UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA

E.T.S. DE INGENIÉRIA INFORMÁTICA

TRABAJO FIN DE MÁSTER

Máster en Ingeniería de Sistemas y Control

RECONOCIMIENTO DE LA ESCALA Y ANILLOS DE DISTANCIAS EN IMÁGENES DE RADAR

Andreas Muñoz Zuara

Directores:

José Miguel Guerrero Hernández

Matilde Santos Peñas

Departamento de Informática y Automática

Septiembre de 2015

Máster en Ingeniería de Sistemas y Control

RECONOCIMIENTO DE LA ESCALA Y ANILLOS DE DISTANCIAS EN IMÁGENES DE RADAR

Proyecto Tipo A:

Proyecto específico propuesto por un profesor

Andreas Muñoz Zuara

Directores:

José Miguel Guerrero Hernández

Matilde Santos Peñas

Autorización

Autorizamos a la Universidad Complutense y a la UNED a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a sus autores, tanto la memoria de este Trabajo Fin de Máster, como el código, la documentación y/o el prototipo desarrollado.

Firmado:

Andreas Muñoz Zuara

RECONOCIMIENTO DE LA ESCALA Y ANILLOS DE DISTANCIAS DE RADAR

RESUMEN

En este trabajo se ha desarrollado un sistema automático de reconocimiento de la escala y los anillos de distancias en las imágenes obtenidas de un radar comercial. La escala a la que son tomadas las imágenes de un radar determina automáticamente el número y la separación entre los anillos de distancia, por lo que una incoherencia entre ambos valores invalidaría la imagen. La detección automática de estos valores nos va a permitir determinar la validez de dicha imagen de forma rápida y efectiva. Para ello se ha dividido el sistema en dos apartados, por un lado se ha realizado la detección de las circunferencias concéntricas y por otro la detección de la escala.

El reconocimiento de círculos concéntricos se ha dividido en tres operaciones. En primer lugar se ha requerido la aplicación de técnicas de detección de bordes para diferenciar las formas geométricas del fondo. Para dicha operación se seleccionó el operador de Sobel. En segundo lugar se realizó la detección del centro de las circunferencias, para lo cual se llevó a cabo un estudio del estado del arte de las técnicas de reconocimiento de patrones, seleccionando algunas de ellas relacionadas con un sistema muy utilizado en este ámbito como es la transformada de Hough. Además de la utilización de diferentes implementaciones de dicha transformada, se diseñó un algoritmo basado en un artículo de la bibliografía para tratar de mejorar los resultados obtenidos con la transformada. También se eligió un método distinto, el cálculo mediante la suma de diferencias cuadradas, para poner en perspectiva los resultados de las técnicas anteriores. Finalmente, para la detección de las circunferencias a partir del centro, se implementó un método que nos permite hacer un barrido de la imagen con diferentes radios en busca de circunferencias que cumplan las condiciones establecidas.

En la detección de escala se realizó una búsqueda entre las familias de soluciones de la detección de caracteres, siendo elegida una solución del tipo reconocimiento de plantillas. Tras una selección previa de las plantillas más adecuadas, se eligieron dos algoritmos muy utilizados, la correlación cruzada normalizada y la suma de diferencias cuadráticas, a fin de poder seleccionar la técnica que más nos convenga.

Para terminar se realizaron una serie de pruebas para cada una de las tecnologías seleccionadas, con diferentes configuraciones y parámetros de entrada en caso de que éstos fueran modificables. Una vez terminadas las pruebas valiéndonos de la base de datos de imágenes de radar de que disponemos, se realizó una comparativa de los resultados y se determinaron cuáles fueron las técnicas más efectivas en la resolución del problema y cuáles serían algunas posibles mejoras realizables al sistema para mejorar su comportamiento de cara al futuro.

Lista de palabras clave: Transformada circular de Hough, reconocimiento de patrones, reconocimiento de caracteres, correlación cruzada normalizada, suma de diferencias cuadráticas, detección de plantillas.

Índice

1 Introducción y objetivos 15

1.1 Motivación 15

1.2 Estado del arte en reconocimiento de formas 18

1.2.1 Detección de circunferencias 18

1.2.2 Reconocimiento de caracteres 20

1.3 Motivación 23

1.4 Objetivos 24

1.5 Organización de la memoria 24

2 Algoritmos para la detección de círculos concéntricos en imágenes de radar 27

2.1 Detección de bordes y transformada circular de Hough 27

2.1.1 Extracción de bordes 27

2.1.2 Transformada lineal y circular de Hough 30

2.2 Definición e implementación de los algoritmos de detección de círculos concéntricos 32

2.2.1 Algoritmo de detección del centro a partir de tres puntos 33

2.2.2 Transformada circular de Hough 35

2.2.3 Detección del centro a partir de la minimización de la suma cuadrada 36

2.3 Definición e implementación del algoritmo de detección de radios 37

3 Algoritmos para la detección de patrones de la escala del radar 39

3.1 Identificación de plantillas (*Template matching*) 39

3.1.1 Correlación cruzada normalizada 41

3.1.2 Suma de diferencias cuadradas 41

3.2 Elección del algoritmo de detección de escala 42

4 Resultados 43

4.1 Escenario de pruebas y material utilizado 43

4.2 Resultados de detección de círculos concéntricos 43

4.2.1 Resultados de la detección de bordes 43

4.2.2 Ajuste paramétrico de los algoritmos 47

4.2.3 Resultados de las pruebas 55

4.3 Resultados de la identificación de la escala mediante reconocimiento de patrones 61

4.3.1 Elección de las plantillas 61

4.3.2 Resultados de las pruebas de detección de escala 68
5 Conclusiones y líneas de trabajo futuras 75
5.1 Conclusiones 75
5.2 Líneas de trabajo futuras 76
6 Bibliografía 79

Índice de figuras

Figura 1.- Esquema de un sistema de reconocimiento de patrones. 15
Figura 2.- Radar Furuno modelo 1715. 17
Figura 3.- Sistema de medición de imágenes del radar. 17
Figura 4.- Esquema de aprendizaje de retropropagación de una red neuronal artificial. 22
Figura 5.- Etapas del sistema ALPR. 22
Figura 6.- Comparación entre la imagen original (izquierda) y la imagen tras el detección de bordes (derecha). 28
Figura 7.- Ejemplo de aplicación del operador de Sobel. 29
Figura 8.- Ejemplo de aplicación del método de Canny. De izquierda a derecha, imagen original; imagen con filtro presuavizado; imagen con gradiente; imagen con supresión de no máximos; resultado final. 30
Figura 9.- Puntos de la recta en el espacio de la imagen (A) generan diferentes rectas en el espacio paramétrico (B) (Illingworth y Kittler, 1988). 31
Figura 10.- Votación en el espacio paramétrico (Chen y col., 2012). 32
Figura 11.- El ángulo $\boldsymbol{ABC}$ debe ser igual que el ángulo $\boldsymbol{DEF}$ si los círculos son concéntricos. 34
Figura 12.-Círculo a detectar usando 4 puntos (izquierda). Se realizan 4 circunferencias con radio R y en la intersección está el centro de la circunferencia a detectar (derecha). 36
Figura 13.- Ejemplo de barrido de radios en la imagen de radar a partir de un centro (x,y). 37
Figura 14.- Caracteres de la escala antes y después de la detección de bordes. 39
Figura 15.- Representación gráfica de la identificación de plantillas. 40
Figura 16.- Imagen original del radar y su histograma. 43
Figura 17.- Detección con Sobel horizontal sin dilatación (izquierda) y ejemplo de borde obtenido (derecha). 47
Figura 18.- a) Imagen original b) Centros detectados en la imagen de bordes. c) Centros válidos. d) Círculos detectados a partir de los centros válidos. 49
Figura 19.- Centros resultantes con un umbral de decisión a) r*pi*0.5, b) r*pi*0.8 y c) r*pi*1. 51
Figura 20.- Resultado obtenido con el método de Peng y diferentes configuraciones de parámetros de entrada, para el caso de circunferencias pequeñas (imágenes superiores) y para circunferencias grandes (imágenes inferiores). 53
Figura 21.- Círculos detectados con una precisión de a) 35% b) 40% c) 65%. 53
Figura 22.- Círculos detectados con un incremento de radio en píxeles de a) 0.5; b) 2; c) 5. 54
Figura 23.- Círculos detectados para un número de pixeles de a) 40 b) 100 c) 200. 55
Figura 24.- Resultado de la función *imfindcircles* para la imagen de Matlab (izquierda) y para la imagen de radar (derecha). 56
Figura 25.- Imágenes utilizadas para la evaluación de los algoritmos seleccionados. 57
Figura 26.- Gráfica comparativa de los tiempos de todos los métodos. 60
Figura 27.- Gráfica comparativa de los centros detectados por todos los métodos. 60
Figura 28.- Gráfica comparativa de los centros válidos con cada método. 61
Figura 29.- Ejemplo de plantillas individuales. 62
Figura 30.- Plantillas individuales tras el procesamiento mediante el método de Otsu. 63
Figura 31.- Recuadro de búsqueda de plantillas a) original y b) filtrado. 63
Figura 32.- Curva de respuesta de la correlación cruzada para una de las plantillas. 64

Figura 33.- Caracteres reconocidos con umbral de 0.8. 64
Figura 34.- Caracteres detectados con umbral de 0.6. 65
Figura 35.- Caracteres detectados con la imágen original y un umbral de 0.8. 66
Figura 36.- Comparación resultados con y sin preprocesado 67
Figura 37.- Patrón a detectar (izquierda) y resultado de detección (derecha). 68
Figura 38.- Plantillas utilizadas en el ejemplo de reconocimiento de caracteres. 68

Índice de tablas

Tabla 1.- Posibles valores de funcionamiento del radar FURUNO 1715. 41
Tabla 2.- Resultados obtenidos con cada uno de los algoritmos de detección de bordes a partir de la figura 16. 44
Tabla 3.- Número promedio de bordes válidos para la detección de círculos en cada detector de bordes. 46
Tabla 4.- Relación comparativa de las diferentes configuraciones testeadas. 50
Tabla 5.- Resultados obtenidos para el algoritmo propio completo con la configuración $\boldsymbol{\delta inicial}$=200, $\boldsymbol{dmin}$=100. 58
Tabla 6.- Resultados obtenidos para el algoritmo propio completo con la configuración $\boldsymbol{\delta inicial}$=800, $\boldsymbol{dmin}$=200. 58
Tabla 7.- Resultados obtenidos para el algoritmo propio parcial con la configuración $\boldsymbol{\delta inicial}$=200, $\boldsymbol{dmin}$=100. 58
Tabla 8.- Resultados obtenidos para el algoritmo propio parcial con la configuración $\boldsymbol{\delta inicial}$=800, $\boldsymbol{dmin}$=200. 58
Tabla 9.- Resultados obtenidos para la transformada circular de Hough. 59
Tabla 10.- Resultados obtenidos para el de detección por mínimos cuadrados. 59
Tabla 11.-Subconjunto de muestra con imágenes testeadas (izquierda) y la mejor coincidencia encontrada (derecha) que supera el umbral 0.8 en la NCC. 69
Tabla 12.- Gráficas comparativas de los resultados para cada plantilla. La barra azul es valor máximo de la SSD y la roja de la NCC. 72

1 Introducción y objetivos

1.1 Motivación

El reconocimiento de patrones es uno de los problemas más importantes a lo largo de la historia del ser humano. Estamos continuamente intentando categorizar la información que recibimos para clasificarla en grupos y así poder tratarla de forma adecuada. Este proceso, que se ha venido realizando de forma natural desde hace miles de años, se ha tratado de llevar a las máquinas, tanto para automatizar el proceso y evitar que necesite un supervisor humano, como para mejorar nuestra comprensión sobre la naturaleza y funcionamiento del reconocimiento de patrones en el mundo natural.

El proceso a seguir para la clasificación de patrones se puede observar en la Figura 1 (Orrite, 2009). En primer lugar, debemos obtener información del objeto en cuestión. A partir de dicha información se hace un procesamiento para obtener una serie de características que nos permitan categorizar dicho objeto. Una vez tenemos esas características principales, éstas son utilizadas por un clasificador, que con los datos almacenados generalmente en una etapa de aprendizaje anterior y aplicando diferentes técnicas de toma de decisión, realizará una hipótesis sobre cuál es la categoría a la que pertenece el objeto de estudio.

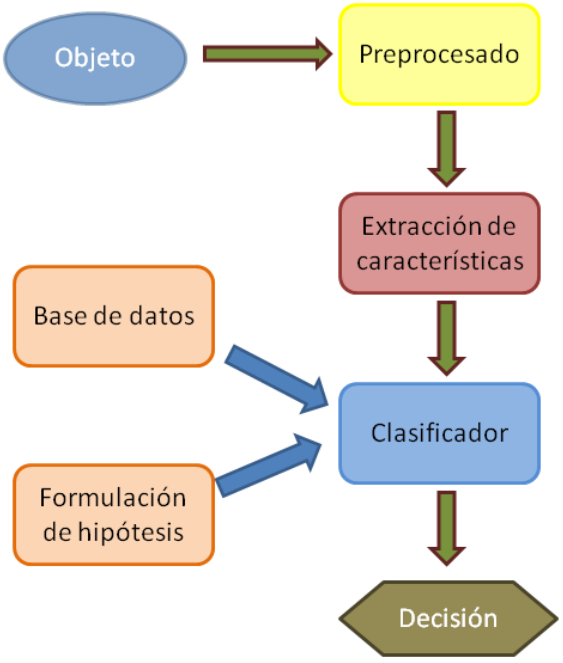


Figura 1.- Esquema de un sistema de reconocimiento de patrones.

Aunque *a priori* el procedimiento parece bastante intuitivo debido a su presencia intrínseca en nuestro comportamiento, surgen una serie de problemas que hacen que la extrapolación a sistemas automáticos sea muy compleja. A continuación presentaremos algunos de los principales retos y limitaciones a la hora de categorizar de forma automática (Duda y col., 2000).

1. En primer lugar, la identificación de las características que vamos a utilizar para clasificar es clave en los resultados que vamos a obtener. Para ello es importante el conocimiento previo del ámbito en que se va a utilizar y los criterios de clasificación. Un extractor de características genérico es, en la mayoría de los casos, menos efectivo que uno específico, además de la enorme complejidad que conlleva.

2. Otro aspecto importante es el ruido, entendiendo como tal cualquier señal que no nos interese y que perturbe la información original. El ruido es inevitable en el tratamiento de señales; lo que hay que intentar es evitarlo o minimizarlo, ya sea modelándolo o mediante la elección de características que no se vean muy afectadas por el ruido.

3. A la hora de tomar las muestras también hay que tener en cuenta otras perturbaciones que no son debidas al ruido, como por ejemplo, la información incompleta o la superposición de objetos. Estas dificultades, sumadas al ruido del punto anterior, puede hacer que la probabilidad de fallo a la hora de obtener una respuesta sea alta.

4. También hay que tener en cuenta la variación temporal del sistema. Es posible que hayamos diseñado un buen clasificador para el sistema actual, pero si este sistema es cambiante en el tiempo, añadiendo o reduciendo posibles clases, nuestro diseño puede no adaptarse bien y empobrecer los resultados hasta que llegue un momento que no sirva como solución aceptable.

Las limitaciones expuestas se pueden tratar de subsanar añadiendo complejidad al algoritmo de decisión y recogiendo una mayor cantidad de información para disponer de más características con las que comparar. Sin embargo, esa mayor complejidad es una limitación más a tener en cuenta, ya que en la mayoría de los casos no podemos mejorar el sistema todo lo deseado debido a las limitaciones tecnológicas y a las del propio sistema, como pueden ser la velocidad de decisión y el acceso a mayor información, entre otros.

En resumen, las variables a tener en cuenta a la hora de diseñar un sistema de reconocimiento de patrones son lo suficientemente variadas y complejas de tratar como para suponer un importante reto tecnológico. Para poder abordar estos problemas es importante estudiar la forma que tiene de resolverlos la naturaleza y tratar de adaptar dichos conocimientos a los sistemas artificiales que más nos interesen en cada caso.

Uno de los campos más interesantes en los que se aplican estas técnicas es el tratamiento de imágenes. Para una persona reconocer figuras o patrones en una imagen es un problema trivial. Sin embargo, una máquina no es capaz de identificar tan fácilmente esos patrones, ya que para ella una imagen es solamente un conjunto de píxeles y carece de las normas de asociación de la naturaleza. En este trabajo fin de máster se propone la aplicación de técnicas de reconocimiento de patrones sobre imágenes de radar para reconocer la escala y los anillos de distancia presentes en dichas imágenes. Dichas técnicas se utilizan para llevar a cabo identificación de formas, en este caso circunferencias por una parte y plantillas predefinidas por otro. El tipo de forma, así como la información disponible para detectarla, van a determinar la clase de soluciones que se van a evaluar y seleccionar. Para ello vamos a contar con datos obtenidos a partir del radar marca Furuno modelo 1715 (FURUNO, 2011), Figura 2.

Figura 2.- Radar Furuno modelo 1715.

Este radar es un sistema comercial, en el cual la información no está accesible para poder tratarla directamente. Por ello, se ha decidido utilizar una cámara situada en posición cenital a la pantalla del radar, de forma que el reconocimiento que se va a llevar a cabo se realizará sobre esas imágenes, Figura 3

Se han seleccionado imágenes de radar debido a que es un sistema de detección de obstáculos muy útil en barcos, más si cabe si pretenden ser instalados en barcos autónomos, ya que permite a dicho barco conocer en todo momento la posición de posibles obstáculos presentes en la ruta o que están cercanos a él. Con el tratamiento de imágenes que planteamos, se le dotará a dicho barco de la capacidad de comprobar la veracidad de la información que está procesando.

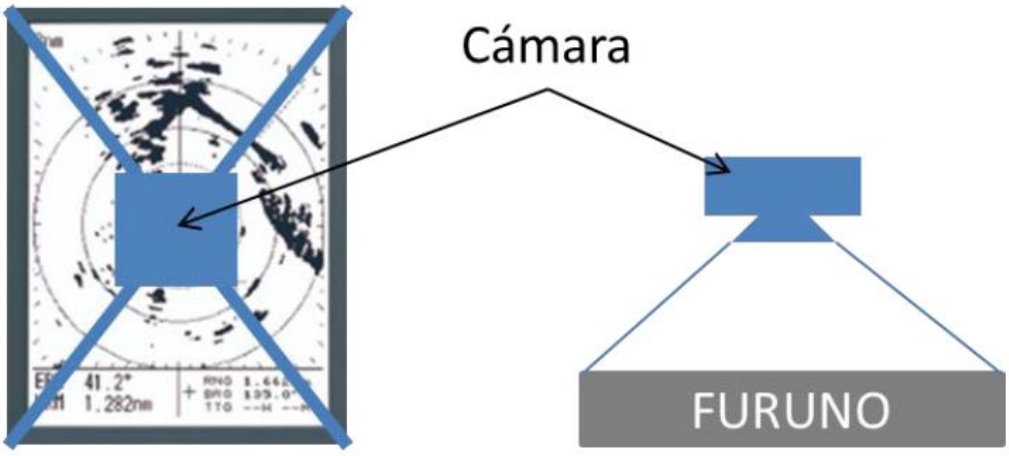


Figura 3.- Sistema de medición de imágenes del radar.

Las imágenes obtenidas por la cámara, tal y como se verá en la sección 4.1, son a color y de alta resolución (1920 x 1080 píxeles cada una). Estas imágenes serán procesadas mediante técnicas de visión por computador, de manera que la información que el usuario ve en pantalla pueda ser interpretada de forma automática por un ordenador.

1.2 Estado del arte en reconocimiento de formas

Dentro de la automatización del reconocimiento de patrones en aplicaciones reales, la detección de formas circulares es uno de los problemas a resolver más extendido en ámbitos muy diversos. Desde reconocimiento de señales de tráfico en seguridad vial (Fleyeh, 2011; Arlicot, 2009) hasta detección de las pupilas en biomedicina (Hiley, 2006; Ebisawa, 2009), prácticamente cualquier ámbito necesita de las técnicas más avanzadas que le permitan resolver este problema de la forma más eficaz.

A la hora de enfrentar el reto de la detección de formas, existen dos grandes grupos de técnicas, las deterministas y las estocásticas. En las primeras se engloban los métodos derivados de la transformada de Hough (Yuen y col., 1990) y técnicas de identificación de plantillas *(template matching)* (Andáloa y col., 2010). Por otra parte, los métodos estocásticos incluyen técnicas de muestreo aleatorio (Fischer y Bolles, 1981), reconocimiento simulado (Bongiovanni y Crescenci, 1995) y algoritmos genéticos (Roth y Levine, 1994).

Las primeras estrategias que se utilizaron para hacer frente a la detección de formas fueron el reconocimiento de patrones y de modelos. Son los procedimientos más intuitivos y más sencillos de aplicar, pero no por ello su efectividad es menor respecto a otros casos tal y como se puede ver en (Line y Chen, 2008). Sus principales limitaciones están relacionadas con la extracción de contornos en imágenes reales y en la variancia de las posiciones de los objetos respecto de los modelos en la mayoría de los casos. Esto es debido a que para detectar la forma, muchos métodos se basan en obtener los contornos de la figura o las características que la pueden definir (circularidad, excentricidad y relación entre ejes, entre otros), y dicha información se compara con una base de datos para buscar formas conocidas que presenten características similares. Si no disponemos en dicha base de datos de formas variadas en múltiples posiciones, al modificar la posición de una figura es posible que no pueda ser reconocida debido a la falta de patrones similares.

1.2.1 Detección de circunferencias

El método más utilizado para la detección de círculos en imágenes digitales es, sin duda, la transformada circular de Hough (TH) (Muammar y Nixon, 1989), para la cual existen múltiples implementaciones y variaciones. La transformada de Hough modifica la información de la imagen real a un espacio donde los píxeles de la imagen están parametrizados, tal y como se verá más adelante. Dichos parámetros dependerán de la forma geométrica que estemos modelando. Una vez realizada la transformación, la detección se reduce a determinar los picos en el espacio transformado, filtrando y promediando los datos. Estas técnicas son muy robustas, pero requieren de un gran espacio en memoria y de una capacidad de cálculo elevada. Además, el ruido hace que la precisión del resultado se vea afectada significativamente, ya que la dirección del gradiente se ve alterada. Para minimizar dichas limitaciones existen variantes del método clásico como son la TH aleatorizada (Xu ycol., 1990), la TH probabilística (Shaked y col., 1996) o la TH *fuzzy* (Han y col., 1993). Aunque dichos

desarrollos sirvieron para reducir el tiempo de procesamiento, la debilidad frente al ruido sigue siendo un factor a tener en cuenta.

Como alternativa a los métodos basados en la TH, una gran variedad de estrategias de optimización surgieron a lo largo del tiempo. Los algoritmos genéticos (AG) se han utilizado de forma importante en tareas de detección de formas. Algunos ejemplos son: a) el algoritmo de Roth y Levine (Roth y Levine, 1994) para el uso de AG en extracción de curvas y superficies parametrizables en imágenes; b) (Ayala-Ramirez y col., 2006), quienes proponen un detector de círculos basado en AG que es capaz de detectar varias circunferencias, aunque no detecta círculos imperfectos; c) (Cuevas, Zaldiwar y col., 2010), quienes presentan otra variante de método heurístico, la optimización de la evolución de la diferencial discreta (DDE). Estos algoritmos producen buenos resultados incluso con ruido, pero los tiempos de ejecución siguen siendo elevados, lo que impediría su uso en estrategias de tiempo real, cuyo tiempo de procesamiento debe ser mínimo.

Los algoritmos aleatorizados también son otra interesante opción como sustituto de la TH. Chen y Chung (Chen y Chung, 2001) proponen un algoritmo aleatorizado de detección de círculos sin necesidad de afinar parámetros (RCD). El método consiste en obtener cuatro puntos aleatorios, tres se utilizan para construir un círculo y el cuarto para comprobar si ese círculo incluye todos los puntos utilizados. Los círculos elegidos se determinan mediante una votación. Tomando este método como base, han surgido otras soluciones tratando de mejorar los tiempos de ejecución, como el GRCD-R (Chung y col., 2012). Esta familia de métodos consigue mejorar la precisión y velocidad, pero no significativamente.

