schmalstieg, 2007) and ARTag (Fiala, 2010).

The work of Kim and Petriu (2010) is the only one, up to our knowledge, in which a classifier is employed for a similar task. However, the differences with our work are relevant. They assumed that all candidates detected are valid markers (i.e., no background), thus limiting the application in realistic scenarios such as this shown in Fig. 1. In addition, they employ letters as identifiers, and a neural network is trained to recognize them under different image transformations. However, instead of feeding the classifier with the raw data, they first calculate twelve features from the letters which are the input to the classifier. In contrast, our approach makes no assumptions about the marker patterns, which can be

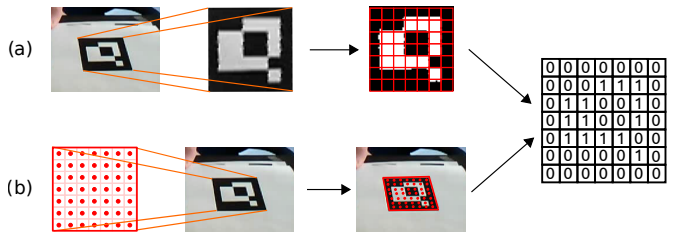

Figure 2: Bit extraction approaches. (a) First, perspective transformation is removed from the marker to obtain its canonical view. The resulting image is binarized and divided in cells. The number of black and white pixels on each cell are counted to determine the binary value of each bit. (b) The inverse perspective transformation of the maker is calculated and employed to obtain the position of the cell centers in the input image. The pixel values in the original image determine the value of each bit.

binary codes, letters or images. Additionally, we are testing multiple types of classifiers to select the best approach.

The remainder of this paper is structured as follows. Section 2 gives a brief explanation of the machine learning methods employed at this work. Section 3 formulates the problem and how the training set has been designed. Section 4 presents the experimentation carried out and Section 5 draws some conclusions.

2. Machine learning algorithms

The goal of machine learning is to turn data into information. Since this work deals with a classification problem, the explanation of the algorithms assumes a supervised learning view.

2.1 Support Vector Machines

SVM are maximum margin classifiers that appeared as a consequence of the research on *structural risk minimisation* (Cortes & Vapnik, 1995). They map input vectors to a higher dimensional space where a maximal separating hyperplane is constructed. Suppose that we are given with the training data $(x_1, y_1), \ldots, (x_m, y_m) \in \mathcal{X} \times \{\pm 1\}$, where $x_i \in \mathcal{R}^d$ represent the patterns and y_i the labels.

Let us define a kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathcal{R}$ that measures the similarity of two patterns and a mapping function $\Phi : \mathcal{X} \to \mathcal{H}$ that translates the patterns to a higher dimensional feature space. Then, the kernel is defined as $k(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle$. The problem of finding the optimal hyperplane $\mathbf{w}$ that separates the classes is solved by the following minimisation problem:

$$\frac{1}{2} \|\mathbf{w}\|^2 \text{ subject to } y_i(\langle \mathbf{w}, \Phi(x_i) \rangle + b) \geq 1, \ i = 1, \ldots, m.$$

Making use of the Lagrangian dual, the optimisation problem can be transformed into:

$$L(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j k(x_i, x_j), \text{ subject to } 0 \leq \alpha_i \leq C, \ i = 1, \ldots, m, \sum_{i=1}^{m} \alpha_i y_i = 0,$$

where α_i are the Lagrangian multipliers and C is its upper bound. The optimal decision function is then expressed as:

$$f(x) = sgn \left(\sum_{i=1}^{m} y_i \alpha_i k(x, x_i) + b \right),$$

where *sgn* is the *signum* function of a real number. Although SVM are binary classifiers, multiclass SVM can be employed using a one-versus-one approach, in which a classifier is created for each pair of classes and the most voted one is considered the winner.

2.2 Multilayer Perceptron

The first neural networks have its origins in attempts to find mathematical representations of information processes in biological systems (McCulloch & Pitts, 1943). The most successful model of this type in the context of pattern recognition is the feed-forward neural network, also known as the MLP (Bishop, 2006). For many applications, the resulting model can be significantly more compact, and hence faster to evaluate, than a SVM having the same generalization performance. The price to be paid for this compactness, as with the relevance vector machine, is that the likelihood function, which forms the basis for network training, is no longer a convex function of the model parameters. In practice, however, it is often worth investing substantial computational resources during the training phase in order to obtain a compact model that is fast at processing new data.

The MLP consists of multiple layers of nodes, with each layer fully connected to the next one, i.e., each node has a connection to every node from the next layer. The connection between two nodes are known as weights, w, and its values are adjusted during the training phase. The first layer is known as the input layer, which feed the network with the input data. Next, there is at least one hidden layer. The number of nodes in each layer can be different. Last, the final decision of the network is given by the output layer.

Let be the two-layer network of the Fig. 3, with inputs of dimension D, $x_1, ..., x_D$, a single hidden layer with M nodes and outputs of size K, $y_1, ..., y_K$.