En contraste con las propuestas anteriores de optimización global han surgido otras soluciones de optimización multimodal que permiten la detección de múltiples puntos óptimos, tanto globales como locales. Los círculos bien definidos se pueden detectar mediante óptimos globales, mientras que para los que están difusos o incompletos será necesaria la búsqueda de máximos locales. Para llevar a cabo esta estrategia, algunas propuestas se inspiran en el comportamiento de la naturaleza: a) en (Poli y col., 2007), la optimización de una partícula en un enjambre (PSO) modela el comportamiento social de una bandada de aves o un banco de peces. Cada partícula del enjambre representa una posible solución, y en cada momento se registra cada posición de las partículas y en especial la que tiene la mejor solución hasta el momento. En cada iteración las partículas se ven atraídas hacia la mejor solución y hacia la posición de la misma; b) en (Cuevas y col., 2012a) se propone la detección de múltiples círculos mediante el uso de optimización basada en una colonia de abejas (ABC). Utilizando el comportamiento de un enjambre de abejas, el algoritmo parte de tres componentes: las fuentes de comida, que serían las posibles soluciones; la cantidad de néctar de cada fuente, que representa la calidad de la posible solución; y las diferentes clases de abejas, que simbolizan las operaciones que permiten generar nuevas fuentes de comida. En (Cuevas y col., 2012b), la optimización por electromagnetismo (EMO) utiliza la teoría física sobre dicho fenómeno para optimizar problemas con un solo objetivo. Cada partícula tiene una carga que depende del valor de su función objetivo, y las cargas se van atrayendo y repeliendo dependiendo de dichos valores. Este algoritmo tiene un coste computacional bajo y una convergencia razonablemente rápida.

Recientemente nuevos algoritmos para la detección de círculos en tiempo real se han desarrollado para mejorar los resultados obtenidos en ambientes donde el tiempo de respuesta es crítico. Frosio y Borghese (Frosio y Borghese, 2008) proponen un detector en tiempo real basado en la estimación de la máxima probabilidad, con el que permiten detectar círculos estableciendo el radio de antemano. Akinlar y Topal (Akinlar y Topal, 2013) presentan también un detector de circunferencias llamado EDCircles. Su funcionamiento consiste en convertir arcos y tramos incompletos en segmentos lineales. Si varios de esos segmentos giran en la misma dirección consecutivamente, se considera que forman parte de un círculo. Además también presenta un control de falsos positivos basado en la fórmula de número de falsas alarmas (NFA) de Desolneux (Desolneux y col., 2001). Otros métodos como (Liu y col., 2014) se basan en este algoritmo para mejorar su precisión y robustez, así como para disminuir el tiempo de convergencia mediante el uso de nuevas técnicas de extracción de bordes y un nuevo algoritmo de detección denominado *"top-down"*, que empieza por los segmentos mayores y los va dividiendo en segmentos más pequeños en caso de no encontrar un círculo.

A pesar de la extendida utilización de círculos concéntricos en la industria, su estudio para el desarrollo de soluciones específicas en la detección de dichos círculos no está muy generalizado. Algunas soluciones se han propuesto para solucionar problemas industriales como la calibración de cámaras en estudios ópticos (Jiang y Quan, 2005) o la identificación de estos círculos en placas electrónicas (PCB) (Nao y col., 2010). Generalmente la mayoría de estos algoritmos parametrizan una circunferencia para obtener el centro y a partir de ahí obtener el resto de circunferencias. En (Chen y col., 2012) parten del gradiente de la transformada de Hough para obtener el centro de la circunferencia. Con el centro y el gradiente se realiza una transformada de Hough en una dimensión para obtener el radio. Este método mejora la solución obtenida mediante TH frente al ruido y las discontinuidades.

En este trabajo nos basaremos en la solución propuesta en (Silveira, 2005) para detectar los círculos concéntricos de la imagen de radar. El planteamiento es similar al caso anterior y se explicará en detenimiento en el capítulo dos sección 2.1.

1.2.2 Reconocimiento de caracteres

El reconocimiento de caracteres ha suscitado mucha atención desde hace décadas debido a su uso en aplicaciones de gran relevancia. No sólo se utiliza en la industria, sino que es clave en el sector financiero. Algunas de las aplicaciones más comunes donde se utilizan estas técnicas son el reconocimiento de matrículas de vehículos o la identificación de números de serie en billetes o cheques.

Las técnicas más extendidas en este ámbito de trabajo se dividen tres grupos: identificación de plantillas, reconocimiento probabilístico y sistemas inteligentes como las redes neuronales. La identificación de plantillas, o *template matching*, consiste en la búsqueda de una imagen considerada la plantilla dentro de una imagen de mayor tamaño. Se trata de una técnica sencilla pero con un coste computacional que puede ser elevado si el número de plantillas que se pretende identificar es alto. Esta estrategia se explicará con detalle en el capítulo tres.

Respecto al reconocimiento estadístico de patrones, la idea principal consiste en la realización de una evaluación estadística de la imagen, de forma que el patrón reconocido será aquel que presente una probabilidad mayor de verosimilitud con la imagen original. Antes de poder ser utilizado, primero será necesario entrenarlo con algunos casos conocidos. De esta forma podemos comprobar que la salida se adecúa a la solución correcta, y en caso contrario, podemos ajustar parámetros y probabilidades que permitan obtener dicha solución. De la imagen se extraen las propiedades que mejor definen los patrones buscados, a través de las cuales se determinará el patrón al que pertenece.

Los algoritmos de clasificación utilizados dependerán de la información de que dispongamos previamente a la evaluación. En caso de que se disponga de las densidades de probabilidad correspondientes a cada clase de patrón, la opción más común es la teoría de decisión de Bayes (Bayes, 1763). Si nos encontramos con que no disponemos de la información probabilística inicial necesaria, existen otros métodos en los que el sistema va aprendiendo, de forma supervisada (Delalleau y col., 2005) o de forma autónoma (Zhu y Chen, 2007), a medida que evalúa patrones.

Por otro lado, las redes neuronales surgen como solución para los problemas no lineales, ya que son difíciles de modelar y resolver con los métodos anteriormente citados. Como su nombre indica, estos sistemas simulan el funcionamiento de un cerebro que está compuesto por una serie de neuronas conectadas entre sí formando una red sináptica. En el caso de una red neuronal artificial, su estructura la componen una serie de autómatas conectados entre sí. El funcionamiento global vendrá determinado por el procesamiento de cada elemento, la arquitectura de la red (las conexiones entre células) y las reglas que gobiernan el funcionamiento del sistema (los pesos asignados a cada unidad).

Una red neuronal estima una salida a partir de una entrada de forma dinámica y no paramétrica. Las redes neuronales artificiales son capaces de aprender de sus errores cambiando sus reglas de funcionamiento, es decir, variando los pesos asignados a cada unidad, como se observa en la Figura 4.

Para poder adaptar su salida a una solución más óptima deben ser entrenadas antes de estar plenamente funcionales. El entrenamiento más común es proporcionarle a la red un conjunto de entradas con sus correspondientes salidas, de forma que la red estime su salida, la compare con la solución óptima, evalúe su error y recalcule sus pesos internos para acercarse a dicha solución. Algunos ejemplos de este entrenamiento sería el uso de datos macroeconómicos como entrada y los valores de la Bolsa de ese día como salida para estimar cómo afectan las variables introducidas en los movimientos bursátiles; o en aplicaciones médicas la introducción de variables del estado físico como entrada y las enfermedades que esa persona posee como salida, para la estimación de enfermedades a partir de unos síntomas previos.

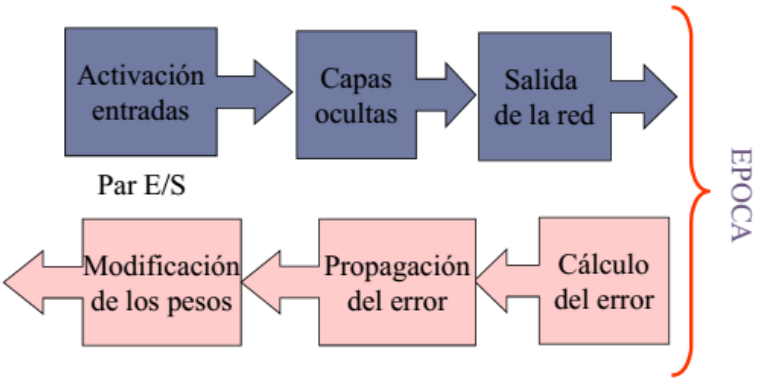


Figura 4.- Esquema de aprendizaje de retropropagación de una red neuronal artificial.

La principal ventaja de esta técnica es que no es necesario un conocimiento del problema extenso, simplemente con algunos ejemplos iniciales la red puede ir adaptando ella sola la salida hasta dar con la mejor posible. Son extremadamente útiles para problemas no lineales y para soluciones en tiempo real, ya que su estructura altamente paralela agiliza el procesado. Además son tolerantes a fallos individuales y a información difusa, con ruido o inconsistencias.

Respecto a las aplicaciones más comunes del reconocimiento de caracteres, el reconocimiento automático de matrículas (ALPR) (Du y col., 2013) tiene numerosas aplicaciones reales, como puede ser el control de acceso de vehículos en un parking (Rashid y col., 2012), la monitorización del tráfico o los radares en carretera (Liu y col., 2011a). Dichos sistemas de reconocimiento se basan principalmente en dos técnicas: identificación de plantillas (Massoud y col., 2013) y los métodos de aprendizaje dinámico (Liu y col., 2011b). En cualquier caso, estas técnicas están limitadas tanto por factores ambientales (iluminación, fondo de la imagen o velocidad del vehículo) como por variaciones en la matrícula (localización en el vehículo, tamaño, color o cambio de estándares). La Figura 5 muestra las etapas en las que se divide un ejemplo de reconocimiento de caracteres en placas de matrícula.

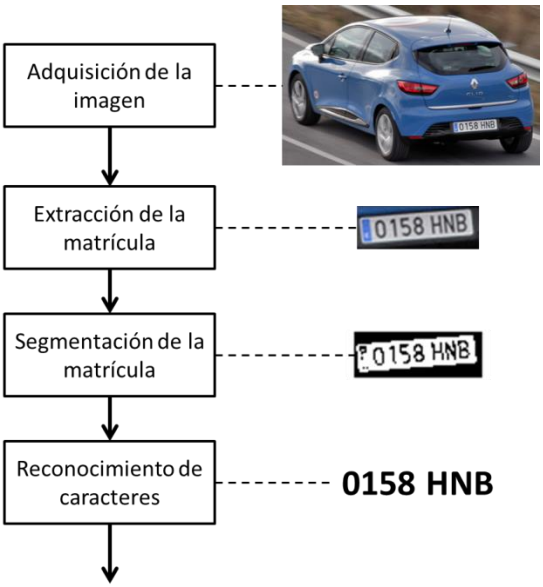


Figura 5.- Etapas del sistema ALPR.

Por otro lado, el reconocimiento de números de serie en documentos bancarios representa un reto muy importante dado que requiere de una fiabilidad completa, más importante en este caso que la velocidad de procesamiento. Esto es debido a que incluso pequeños fallos en el reconocimiento de caracteres pueden acarrear grandes problemas financieros. Por ello, en estos casos, se utilizan técnicas adaptativas como las redes neuronales o algoritmos genéticos. En (Zhao y col., 2010) utiliza una red neuronal MLP con propagación hacia atrás o retro-propagación (*backpropagation*) entrenada mediante un algoritmo genético. Las redes neuronales tienen una excelente habilidad para minimizar fallos y por eso son muy utilizadas en el reconocimiento de patrones. La propagación hacia atrás es beneficiosa en la búsqueda local de datos, mientras que el algoritmo genético es útil en la búsqueda de mínimos globales en espacios complicados con muchos máximos locales y no lineales. La utilización de ambos métodos conjuntos es un sistema muy efectivo a la hora de entrenar la red neuronal. Sin embargo, este sistema no consigue superar una fiabilidad del 95%. Para mejorar esta propuesta, en (Feng y col., 2014) se propone realizar un estudio exhaustivo de esta problemática aplicada al mismo tipo de numeración del caso anterior, la RMB, notación bancaria utilizada en China. Para ello comparan los resultados obtenidos con tres métodos de extracción de características, cuatro tipos de clasificadores y cinco estrategias de clasificación diferentes. Además incluyeron tres clases de distorsionadores para mejorar la precisión del reconocimiento, y finalmente se probaron tres esquemas de rechazo para evitar muestras con baja probabilidad de éxito. De esta forma obtuvieron un método de clasificación en cascada que era capaz de reconocer números con un 100% de fiabilidad, aunque para ello tenía que rechazar el 1% de las muestras por falta de fiabilidad.

1.3 Motivación

Las imágenes de radar proporcionan una información necesaria para la navegación. Esta información tiene que ser analizada, clasificada e interpretada de forma rápida y fiable, pues es de vital importancia cuando se tratan de sistemas que van a ir instalados en barcos autónomos.

En este Trabajo Fin de Máster se propone la implementación de varios algoritmos de detección de patrones con el fin de determinar el número de circunferencias concéntricas existentes en las imágenes de radar proporcionadas por una cámara, así como la escala a la que está funcionando el radar, todo ello con el fin de que dado un punto en la imagen, se pueda obtener la distancia real a dicho punto con respecto al vehículo que lleva instalado el radar. De esta forma se diseña un sistema de bajo coste y versátil, ya que puede ser utilizado para la mayoría de radares existentes en el mercado simplemente ajustando los detalles que caracterizan a cada modelo.

La detección de circunferencias concéntricas se dividirá en dos pasos, primero la detección del centro y posteriormente la determinación de los radios de cada una de las circunferencias. En concreto, para la detección del centro, en primer lugar se plantea la implementación de un algoritmo novedoso basándonos en el descrito en el artículo (Silveira, 2005). Más tarde dicha implementación se comparará, por un lado, con uno de los métodos más utilizados en este

entorno, la transformada circular de Hough, además de con otros algoritmos existentes en la literatura (Bone, 2010 y Peng, 2005). Para la detección de las circunferencias a partir de los centros calculados previamente se propone el desarrollo de un algoritmo propio que parte del centro y a través de iteraciones variando el radio busca todos los círculos que comparten dicho centro.

En cuanto a la delimitación de la escala utilizada, se van a aplicar dos de las técnicas más utilizadas en la identificación de patrones, como son la suma de diferencias absolutas (SSD) y la correlación cruzada normalizada (NCC). Ambos procedimientos serán evaluados con diferentes plantillas y procesados de imagen para comparar resultados en cada caso.

1.4 Objetivos

El objetivo principal de este trabajo es el diseño de un algoritmo que permita la identificación de círculos concéntricos en imágenes de radar, e identificar la escala a la que está funcionando el mismo. Además, se realizará una comparación con diferentes métodos tanto para la detección de los círculos como el reconocimiento de los números de escala. Para conseguirlo se han completado los siguientes objetivos:

1) Estudio y análisis de técnicas basadas en visión por computador:
   - Métodos de reconocimiento de círculos.
   - Técnicas de identificación de circunferencias concéntricas.
   - Técnicas de reconocimiento de patrones numéricos.
2) Propuesta de soluciones, mejoras y adaptaciones de los métodos estudiados:
   - Métodos de detección del centro en circunferencias concéntricas y los radios de las mismas.
   - Métodos para reconocer patrones numéricos.
3) Análisis y validación de los resultados obtenidos:
   - Análisis de los métodos en diferentes escenarios.
   - Comparativa de resultados entre los diferentes métodos, tanto para detección de círculos como para reconocimiento de caracteres.
   - Selección de los mejores métodos para abordar este problema.

1.5 Organización de la memoria

En el capítulo uno se realiza una introducción al problema al que se pretende dar solución en este trabajo fin de máster, incluyendo el estado del arte de las tecnologías utilizadas y los objetivos planteados para abordar dicho problema.

En el capítulo dos se explican de forma detallada las técnicas aplicadas para la determinación de circunferencias concéntricas y se presenta el proceso seguido para la implementación de

dichas técnicas. Cabe destacar la aportación de un algoritmo propio novedoso para la detección del centro de las circunferencias. También se ha desarrollado un algoritmo original para la detección de circunferencias.

En el capítulo tres se exponen las metodologías utilizadas para la obtención de la escala de las imágenes. De forma análoga al capítulo anterior, se explica el proceso seguido para llevar a cabo la detección.

En el capítulo cuatro se muestran los resultados obtenidos de la implementación de los diferentes desarrollos que se han explicado en los capítulos anteriores.

Finalmente, en el capítulo cinco se exponen las conclusiones obtenidas sobre el resultado del trabajo, así como las posibles líneas de mejora que se podrían seguir.

2 Algoritmos para la detección de círculos concéntricos en imágenes de radar

La detección de círculos concéntricos en imágenes es un caso específico de la detección de patrones, y más específicamente de la detección de círculos genérica. No sólo se trata de encontrar todos los círculos presentes, sino que además hay que detectar un centro común para todos ellos. Aunque existen multitud de soluciones de todo tipo para el caso general de encontrar un círculo, las soluciones para este caso en concreto son muy reducidas ya que no han sido objeto de un amplio estudio.

En este capítulo explicaremos en primer lugar las bases y el funcionamiento de las técnicas elegidas para la resolución de este problema. A continuación se expondrán los algoritmos utilizados y los resultados obtenidos con cada uno de ellos, para finalmente resumir todo el proceso en las conclusiones finales.

2.1 Detección de bordes y transformada circular de Hough

Para afrontar el problema de detectar círculos concéntricos en las imágenes de radar, vamos a enfocarlo hacia soluciones derivadas de la transformada circular de Hough, la cual ha sido ampliamente utilizada en la literatura (Muammar y Nixon, 1989). Como ya se ha expuesto brevemente en el estado del arte, esta transformada constituye uno de los pilares más importantes en el tratamiento de imágenes.

Sin embargo, antes de poder utilizar dicha técnica, es necesario un preprocesado de la imagen que mejore el funcionamiento de la transformada. El principio de funcionamiento de la transformada de Hough necesita trabajar con los bordes de la imagen. Por puntos de borde entendemos los puntos que representan un cambio abrupto con respecto al valor de los pixeles adyacentes. Por tanto un borde será una cadena continua de puntos. Para poder resaltar estos bordes respecto del fondo de la imagen, es necesaria la aplicación de las que se conocen como técnicas de extracción de bordes o *edge detection*.

2.1.1 Extracción de bordes

La detección de bordes muchas veces es uno de los primeros pasos a la hora de realizar tratamiento de imágenes. Se compone de una serie de operaciones matemáticas que nos permiten detectar puntos de discontinuidad en las imágenes (Pajares y Cruz, 2007) Generalmente esos puntos de discontinuidad suelen pertenecer a bordes, y la detección de los mismos nos permite eliminar factores ambientales de la imagen y mejorar en gran medida el tratamiento que se podrá hacer sobre dicha imagen posteriormente, ya que en una imagen a

color como es nuestro caso existe una gran variedad de valores para cada pixel, lo que hace mucho más difícil detectar patrones y geometrías en la misma.

Tras la detección de bordes se obtiene una imagen en blanco y negro, donde generalmente el negro será el fondo de la imagen y el blanco serán los puntos de discontinuidad que nos permitirán detectar formas. En la Figura 6 podemos observar la diferencia entre la imagen original y el de la imagen de bordes obtenida a partir de la misma.

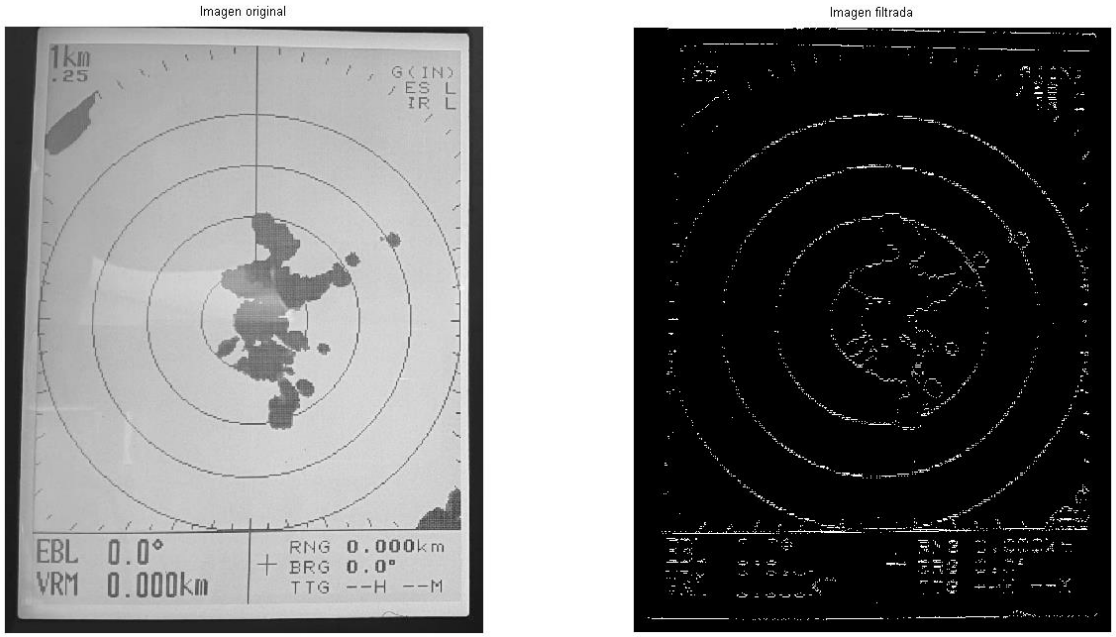


Figura 6.- Comparación entre la imagen original (izquierda) y la imagen tras el detección de bordes (derecha).

Existen una gran variedad de métodos para la determinación de bordes en imágenes (Ziou y Tabbone, 1998). La mayoría de estos métodos se basan en la detección de los gradientes de los pixeles, y las principales diferencias radican en el tipo de filtro utilizado para determinar dichos gradientes. Aunque recientemente se han ido desarrollando nuevos métodos de detección de bordes, como es el caso de (Desolneux y col., 2001) que se basa en el principio de Helmholtz, o (Topal y col., 2011) quien presenta una solución para tiempo real utilizando el algoritmo Edge Drawing, existen una serie de técnicas que son las más utilizadas. A continuación se describirán dos de los métodos más comunes: el operador de Sobel y el método de Canny.

a) Operador de Sobel: Desarrollado por Irwin Sobel y Gary Feldman, es uno de los operadores más utilizados a la hora de obtener bordes en imágenes.

El proceso consiste en convolucionar la imagen con dos kernels o núcleos del operador, como se muestra en la ecuación (1). Cada uno de ellos nos permitirá aproximar las derivadas de cada pixel respecto a la intensidad para obtener los gradientes, en el eje vertical en un caso, y en el eje horizontal en el otro. Una vez obtenidas ambas componentes, se puede calcular la dirección y magnitud del gradiente, permitiéndonos a su vez obtener los bordes a partir de dicha información. En la Figura 8 observamos el resultado de este operador al aplicarse a una imagen genérica.

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * A \quad G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * A \qquad (1)$$

Figura 7.- Ejemplo de aplicación del operador de Sobel.

b) El método de Canny: fue desarrollado por John F. Canny (Canny, 1986) junto con la teoría matemática que explica su funcionamiento. Este operador utiliza un algoritmo con diferentes etapas para detectar gran cantidad de tipos de límites. El proceso consta de varios pasos:

1. Se aplica un filtro Gaussiano a la imagen original para suavizar la imagen y eliminar ruido.
2. A continuación se utiliza alguno de los operadores matemáticos para la detección de bordes (Roberts, Prewitt, Sobel) para la obtención del gradiente de cada pixel tanto en el eje horizontal como vertical.
3. El siguiente paso consiste en la supresión de picos que no sean máximos. Este proceso sirve para eliminar espurios y afinar los bordes, dejando solamente los gradientes más pronunciados. Para mejorar la detección se introduce un nuevo filtrado con un doble umbral. Los puntos con un gradiente mayor al umbral superior son marcados como bordes fuertes; los que se encuentren entre el umbral superior y el inferior serán almacenados como bordes débiles, y los que queden por debajo del umbral inferior serán eliminados.
4. Finalmente, los bordes resultantes serán fruto de la unión de los puntos fuertes del paso anterior, además de los puntos débiles que estén unidos a algún punto fuerte. En caso contrario se eliminan también.

En la Figura 8 se observa el resultado de aplicar el método de Canny a la misma imagen del ejemplo anterior.

Figura 8.- Ejemplo de aplicación del método de Canny. De izquierda a derecha, imagen original; imagen con filtro pre-suavizado; imagen con gradiente; imagen con supresión de no máximos; resultado final.

Como podemos observar, aunque el objetivo es el mismo, los resultados difieren dependiendo del método que utilicemos. En el caso de Canny los bordes son más definidos debido al filtrado previo a la aplicación del operador. Sin embargo, y debido al mismo motivo, la nitidez de los mismos es menor a la obtenida por Sobel. Por ello, no existe un método que supere a otros en toda situación, sino que será la imagen con la que tengamos que trabajar, y su respuesta a cada método, la que nos determinará qué operador es el más adecuado en cada caso.

2.1.2 Transformada lineal y circular de Hough

La transformada de Hough (Hough, 1962) es una técnica originariamente empleada en el tratamiento de imágenes diseñada para detectar patrones y formas complejas en imágenes binarias (Illingworth y Kittler, 1988). La idea básica de este método es la de transformar los datos del espacio de la imagen a un espacio paramétrico, obtenido a partir de los datos de la misma. De esta forma se pasa de un problema bastante complejo en el espacio de la imagen a otro más sencillo en el espacio paramétrico.

Al emplear el conjunto total de la imagen, resulta ser un método bastante robusto ante ruido y discontinuidades. Necesita que la imagen de entrada sea binarizada (en blanco y negro), lo que implica que antes es necesario realizar la detección de bordes de la imagen. A grandes rasgos, el algoritmo tratará de obtener geometrías como líneas, circunferencias, elipses o cualquier otra curva. Como contrapartida, requiere de una gran capacidad de memoria y el coste computacional y temporal es muy elevado. Para ilustrar el funcionamiento de este algoritmo vamos a explicar cómo se obtiene una recta que pasa por dos puntos de la imagen.

Dado el punto de la imagen de coordenadas (x_i, y_i), sobre el mismo pasarán infinitas rectas, representadas por la ecuación (2):

$$y_i = mx_i + c \qquad (2)$$

Cada una de las posibles rectas viene definida por los valores de pendiente *m* y término independiente *c*. Si transformamos dicha expresión al espacio paramétrico, la ecuación de la recta vendrá definida por la ecuación (3):

$$c = y_i - mx_i \qquad (3)$$

Variando el valor de la pendiente *m* obtendremos diferentes valores de término independiente *c*, que conformarán una recta cuya pendiente dependerá de los valores *x* e *y*. Eso quiere decir que en espacio paramétrico, cada recta vendrá definida por el punto de la imagen. Si evaluamos diferentes puntos de la imagen, obtendremos una familia de rectas, como se observa en la Figura 9. Lo interesante del caso es que si dos puntos pertenecen a la misma recta en la imagen, las rectas que se crearán en el espacio paramétrico se cortarán en el punto (*m*,*c*) que parametriza la recta en el espacio de la imagen a la que pertenecen ambos puntos. De esta forma, buscando los puntos donde las intersecciones de rectas sean máximas, obtendremos las rectas de la imagen. Como en una imagen real la intersección no va a ser perfecta, se realizará un proceso de votación y se elegirán los puntos cuyos votos superen un umbral definido.

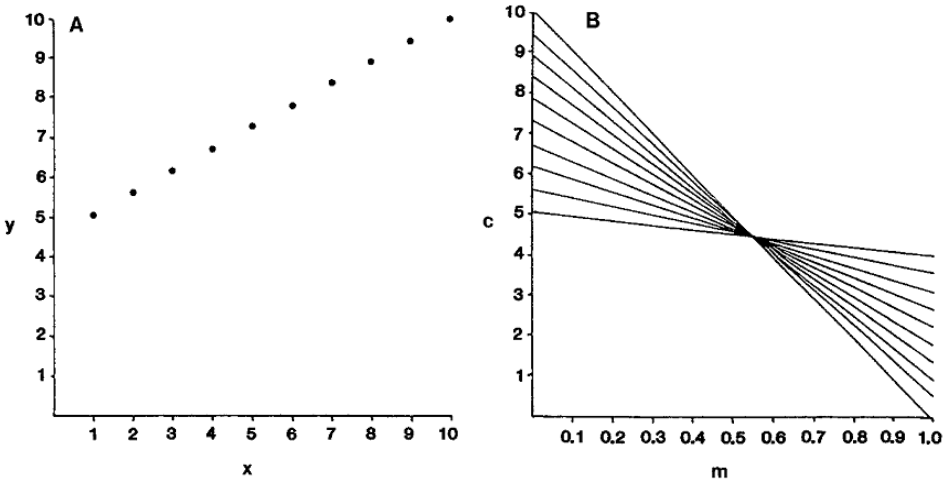


Figura 9.- Puntos de la recta en el espacio de la imagen (A) generan diferentes rectas en el espacio paramétrico (B) (Illingworth y Kittler, 1988).

Aunque originariamente el método estaba pensado únicamente para encontrar rectas, posteriormente se ha utilizado para la detección de otras formas geométricas más complejas. Para el caso que nos ocupa, la detección de círculos, el procedimiento a seguir sería análogo al explicado anteriormente. Primero parametrizamos un círculo a partir de su ecuación, obteniendo una ecuación de 3 dimensiones formada por las coordenadas del centro (a, b) y el radio r (4).

$$(x_i - a)^2 + (y_i - b)^2 = r^2 \qquad (4)$$

Donde (x_i, y_i) vuelven a ser las coordenadas de cada punto del borde. La posición del centro del círculo se irá modificando en el espacio paramétrico y con la distancia al punto de la imagen obtendremos el radio en cada caso. Utilizando en la rejilla de acumulación cada centro con cada radio calculado, se procede a la votación. En la Figura 10 observamos cómo para ciertos puntos de la imagen se produce una intersección en el espacio paramétrico en un centro y radio determinados.

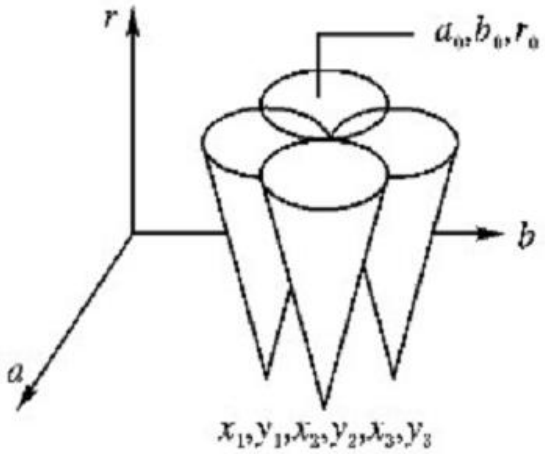


Figura 10.- Votación en el espacio paramétrico (Chen y col., 2012).

Como ya apuntábamos al principio, es mucho más sencillo buscar picos de intersecciones en el espacio paramétrico que realizar complejas operaciones para buscar y reconocer geometrías en el espacio de la imagen.

La robustez de esta técnica ante el ruido se debe a que un ruido aleatorio es capaz de deformar u ocultar en parte formas ya existentes, pero es realmente complicado que genere nuevas formas donde no las hay. Sin embargo, también hemos comprobado que el paso al espacio paramétrico implica aumentar las dimensiones de trabajo, lo que aumenta la carga de trabajo y de memoria necesarias. Por esta razón, estos sistemas no son recomendables en entornos de trabajo en tiempo real.

2.2 Definición e implementación de los algoritmos de detección de círculos concéntricos

Ya hemos comprobado en el capítulo anterior que existen numerosos trabajos sobre métodos de detección de círculos. Aunque cada artículo explica un método diferente de resolución dependiendo del problema que se aborda en él, cuando se trata de soluciones derivadas de la transformada circular de Hough, la mayoría de autores coinciden en separar el problema en dos etapas. La primera se dedicará a buscar el centro de la circunferencia, mientras que la segunda partirá de ese centro para buscar el radio de la misma. De esta forma convertimos un problema en tres dimensiones en dos problemas separados, uno de dos dimensiones y otro de

una, lo que simplifica notablemente el proceso y acelera la convergencia de la solución. El problema en tres dimensiones realiza búsquedas en un número N de matrices a la vez para encontrar la mejor solución total, que es mucho más costoso que buscar un centro a partir de un vector de puntos y posteriormente usar ese centro para barrer una única matriz, la imagen, para hallar los círculos.

En este apartado vamos a seleccionar dos propuestas en concreto para la detección del centro diferentes a la transformada de Hough.

2.2.1 Algoritmo de detección del centro a partir de tres puntos

En el estado del arte ya se comentaba que para este trabajo se ha trabajado en base al algoritmo expuesto en (Silveira, 2005). La solución propuesta está basada a su vez en el método descrito por Cao-Deravi en (Cao, 1992), en el cual se postula una detección de centros mediante el uso de tres puntos pertenecientes a un mismo borde. Para la selección de dichos puntos se ayudan de la dirección del gradiente del borde y de una serie de reglas de búsqueda propias. Como el gradiente se ve afectado por el ruido, este sistema es bastante sensible al ruido.

Con el fin de mejorar esta propuesta, Silveira propone el mismo sistema de detección de centros utilizando tres puntos del borde, pero utilizando reglas y restricciones geométricas a fin de reducir el efecto del ruido. Además también permite detectar círculos parcialmente ocultos o irregulares. A continuación se detalla cómo llevar a cabo esta detección.

Partiendo de la imagen de bordes, se etiqueta cada uno de estos bordes y se selecciona aleatoriamente uno de ellos para su evaluación. De ese borde se seleccionan de manera aleatoria tres puntos, los cuales deben cumplir las siguientes restricciones:

$$\begin{aligned} d_{min}^2 \leq (x_A - x_B)^2 + (y_A - y_B)^2 \leq 4r_{max}^2 \\ d_{min}^2 \leq (x_A - x_C)^2 + (y_A - y_C)^2 \leq 4r_{max}^2 \\ d_{min}^2 \leq (x_B - x_C)^2 + (y_B - y_C)^2 \leq 4r_{max}^2 \end{aligned} \qquad (5)$$

Donde r_{max} es el radio máximo y d_{min} es la distancia mínima que debe existir entre los entre puntos elegidos. Si se diera el caso de que los puntos seleccionados no cumplen estas restricciones, se seleccionan nuevos puntos hasta encontrar tres que las satisfagan.

Una vez se han obtenido los tres puntos, para obtener el centro se va a resolver la ecuación general de la circunferencia, a partir de la cual obtendremos una matriz de coeficientes que permitirán calcular las coordenadas del centro. La formulación matemática es la siguiente.

$Ecuación\ general\ de\ la\ circunferencia$
$x^2 + y^2 + Ax + By + C = 0$

$En\ el\ punto\ A: Ax_a + By_a + C = -(x_a^2 + y_a^2)$

$Matriz\ de\ coeficientes\ para\ los\ puntos\ A, B\ y\ C$

$$\begin{bmatrix} x_a & y_a & 1 \\ x_b & y_b & 1 \\ x_c & y_c & 1 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \end{bmatrix} = \begin{bmatrix} -(x_a^2 + y_a^2) \\ -(x_b^2 + y_b^2) \\ -(x_c^2 + y_c^2) \end{bmatrix}$$

$Coordenadas\ del\ centro: x_c = -A/2, \quad y_c = -B/2$

Como estamos buscando centros para círculos concéntricos, solo nos interesarán los centros que permitan detectar varios círculos. Para comprobarlo vamos a utilizar operaciones geométricas sobre la imagen.

Primero, se trazarán segmentos que unan cada uno de los puntos con el centro seleccionado. Si existe una circunferencia concéntrica, la cual comparte el mismo centro, cada uno de los segmentos cortará dicha circunferencia de forma que los tres nuevos puntos formarán el mismo ángulo que los puntos originales. En la Figura 11 podemos observar este fenómeno.

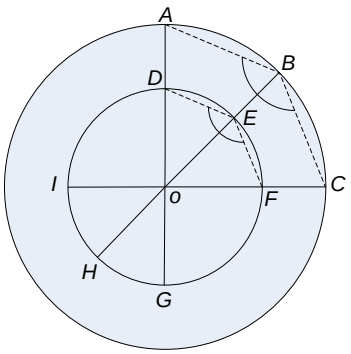


Figura 11.- El ángulo $\widehat{ABC}$ debe ser igual que el ángulo $\widehat{DEF}$ si los círculos son concéntricos.

Sobre el papel es sencillo dibujar un segmento que conecte dos puntos porque el espacio es continuo; sin embargo, en una imagen digital nos encontramos con la limitación de la resolución en píxeles que tengamos. Por lo tanto, tenemos que dibujar un segmento que una dos píxeles a través de un conjunto de píxeles. Para este proceso vamos a utilizar el algoritmo de Bresenham (Foley, 1980), cuyo pseudocódigo puede verse a continuación:

[1] $Ecuación\ de\ la\ recta: y = mx + c$
[2] $Incrementamos\ x: x = x + 1$
[3] $Dos\ opciones: \begin{cases} m + c < 0.5 & y = y\ ; c = c + m; \\ m + c > 0.5 & y = y + 1; c = 1 - (c + m); \end{cases}$
[4] $Volvemos\ al\ paso$ [2] $hasta\ que\ x = x_{final}$

Destacar que este algoritmo sólo sirve en caso de la pendiente $0 < m < 1$. En caso contrario, los incrementos se realizarán en *y,* y la evaluación de la pendiente servirá para incrementar *x* o no. También hay que tener en cuenta posibles pendientes negativas, que nos obligarán a intercambiar los puntos de inicio y fin para obtener una pendiente positiva.

Una vez obtenidos los segmentos, si se da el caso de que encontramos tres puntos que corresponden a un borde en cada uno de ellos, para calcular los ángulos que forman los segmentos que unen cada trío de puntos vamos a utilizar la siguiente ecuación.

$$\phi(\overline{BA}, \overline{BC}) = arc\ cos \frac{\overline{BA}.\overline{BC}}{\|BA\|\|BC\|} \qquad (6)$$

De esta forma, si se cumple la relación $\phi(\overline{BA}, \overline{BC}) = \phi(\overline{ED}, \overline{EF})$, se considera un centro válido como candidato y se almacena para ser evaluado en la segunda parte del desarrollo, donde se buscarán los radios de los círculos concéntricos. Para cada borde se seleccionarán varios centros a partir de diferentes ternas de puntos. De esta forma evitamos la posibilidad de elegir un sólo centro y que debido al ruido no sea válido.

2.2.2 Transformada circular de Hough

Dado que el método anterior se postula como una mejora de la transformada circular de Hough (Hough, 1962), pensada para ambientes ruidosos, se hace necesario estudiar si realmente mejora esa transformada y hasta qué punto la relación entre la precisión y el tiempo de procesamiento empieza a ser relevante.

Uno de los principales problemas es que el funcionamiento de esta técnica requiere el conocimiento a priori del radio del círculo que estamos buscando, por lo que la posición de la cámara con respecto a la pantalla debe ser precisa con el fin de no incrementar considerablemente el tiempo de cómputo, necesidad que en el caso anterior no es requerida. Como se ha explicado previamente, esto es debido a que de esta forma se reduce la dimensión del espacio paramétrico a dos dimensiones, reduciendo la complejidad de la solución. El proceso consiste en el trazado de circunferencias de radio R con centro cada uno de los puntos de los bordes. Tras el proceso de votación, los puntos de intersección que superen un determinado umbral serán considerados candidatos a centros de circunferencia. En la Figura 12 podemos observar un ejemplo gráfico de este proceso.

A la hora de utilizar la transformada circular de Hough existen múltiples variantes. Una ellas es la presentada por (Peng, 2005), quien propone la búsqueda de círculos concéntricos sin utilizar bucles en su implementación, lo que reduce el tiempo de ejecución. Asimismo, otra propuesta que utilizaba el mismo procedimiento sin bucles es la implementación rápida de la transformada (Pfragner, 2004).

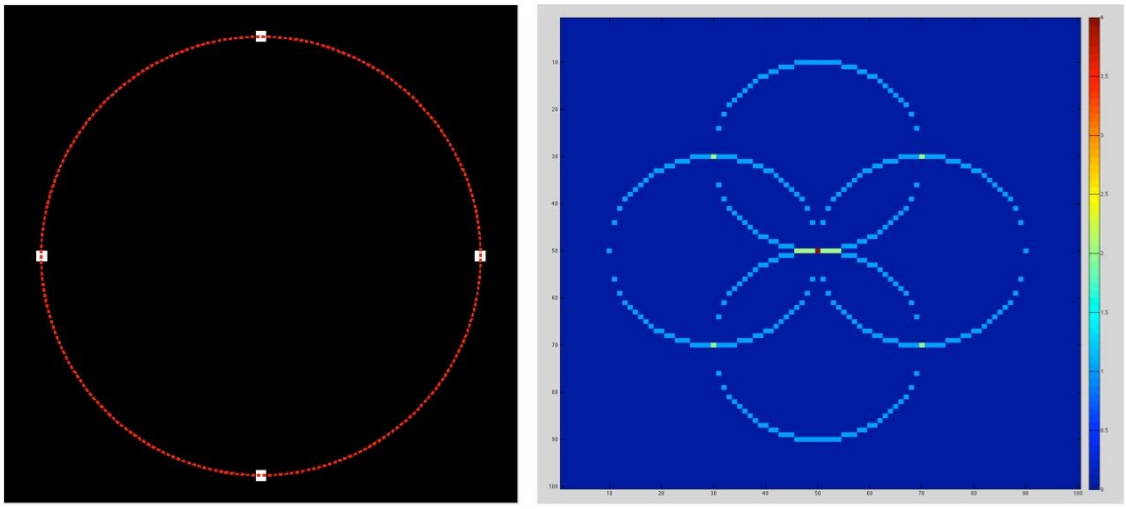


Figura 12.-Círculo a detectar usando 4 puntos (izquierda). Se realizan 4 circunferencias con radio R y en la intersección está el centro de la circunferencia a detectar (derecha).

Por otro lado, David Young (Young, 2014) propone una nueva solución basada en otra implementación optimizada de (Bone, 2010) para detectar circunferencias en imágenes binarizadas, como por ejemplo aquellas resultado de una detección de bordes. Estos métodos se basan en el algoritmo del punto medio de un círculo para encontrar circunferencias más rápidamente y sin huecos.

2.2.3 Detección del centro a partir de la minimización de la suma cuadrada

Por último, se ha seleccionado la solución propuesta por Coope (Coope, 1993) quien realiza un ajuste mediante mínimos cuadrados lineales y no lineales para aproximar una circunferencia mediante un conjunto de puntos.

Para seleccionar los puntos, se realiza un etiquetado de las zonas obtenidas en la imagen de bordes. Para cada una de las zonas se obtienen los puntos que la definen y se procede a realizar el ajuste de la circunferencia que mejor aproxima la nube de puntos mediante la minimización de la suma cuadrada.

En este caso, el problema de determinar el ajuste de la mejor circunferencia que pasa por una serie de puntos consiste en determinar los valores de $x \in R^2$ y $r \in R^+$ a partir del conjunto de puntos $a_j \in R^2, j = 1,2,...,m$, para lo que es necesario resolver el problema:

$$\min_{x,r} \sum_{j=1}^{m} \left\{ F_j(x,r) \right\}^2 \qquad (7)$$

donde $F_j(x,r)$ es la distancia del punto a_j al círculo ajustado, y *x* denota el centro del círculo y *r* su radio.

2.3 Definición e implementación del algoritmo de detección de radios

Como resultado de los desarrollos del apartado anterior, se ha obtenido en cada caso un vector con las coordenadas de los centros candidatos, cantidad que dependerá del procesado de cada algoritmo.

Partiendo de esos datos, el siguiente paso va a consistir en determinar para cada centro si corresponde al centro de un círculo o si el algoritmo nos ha devuelvo un candidato inadecuado. En este caso se ha diseñado un proceso novedoso que permite identificar las circunferencias existentes en la imagen cuyo centro sea el punto evaluado a partir de la ecuación general de la circunferencia. La idea subyacente de este algoritmo es sencilla y se basa en consideraciones geométricas sobre la imagen tal y como se detalla a continuación.

La idea de propagación de radios se puede ver reflejada en la Figura 13. Partiendo del punto (x,y) a evaluar, se va a realizar un barrido en un rango especificado de radios. Para cada radio, se va a calcular una circunferencia utilizando la ecuación general vista en (2). La cantidad de puntos que formarán dicha circunferencia se determinará como un parámetro de entrada de la función, así como el salto que va a haber entre los radios y la precisión para establecer si se ha detectado un círculo o no. Estos parámetros son los que nos permiten que una circunferencia deformada, parcialmente oculta o simplemente una sección circular, sea correctamente identificada, pero hay que tener en cuenta que disminuir mucho la precisión puede hacer que se detecten formas que realmente no son circunferencias.

Una vez se ha dibujado el círculo en cuestión, se evalúan los valores de los píxeles que forman parte del mismo. Dado que la imagen de entrada ha de ser una imagen binarizada, los pixeles blancos serán parte de un borde y los negros parte del fondo. De esta manera, si el porcentaje de pixeles blancos coincidentes con los puntos de la circunferencia trazada supera el nivel de precisión establecido a la entrada, ese punto será centro de una circunferencia de radio R.

Figura 13.- Ejemplo de barrido de radios en la imagen de radar a partir de un centro (x,y).

En nuestra implementación hemos establecido los valores de máximo y mínimo del radio de forma que abarcan toda la imagen, empezando con r_{min} cero y terminando con r_{max} la mitad del lado más corto de la imagen con el fin de asegurarnos que las circunferencias sean identificadas dentro de la imagen.

Una vez terminado el barrido en el rango establecido, se va a realizar una comprobación para asegurar que ningún círculo se ha contabilizado más de una vez. Como el salto entre radios es variable y no conocemos el grosor de los círculos tras la extracción de bordes, existe la posibilidad de que exista una detección múltiple del mismo círculo. Para solventarlo, se va a recorrer el vector de radios y se van a agrupar los radios cuya separación sea menor que un umbral establecido, ya que se considerará que están demasiado cercanos y corresponden a la misma circunferencia. Cuando se supera ese umbral se considera ya un círculo nuevo. Una vez se tienen agrupados los radios, de cada grupo se selecciona aquel que mayor porcentaje de puntos de borde posee, pues mayor cantidad de puntos caen en un borde y por lo tanto la precisión de detección es mayor, y de ahora en adelante se considera que dicha circunferencia tiene ese radio.

3 Algoritmos para la detección de patrones de la escala del radar

El reconocimiento de caracteres en imágenes es tan importante como la detección de formas geométricas, y aunque también se trata de reconocer objetos en una imagen, las técnicas utilizadas en el apartado anterior no son adecuadas para este problema. Como podemos ver en la Figura 14 la detección de bordes convierte los caracteres de la escala en una amalgama de bordes que difícilmente podrían ser interpretados como números.

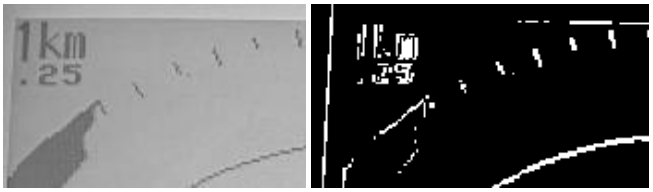


Figura 14.- Caracteres de la escala antes y después de la detección de bordes.

Como se dijo en la sección 1.2.2, para el reconocimiento de patrones existen tres grandes familias de soluciones: identificación de plantillas, técnicas probabilísticas y redes neuronales. La utilización de unas u otras depende en gran medida de la información que tenemos de inicio para lidiar con el problema, aunque muchas veces se pueden utilizar indistintamente, y será la respuesta que obtengamos de cada una la que nos haga elegir la más adecuada para cada caso.

En nuestro caso hemos elegido la técnica de identificación de plantillas debido a la necesidad de diseñar un sistema rápido, fiable y escalable, de manera que pueda ser utilizado para distintos modelos de radar. A continuación vamos a explicar el funcionamiento de dicho procedimiento.

3.1 Identificación de plantillas (*Template matching*)

La identificación de plantillas es una técnica utilizada en el procesamiento de imágenes para buscar un patrón determinado dentro de una imagen de tamaño superior. Para realizar dicha búsqueda se parte de una serie de plantillas que representan los patrones que estamos buscando. Esta primera consideración nos obliga a disponer de información basada en la experiencia previa, es decir, necesitamos disponer de los patrones a buscar y tener a nuestra disposición plantillas iguales o similares a dichos patrones para poder realizar la búsqueda.