A linear combination is constructed in each node from the hidden layer as:

$$a_j = \sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)},$$

where $j = 1, ..., M$, the superscript (1) indicates that the corresponding parameters are in the first layer of the network. The parameters $w_{ji}^{(1)}$ are the *weights* from the input layer to the first hidden layer and $w_{j0}^{(1)}$ are known as *biases*. The quantities a_j are the activations of each node, which are then transformed using a nonlinear activation function $f(\cdot)$:

$$z_j = f(a_j).$$

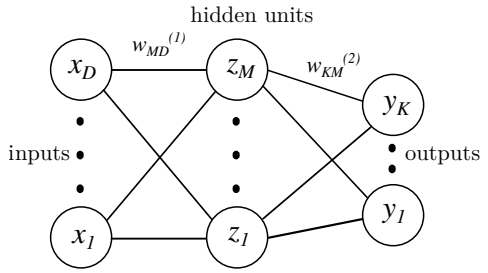

Figure 3: Network diagram for the two-layer neural network.

These quantities correspond to the *hidden units*. The Rectified Linear Unit (ReLU) is an example of nonlinear activation function which is defined as $f(x) = \max(0, x)$. For standard regression problems, the activation function is the *identity* $f(x) = x$. The values of z_j are again linearly combined in the next layers to give *output unit activations*:

$$a_k = \sum_{j=1}^{M} w_{kj}^{(2)} z_j + w_{k0}^{(2)},$$

where $k = 1, ..., K$, and K is the total number of outputs. This transformation correspond to the second layer of the network. Finally, the output unit activations of the last layer are transformed using an appropriate loss function, σ, to give a set of network outputs y_k. These various stages can be combined to give the overall network function:

$$y_k(x, w) = \sigma(\sum_{j=1}^{M} w_{kj}^{(2)} f(\sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}) + w_{k0}^{(2)}).$$

Thus the neural network model is a nonlinear function from a set of input variables x_i to a set of output variables y_k controlled by a vector w of adjustable parameters.

For multiclass problems, a softmax loss function can be used. The softmax function is a generalization of the logistic function to handle multiple classes. This function take a K-dimensional vector, being K the number of classes, and return a K-dimensional vector of real values $0 \leq y_k \leq 1$that add up to 1, $\sum_k y_k = 1$. The function is given by:

$$y_k(x, w) = \frac{e^{a_k(x,w)}}{\sum_j e^{a_j(x,w)}},$$

The common way to train a neural network is through the backpropagation algorithm (Rumelhart et al., 1986). The backpropagation is a supervised training algorithm that is divided in two steps. First, in the forward step, each neuron of the network generates its output and propagates it thought the next layer. Given a training set comprising a set of input vectors x_n, where $n = 1, ..., N$, together with a corresponding set of target vectors t_n, the backpropagation tries to minimize the error function. Considering a multiclass problem the error function has the form:

$$E(w) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{kn} \ln y_k(x_n, w) \tag{1}$$

Then, the error is used to update the weights of each neuron in a step known as backward,

$$w^{(\tau+1)} = w^{\tau} + \Delta w^{\tau},$$

where τ labels the iteration step. Different algorithms involve different choices for the weight vector update $\Delta w(\tau)$.

The backpropagation algorithm recursively alternates between forward and backward until the outpute error be lower than a fixed threshold.

2.3 Convolutional Neural Networks

CNN are variations of MLP designed to use minimal amounts of preprocessing. Although the first CNN date back decades (LeCun et al., 1989), they have recently shown an explosive popularity due to the use of modern GPUs, which allow to process a huge amount of data in a reasonable time. They have proved their success in image classification tasks (Krizhevsky et al., 2012), object detection (Ouyang et al., 2014), face recognition (Sun et al., 2014) and super-resolution (Dong et al., 2016) among others. In general, CNN has been widely applied to image data, this is mainly due to the fact that CNNs exploits the strong spatially-local correlation present in images.

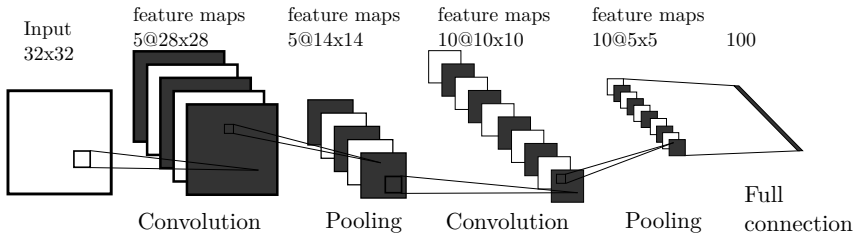

Figure 4: Example architecture of a convolutional neural network.

Let consider the example architecture of Fig. 4 for the explanation. First, as its name suggests, the convolutional layer performs a convolution with the input data and a set of filters generating the feature maps. The first convolution layer from the example architecture uses 5 filters with kernel size 5×5. The parameters of these filters are learned during the training phase and the same filters are convolved across the entire input data. So, the parameters from the learned filters are shared. This reduces the numbers of parameters required to train the network compared to MLP when the input data is large enough. The pooling layers perform a subsampling on the data. Max-pooling or avg-pooling can be applied. The example architecture uses kernel size 2×2 at pooling layers, reducing by half the feature maps.