El objetivo de las técnicas de identificación de plantillas es determinar dentro de una imagen la posición cuya semejanza a la plantilla es máxima. Para ello, se aplican una serie de operaciones matemáticas sobre el conjunto de píxeles que definen la plantilla y la imagen. Antes de llevar a cabo esta comparación, es necesario añadir una serie de restricciones como que la imagen original tenga un tamaño superior al de la plantilla, o que la iluminación entre la imagen y la

plantilla sea similar, ya que si no se tienen en cuenta estas condiciones, la identificación de la plantilla sobre la imagen original no sería posible.

Dado que con esta técnica lo que se pretende es comparar la imagen original f y la plantilla w para evaluar el grado de similitud entre ambas, el algoritmo utilizado recorre la imagen objetivo en busca de la plantilla. En este recorrido, se va calculando la diferencia entre ambas imágenes mediante métodos de cálculos de similitud, donde los más utilizados son la suma de diferencias absolutas (SAD) (Steinmetz, 2000), la suma de diferencias cuadradas (SSD) (Lehmann y Casella, 1998; Wackerly y col., 2008). y la correlación cruzada normalizada (NCC) (Lewis, 1995; Haralick y Linda, 1992).

Para llevar a cabo este proceso se ha de tener las dos imágenes, la original f con unas dimensiones *NxM*, y la plantilla a identificar *w* con sus dimensiones *JxK*.

Partiendo de un origen conocido (x_o, y_o) y para cada píxel de la imagen original, se superpone la plantilla en esa posición y se desplaza la plantilla adquiriendo los valores de similitud de ambas imágenes, a los cuales se les aplica una función F (Figura 15).

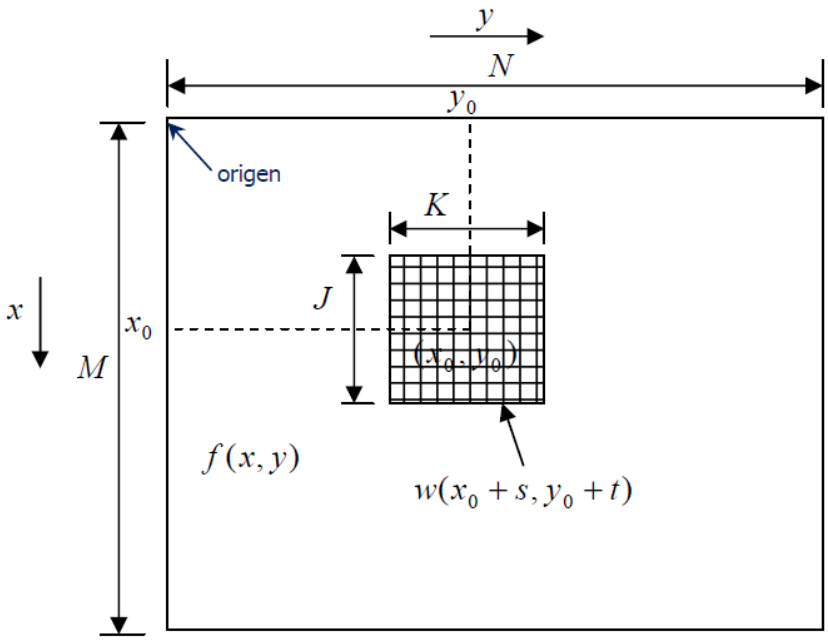


Figura 15.- Representación gráfica de la identificación de plantillas.

Donde $x \in [0, M-1]$; $y \in [0, N-1]$; $s \in [0, J-1]$; $t \in [0, K-1]$, siendo t el desplazamiento en el eje horizontal y s el desplazamiento en el eje vertical. Existen varias funciones F que se pueden aplicar como puede consultarse en (Ahuja, 2010).

Una vez calculada la función para cada punto de la imagen original, dependiendo de dicha función, tendremos que buscar el máximo valor o el mínimo, siendo dicho punto el elegido por el algoritmo para situar en esa posición el patrón seleccionado.

Esta técnica es muy útil en caso de que tengamos acotados los posibles valores que estamos intentando identificar y dispongamos de plantillas para cada uno de ellos. En caso contrario, el

alto coste computacional resultado de recorrer la imagen entera buscando el patrón puede hacer que esta técnica sea inviable. Dado que el radar FURUNO 1715 tiene limitados los valores de funcionamiento, como puede verse en la Tabla 1, esta técnica es una opción viable para el problema que se aborda en este trabajo.

Tabla 1.- Posibles valores de funcionamiento del radar FURUNO 1715.

Rango(nm/km)	0.125	0.25	0.5	0.75	1	1.5	2	3	4	6	8	12	16	24	36
Intervalo de anillos(nm/km)	0.0625	0.125	0.125	0.25	0.25	0.5	0.5	1	1	2	2	3	4	6	12
Número de anillos	2	2	4	3	4	3	4	3	4	3	4	4	4	4	3

Para nuestro trabajo hemos elegido dos técnicas diferentes a fin de poder comparar resultados, la correlación cruzada normalizada (NCC) y la suma de diferencias cuadradas (SSD).

3.1.1 Correlación cruzada normalizada

La correlación cruzada es una medida de la similaridad entre dos imágenes con un desfase relativo entre ellas. La ecuación (8) muestra cómo se calcula, referenciada según los datos de la Figura 15, donde vemos que el desfase entre la imagen $f(s,t)$ y la plantilla $w(x+s,y+t)$ es el punto donde medimos la correlación, (x,y). En su naturaleza y definición es similar a la convolución de dos funciones.

$$NCC(x,y) = \sum_s \sum_t f(s,t)w(x+s,y+t) \qquad (8)$$

Es una técnica muy utilizada en imágenes cuando la variabilidad del brillo y la iluminación entre la imagen y la plantilla es grande (Lewis, 1995). Para evitar verse afectada por esas condiciones, las imágenes primero son normalizadas (restando la media y dividiendo por la desviación estándar) y posteriormente se evalúa su correlación mutua.

Al haber sido normalizada, el resultado obtenido de esta función varía entre 0 y 1, siendo 0 cuando ambas imágenes no tienen ningún pixel en común y un valor máximo de 1 cuando son idénticas. Por lo tanto lo que buscaremos será maximizar la salida para que el parecido entre ambas imágenes sea el mayor posible.

3.1.2 Suma de diferencias cuadradas

La suma de diferencias cuadradas es otro de los métodos más importantes de reconocimiento de patrones y su funcionamiento es aún más intuitivo que la correlación cruzada. Como vemos en la ecuación (9), consiste en calcular las diferencias entre ambas imágenes evaluadas. Se realiza una resta entre el píxel $f(s,t)$ y el $w(x+s,y+t)$, donde se puede apreciar una diferencia entre puntos de (x,y), como en el método anterior. El resultado de esa resta es el

error entre ambas imágenes, y se eleva al cuadrado para que todos los valores se contabilicen en valor absoluto. De lo contrario, podría ocurrir que errores de signo opuesto se cancelasen y el resultado diera que ambas imágenes tuvieran menos diferencias de las que realmente existen.

$$SSD(x,y) = \sum_s \sum_t (f(s,t) - w(x+s, y+t))^2 \qquad (9)$$

Como en este caso la función nos devuelve las diferencias entre la imagen y la plantilla, el punto que buscaremos será aquel en el que el error sea menor. Como comprobaremos con los resultados obtenidos, al no realizar el proceso de normalización este sistema es más sensible a los cambios provenientes de las condiciones ambientales.

3.2 Elección del algoritmo de detección de escala

Con el fin de seleccionar el método que mejor pueda llevar a cabo la identificación de la escala a la que está trabajando el radar, en el menor tiempo posible y con la menor cantidad de recursos, en primer lugar hay destacar que la variedad de plantillas que vamos a poder detectar viene delimitada por la cantidad de escalas del radar. Los posibles valores que vamos a encontrar son números que presentan 10 valores diferentes, y el punto, lo que hacen un total de 11 plantillas. Como además los números son de dos tamaños diferentes, suponiendo que la misma plantilla no sirva para detectar ambos tamaños, nos encontraríamos con un total de 21 patrones. Esta es una cantidad de casos asumible.

En segundo lugar hay que tener en cuenta que la posición de la escala es fija, siempre se encuentra en la esquina superior izquierda de la imagen. Este hecho hace que la búsqueda sea mucho más acotada y reducirá en todo caso el tiempo de procesamiento del algoritmo elegido, ya que tras la detección de las circunferencias concéntricas es posible delimitar la zona en la que identificar la plantilla.

La información de entrada son una serie de capturas de la cámara situada sobre el radar, que utilizaremos a lo largo de todo el proceso. De dichas imágenes podemos sacar las plantillas en las que se encuentran todos los patrones posibles. Dadas las características del problema y la información disponible, parece un caso bastante adecuado para el uso de las técnicas de identificación de plantillas estudiadas en la sección 3.1. Tanto las soluciones probabilísticas como las redes neuronales explicadas en la sección 1.2.2, requieren de un entrenamiento previo para optimizar sus salidas. Además, en el primer caso, sería necesario conocer información sobre las densidades de probabilidad de los patrones, información de la que no disponemos. En cuanto a las redes neuronales, sería posible obtener una solución tras la etapa de entrenamiento si la cantidad de imágenes que tenemos para entrenarla es suficiente, pero requeriría un coste temporal mayor.

De esta forma, se van a utilizar las técnicas de reconocimiento de plantillas para detectar la escala de la imagen. Posteriormente se comparará la cantidad de círculos que teóricamente debería haber según la plantilla con los círculos que se han detectado mediante las técnicas propuestas de detección de círculos concéntricos con el fin de validar la solución final.

4 Resultados

En este capítulo se van a mostrar los resultados obtenidos con cada una de las técnicas descritas en los capítulos dos y tres para diferentes situaciones y configuraciones posibles del radar. En primer lugar se mostrará el escenario en el que se realizaron las pruebas, para a continuación exponer de forma ordenada con gráficas y tablas las soluciones obtenidas.

4.1 Escenario de pruebas y material utilizado

Para la adquisición de imágenes se utilizó una Webcam modelo Logitech HD Pro Webcam C920, que permite vídeo con una calidad de alta definición a 1080p y una tasa de adquisición de 30 fps. Las imágenes tienen un tamaño de 1920 x 1080 píxeles en el formato de color RGB con una representación de 24 bits por píxel en el formato BMP.

Para el desarrollo de los códigos se ha utilizado la *toolbox* de procesamiento de imagen de Matlab 2014b (Mathworks, 2014). El ordenador que se empleó para realizar los cálculos de tiempos y obtener las gráficas es un Inter Core i7-3770k 3.5GHz con 32 Gb de RAM, utilizando Windows 7 Professional.

4.2 Resultados de detección de círculos concéntricos

4.2.1 Resultados de la detección de bordes

Como ya hemos explicado, el primer paso es la detección de los candidatos a centro de todas las circunferencias. Para ello vamos a partir de las imágenes de radar que tenemos como base de datos. Dichas imágenes son en color, y además presentan brillos y reflejos sobre la superficie que pueden dificultar la selección de puntos. La Figura 16 muestra la gran varianza de valores de los píxeles que presenta la imagen original.

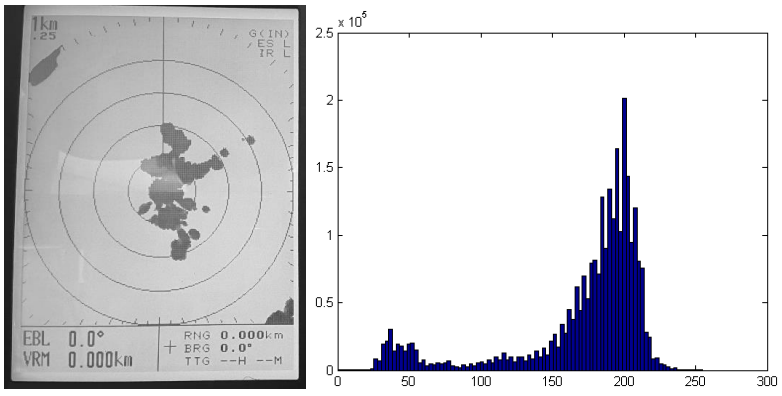
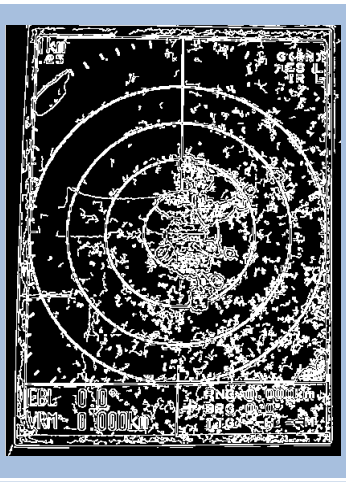
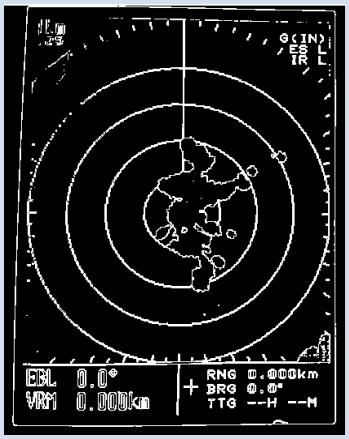
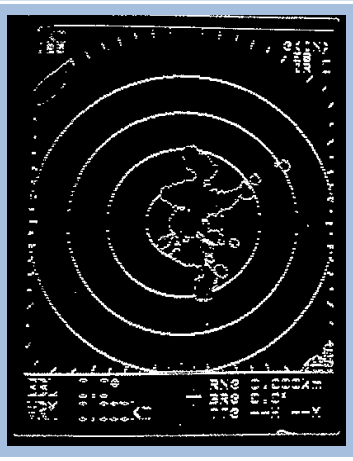
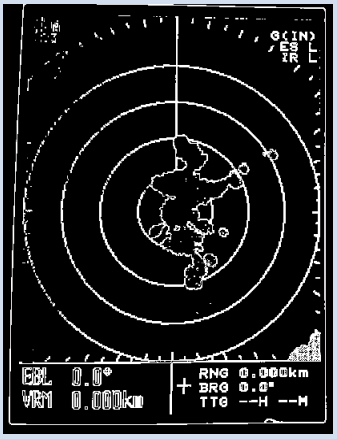
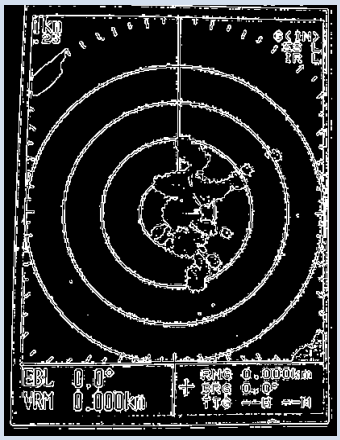
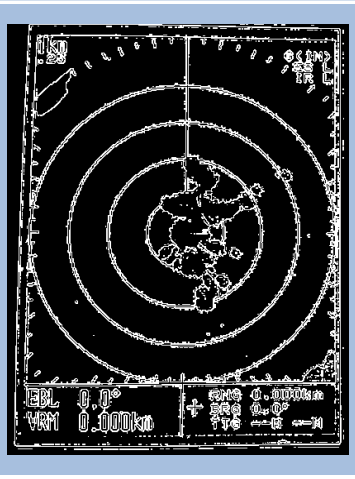


Figura 16.- Imagen original del radar y su histograma.

Para reducir esa varianza de valores, y porque muchos algoritmos de detección de círculos exigen una imagen binarizada de entrada, se va a preprocesar la imagen utilizando un método de detección de bordes. Para este proceso vamos a utilizar la función *edge* de Matlab. La nueva imagen va a poder separarse en diferentes bordes, que serán las fuentes a partir de las cuales trataremos de encontrar los centros de las circunferencias. En la Tabla 2 se muestran los resultados obtenidos con cada uno de los operadores que permite elegir la función, así como el borde más significativo por ser el que mayor cantidad de píxeles contiene.

Tabla 2.- Resultados obtenidos con cada uno de los algoritmos de detección de bordes a partir de la figura 16.

Algoritmo de detección de bordes	Resultado del algoritmo	Borde significativo
Canny		
Sobel		
Sobel horizontal		

Roberts		
Prewitt		
Log		
Zerocross		

Como se exponía en el análisis teórico, aunque la filosofía de todos los sistemas utilizados es similar, los operadores matemáticos utilizados en cada caso provocan que los resultados obtenidos difieran notablemente. También se apuntaba en capítulos anteriores que no hay un método sobresaliente en todo caso. Mientras que en el ejemplo observado en el capítulo dos, apartado 2.1.1, parecía que la solución de Canny era mejor que la de Sobel, en este caso es al contrario. Ahora observamos que al menos 3 métodos, Sobel, Roberts y Prewitt, presentan una respuesta similar que es la mejor de todos los casos para el problema que se aborda en este trabajo.

Para poder elegir el método que mejor se adapta a nuestro entorno de trabajo vamos a utilizar los bordes detectados como discriminante visual. Observamos que en todos los algoritmos genéricos y obteniendo los gradientes en ambas direcciones (horizontal y vertical), el borde principal obtenido incluye prácticamente toda la información relevante de la imagen, desde todos los círculos existentes, hasta ejes o cuadrículas de leyenda pertenecientes a la forma de mostrar la información del radar.

Esta característica resulta un inconveniente en nuestro caso, ya que los ejes que sirven para escalar la imagen y ver representada correctamente la información en pantalla, atraviesan todas las circunferencias. Dado que el funcionamiento básico del algoritmo que se ha diseñado para detectar centros de círculos concéntricos se basa en recorrer los bordes buscando los posibles centros de cada uno, este comportamiento supone un grave inconveniente ya que, tras el etiquetado de las zonas de la imagen, el algoritmo detectará todas las circunferencias como un único borde al estar conectadas entre sí. Por esta razón, se han utilizado opciones específicas dentro de los casos generales.

Se ha elegido el algoritmo de Sobel horizontal, en el que sólo se convoluciona la imagen con el núcleo horizontal del operador, lo que permite eliminar los bordes verticales. El resultado lo podemos apreciar en la misma tabla 2 (Sobel horizontal). Como se puede ver, la imagen obtenida es de la misma calidad que con el operador de Sobel, pero si nos fijamos en el borde vertical que une todas las circunferencias, se observa que este algoritmo permite eliminar el problema de que el eje principal unía todas las circunferencias, lo que nos permitirá obtener los centros con mayor fiabilidad.

Además, a la hora de analizar los bordes, se desechan los que tienen un tamaño demasiado pequeño para obtener los tres puntos a la suficiente distancia según el método explicado en la sección 2.2.1.

La Tabla 3 muestra que el método con la menor cantidad de bordes válidos con este criterio de selección es el Sobel horizontal. Esto indica que dicho método deberá analizar una menor cantidad de bordes para obtener el mismo centro que el resto de técnicas.

Tabla 3.- Número promedio de bordes válidos para la detección de círculos en cada detector de bordes.

	Canny	Sobel	Sobel horiz.	Roberts	Prewitt	Log	Zerocross
Número de bordes válidos	92	48	30	46	51	50	50

Otra problemática a tener en cuenta es el grosor del borde. En las imágenes anteriores observamos que los bordes son bastante gruesos. Eso es debido a que hemos aplicado una operación de dilatación (*dilate*) sobre los bordes a la hora de obtenerlos. La razón por la que se ha aplicado es que si no los bordes que se obtenían quedaban muy desdibujados e inconexos entre sí lo que dificultaba la detección de círculos, como se pone de manifiesto en la Figura 17.

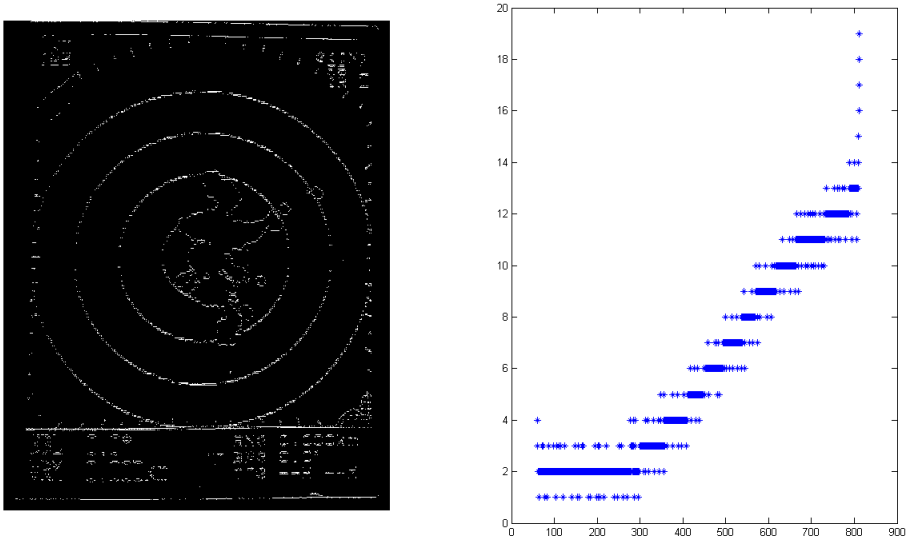


Figura 17.- Detección con Sobel horizontal sin dilatación (izquierda) y ejemplo de borde obtenido (derecha).

De esta forma se soluciona el problema de la integridad de los bordes. Sin embargo añadimos más píxeles como entrada a nuestro algoritmo.

4.2.2 Ajuste paramétrico de los algoritmos

Una vez hemos sido capaces de separar los bordes del fondo de la imagen antes de empezar a probar los algoritmos para obtener resultados definitivos se realizará un estudio mediante el cual se obtienen los mejores valores para los parámetros de configuración del algoritmo siempre con el fin de que éste sea lo más rápido posible sin perder precisión en la detección.

En cuanto a la solución propuesta por Coope (Coope, 1993) de minimización de diferencias cuadradas mediante el ajuste de mínimos cuadrados de una circunferencia a partir de un conjunto de puntos, al no tener posibilidades de configuración porque utiliza como entrada únicamente los datos del conjunto de puntos que definen cada borde, no es necesario que se estudie cómo varía la solución en función de los parámetros del método.

4.2.2.1 Algoritmo de detección del centro a partir de tres puntos

El algoritmo de detección de centros diseñado recorre cada borde válido en busca de ternas de puntos que cumplan la condición de distancia mínima entre ellos, así que cuantos más puntos mayor tiempo de computación. El engrose de los bordes genera muchos puntos que se encuentran contiguos a los ya existentes, por lo que evaluar todos supone en muchos casos seleccionar algunos que están prácticamente al lado de los anteriores. Esto crea un sobremuestreo de puntos que no aportan gran información y que ralentizan el proceso. Para evitar esta situación, cada vez que evaluamos un punto inicial y encontremos otros dos que cumplan también las condiciones iniciales del algoritmo, el siguiente punto inicial se elegirá a una distancia $\delta_{inicial}$ que se determinará de forma empírica a través de las pruebas realizadas. La distancia que separa los tres puntos entre sí es la especificada en la ecuación (5), d_{min}. El objetivo es encontrar el número suficiente de centros que nos permitan encontrar aquel que sea correcto, pero sin que ese número sea tan elevado como para incrementar mucho el tiempo necesario de búsqueda.

Para realizar el ajuste vamos a ejecutar el algoritmo sobre varias imágenes significativas de la base de datos del radar. Para comparar las diferentes configuraciones se va a realizar el proceso completo, incluyendo la obtención de bordes y la detección de los círculos concéntricos. Esto se debe a que, por un lado, la detección de bordes es prácticamente instantánea y no afecta al total, y, por otro lado, a que no se puede simplemente comparar el tiempo de computación del algoritmo de detección de centros por separado, ya que el número de centros encontrados en cada caso puede hacer que una configuración más rápida individualmente conlleve mayor trabajo para el algoritmo de detección de círculos concéntricos y, en el proceso completo sea peor solución que otra más lenta por separado pero que detecte menos centros. Esta consideración se realizará para los ajustes de todos los algoritmos de detección de centros.

Para seleccionar la mejor configuración, vamos a realizar diferentes simulaciones del algoritmo con diferentes valores para $\delta_{inicial}$ y d_{min}. De cada ejecución compararemos el tiempo de ejecución, los centros detectados y los centros realmente válidos. Como ya se ha dicho, el tiempo de ejecución incluye el proceso completo. Los centros detectados son los centros que se obtienen como posibles candidatos a la salida del algoritmo de detección de centros, y los centros válidos son aquellos centros que han conseguido detectar todos los círculos presentes en la imagen. En la Figura 18 podemos realizar el seguimiento del proceso total que se ha seguido: partiendo de la imagen inicial (a), se detectan para cada borde los centros candidatos (b). Una vez se han evaluado todos los bordes, se aplica el algoritmo de detección de círculos y obtenemos los centros válidos para detectar todos los círculos de esta imagen (c), así como la estimación del radio de dichos círculos (d).