In practise, there usually are several pairs of convolution and pooling layers. At each stage there is a larger degree of invariance to input transformations compared to the previous

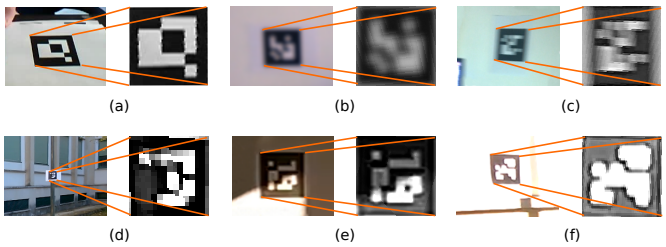

Figure 5: Marker images obtained after removing perspective distortion under extreme conditions. (a) Ideal conditions. (b) Defocus. (c) Motion blur. (d) Small scale. (e) Non-uniform lighting. (f) Overexposure. The low quality images in (b-f) lead to erroneous binary code extractions, producing errors during the marker identification process.

layer. There may be several feature maps in a given convolutional layer for each plane of units in the previous subsampling layer, so that the gradual reduction in spatial resolution is then compensated by an increasing number of features. Finally, before the loss layer there typically is at least one fully connected layer.

The whole network can be trained using the backpropagation algorithm.

3. Problem formulation and dataset generation

Let us denote by $\mathbb{D}$ the dictionary of n markers to be employed in a particular application. Also, let us assume that a method for detecting marker candidates in images is available. For each candidate, the homography of the projected square is calculated and its canonical view obtained as shown in Fig. 2(a). Then, instead of extracting the marker bits, we directly feed a classifier with the canonical image in order to identify the markers.

Please notice that four different canonical views of a marker are possible, namely, these corresponding to rotations of 0, 90, 180 and 270 degrees. For a correct camera pose estimation, it is required to know the rotation so as to identify each marker corner correctly. Thus, apart of determining if the candidate belongs to any of the markers in $\mathbb{D}$, we need to determine the rotation of the canonical marker view.

Formulated as a classification problem, our input consists in the pixels of the grey-level canonical image, and a total $4 \times n + 1$ classes are defined: one for each possible marker rotation, and an additional one representing the negative case, i.e., indicating that the input is not a marker but part of the background.

3.1 Training Dataset Generation

In order to apply classification techniques, a labeled dataset including samples from all the considered classes is required. In our problem, it consists of a collection of canonical images of all the dictionary markers (considering the four possible rotations) and also non-marker images for the negative class. For the latter, a large amount of patches from the

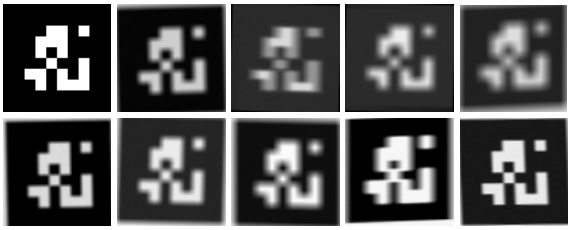

Figure 6: Some random marker transformations generated for training. The top-left image corresponds to the original marker and the rest are synthetic transformations.

MIRFLICKR-25000 image dataset (Huiskes & Lew, 2008) was collected. However, since obtaining a large dataset of markers representing a wide variety of conditions from real images is not practical, we propose the creation of a synthetic training dataset by applying different transformations to the original marker images. These transformations aim at emulating those happening in real situations, including extreme conditions such as those shown Fig. 5.

Let us denote by I the original marker image of size $s \times s$ and $I(p)$ the intensity value of pixel p. Let us also denote I' the synthetic image generated with the proposed transformations. In total, five different transformations are considered. For a given input image I, we first randomly select which transformations are applied and their order, since they are not commutative. The chance of selecting each transform varies with the possibility of finding it in real scenes, as will be explained later. Below are the transformations employed.

- Blur: simulates the motion blur and defocus. A box blur kernel was used with width and height randomly selected following a bidimensional uniform distribution $\mathbf{U_2}((1, \eta_1 s)^2)$.

- Dynamic range compression: since the original input image has only the values $\{0, 255\}$, which are usually not found in real images, we randomly perform a reduction of this range by applying the following transformation to the pixels intensities

$$I'(p) = min(255, aI(p) + b),$$

  where a and b are sampled from the following uniform probability distributions $a \backsim \mathbf{U}(\eta_2, 1)$ and $b \backsim \mathbf{U}(0, \eta_3)$. The min function ensures that the resulting values are not greater than 255.

- Affine transformation: a projective transformation simulating the errors in the corner estimation of the marker candidates is applied. To do so, the image corners of I are randomly translated adding an offset following the uniform distribution $\mathbf{U_2}((-\eta_4 s, \eta_4 s)^2)$.

- Non-uniform light: is simulated by adding a radial gradient light modelled as a bidimensional Gaussian distribution $\mathbf{N}(c, \Sigma)$. The transformation is then defined as:

$$I'(p) = min\left(255, I(p) + 50 \cdot e^{-\frac{1}{2}(p-c)^t \Sigma^{-1}(p-c)}\right).$$

It is assumed that the distribution is symmetric, i.e., $\Sigma = \sigma^2\mathbf{I}$, where $\mathbf{I}$ stands for the identity matrix, and that $\sigma \backsim \mathbf{U}(s/4, s/2)$ and $c \backsim \mathbf{U_2}(\Omega^2)$, where:

$$\Omega = [-s, 0] \cup [s, 2s],$$

so that the light origin, c is out of the image.

- Dilate: simulates the effect produced by overexposure, which increases the inner white borders of the markers (see Fig. 5(f)). The size of the structuring element used for dilation is randomly selected following the distribution $\mathbf{U_2}((1, \eta_5 s)^2)$.

Based on our experience, we have considered each transformation to be selected with a predefined probability. The first three transformations have a 75% chance of being applied, while the fourth and fifth have a 25% and 15% chance respectively. The lower probabilities of the last two are because, when applied, they produce stronger effects. Also, they correspond to cases not as frequently observed in real scenes as the rest of transformations. Likewise, the parameters employed for the different distributions have been empirically determined as $\eta_1 = 0.2$, $\eta_2 = 0.4$. $\eta_3 = 25$, $\eta_4 = 0.025$ and $\eta_5 = 0.08$. Figure 6 shows the original image of a marker and some of the synthetic images automatically generated.

4. Experimentation

This section describes the experimentation carried out to validate our proposal. In order to do a rigorous evaluation of the results, statistical analyses of the different alternatives have been done. First, it is explained how the dataset for training, validation and test has been elaborated. Then, the methodology employed to compare the different methods tested is described. Following, the different configurations of SVM, MLP and CNN are evaluated. Next, the best options of each approach are compared to the state-of-the-art marker systems ArUco (Garrido-Jurado et al., 2014) and AprilTags (Olson, 2011). Finally, we analyze the computing times of each method so as to determine their suitability for real-time applications.

4.1 Dataset Elaboration

For our tests, a dictionary $\mathbb{D}$ composed by fifty markers of 6×6 bits has been generated with the method proposed in (Garrido-Jurado et al., 2016), which guarantees the best error correction capabilities for ArUco and AprilTags approaches. We consider that fifty markers cover a wide range of real applications. In total, there are $4 \times n + 1 = 201$ different classes.

Our dataset is divided in three parts: training, validation and test. The two first are synthetically generated as explained in Sect. 3.1, while the third one is comprised by candidates extracted from real-world videos. During the training, the first part is used to determine the optimal parameters and the second one as stop criterion to avoid overfitting Our training and validation sets are comprised by 250 synthetic images for each class (200 for training and the rest for validation). Additionally, there are 210 000 images for the negative class, extracted from the MIRFLICKR-25000 image collection (Huiskes & Lew, 2008) (out of which 200 000 are for training).

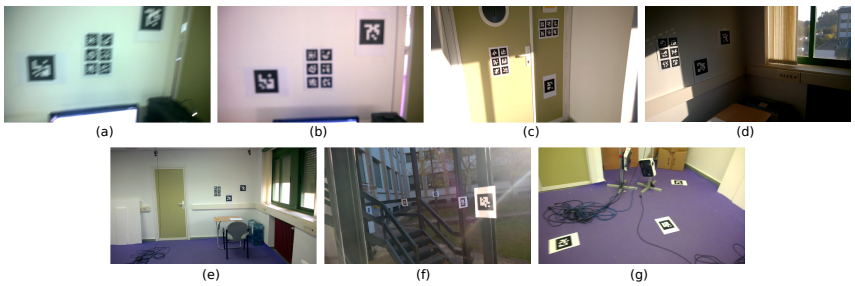

Figure 7: Sequences for testing: (a) blur, (b) defocus, (c) overexposure, (d) non-uniform lighting, (e) scale, (f) outdoor, (g) indoor

The test dataset consists of seven sequences recorded both in outdoor and indoor scenarios, covering challenging situations in real-life applications. The first one (Fig. 7a), shows significant motion blur due to fast camera movement. In the second one (Fig. 7b), the camera is forced to see the markers out of focus. In the third one (Fig. 7c), the environment is highly illuminated and camera overexposure is frequent. In the fourth one (Fig. 7d), some markers are illuminated by non-uniform lighting. The fifth sequence (Fig. 7e) focuses on the identification capability at large distances, i.e. identifying small markers. The sixth sequence (Fig. 7f) shows a large outdoor scenario labeled with a high number of markers. The same is reproduced in the seventh sequence (Fig. 7g) for an indoor scenario.

The method proposed in (Garrido-Jurado et al., 2014) has been used to extract the candidates from the test sequences (see Fig. 1). The original size of the candidates is 40×40 pixels, and they have been manually labelled. In total, there are 3924 frames from which 102 829 candidates have been extracted, out of which 84 030 belong to background.