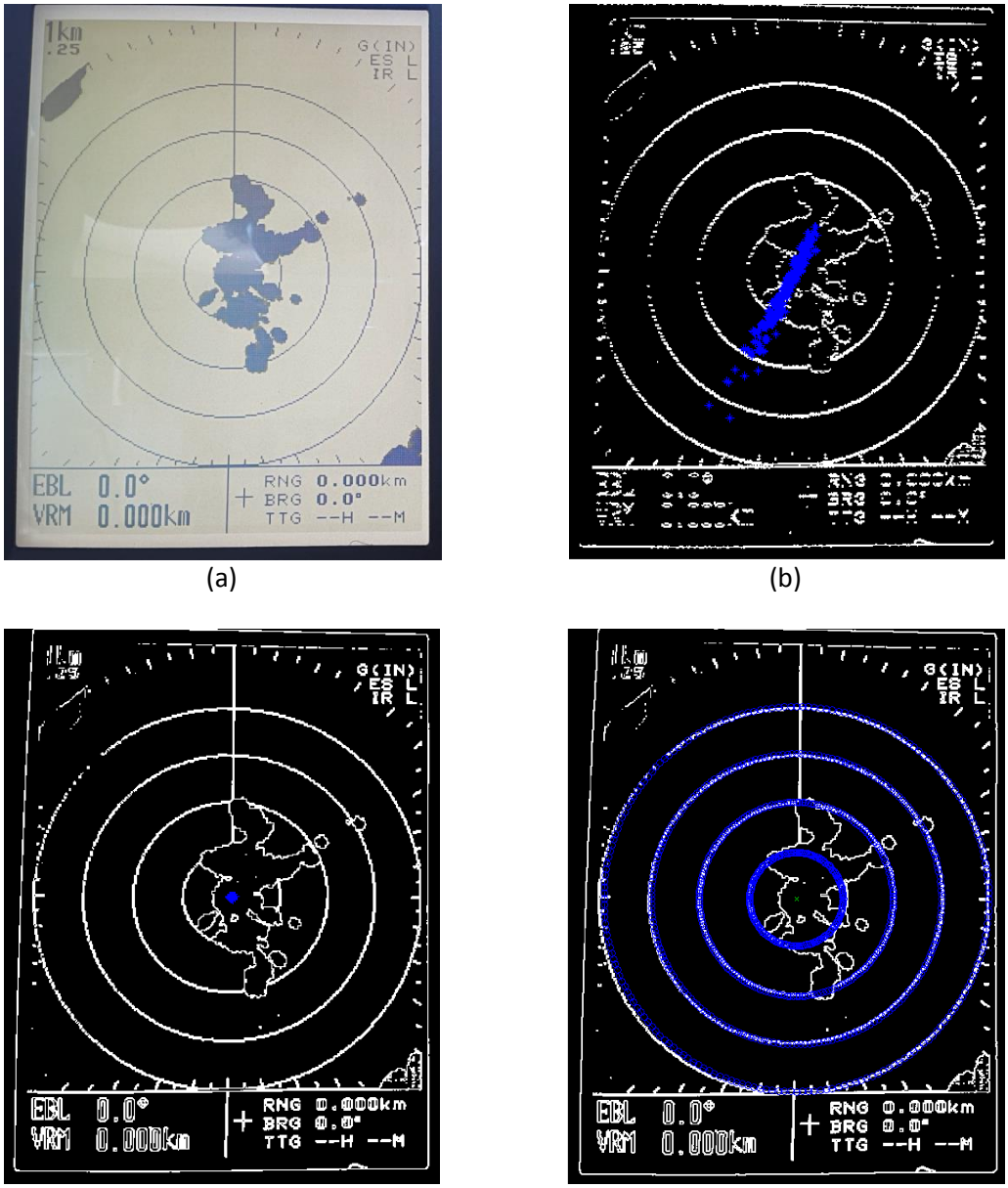


Figura 18.- a) Imagen original b) Centros detectados en la imagen de bordes. c) Centros válidos. d) Círculos detectados a partir de los centros válidos.

Una vez descrito el proceso a seguir, los resultados de este ajuste paramétrico son los mostrados en la Tabla 4. En esta tabla se puede observar, para cada configuración determinada, la cantidad promedio de centros detectados, la cantidad promedio de centros válidos, y el tiempo promedio en segundos que tarda el proceso en detectar los círculos.

Tabla 4.- Relación comparativa de las diferentes configuraciones testeadas.

	Tiempo ejecución (s)	Centros detectados	Centros válidos
$\delta_{inicial}$=200 d_{min}=100	2619	2125	29
$\delta_{inicial}$=200 d_{min}=200	2059	1775	12
$\delta_{inicial}$=200 d_{min}=300	6692	48	0
$\delta_{inicial}$=300 d_{min}=200	1871	1775	12
$\delta_{inicial}$=400 d_{min}=200	1661	1775	12
$\delta_{inicial}$=500 d_{min}=200	1514	1775	12
$\delta_{inicial}$=800 d_{min}=200	1222	1775	12
$\delta_{inicial}$=1000 d_{min}=200	1206	1775	12
$\delta_{inicial}$=1500 d_{min}=200	857	1572	2

En la Tabla 4 observamos que en las tres primeras filas se presentan las variaciones en la d_{min} dejando la $\delta_{inicial}$ constante. A medida que aumenta el valor de la distancia, la cantidad de centros detectados y válidos disminuye, hasta llegar a un punto donde para una d_{min} de 300 píxeles no se detecta ningún centro válido. El resto de casos corresponden a las variaciones en el valor de la $\delta_{inicial}$ dejando d_{min} constante, donde se observa que a medida que aumenta ese valor, sólo se ve afectado el tiempo de ejecución. Hay que irse al caso más extremo para ver reducida la detección de círculos hasta casi no obtener ningún centro válido.

Es decir, el cambio de d_{min} afecta mucho más que el de $\delta_{inicial}$. Esto es debido a que $\delta_{inicial}$ realiza un desplazamiento por el vector de puntos del borde. Con el engrosamiento de bordes, esto implica que saltaremos los puntos más cercanos del borde y empezaremos a buscar la terna más alejada del punto anterior. Sin embargo el aumento de d_{min} implica que la distancia entre sí de los putos elegidos en la imagen debe ser como mínimo d_{min}, aunque en el vector de puntos del borde eso puede significar recorrer un número de puntos mucho mayor. Si aumentamos mucho este valor, será imposible detectar circunferencias cuyo radio sea menor que d_{min}. Por eso a medida que se aumenta este valor se deja de detectar circunferencias.

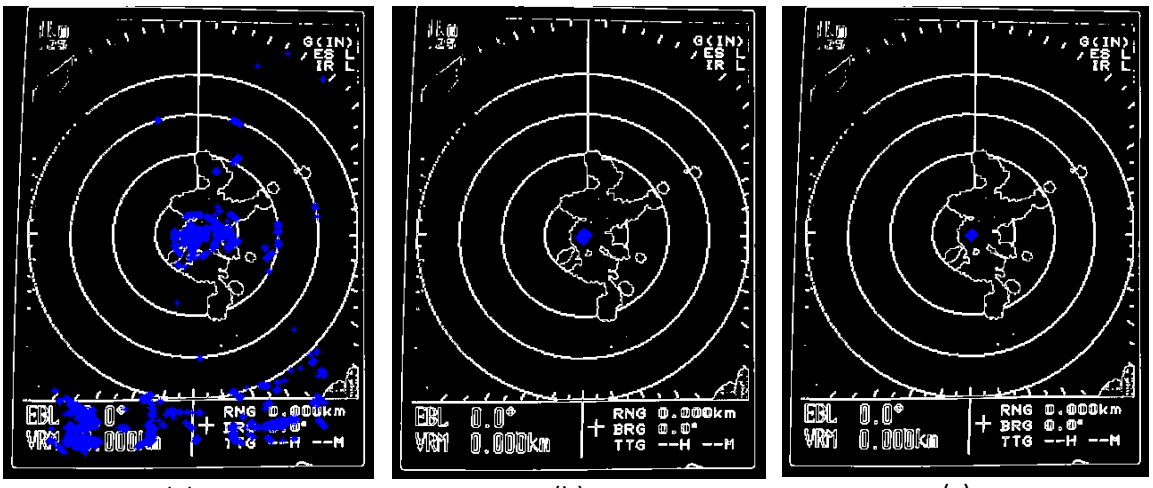
A la vista de los resultados, la configuración elegida para el resto de pruebas ha sido $\delta_{inicial}$=800, d_{min}=200. Hemos descartado el caso más rápido ($\delta_{inicial}$=1000) porque al detectar sólo 2 centros de media, nos arriesgamos a que en otras imágenes con más ruido no detectemos ninguno. A pesar de que la para $\delta_{inicial}$=1000 los tiempos son ligeramente inferiores y la detección de centros es la misma, nos hemos decidido por un caso algo menos extremo porque la reducción de tiempos es prácticamente inapreciable, y, en situaciones ruidosas, las probabilidades de fallo del caso elegido serán menores.

Para las pruebas posteriores ejecutaremos dos versiones de nuestro algoritmo, una con el algoritmo de Bersenham (Foley, 1980) incluido y otra sin él, de forma que podamos comparar resultados. Además se ejecutarán bajo dos configuraciones diferentes: a) $\delta_{inicial}$=800, d_{min}=200, que es la elegida como óptima, y b) $\delta_{inicial}$=200, d_{min}=100, que es más lenta, pero al detectar más centros consideramos que obtiene la mejor solución.

4.2.2.2 Transformada circular de Hough

Para la implementación de la transformada circular de Hough se han estudiado tres algoritmos diferentes: la función *imfindcircles* (encuentra círculos en imágenes) de Matlab (MathWorks), el algoritmo desarrollado por Tao Peng en (Peng, 2005), y la solución propuesta por Peter Bone (Bone, 2010), las cuales ya se han mencionado en capítulos anteriores. En la primera solución no hay ajuste posible, ya que sólo nos deja establecer el rango de radios a buscar, así que pasaremos a explicar las otras dos soluciones.

El algoritmo de Bone nos permite acotar el rango de radios de las circunferencias que vamos a buscar y, además, podemos configurar un umbral que determine el mínimo número de puntos que deben pertenecer a un borde para considerarlo como un círculo real de la imagen, así como la región de la imagen donde se va a realizar la búsqueda. Como en nuestro caso los círculos se pueden encontrar a lo largo de toda la imagen, ese último parámetro lo dejaremos por defecto para que analice la imagen completa. Respecto al umbral de decisión, vamos a realizar diferentes simulaciones para observar a través de imágenes cómo afecta al resultado final.



(a) (b) (c)

Figura 19.- Centros resultantes con un umbral de decisión a) r*pi*0.5, b) r*pi*0.8 y c) r*pi*1.

En la Figura 19 se muestra cómo al ir aumentando el umbral necesario para detectar una circunferencia, los centros detectados, además de disminuir, son más precisos. Como los umbrales establecidos van en relación al radio de la circunferencia buscada, en zonas con muchos bordes se pueden detectar los puntos necesarios para que se detecten circunferencias pequeñas que en realidad no existen. Por eso, al aumentar el umbral todos los puntos alejados del centro y que no interesan a nuestro sistema desaparecen. Además los tiempos de ejecución de todos los casos apenas difieren en unas décimas de segundo, así que no será un

factor determinante a la hora de elegir el umbral. Para las pruebas se escogió el valor *r*pi*0.8*, porque a partir de ese punto (80% de puntos detectados en la circunferencia) prácticamente la cantidad de centros es la misma y supone el umbral menos restrictivo para la máxima precisión de salida.

La solución de Peng, aparte del rango de radios a buscar, nos permite establecer algunas restricciones de entrada para configurar la búsqueda. Se puede establecer un límite para el gradiente de cada pixel a partir del cual se interpreta como borde, así como filtros para definir cómo de perfectos queremos que sean los círculos o la cantidad de ruido que se va a tolerar, y, finalmente, existe la posibilidad de configurar la separación entre radios para la detección de círculos concéntricos. En total son 4 parámetros, una cantidad que nos permite un amplio estudio.

Tras las pruebas de calibración iniciales de los parámetros de entrada adecuados para poder obtener resultados óptimos, observamos que si bien una vez configurado correctamente en imágenes con círculos pequeños el resultado era satisfactorio, en las imágenes del radar, donde las circunferencias son sensiblemente más grandes, no se obtiene ningún centro válido a pesar de probar diferentes configuraciones. En la Figura 20 puede verse este comportamiento. Para las pruebas se utilizado un rango de radios amplio que abarcara todos los posibles círculos, y se cambiaron dos parámetros: el umbral del gradiente (grdthres), que determina el gradiente mínimo de un pixel para ser considerado por el algoritmo como borde; y el radio del filtro usado para buscar máximos (fltr4LM_R), que a medida que aumenta nos permite detectar círculos menos perfectos. Como para nuestra problemática este algoritmo no tiene utilidad, queda descartada esta solución.

4.2.2.3 Detección de radios en círculos concéntricos

Para terminar esta etapa de ajustes, vamos a configurar el algoritmo de detección de círculos. Éste nos presenta la posibilidad de variar tres parámetros: 1) la precisión, un valor que nos indica el porcentaje de círculo detectado que necesitamos encontrar para determinar que se trata un círculo; 2) el incremento de radio, que determina la distancia entre un radio y el siguiente en número de píxeles; y 3) el número de puntos, que determina el número de puntos que definen cada circunferencia y que buscamos en cada iteración para determinar si es o no un círculo. Partiremos en todos los casos de una base de: precisión 35%, incremento de radio de 0.5 píxeles y un número de puntos fijo de 200 píxeles.

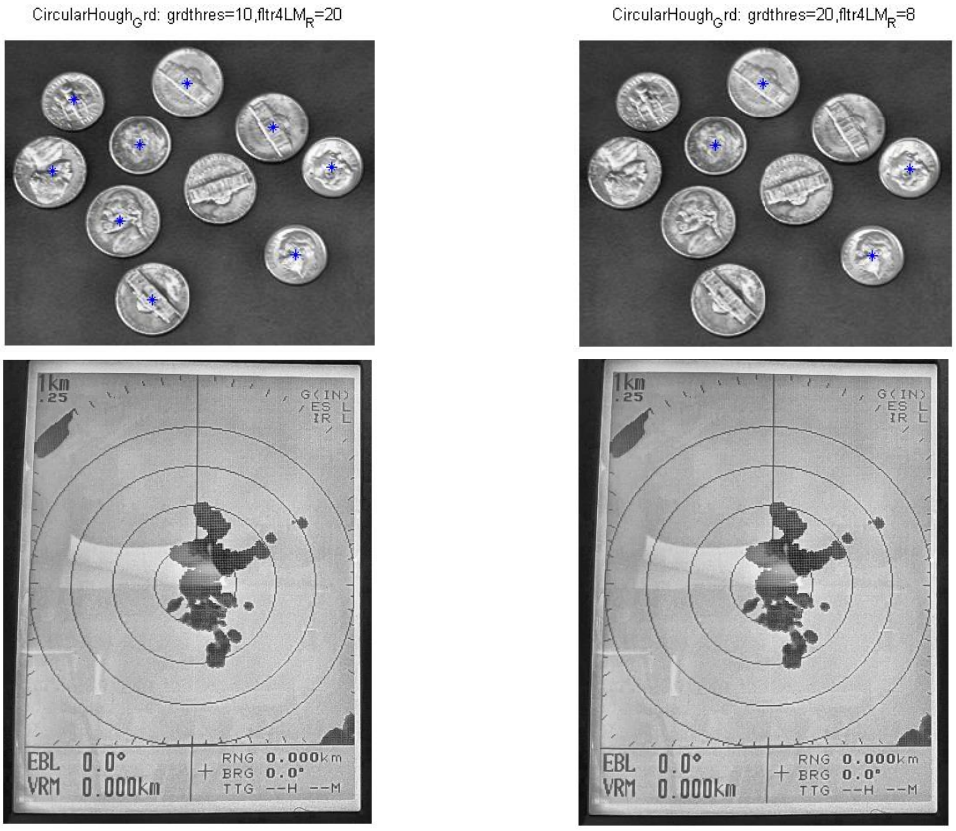
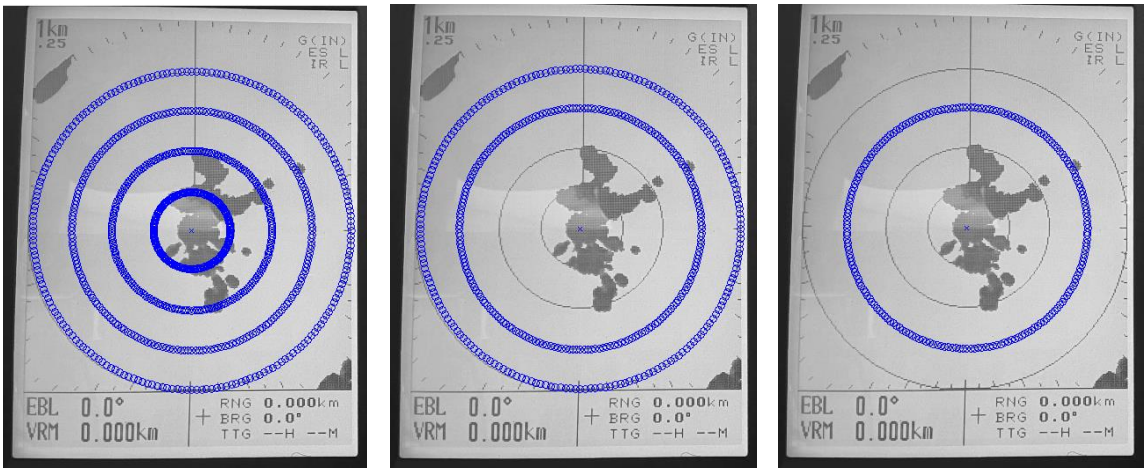


Figura 20.- Resultado obtenido con el método de Peng y diferentes configuraciones de parámetros de entrada, para circunferencias pequeñas (imágenes superiores) y para circunferencias grandes (imágenes inferiores).

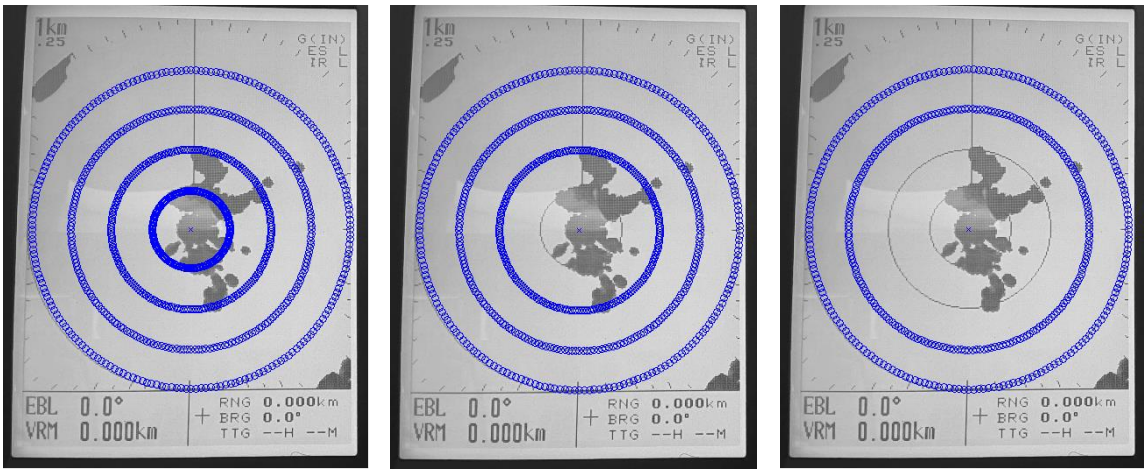
Para la precisión en la Figura 21 observamos que para un 35% detectamos todos los círculos, pero en cuanto aumentamos a 40% se pierde información, y a partir del 65% la detección de algún centro ya es esporádica, dependiendo de la imagen y los puntos de borde seleccionados por el algoritmo. Esto se debe a que el método de extracción de bordes no extrae circunferencias perfectas, sino que extrae secciones de la misma, y por lo tanto si se quiere detectar círculos que no son perfectos, no es recomendable incrementar mucho la precisión ya que el sistema descartará todas las secciones circulares que no sean círculos completos. Por lo tanto para las pruebas utilizaremos la precisión de 35% ya que es la precisión que nos permite detectar todas las circunferencias manteniendo un nivel elevado de éxito.



(a) (b) (c)

Figura 21.- Círculos detectados con una precisión de a) 35% b) 40% c) 65%.

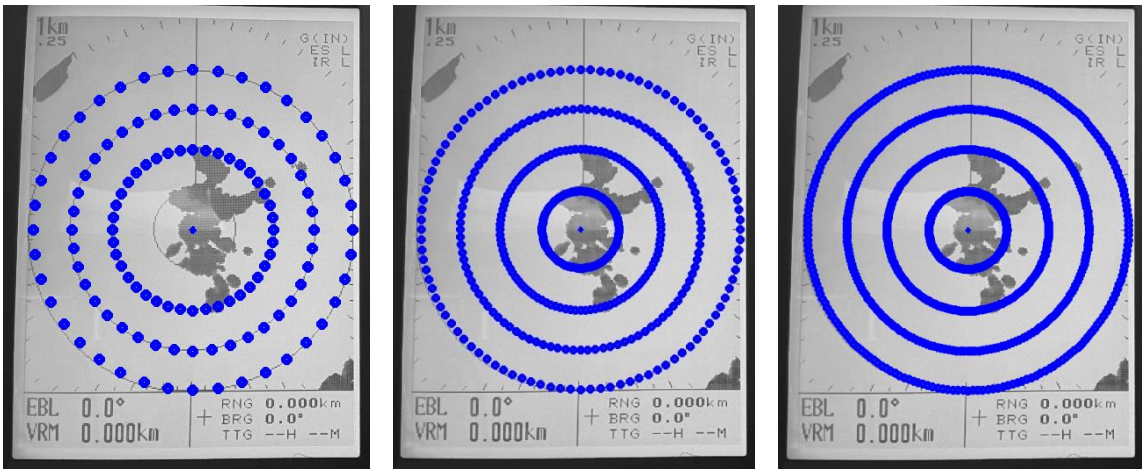
Respecto al incremento mínimo del radio que vamos a elegir, en la Figura 22 se muestran los resultados para tres configuraciones distintas, que se corresponden con un incremento de 0.5, 2 y 5 píxeles respectivamente. Podemos observar que con un incremento de 2 píxeles entre radios ya perdemos una circunferencia y, como es lógico, a medida que aumentamos ese espaciado es más probable que algún círculo no sea detectado. Cabe destacar que esta elección depende mucho del grosor de los círculos de las imágenes de bordes. En nuestro caso son bastante delgados, pero para otro tipo de imágenes, si fueran más gruesos, se podría incrementar más este parámetro sin dejar de detectar ninguna circunferencia. Como los tiempos de ejecución de cada prueba son muy parecidos, situándose las diferencias en unos 0.5 segundos, se considera que es un valor despreciable en el cómputo global del proceso y se elige finalmente el incremento de 0.5 píxeles con el fin de trabajar con el valor de incremento mínimo que nos permite una mayor precisión.



(a) (b) (c)

Figura 22.- Círculos detectados con un incremento de radio en píxeles de a) 0.5; b) 2; c) 5.

Finalmente, para la determinación del número de puntos por círculo, en la Figura 23 se muestran los resultados obtenidos cuando representamos cada uno con 40, 100 y 200 puntos respectivamente. La primera de las imágenes nos muestra unos círculos que sólo se pueden intuir al unir los puntos, y que falla al detectar el círculo más pequeño debido a que la figura que oculta parte del mismo unido a los escasos puntos calculados. En el segundo caso la detección es correcta, pero la separación entre puntos aún es apreciable y podría llevarnos a detecciones erróneas sobre todo a medida que aumenta la precisión. La última imagen sí que muestra prácticamente una circunferencia continua, además de detectar todos los círculos. Por ello, vamos a utilizar la configuración de 200 píxeles por circunferencia para asegurar una mayor precisión a la hora de realizar el resto de pruebas.



(a) (b) (c)

Figura 23.- Círculos detectados para un número de pixeles de a) 40 b) 100 c) 200.

4.2.3 Resultados de las pruebas

A continuación se van a exponer los resultados obtenidos con cada uno de los algoritmos seleccionados al analizar una serie de imágenes de radar obtenidas con la webcam anteriormente mencionada. Para la comparación de dichos algoritmos se han seleccionado diferentes resultados obtenidos con imágenes con varias condiciones, como diferente número de circunferencias, diferentes niveles de ruido y oclusión de los círculos, o diferentes niveles de luminosidad.

De los métodos ajustados en el apartado anterior, se ha comentado que el algoritmo de Peng no era posible utilizarlo porque en la etapa de configuración para imágenes grandes no fue capaz de detectar ningún círculo con ninguna de las parametrizaciones probadas. Otro de los métodos que se ha comentado es el proporcionado por Matlab a través de la función *imfindcircles*, que sólo permite seleccionar el rango de radios, y dado que buscamos a lo largo de toda la imagen, no hay posibilidad de ajuste en este aspecto.

Sin embargo, los resultados obtenidos no fueron los esperados. Después de múltiples pruebas iniciales con una imagen estándar de Matlab, en las que la detección de centros y radios fue perfecta, probamos con nuestras imágenes de radar y comprobamos que la función no devolvía ningún centro como solución, como puede apreciarse en la Figura 24. Como esta función no permite ningún tipo de configuración con el fin de obtener mejores resultados, se descarta también este método.

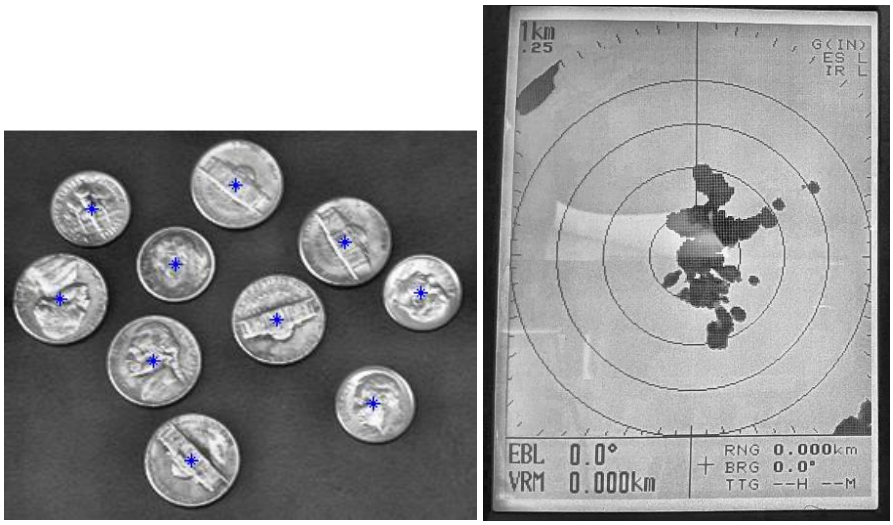
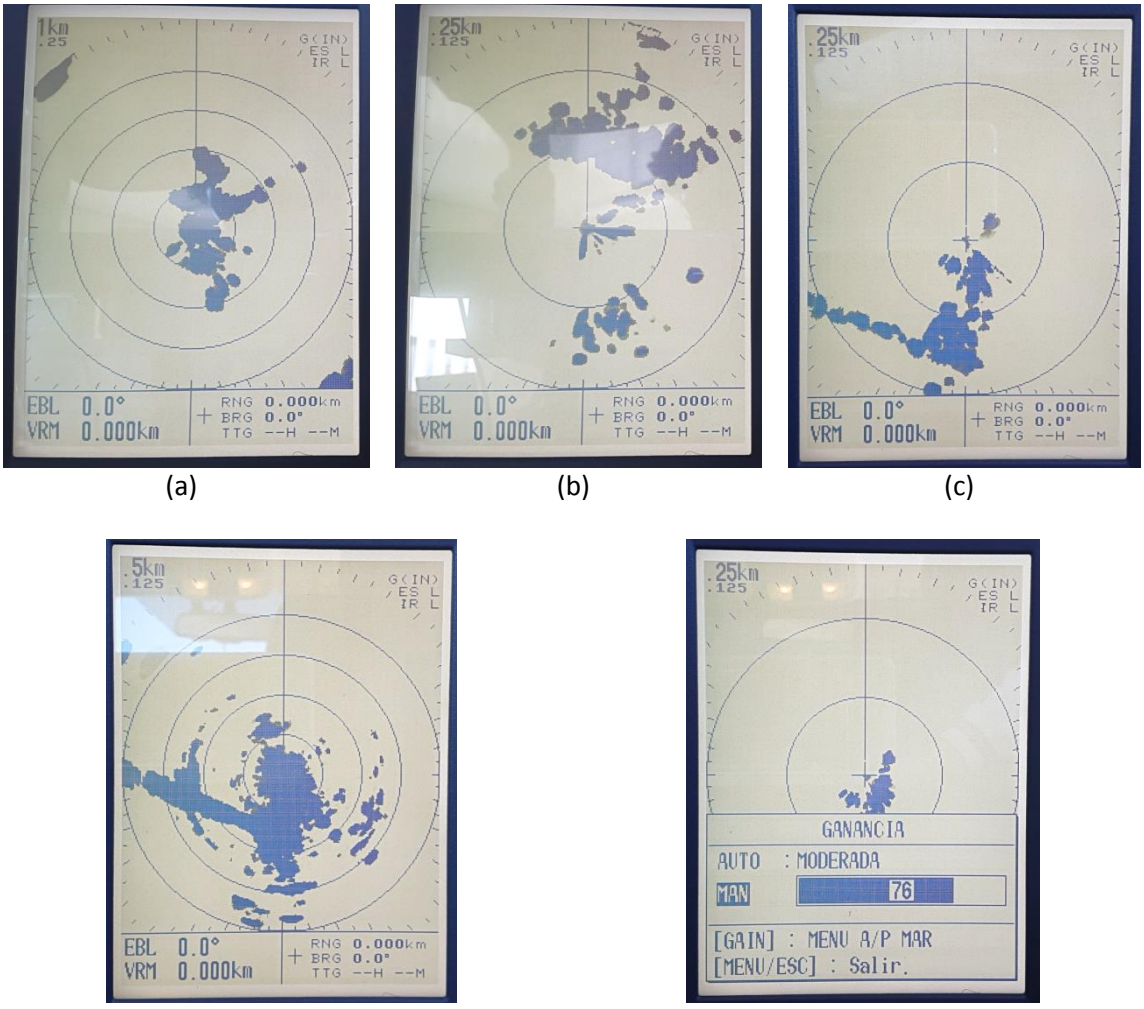


Figura 24.- Resultado de la función *imfindcircles* para la imagen de Matlab (izquierda) y para la imagen de radar (derecha).

En definitiva, en base a las consideraciones mencionadas, para las pruebas finales se van a utilizar los siguientes métodos: el algoritmo que hemos diseñado basándonos en la idea de (Silveira, 2005); la transformada circular de Hough a través del algoritmo de Peter Bone (Bone, 2010), y la solución propuesta por Coope (Coope, 1993) de la detección del centro mediante la minimización de las sumas cuadradas. Además con nuestro algoritmo se van a comprobar las soluciones cuando incluimos el algoritmo de Bersenham y la comparación de ángulos entre segmentos, y cuando no lo utilizamos, así como dos configuraciones distintas de parámetros: la más óptima, $\delta_{inicial}$=800, d_{min}=200, y la más conservadora, $\delta_{inicial}$=200, d_{min}=100. En total, se van a estudiar cuatro resultados distintos para comparar cuál es el mejor.

En la Figura 25 se presentan cinco imágenes que se han seleccionado del conjunto total de 100 disponibles con el fin de comprobar la eficacia de cada algoritmo. De izquierda a derecha y de arriba a abajo encontramos: a) imagen con 4 círculos con bastante oclusión en los más pequeños; b) imagen con 2 círculos, algo de oclusión y, sobre todo, diferentes grados de luminosidad; c) una tercera imagen con 2 círculos y algo de ruido; d) la cuarta imagen tiene 4 círculos con el mayor ruido posible entre las que disponíamos; y e) finalmente una imagen con 2 círculos donde la mitad de la misma está tapada por una pantalla desplegable del radar, con el fin de mostrar que el radar puede ser manipulado por el usuario y el algoritmo funciona correctamente.



(a) (b) (c)

(d) (e)

Figura 25.- Imágenes utilizadas para la evaluación de los algoritmos seleccionados.

Con esta variada selección se ha cubierto lo mejor posible todas las posibles situaciones de cantidad de círculos, luminosidad, ruido y oclusión. A continuación se presentarán los resultados obtenidos en una tabla según el método y por imágenes.

Se han seleccionado los siguientes datos para realizar la comparativa: tiempo de ejecución en segundos, centros detectados por el algoritmo de detección de centros como posibles y centro válidos de círculos concéntricos de cada imagen.

Se muestran los datos recogidos en la ejecución de cada método en la Tabla 5. Primero se presentan para cada imagen, en formato de tabla, y después se mostrarán unas gráficas comparativas que permitirán evaluar el comportamiento de cada método comparado con el resto.

Tabla 5.- Resultados obtenidos para el algoritmo propio completo con la configuración $\delta_{inicial}$=200, d_{min}=100.

	Imagen 1	Imagen 2	Imagen 3	Imagen 4	Imagen 5
Tiempo ejecución (s)	5428	3159	3901	9901	1204
Centros candidatos	651	1062	2703	2996	1837
Centros válidos	14	26	176	14	206

Tabla 6.- Resultados obtenidos para el algoritmo propio completo con la configuración $\delta_{inicial}$=800, d_{min}=200.

	Imagen 1	Imagen 2	Imagen 3	Imagen 4	Imagen 5
Tiempo ejecución (s)	2500	1115	315	1284	360
Centros candidatos	892	655	445	348	303
Centros válidos	12	20	87	0	91

Como se puede observar en las tablas 5 y 6, el algoritmo propio completo, el cual incluye el algoritmo de Bersenham y la comparación de ángulos entre segmentos, muestra que con la solución conservadora de la tabla 5, el tiempo de ejecución es demasiado elevado para trabajos en tiempo real. Es mejor la propuesta mostrada en la tabla 6, la cual reduce considerablemente el tiempo y mantiene una cantidad de centros válidos suficiente en la mayoría de los casos.

Tabla 7.- Resultados obtenidos para el algoritmo propio parcial con la configuración $\delta_{inicial}$=200, d_{min}=100.

	Imagen 1	Imagen 2	Imagen 3	Imagen 4	Imagen 5
Tiempo ejecución (s)	2619	1635	1339	1587	966
Centros candidatos	2125	3244	2703	3470	1976
Centros válidos	29	40	174	14	206

Tabla 8.- Resultados obtenidos para el algoritmo propio parcial con la configuración $\delta_{inicial}$=800, d_{min}=200.

	Imagen 1	Imagen 2	Imagen 3	Imagen 4	Imagen 5
Tiempo ejecución (s)	1222	1038	257	226	335
Centros candidatos	1775	1477	445	348	303
Centros válidos	12	32	87	0	91

En las tablas 7 y 8 se pueden ver los resultados del algoritmo propio sin el algoritmo de Bersenham y sin la comparación de ángulos entre segmentos, los cuales reducen considerablemente el tiempo con respecto a las tablas 5 y 6. Como en el caso anterior, utilizando la solución conservadora de la tabla 7, el tiempo de ejecución es más elevado que utilizando la propuesta mostrada en la tabla 8, la cual reduce el tiempo y también mantiene una cantidad de centros válidos suficientes en la mayoría de los casos.

Tabla 9.- Resultados obtenidos para la transformada circular de Hough.

	Imagen 1	Imagen 2	Imagen 3	Imagen 4	Imagen 5
Tiempo ejecución (sg)	1070	1069	728	889	909
Centros candidatos	44	12	14	18	15
Centros válidos	11	8	14	12	15

En cuanto a la transformada circular de Hough, Tabla 9, se reduce el tiempo de ejecución y la cantidad de centros válidos, sin perder precisión en la detección. Pero los tiempos mostrados siguen siendo elevados para tareas en tiempo real.

Tabla 10.- Resultados obtenidos para el de detección por mínimos cuadrados.

	Imagen 1	Imagen 2	Imagen 3	Imagen 4	Imagen 5
Tiempo ejecución (sg)	32	40	32	49	24
Centros candidatos	62	80	64	109	52
Centros válidos	1	1	2	4	3

En cuanto al algoritmo de minimización de la suma cuadrada, Tabla 10, se puede ver que es el que menor tiempo de procesamiento necesita, reduciendo la cantidad de centros válidos y manteniendo la precisión en cuanto a la detección.

A continuación se muestra tres imágenes donde se pueden ver los seis métodos propuestos y los resultados obtenidos en cuanto al tiempo de procesamiento, cantidad de centros obtenidos y cantidad de centros válidos:

1. Algoritmo completo subóptimo (algoritmo de Bersenham y comparación de segmentos con $\delta_{inicial}$=200 y d_{min}=100).
2. Algoritmo completo óptimo (algoritmo de Bersenham, y comparación de segmentos con $\delta_{inicial}$=800 y d_{min}=200).
3. Algoritmo parcial subóptimo (sin algoritmo de Bersenham ni comparación de segmentos con $\delta_{inicial}$=200 y d_{min}=100).
4. Algoritmo parcial óptimo (sin algoritmo de Bersenham ni comparación de segmentos con $\delta_{inicial}$=800 y d_{min}=200).
5. Transformada circular de Hough.
6. Algoritmo de minimización de la suma cuadrada (circfit).

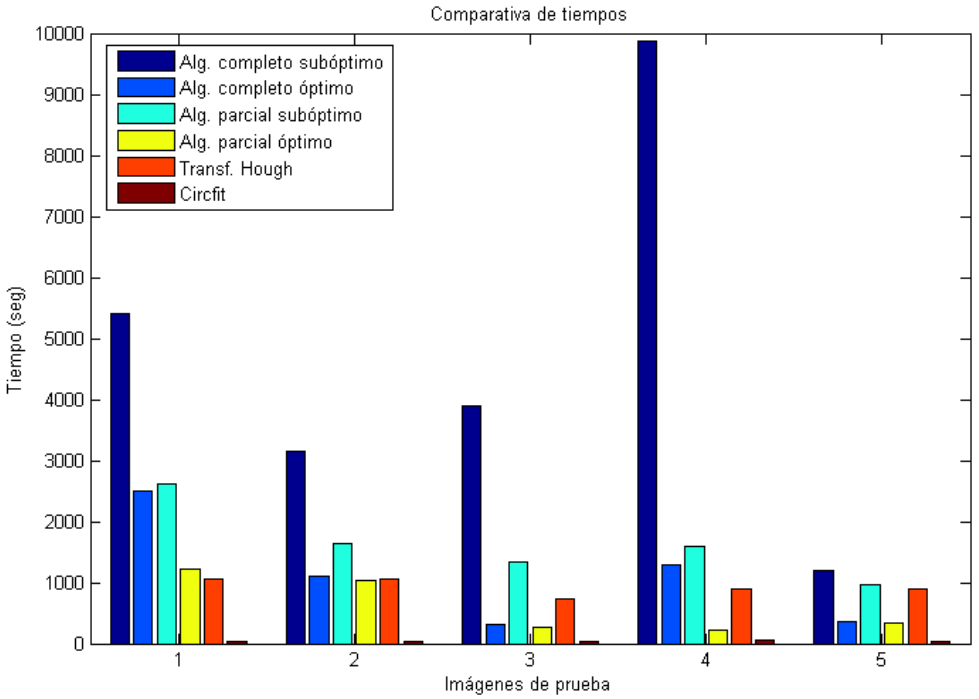


Figura 26.- Gráfica comparativa de los tiempos de todos los métodos.

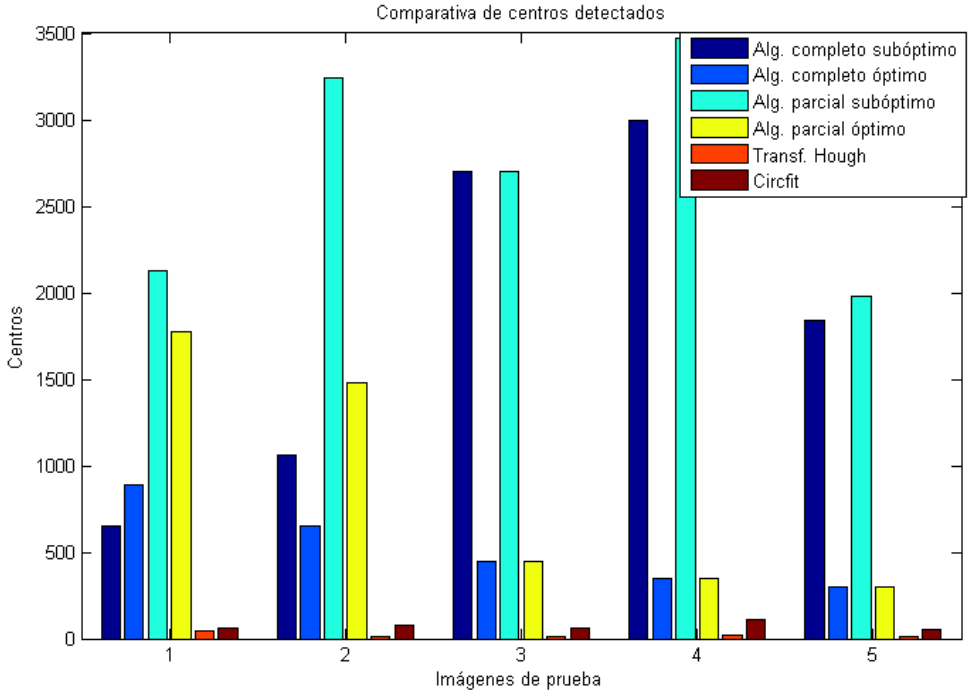


Figura 27.- Gráfica comparativa de los centros detectados por todos los métodos.

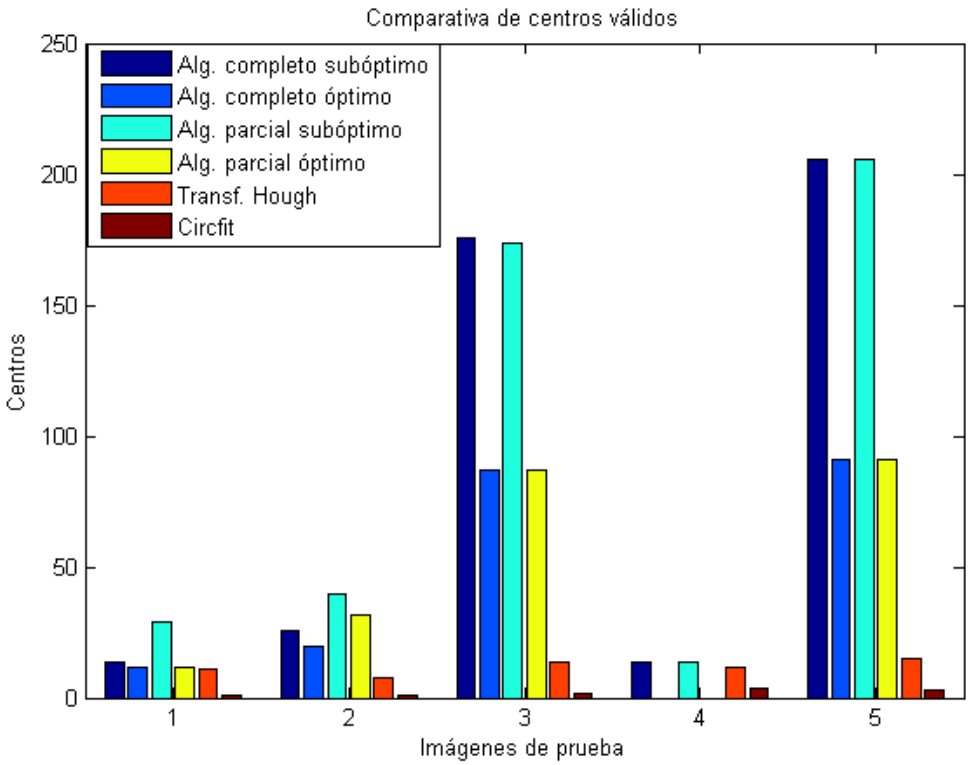


Figura 28.- Gráfica comparativa de los centros válidos con cada método.

En la Figura 26 observamos una comparativa entre todos los métodos de los tiempos de ejecución en cada imagen. Observamos una solución claramente más costosa en tiempo que las demás, el algoritmo propio completo con configuración subóptima. En el resto los tiempos son parecidos, excepto en la minimización de la suma cuadrada que tiene unos tiempos sensiblemente menores que el resto.

En cuanto a los centros detectados, Figura 27, el algoritmo propio tiene una tasa de detección muy superior a la transformada circular y al método circfit, siendo mucho mayor en el caso de la configuración subóptima. Debido a la mayor cantidad de centros, sus tiempos de ejecución son más elevados. El mismo resultado obtenemos en la gráfica de centros válidos, Figura 28, con nuestro algoritmo destacando por la mayor detección. De nuevo este resultado se debe a que a mayor número de centro candidatos, existe una mayor posibilidad de encontrar centros válidos.

4.3 Resultados de la identificación de la escala mediante reconocimiento de patrones

4.3.1 Elección de las plantillas

Como ya se explicó en el capítulo tres, para identificar los valores numéricos de la escala se va a utilizar la técnica de identificación de patrones. Para ello, debemos disponer de un conjunto

de plantillas que nos permita cubrir todas las posibilidades que se puedan encontrar en las imágenes. La elección de dichas plantillas es un factor muy importante en este método. Las plantillas debe ser lo más reconocibles posible, porque a la hora de identificar números, el resultado de la correlación entre un 5 y 6, por ejemplo, no se diferenciará mucho de la resultante de un 6 con un 6, así que tenemos que conseguir que esas pequeñas diferencias se hagan tangibles a la hora de decidir.

Partiendo de la base de datos de imágenes de radar, se buscaron y seleccionaron los mejores candidatos para cada caso, tratando de minimizar las condiciones ambientales para que la diferencia entre el fondo y el carácter fuese lo mayor posible. Con esas condiciones se seleccionaron unas plantillas como las que se presentan en la Figura 29. Como podemos apreciar, a pesar de las consideraciones anteriores las plantillas no tan homogéneas como se desearía. Por otro lado, se han seleccionado dos plantillas diferentes para el punto, ya que es la que con mayor probabilidad puede verse afectada por una falsa detección, así, es posible comparar ambas posibilidades y ver cuál ofrece mejores resultados.

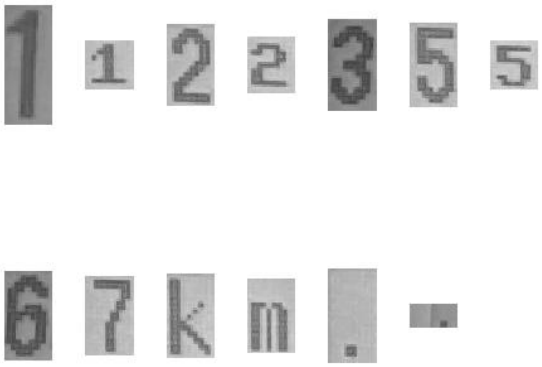


Figura 29.- Ejemplo de plantillas individuales.

Para mejorar la separación con el fondo de la imagen, se va a someter a las plantillas a un preprocesado antes de comenzar el método. Este procedimiento va a consistir en evaluar el valor de los píxeles de cada plantilla. Al ser los caracteres oscuros, sus píxeles tendrán valores de color mayores que el fondo, de forma que se establecerá un umbral a partir del cual si el píxel tiene valor mayor, se considerará parte del carácter y cambiará su color a negro, y si es inferior pasará a color blanco obteniendo una imagen binarizada. Este umbral dependerá del máximo valor de cada imagen, ya que debido a las condiciones de cada uno variarán y un umbral fijo no serviría bien para todos los casos, por ello, se aplica la técnica conocida como método de Otsu (Otsu, 1979) con el fin de obtener un umbral adaptativo para cada imagen a través del estudio del propio histograma. En la Figura 30 se muestran los resultados, que en la mayoría de los casos mejora la percepción del carácter, pero como se puede observar, en algunos casos deforma ligeramente el carácter.