Please notice that while training is done exclusively on synthetic images, testing is performed on real ones.

4.2 Comparison Methodology

In order to obtain a formal comparison of the considered classification methods, non-parametric tests have been employed (García et al., 2010). First, the Friedman omnibus statistical test (Friedman, 1937) has been applied to figure out whether the results of the methods are statistically different or not. In case of finding statistically significant differences (the null hypothesis of equality of means is rejected) the Finner post-hoc test (Finner, 1993) is employed to compare the best ranked method (control method) against the others, so as to find the concrete pairwise comparisons producing the differences. For pairwise comparison the adjusted p values of Finner test have been computed. The value $p = 0.05$ has been employed, indicating that results of the test are reliable at 95.5%.

The F-measure, which is defined as the harmonic mean of precision and recall, has been employed to analyze the performance of the tested methods.

$$F = 2\frac{precision \cdot recall}{precision + recall},$$

$$precision = \frac{tp}{tp + fp}; \quad recall = \frac{tp}{tp + fn},$$

where tp, fp and fn stands for true positives, false positives and false negatives, respectively. In our problem tp correspond with markers correctly identified, fp refers to errors, background elements identified as a marker or inter-marker confusion, and fn correspond with those markers that are considered background elements. Precision reflects a method's capacity for not selecting background elements or avoiding inner-marker confusion, while recall indicates how many correct markers a method is able to find, without take consideration of the errors. As these two measures are opposite, we focus our analysis on the F-measure, which is a unique indicator to determine the overall performance of a method.

4.2.1 SVM Classifiers

This section aims at determining the best SVM configuration for the given problem. To that end, we have tested the original images of 40×40 pixels and their reduced versions of 20×20 and 10×10 pixels. Also, both linear and Radial Basis Function (RBF) kernels have been employed. Additionally, Principal Component Analysis (PCA) (Pearson, 1901) has been applied to further reduce the data dimensionality to the 25, 50 or 100 principal components when possible.

Table 1: Average rankings of the SVM configurations by Friedman test. In bold, configurations that are rejected by the Finner's procedure.

Configuration	Ranking	p_{Finner}
10-RBF	4.28	-
20-LIN	5.07	$8.20E^{-1}$
20-LIN-100	5.50	$7.43E^{-1}$
10-LIN-50	6.21	$6.15E^{-1}$
20-RBF	7.64	$3.77E^{-1}$
20-RBF-100	8.07	$3.28E^{-1}$
10-LIN	8.28	$3.20E^{-1}$
40-LIN	8.35	$3.20E^{-1}$
10-RBF-50	9.78	$1.64E^{-1}$
40-LIN-100	11.42	$6.31E^{-2}$
20-LIN-50	11.85	$5.04E^{-2}$
40-RBF	12.00	$\mathbf{4.95}E^{-2}$
40-LIN-50	13.00	$\mathbf{2.51}E^{-2}$
20-RBF-50	13.92	$\mathbf{1.27}E^{-2}$
20-LIN-25	14.35	$\mathbf{9.71}E^{-3}$

A SVM classifier has been trained for each parameter combination. Table 1 summarizes the results of the fifteen best classifiers, sorted by their Friedman's ranking. The naming convention employed in the first column is $XX - KER(-PCA)$, where XX indicates the size of the image patches, KER stands for the kernel employed (linear or RBF) and,

optionally, PCA indicates the number of PCA components employed. For instance, the first line (10-RBF) is a SVM classifier using input images of 10×10 pixels, a RBF kernel, and no PCA reduction.

The second column indicates the ranking values of each combination, an indication of how good a SVM performs compared to the others. A value of one would indicate that a configuration obtains the best F-measure in the seven sequences tested.

Finally, the third column shows the adjusted p values used for the Finner test. The methods with p value ≤ 0.05 are rejected by the Finner procedure and highlighted in bold to ease their localization. In other words, the elements in bold are those for which the Finner test finds statistically relevant differences compared to the best ranked method, i.e., they are clearly worst than the best method. Whereas for the methods not in bold font, the differences found cannot be considered statistically relevant.

It can be observed that, although the best classifier is 10-RBF, the Finner test has not found strong statistical differences compared to the combinations above the middle horizontal line. However, it can be assured that the elements below this line are worst than the best one.

From this analysis we can assert that the results are significantly worst when PCA is employed for a reduction of only twenty five components. The kernel choice does not seem to affect, although the lineal one ranks slightly better. It is also evident that images of only 10×10 pixels are enough to obtain the best ranking results.

Although it will be explained later in more detail (Sect 4.3), the performance of the best combination, 10-RBF, can be seen in Table 4 along with results of the rest of the methods tested. Finally, it must be indicated that the best SVM is comprised by 8546 support vectors.

4.2.2 MLP Classifiers

For the MLP case, input images of 10×10, 20×20 and 40×40 pixels were tested using both Identity and ReLU activation functions. The number of hidden layers varied between zero and two using 200 neurons. Finally, in all cases, the output layer consist of 201 neurons (the number of classes in our problem) connected to a softmax function of the same size. The maximum number of training epochs was fixed to fifty.

The Friedman's ranking and p values from Finner test are shown in Table 2 for the fifteen best classifiers. The naming convention employed is $XX - AF(-H*)$, where XX indicates the size of the image patches, AF stands for the activation function (Identity or ReLU), and, finally, the amount of Hs indicates the number of hidden layers contained in the network.

From this post-hoc analysis, we can affirm that the use of ReLU activation function deteriorates the results and, in general, larger images improve them. The performance of the best MLP, 40-Identity-H, can be seen in Table 4 for each of the test sequences along with the results of the other approaches.

Table 2: Average rankings of the MLP configurations by Friedman test. In bold, configurations that are rejected by the Finner's procedure

Configuration	Ranking	p_{Finner}
40-Identity-H	1.57	-
20-Identity-H	2.85	$5.91E^{-1}$
40-Identity	3.00	$5.76E^{-1}$
20-Identity	3.42	$4.88E^{-1}$
10-Identity-H	5.14	$1.68E^{-1}$
40-Identity-HH	5.57	$1.29E^{-1}$
20-Identity-HH	6.99	$\mathbf{3.57}E^{-2}$
10-Identity	7.57	$\mathbf{2.13}E^{-2}$
10-Identity-HH	8.85	$\mathbf{4.61}E^{-3}$
20-ReLU	10.28	$\mathbf{6.22}E^{-4}$
40-ReLU-H	11.42	$\mathbf{1.04}E^{-4}$
10-ReLU	12.28	$\mathbf{2.58}E^{-5}$
40-ReLU	12.57	$\mathbf{1.95}E^{-5}$
10-ReLU-H	14.14	$\mathbf{1.46}E^{-6}$
20-ReLU-H	14.28	$\mathbf{1.46}E^{-6}$

4.2.3 CNN Classifiers

For the CNN case, we have tested the original input data size of 40×40 pixels and the reduced version of 20×20 pixels. Using smaller images was not possible due to the sizes of the employed filters.

The Identity and ReLU activation functions were tested and, in all cases, convolution kernels of size 5×5 were employed. We have tested using one and two convolutional steps with a different number of filters, namely $\{10, 25, 50\}$. A max-pooling layer of kernel 2×2 was applied after each convolution. Then, up to one fully connected hidden layer of 200 neurons was inserted before the output layer, which is connected to a the softmax layer. As in the MLP case, all configurations were trained up to fifty epochs.

Table 3 shows the fifteen best CNN configurations ranked by the Friedman test. The naming convention used for the CNN is $XX - AF - YY(-ZZ)(-H?)$, where XX refers to the input size of the image patches, AF stands for the activation function (Identity or ReLU), YY indicates the number of filters for the first convolution layer, ZZ indicates the number of filters for the second convolution layer (if used), and finally the addition of the hidden layer is indicated with H.

From the analysis, it can be observed that the use of the ReLU function deteriorates the results and the addition of a second convolution layer does not improve them either. When the input data is 20×20, the inclusion of one hidden layer produces worst results, while for the 40×40 case, the opposite occurs. As previously, the F-Measures of the best CNN, 40-Identity-10-H, are detailed for each test sequence in Table 4.

Table 3: Average Rankings of the CNN configurations using the Friedman test. In bold, configurations that are rejected by the Finner's procedure

Configuration	Ranking	p_{Finner}
40-Identity-10-H	3.28	-
40-Identity-25-H	3.42	$9.52E^{-1}$
40-Identity-50-H	3.85	$8.33E^{-1}$
20-Identity-25	5.28	$4.51E^{-1}$
20-Identity-50	5.42	$4.44E^{-1}$
40-Identity-50	7.35	$1.21E^{-1}$
20-Identity-10	7.50	$1.18E^{-1}$
40-Identity-25	7.78	$1.02E^{-1}$
40-Identity-50-25-H	9.00	$\mathbf{3.37}E^{-2}$
40-Identity-25-25-H	9.57	$\mathbf{1.98}E^{-2}$
20-Identity-10-H	10.00	$\mathbf{1.38}E^{-2}$
40-Identity-10	10.21	$\mathbf{1.31}E^{-2}$
20-Identity-25-H	10.57	$\mathbf{1.07}E^{-2}$
20-Identity-50-H	11.71	$\mathbf{2.95}E^{-3}$
40-ReLU-10-H	14.99	$\mathbf{1.33}E^{-5}$

4.3 Comparison with the state of the art

In this section, we evaluate the best classifiers previously obtained against the fiducial marker systems ArUco (Garrido-Jurado et al., 2014) and AprilTags (Olson, 2011) in the test sequences. The F-measures of each method are shown in Table 4.