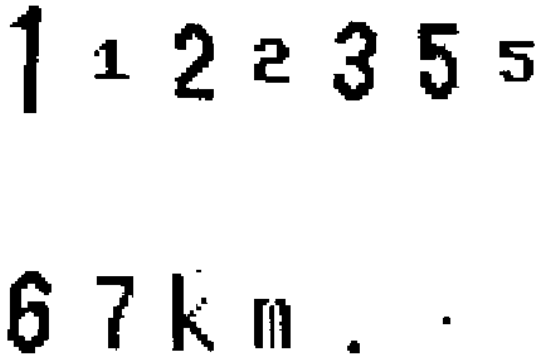


Figura 30.- Plantillas individuales tras el procesamiento mediante el método de Otsu.

Para que la detección sea correcta vamos a someter al mismo proceso de binarización a la imagen original con el fin de que los valores de los píxeles sean los mismos en ambas imágenes. Como conocemos de antemano la posición de la escala dentro de la imagen total, vamos a seleccionar un rectángulo que incluya la posición de la escala de forma ajustada para reducir el tiempo de ejecución, como el de la Figura 31.

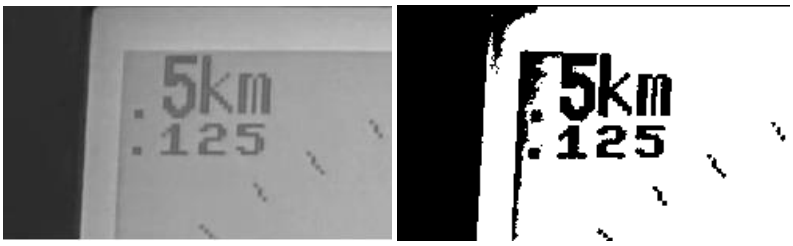


Figura 31.- Recuadro de búsqueda de plantillas a) original y b) filtrado.

Para evaluar el desempeño de este conjunto de plantillas vamos a realizar la identificación de patrones aplicando la correlación cruzada normalizada (NCC) descrita en la sección 3.1.1 sobre el conjunto. Para ello, utilizaremos la imagen anteriormente mostrada, y evaluaremos las plantillas mostradas en la figura 30 para comprobar si esta técnica es capaz de reconocer todos los caracteres existentes en la imagen. Para determinar si una plantilla ha sido detectada o no, se establece un umbral de decisión en la respuesta de la NCC en cada plantilla, de manera que si el valor máximo de la NCC supera el umbral, se considera que la plantilla ha sido encontrada.

En la Figura 32 se observa la curva de respuesta de la NCC al aplicar la plantilla sobre toda una imagen completa, donde se aprecia claramente un pico que marca el lugar de la máxima correlación. Que en ese punto la correlación sea máxima no quiere decir que se haya detectado la plantilla, porque puede tratarse de un valor demasiado pequeño para ser considerado una identificación correcta.

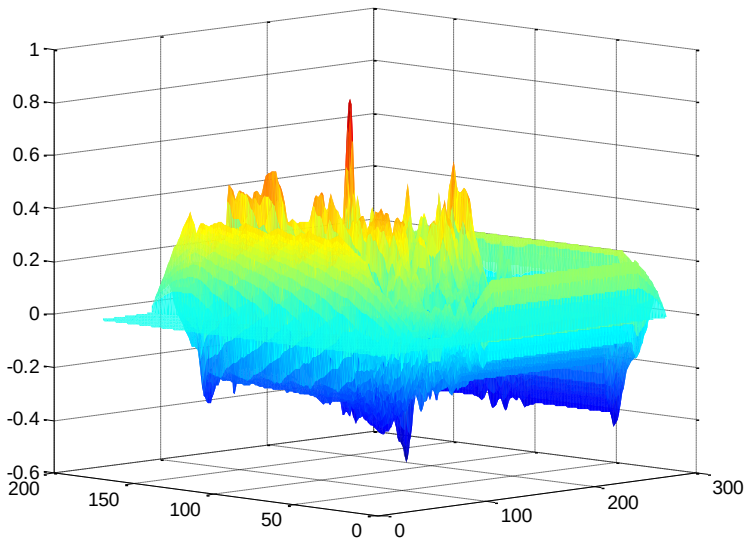


Figura 32.- Curva de respuesta de la correlación cruzada para una de las plantillas.

Para las primeras pruebas se estableció un umbral más restrictivo de 0.8 como umbral de decisión y otro más laxo de 0.6 para evaluar las diferencias entre ambos casos. En la Figura 33 se muestran los caracteres encontrados por el algoritmo con el umbral más restrictivo de 0.8, y en la siguiente, la Figura 34, los resultados de la detección con el umbral en 0.6 al aplicar cada una de las plantillas mostradas en la Figura 30 sobre la imagen mostrada en la Figura 31.

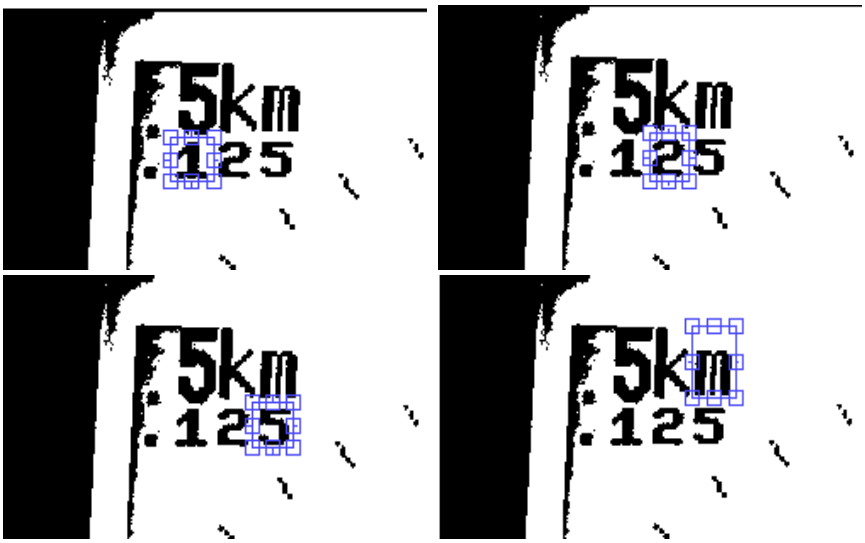


Figura 33.- Caracteres reconocidos con umbral de 0.8.

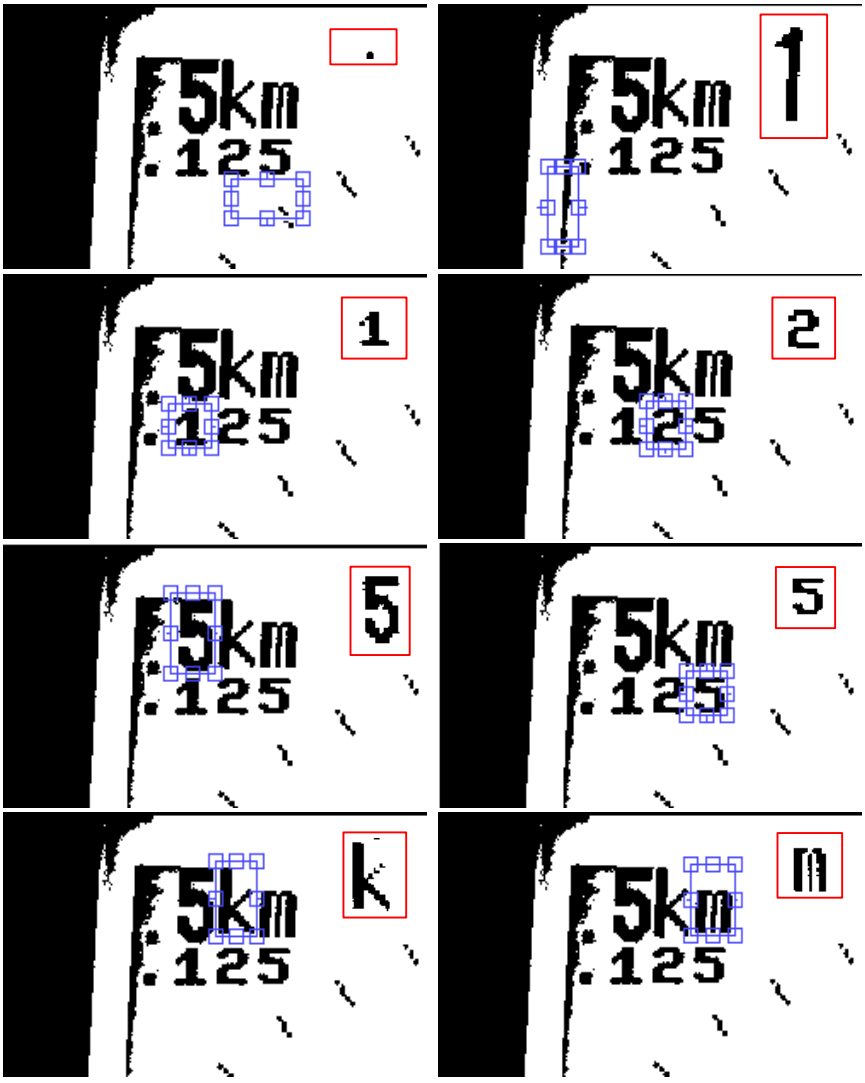


Figura 34.- Caracteres detectados con umbral de 0.6.

En primer lugar vemos que el número de caracteres detectados en cada caso no es el mismo. En el primer caso sólo logramos detectar cuatro de los ocho caracteres de la imagen. En el segundo caso detectamos ocho caracteres, pero dos de ellos son falsas detecciones, por ello, no tiene sentido disminuir el umbral por debajo de 0.6, ya que aunque consiguiéramos detectar todos correctamente, habría una mayor cantidad de falsos positivos, además de que no tendría sentido afirmar que dos imágenes son iguales cuando su correlación es baja. Por lo tanto, se hace necesario encontrar otro conjunto de plantillas que nos ofrezca mejores resultados.

En el caso anterior hemos visto que realizar un preprocesado para intentar separar el fondo de los caracteres a fin de mejorar su detección por el algoritmo de correlación cruzada normalizada no ha dado los resultados esperados. Por ello, ahora vamos a partir de las mismas plantillas, pero en este caso vamos a realizar la correlación sin realizar ningún tratamiento anterior y evaluaremos si los resultados mejoran los anteriores.

La primera prueba se realizó de nuevo con un umbral de 0.8, cuyos resultados de aplicar las plantillas de la Figura 29 sobre la Figura 31 se pueden observar en la Figura 355. Observamos que la detección de los ocho caracteres es correcta, así que no es necesario reducir el umbral a uno menos restrictivo, ya que aumentaría la cantidad de falsos positivos.

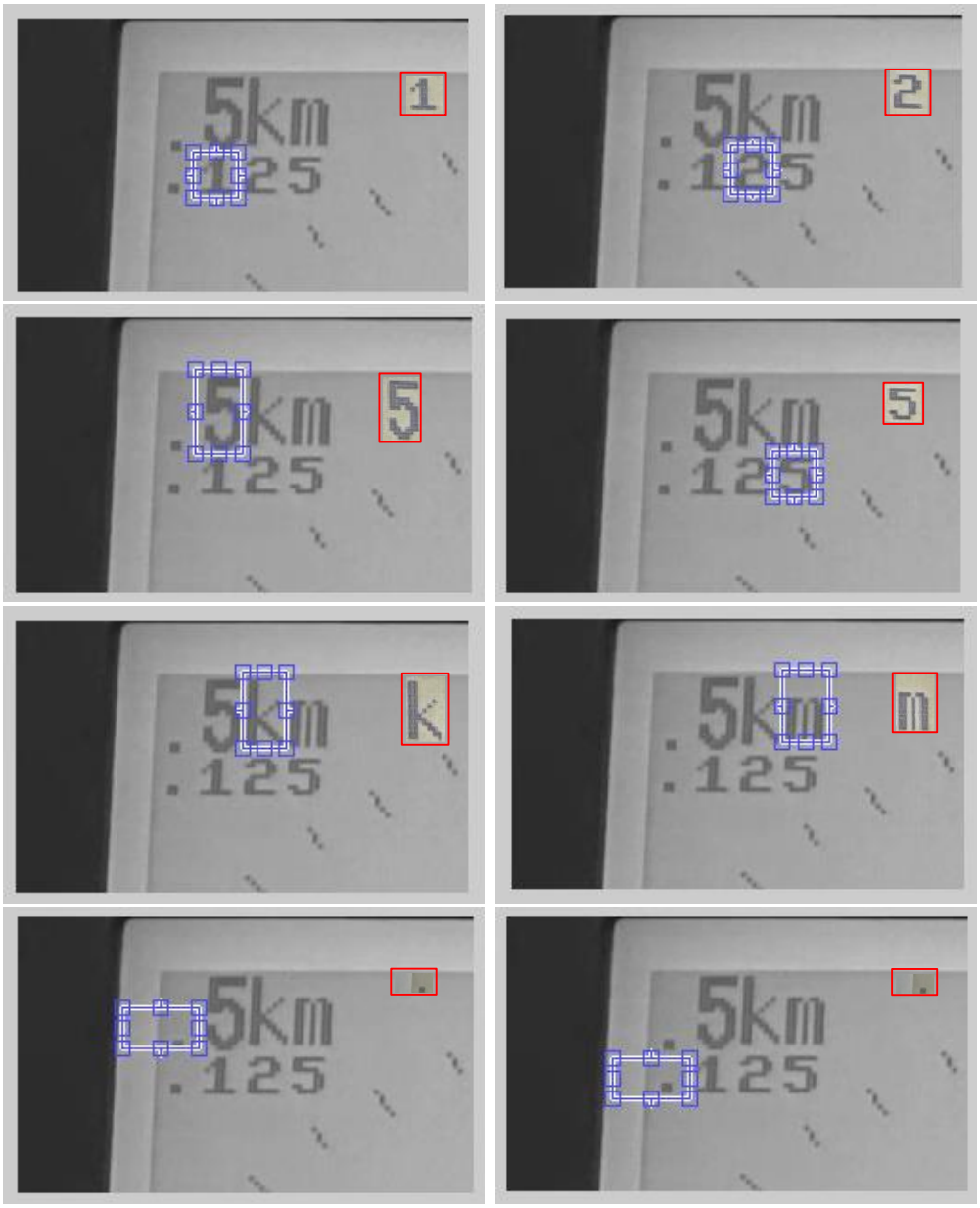


Figura 35.- Caracteres detectados con la imagen original y un umbral de 0.8.

En este último caso, además de ser capaz de reconocer todos los caracteres, si nos fijamos en los valores máximos de la correlación mostrados en la Figura 36 para cada una de las plantillas utilizadas, observaremos que para este conjunto donde las imágenes no son preprocesadas, los valores son mayores, lo que quiere decir que se detecta mejor la correspondencia entre las imágenes sin procesar.

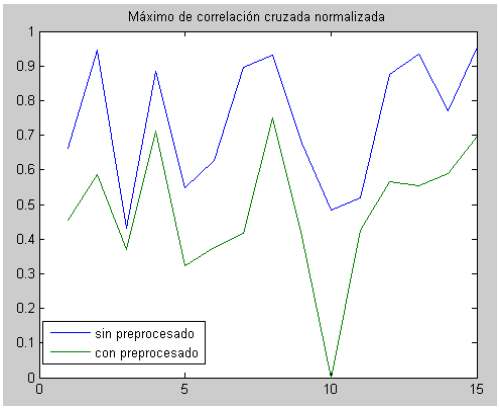


Figura 366.- Comparación resultados con y sin preprocesado

Aunque este conjunto de plantillas es válido para realizar el resto de pruebas de forma general, dadas las especificaciones concretas del radar FURUNO 1715 existe otra posibilidad de adquirir las plantillas. En este caso, vamos a seleccionar como plantilla aquella porción de imagen que incluye la escala con todos los caracteres que la conforman para cada una de las 15 posibilidades mostradas en la tabla 1. De esta forma, en lugar de buscar carácter a carácter entre los caracteres posibles, vamos a buscar una escala completa entre todas las escalas posibles.

Esto es factible debido a que las posibles escalas están limitadas de acuerdo a las especificaciones de este radar en concreto, de forma que solo tenemos que preocuparnos de las escalas que disponemos en la base de datos porque no va a aparecer otra que no esté contemplada.

En el reconocimiento de caracteres individuales, cuando se encuentra una coincidencia, hay que almacenar el tipo de carácter encontrado y la posición del mismo. Así, una vez se ha terminado este proceso, es necesario montar la escala determinando la posición relativa de cada carácter respecto a los demás para poder reconocer el número total que forman. Una vez se ha determinado el número superior y el inferior, podemos calcular el número de círculos que nos vamos a encontrar en la imagen. Este es un proceso laborioso, porque además de guardar la posición y el tipo de plantilla encontrado, hay que tener en cuenta otras situaciones como la repetición de plantillas o que no seamos capaces de encontrar una de ellas porque la imagen está demasiado alterada, lo que limita la cantidad de veces que debe repetirse la búsqueda antes de dar por finalizada la detección de la escala. Sin embargo, trabajando con plantillas de escala completa, cuando tenemos una coincidencia y al conocer de antemano la plantilla, no tenemos más que consultar la escala de esa plantilla y con ella sabremos el número de círculos que vamos a encontrar en la imagen. El procesado posterior se reduce a consultar en la base de datos la información necesaria asociada a dicha plantilla, por lo que se reduce considerablemente el tiempo de ejecución.

De esta forma, para el mismo caso que hemos utilizado anteriormente, si ahora utilizamos plantillas completas, el resultado que obtendremos será el mostrado en la Figura 37. Observamos que la detección de plantillas es correcta, por lo que este sistema es viable en el caso de que el radar utilizado tenga acotados los rangos de escala como los tiene el modelo utilizado en estas pruebas.

Figura 37.- Patrón a detectar (izquierda) y resultado de detección (derecha).

Como resultado de este proceso de selección de plantillas, hemos podido determinar que con estos algoritmos de detección es mejor trabajar con imágenes a color antes que con aquellas preprocesadas, y hemos encontrado dos conjuntos de plantillas que podrían servirnos para realizar las pruebas. Para seguir con el proceso vamos a utilizar las plantillas con la escala completa debido a las ventajas que nos ofrecen sobre las plantillas individuales.

4.3.2 Resultados de las pruebas de detección de escala

Para las pruebas de la identificación de caracteres en las imágenes de radar vamos a partir del conjunto de plantillas de escala completa, de las cuales se muestran varias en la Figura 38. Las pruebas van a consistir en buscar el patrón adecuado en 100 imágenes de radar, de modo que podamos comprobar si se escoge la plantilla adecuada en cada uno de los casos. Para dificultar lo máximo posible la detección, dentro de estas 100 imágenes hay varias en las que se hace más complicado reconocer el patrón adecuado, como por ejemplo aquellas que tienen una luminosidad distinta al patrón, o en el peor de los casos, cuando se varía la escala y en la transición entre una y otra hay un solapamiento de dos patrones.

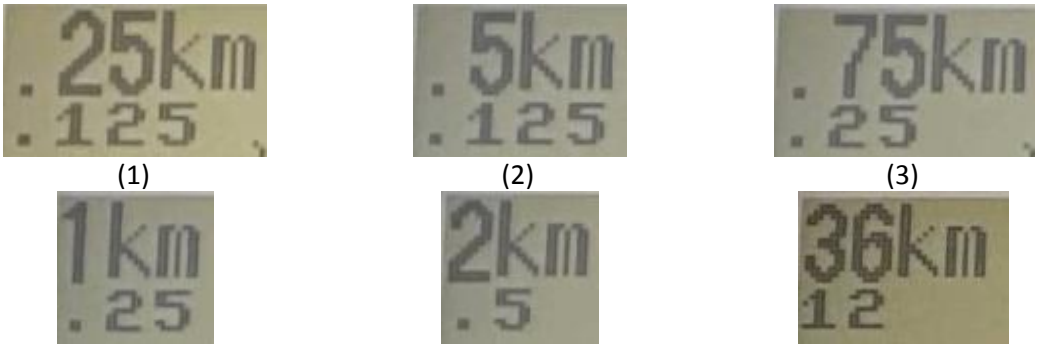
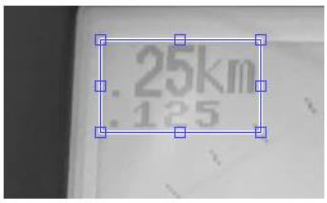
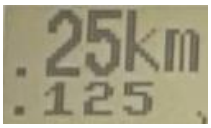
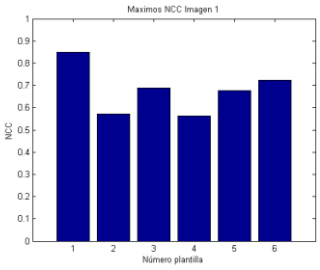
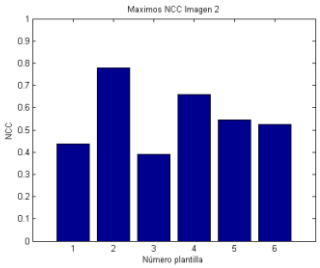
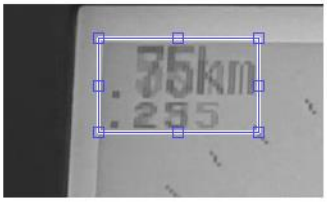
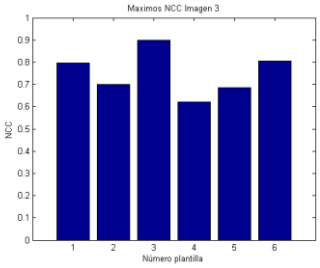
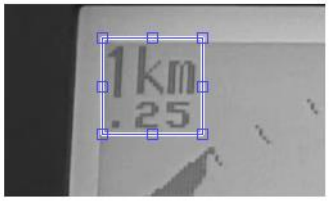
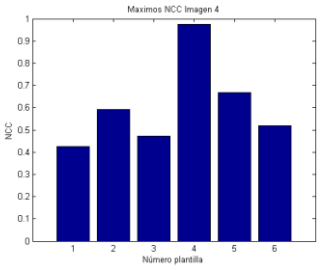
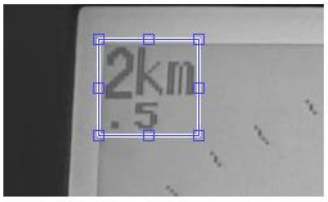
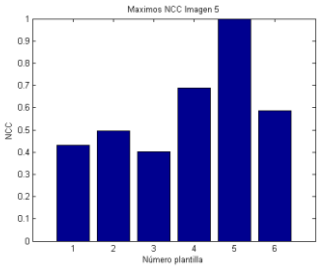
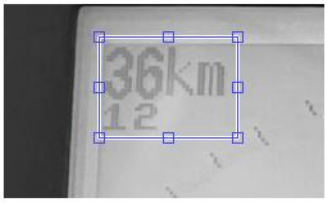
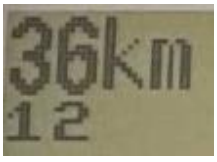
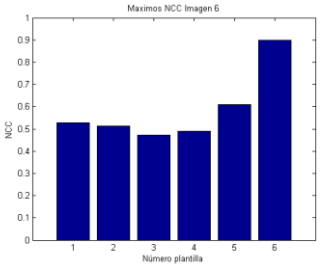


Figura 38.- Plantillas utilizadas en el ejemplo de reconocimiento de caracteres.

4.3.2.1 Detección de la escala aplicando la correlación cruzada normalizada

A continuación se van a mostrar los resultados obtenidos con el conjunto de imágenes al aplicar el método de la correlación cruzada normalizada descrito en el capítulo tres sección 3.1.1.

Tabla 11.-Subconjunto de muestra con imágenes testeadas (izquierda) y la mejor coincidencia encontrada (derecha) que supera el umbral 0.8 en la NCC.

IMAGEN CON EL PATRON A BUSCAR	PATRON DE MÁXIMA NCC	ÉXITO	NCC CON LOS PATRONES DE LA FIGURA 38
		SI	
		NO	
		SI	
		SI	
		SI	
		SI	

En la Tabla 11 se puede observar la imagen recortada en la que se pretende encontrar la plantilla, la plantilla que ha sido identificada con el mayor valor de NCC, si el valor obtenido supera con éxito el umbral establecido a 0.8 y la gráfica obtenida al testear el conjunto de patrones disponibles con esa imagen donde se muestra el valor máximo de NCC obtenido para cada uno de los patrones mostrados en la Figura 39. Esta representación se trata de un subconjunto de las 15 plantillas disponibles.