Table 4: F-Measure of each method in the test sequences. In bold, the best raking method for each sequence

	SVM	MLP	CNN	ArUco	AprilTags
blur	**0.999**	0.996	0.996	0.894	0.890
defocus	0.942	0.910	**0.961**	0.306	0.149
overexposure	0.988	0.994	**0.996**	0.946	0.807
n. u. l.	0.922	**0.993**	0.988	0.900	0.914
zoom out	**0.871**	0.825	0.741	0.492	0.029
room	**0.997**	0.990	0.990	0.816	0.774
outdoor	**0.952**	0.933	0.936	0.856	0.691

It can be observed that the results of the three trained classifiers clearly surpass those obtained by ArUco and AprilTags. Also, SVM ranks better in four of the seven sequences. Table 5 shows the average values of precision, recall and F-Measure for the seven sequences.

It can be observed that ArUco and AprilTags are highly restrictive during the marker identification process, leading to a high precision at the expense of a low recall. This is especially remarkable in the AprilTags case, which can correct up to three erroneous bits.

Table 5: Average precision, recall and F-Measure for the test sequences.

	SVM	MLP	CNN	ArUco	AprilTags
precision	0.986	0.951	0.972	0.996	**0.997**
recall	0.924	**0.949**	0.922	0.639	0.511
F-Measure	**0.953**	0.948	0.944	0.744	0.608

However, when the F-measure is analyzed, the three trained classifiers perform significantly better than ArUco and AprilTags.

As in previous sections, a Friedman test has been employed to rank the methods. In this case, the Wilcoxon signed-rank test (Wilcoxon, 1945) has been used in order to determine if there are significant differences between any pair of methods. Table 6 shows the Friedman's ranking and also the p-value for all pairwise comparisons. Methods that are significantly worst than others are shown in bold in their respective columns. The Table shows that, in all cases, the classifiers obtain better results than ArUco and AprilTags, and that these differences are statistically relevant. Also, it can be observed that, although SVM ranks better than MLP and CNN, the differences are not statistically relevant.

Table 6: Average Rankings of all compared methods by Friedman test. In bold, methods that are significantly different according to Wilcoxon test.

	Rank	*Wilcoxon*			
		SVM	CNN	MLP	ArUco
SVM	1.71	-	-	-	-
CNN	1.99	1.0	-	-	-
NN	2.28	$3.98E^{-1}$	$8.65E^{-1}$	-	-
ArUco	4.14	$\mathbf{1.79}E^{-2}$	$\mathbf{1.79}E^{-2}$	$\mathbf{1.79}E^{-2}$	-
AprilTags	4.85	$\mathbf{1.79}E^{-2}$	$\mathbf{1.79}E^{-2}$	$\mathbf{1.79}E^{-2}$	$\mathbf{4.25}E^{-2}$

In order to exemplify the impact of using our proposal in a real-life application, Fig. 8 shows the recall of the different methods for the *zoom out* scene as a function of the observed marker size. We can see that the results obtained by the three machine learning approaches clearly surpass those obtained by ArUco and AprilTags, specially for the smallest sizes. In this particular case, the MLP method achieves better recall than SVM and CNN, but not better F-measure, as previously shown in Table 4.

Finally, we present in Table 7 the computational times required by each method to process a single candidate. Tests have been performed in an Intel i7-3930K processor using a single thread. In those cases where GPU implementations of the methods were available, a Nvidia GeForce GTX 770 was employed. The results show that ArUco is orders of magnitude faster than the other methods. However, the computing times of the proposed approaches are also valid for real-time applications.

Table 7: Computing times of each method in classifying an image (ms)

Method	SVM	CNN	MLP	ArUco	AprilTags
CPU	1.47	3.95	$6.07E^{-1}$	$9.56E^{-2}$	4.52
GPU		$7.99E^{-1}$	$5.27E^{-1}$		

5. Conclusions

This work has proposed to handle the fiducial marker identification problem as a classification one in order to increase the performance under difficult image conditions, such as camera defocus, motion blur, small scale or non-uniform lighting. Since obtaining a large training dataset, from real images, covering realistic and challenging situations is not practical, we propose a method to create a synthetic one by applying a set of transformations that emulates those happening in real situations.

We have evaluated different type of classifiers, namely SVM, MLP and CNN using a wide variety of configurations, and the best configuration of each type has been then evaluated against the state-of-the-art fiducial marker systems ArUco and AprilTags. The results show that, in all cases, the proposed classifiers obtains results significantly better than ArUco and AprilTags. However, the differences found between the classifiers are not statistically relevant, i.e., the three classifiers perform similarly.

Finally, it must be indicated that the tools employed and the best classifier obtained in this work will be set publicly available as a part of the ArUco library (Muñoz-Salinas. & Garrido-Jurado, 2013) upon its publication.

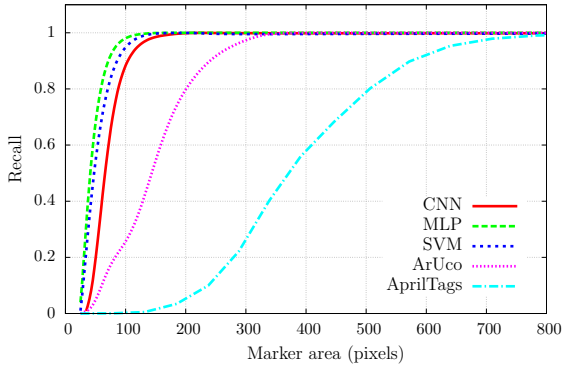

Figure 8: Evolution of recall as a function of the visible marker area in the *zoom out* sequence.

Acknowledgement

This research has been supported by the Spanish Ministry of Science and Technology under project TIN2015-00182.

References

Bishop, C. (2006). *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. In *Machine Learning*, pp. 273–297.

Cubillo, J., Martin, S., Castro, M., Diaz, G., Colmenar, A., & Boticki, I. (2014). A learning environment for augmented reality mobile learning. In *Frontiers in Education Conference (FIE), 2014 IEEE*, pp. 1–8. IEEE.

Dong, C., Loy, C. C., He, K., & Tang, X. (2016). Image super-resolution using deep convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *38*(2), 295–307.

Fiala, M. (2010). Designing highly reliable fiducial markers. *IEEE Trans. Pattern Anal. Mach. Intell.*, *32*(7), 1317–1324.

Finner, H. (1993). On a monotonicity problem in step-down multiple test procedures. *Journal of the American Statistical Association*, *88*(423), 920–923.

Friedman, M. (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, *32*(200), 675–701.

García, S., Fernández, A., Luengo, J., & Herrera, F. (2010). Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Information Sciences*, *180*(10), 2044 – 2064. Special Issue on Intelligent Distributed Information Systems.

Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F., & Marín-Jiménez, M. (2014). Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, *47*(6), 2280 – 2292.

Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F., & Medina-Carnicer, R. (2016). Generation of fiducial marker dictionaries using mixed integer linear programming. *Pattern Recognition*, *51*, 481–491.

Huiskes, M. J., & Lew, M. S. (2008). The mir flickr retrieval evaluation. In *MIR '08: Proceedings of the 2008 ACM International Conference on Multimedia Information Retrieval*, New York, NY, USA. ACM.

Khattak, S., Cowan, B., Chepurna, I., & Hogue, A. (2014). A real-time reconstructed 3d environment augmented with virtual objects rendered with correct occlusion. In *Games Media Entertainment (GEM), 2014 IEEE*, pp. 1–8. IEEE.

Kim, G., & Petriu, E. (2010). Fiducial marker indoor localization with artificial neural network. In *Advanced Intelligent Mechatronics (AIM), 2010 IEEE/ASME International Conference on*, pp. 961–966.

Klopschitz, M., & Schmalstieg, D. (2007). Automatic reconstruction of wide-area fiducial marker models. In *Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pp. 1–4. IEEE Computer Society.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Comput.*, *1*(4), 541–551.

McCulloch, W., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biology*, *5*(4), 115–133.

Muñoz-Salinas., R., & Garrido-Jurado, S. (2013). ArUco library. `http://www.uco.es/investiga/grupos/ava/node/26/`. [Online; accessed 15-December-2015].

Olivares-Mendez, M., Kannan, S., & Voos, H. (2015). Vision based fuzzy control autonomous landing with uavs: From v-rep to real experiments. In *Control and Automation (MED), 2015 23th Mediterranean Conference on*, pp. 14–21.

Olson, E. (2011). AprilTag: A robust and flexible visual fiducial system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3400–3407. IEEE.

Ouyang, W., Luo, P., Zeng, X., Qiu, S., Tian, Y., Li, H., Yang, S., Wang, Z., Xiong, Y., Qian, C., Zhu, Z., Wang, R., Loy, C., Wang, X., & Tang, X. (2014). Deepid-net: multi-stage and deformable deep convolutional neural networks for object detection. In *arXiv preprint arXiv:1409.3505*.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, *2*(6), 559–572.

Rekimoto, J. (1998). Matrix: A realtime object identification and registration method for augmented reality. In *Third Asian Pacific Computer and Human Interaction, Kangawa, Japan*, pp. 63–69. IEEE Computer Society.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning internal representations by error propagation. In Rumelhart, D. E., & Mcclelland, J. L. (Eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*, pp. 318–362. MIT Press, Cambridge, MA.

Rumpler, M., Daftry, S., Tscharf, A., Prettenthaler, R., Hoppe, C., Mayer, G., & Bischof, H. (2014). Automated end-to-end workflow for precise and geo-accurate reconstructions using fiducial markers. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, *3*, 135–142.

Sanchez-Lopez, J. L., Pestana, J., de la Puente, P., & Campoy, P. (2015). A reliable open-source system architecture for the fast designing and prototyping of autonomous

multi-uav systems: Simulation and experimentation. *Journal of Intelligent & Robotic Systems*, 1–19.

Sun, Y., Chen, Y., Wang, X., & Tang, X. (2014). Deep learning face representation by joint identification-verification. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., & Weinberger, K. Q. (Eds.), *Advances in Neural Information Processing Systems 27*, pp. 1988–1996. Curran Associates, Inc.

Wagner, D., & Schmalstieg, D. (2007). ARToolKitPlus for pose tracking on mobile devices. In *Computer Vision Winter Workshop*, pp. 139–146.

Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin*, *1*(6), 80–83.