Como se puede observar en dicha tabla, en casi todos los casos se ha encontrado como mejor solución la plantilla correcta, incluso en casos donde las condiciones no son adecuadas, con cambios de luminosidad acusados o incluso con caracteres borrosos. Tan sólo hay un único caso en el que no se ha encontrado solución (segunda imagen). Esto se debe a que la diferencia de luminosidad con la plantilla ha sido muy elevada, y pese a ser la plantilla con mayor valor de NCC, éste no ha superado el valor mínimo establecido a 0.8, por lo que se trata de un falso negativo.

Si nos fijamos en la última columna de la tabla, es fácilmente apreciable ver que en todas las imágenes hay una plantilla claramente ganadora. Aunque como se puede ver, es posible que algún caso haya más de una solución que supera el umbral, como en la tercera imagen. Para evitar esos casos, además de aplicar el umbral establecido, será necesario discriminar la mejor solución de entre las que lo superen, quedándonos siempre con el máximo de todas

En general, de las 100 imágenes probadas los resultados obtenidos son:

- 87 imágenes correctamente identificadas.
- 8 imágenes correctamente clasificadas que no superaron el umbral de 0.8.
- 5 imágenes incorrectamente clasificadas.
- Tiempo de procesamiento promedio de 12 segundos.

En base a esto se puede concluir que en el peor de los casos, con un umbral máximo de 0.8, esta técnica alcanza un 87% de éxito, pudiendo aumentarse hasta un 95% si se realiza un ajuste del umbral y se establece a 0.65, valor obtenido como promedio de los valores máximos de NCC para las 8 imágenes que generaron los falsos negativos. En cualquier caso, el tiempo de procesado permite que esta técnica pueda ser utilizada en tareas que requieran una limitación del tiempo de ejecución no muy elevada.

4.3.2.2 Detección de la escala aplicando la suma de diferencias cuadráticas

Con el fin de comparar la normalización cruzada normalizada (NCC) con la técnica de la suma de diferencias cuadráticas (SSD), se ha utilizado el método propuesto por (Kroon, 2011), quien presenta una solución robusta para la detección de plantillas.

Para poder comparar los resultados entre ambas técnicas, dado que en el caso de la SSD el valor a identificar debe ser el mínimo tal y como se ha explicado en el capítulo tres sección 3.1.2, el signo es invertido y los valores escalados en el rango [0,1], de esta forma podremos comparar los resultados de ambos métodos a la vez. Por ello, tal y como hemos hecho en el caso anterior de la NCC, vamos a seleccionar el mejor valor de la SSD para cada imagen y cada plantilla.

Primero vamos a visualizar la respuesta de los dos algoritmos en la Figura 39 cuando a la misma imagen se le aplican las diferentes plantillas. Esta imagen nos muestra el resultado final con el punto de máxima SSD y de máxima NCC, la plantilla utilizada en cada caso y los resultados de ambos métodos a lo largo de toda la imagen. En estas imágenes de NCC y SSD, los puntos más brillantes representan los máximos valores en cada caso.

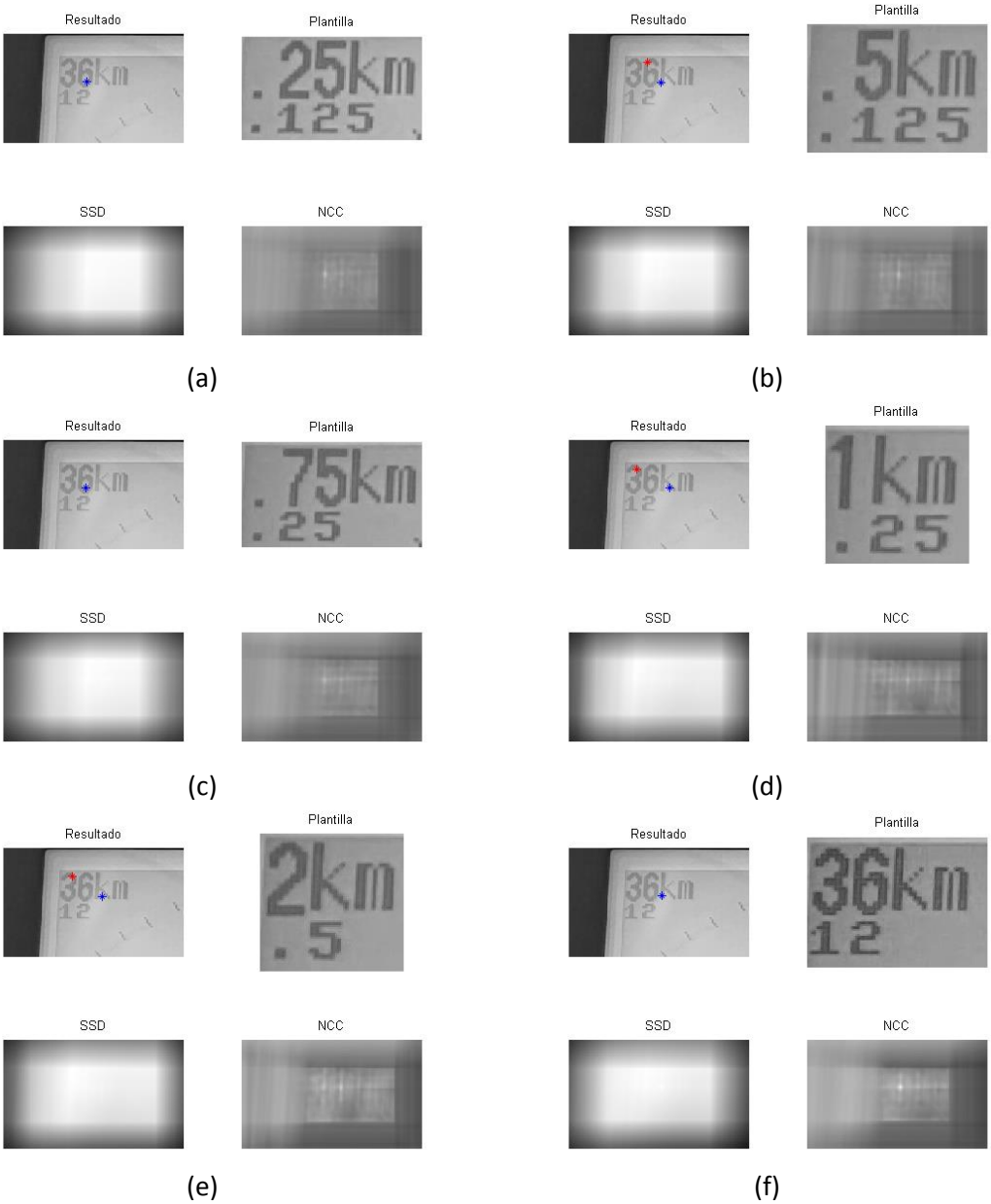
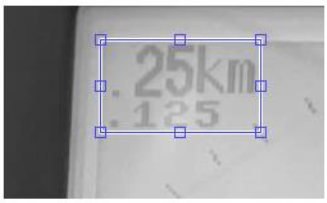
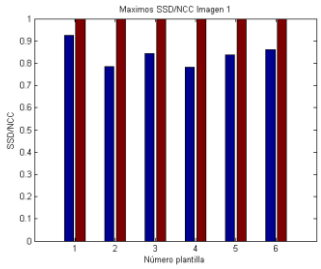
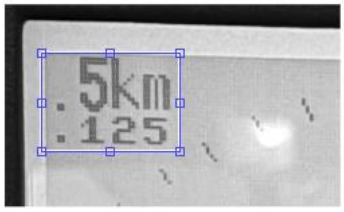
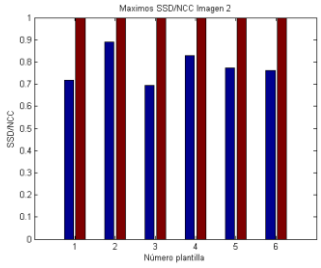
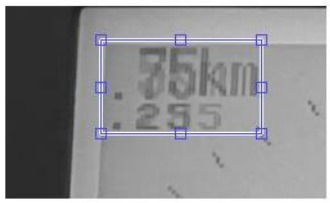
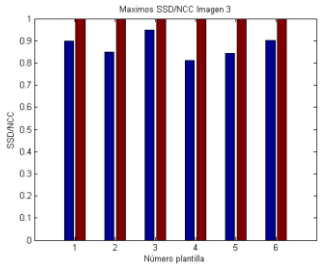
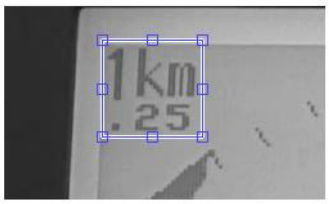
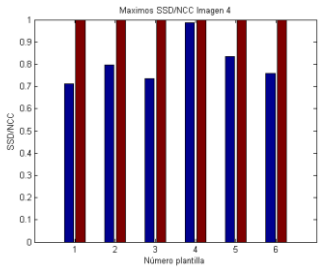
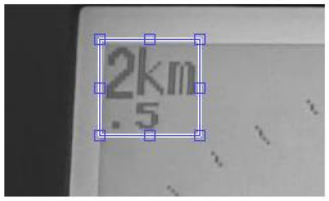
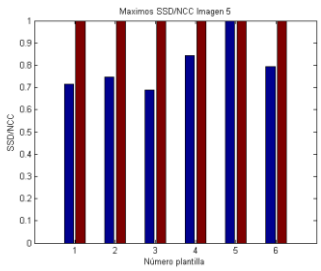
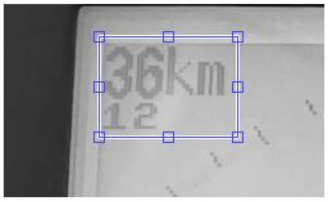
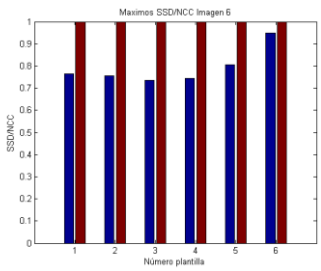


Figura 39.-Resultados obtenidos para la SSD y la NCC en una misma imagen con todas las plantillas. El punto rojo en "Resultado" es el máximo de la SSD y el azul el máximo de la NCC.

Como se puede observar en las imágenes de la Figura 39, en primer lugar los puntos de máxima correlación cruzada y de máxima diferencia cuadrática no suelen coincidir, aunque sí lo hacen en la plantilla que coincide con la imagen (f).

A continuación se muestran en la Tabla 12 las gráficas con los resultados comparativos entre el máximo valor de la NCC y el de la SSD en cada plantilla para distintas imágenes.

Tabla 12.- Gráficas comparativas de los resultados para cada plantilla. La barra azul es valor máximo de la SSD y la roja de la NCC.

IMAGEN CON EL PATRON A BUSCAR	SSD Y NCC CON LOS PATRONES DE LA FIGURA 38
	
	
	
	
	
	

Observando las gráficas de la Tabla 12, lo más llamativo es que en todas ellas el valor máximo de la SSD es 1, es decir, encuentra al menos un punto donde la detección es perfecta. Sin embargo, los resultados de la NCC se mantienen, de manera que los valores de los máximos varían dependiendo de la plantilla y de la imagen, y siempre la máxima correlación corresponde a la plantilla adecuada tal y como se mostró en la Tabla 11.

Por todo ello, queda descartada la opción de utilizar la suma de diferencias cuadráticas debido a que en su mayoría devuelve falsos positivos. Pese a tratarse de una implementación robusta, la correlación cruzada normalizada devuelve los mejores valores al no ser tan dependiente de la iluminación y ser más dependiente de la textura de la imagen

5 Conclusiones y líneas de trabajo futuras

5.1 Conclusiones

Respecto a la detección de círculos concéntricos, podemos sacar varias conclusiones atendiendo a los resultados obtenidos. En todos los métodos probados observamos que la correcta parametrización inicial juega un papel importante en la ejecución del algoritmo. Tanto para el algoritmo de diseño propio como la transformada circular de Hough, tanto el tiempo de ejecución como los centros obtenidos se ven afectados por los umbrales de decisión establecidos en cada caso. Por lo tanto, antes de ejecutar cualquier método con los parámetros genéricos que incluya, es necesario dedicar un tiempo al ajuste de los mismos, sobre todo porque dependiendo de las imágenes que estemos tratando los parámetros que mejor rendimiento van a obtener cambiarán.

Si nos fijamos en las dos implementaciones distintas del algoritmo propio, es decir, la que incluye el algoritmo de Bersenham y la comparación de ángulos entre segmentos y la que no los utiliza, podemos concluir que desarrollar esos algoritmos no nos aporta mejoras en ningún aspecto. Es más, supone un sobrecoste en tiempo computacional que luego no repercute en la calidad de la información obtenida. Aunque utilizar estas técnicas provoca que los centros detectados disminuyan notablemente, el número de centros válidos con ambas opciones es prácticamente el mismo, y la diferencia de tiempos de ejecución es muy grande. Por lo tanto, para esta solución no se hace necesario realizar el procesamiento del algoritmo de Bersenham ni la comparación de ángulos entre los segmentos obtenidos.

Siguiendo con esta implementación, también presentamos dos configuraciones de parámetros diferentes, una llamada óptima por su mejor respuesta, y otra subóptima. Los tiempos de ejecución de la óptima se reducen drásticamente respecto a la subóptima, debido en mayor parte a que la cantidad de centros detectados es mucho más reducida. Sin embargo, la configuración óptima no fue capaz de detectar todas las circunferencias en todos los casos. Por lo tanto, aunque en líneas generales se perfila como una solución mejor, hay que tener en cuenta que la tasa de fallos es mayor debido a que sus parámetros son más restrictivos. La conclusión en este aspecto es que en cada situación hay que realizar una evaluación entre la restricción en los parámetros del algoritmo y la tasa de fallos admitida. Cuando más restrinjamos los parámetros de entrada, mejor funcionamiento tendrá el algoritmo, pero también es posible que en algunas situaciones perdamos información determinante. Por ello, es necesario que se alcance un compromiso, siendo una opción factible la de utilizar la opción más restrictiva y dando suficiente margen a los parámetros para los posibles casos más difíciles de detectar.

En el artículo en el cual nos basamos para diseñar nuestro algoritmo (Silveira, 2005), la razón del desarrollo de dicho algoritmo era la mejora en coste computacional respecto a la transformada circular de Hough debido a que realizaba más cálculos en el espacio de la imagen que en el paramétrico. Analizando los resultados, con la correcta configuración de los

parámetros, nuestro algoritmo es igual o más rápido que la transformada de Hough utilizada, así que en ese aspecto se comprueba que dicha afirmación es cierta.

Finalmente, resaltar que pese a las mejoras en tiempo de ejecución de algunos casos en el algoritmo diseñado y en la transformada circular de Hough, el único algoritmo que podría usarse en aplicaciones de tiempo real sería el de detección por mínimos cuadrados, ya que en todos los casos el tiempo de ejecución dura menos de un minuto y hace que sea una opción fiable para sistemas que requieran de una detección rápida. Este método supera a todos los demás algoritmos, pero también tiene el problema de en ciertas ocasiones falla en la detección, por lo que en líneas generales sería el algoritmo óptimo para tareas en tiempo real pero habría que tener en consideración que puede conllevar fallos en la detección, y dado que no existe la posibilidad de ajustes de parametrización, puede ser un descalificante en los casos en los que sea primordial obtener la máxima precisión.

En cuanto a la detección de caracteres, a la vista de los resultados está claro que la correlación cruzada normalizada ofrece mayores prestaciones que la suma de diferencias cuadráticas utilizando plantillas completas en este modelo de radar. Esto es debido a que la NCC, al normalizar los datos de la imagen, es una solución robusta frente a cambios en la iluminación. Además, al realizar operaciones más complicadas que la SSD, es menos sensible a los cambios de los píxeles y por ello en general permite una mejor respuesta. El parámetro que va a decidir si la detección es correcta o no es el umbral de decisión establecido, ya que como hemos podido comprobar, la respuesta ante diferentes plantillas puede ser lo suficientemente buena aunque puede inducir a error. En este caso sabemos que no existe la posibilidad de encontrar la misma plantilla dos veces en la imagen, pero en otros casos en los que no sea así, no podremos discriminar simplemente eligiendo la plantilla que mejor respuesta de NCC tenga.

5.2 Líneas de trabajo futuras

Como líneas de investigación se proponen las siguientes:

1.- Optimizar el código que se ha diseñado para la implementación del algoritmo, de forma que se pueda reducir el tiempo de ejecución y mejorar la detección de centros para reducir el número de falsos candidatos. Para llevar a cabo este punto, la solución que se plantea es la de implementar el algoritmo en lenguajes de programación orientados a tiempo real, como puede ser el caso de C.

2.- Comparar el algoritmo con otros tipos de soluciones, como las redes neuronales, para poner en perspectiva su funcionamiento y evaluar si es mejor para este problema el uso de un tipo distinto de soluciones.

3.- Búsqueda de soluciones en tiempo real que nos permitan el tratamiento de las imágenes con tan solo unos segundos de retardo. Para ello, sería necesario estudiar las herramientas existentes en el mercado para poder desarrollar una solución factible en una plataforma orientada a tiempo real.

4.- La técnica de detección de círculos concéntricos y detección de plantillas puede ser orientada a no realizarse a cada instante. La detección de radios propuesta nos permite identificar la cantidad de círculos concéntricos existentes en la imagen. Por otro lado, la detección de la escala nos permite identificar la posición de la misma y el rango de trabajo. De esta forma, una vez situada la cámara al inicio de la aplicación, es posible realizar los pasos necesarios para la identificación del centro de las circunferencias y de la escala una única vez. De esta manera, se identificarían las circunferencias existentes en la imagen a partir del centro identificado al inicio de la ejecución, y conociendo la posición de la escala se calcularía el rango de trabajo. Cada vez que los datos de la escala y de los radios no coincidan, se realizaría un nuevo ajuste en la escala en la posición en la que se encuentra manteniendo el centro ideal. Esta implementación reduciría drásticamente el tiempo de ejecución y podría utilizarse en sistemas que requieran de tratamiento de imágenes en tiempo real.

6 Bibliografía

Ahuja, S. (2010). *Correlation based similarity measures. Summary*. Obtenido de siddhantahuja.wordpress.com: https://siddhantahuja.wordpress.com/2010/04/11/correlation-based-similarity-measures-summary/

Akinlar, C., & Topal, C. (2013). EDCircles: a real-time circle detector with a false detection control. En *Pattern Recogn 46* (págs. 725-740).

Andaléa, F., Miranda, P., Torres, R., & Falcão, A. (2010). Shape feature extraction and description based on tensor scale. En *Pattern Recognition 43* (págs. 26-36).

Arlicot, A., Soheilian, B., & Paparoditis, N. (2009). Circular road sign extraction from street level images using color, shape and texture database maps. En *Proceedings of IAPRS* (Vol. XXXVIII, págs. 205-210).

Ayala-Ramirez, V., Garcia-Capulin, C., Perez-Garcia, A., & Sanchez-Yanez, R. (2006). Circle detection on images using genetic algorithms. En *Pattern Recognition Letters 27* (págs. 652-657).

Bayes, T. (1763). An Essay towards solving a Problem in the Doctrine of Chances. En *Philosophical Transactions of the Royal Society of London 53* (págs. 370-418).

Bone, P. (2010). *Hough Transform for circle detection*. Obtenido de http://www.mathworks.com/matlabcentral/fileexchange/9833-hough-transform-for-circle-detection

Bongiovanni, G., & Crescenzi, P. (1995). Parallel simulated annealing for shape detection. En *Computer Vision and Image Understanding 61* (págs. 60-69).

Bucher, I. (2004). *Circle fit*. Obtenido de http://www.mathworks.com/matlabcentral/fileexchange/5557-circle-fit

Canny, J. (1986). A Computational Approach To Edge Detection. En *IEEE Trans. Pattern Analysis and Machine Intelligence 8* (págs. 679-698).

Cao, X. D. (1992). An Efficient Method for the Detection of Multiple Concentric Circles. En *Proc. International Conference on Acoustic, Speech and Signal Processing* (págs. III-137-III-140).

Chen, T., & Chung, K. (2001). An efficient randomized algorithm for detecting circles. En *Computer Vision and Image Understanding 83* (págs. 172-191).

Chen, X., Lu, L., & Gao, Y. (2012). *A New Concentric Circle Detection Method Based on Hough Transform.*

Chung, K., Huang, Y., Shen, S., Krylov, A., Yurin, D., & Semeikina, E. (2012). Efficient sampling strategy and refinement strategy for randomized circle detection. En *Pattern Recognition 45* (págs. 252-263).

Coope, I. D. (1993). Circle fitting by linear and nonlinear least squares. En *Journal of optimization theory and applications.* New York.

Cuevas, E., Oliva, D., Zaldivar, D., Perez-Cisneros, M., & Sossa, H. (2012). Circle detection using electro-magnetism optimization. En *Information Science 182* (págs. 40-55).

Cuevas, E., Sención-Echauri, F., Zaldiwar, D., & Perez-Cisneros, M. (2012). Multi-circle detection on images using artificial bee colony (ABC) optimization. En *Software Computation 16* (págs. 281-296).

Cuevas, E., Zaldiwar, D., Perez-Cisneros, M., & Ramirez-Ortegon, M. (2010). Circle detection using discrete differential evolution optimization. En *Pattern Analysis Applications 14* (págs. 93-107).

Delalleau, O., Bengio, Y., & Le Roux, N. (2005). Efficient Non-Parametric Function Induction in Semi-Supervised Learning. En *AISTATS 2005 Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics* (págs. 96-103).

Desolneux, A., Moisan, L., Morel, & Jean-Michel. (2001). Edge Detection by Helmholtz Principle. En *Journal of Mathematical Imaging and Vision 14* (págs. 271-284-).

Du, S., Ibrahim, M., Shehata, M., & Badawy, W. (2013). Automatic License Plate Recognition (ALPR): A State-of-the-Art Review. En *IEEE Transactions on circuits and systems for video technology,vol.23* (págs. 311-325).

Duda, R. O., Hart, P. E., & Stork, D. G. (2000). *Pattern Classification, 2nd Edition.*

Ebisawa, Y. (2009). Robust pupil detection by image difference with positional compensation. En *Proceedings of IEEE VECIMS* (págs. 143-148).

Essannouni, F., Oulad Haj Thami, R., Aboutajdine, D., & Salam, A. (2007). Simple noncircular correlation method for exhaustive sum square difference matching. En *Opt. Eng. 46.*

Feng, B.-Y., Ren, M., Zhang, X.-Y., & Suen, C. Y. (2014). Automatic recognition of serial numbers in bank notes. En *Pattern Recognition 47* (págs. 2621-2634).

Fischer, M., & Bolles, R. (1981). Random sample consensus: a paradigm to model fitting with applications to image analysis and automated cartography. En *CACM 24* (págs. 381-395).

Fleyeh, H., & Davami, E. (Intel Trans Syst 2011;5(3):190–6.). *Eigen-based traffic sign recognition.*

Foley, J. D. (1980). Bresenham algorithm with grey scale. In *Communications of the ACM. Technical Note.*

Frosio, I., & Borghese, N. (2008). Real-time accurate circle fitting with occlusions. En *Pattern Recogn 41* (págs. 1041-1055).

FURUNO. (2011). *Manual de operario para el radar marino modelo 1715.*

Han, J., Koczy, L., & Poston, T. (1993). Fuzzy Hough transform. En *Proceedings of the 2nd International Conference on Fuzzy Systems* (págs. 803-808).

Haralick, R. M., & Shapiro, L. G. (1992). Computer and Robot Vision, Volume II.

Hiley, J. B., Redekopp, A. H., & Fazel-Rezai, R. (2006). A low cost human computer interface based on eye tracking. En *Proceedings of IEEE EBMS* (pág. 3226).

Hough, P. (1962). A method and means for recognizing complex patterns. En *U.S. Patent Office No 306954.*

Illingworth, J., & Kittler, J. (1988). A Survey of the Hough Transform. En *Computer vision, graphics and image processing 44* (págs. 87-116).

Jiang, G., & Quan, L. (2005). Detection of Concentric Circles for Camera Calibration. En *Proceedings.Tenth IEEE International Conference on Computer Vision (ICCV), vol. 1* (págs. 331-340).

Kroon, D.-J. (2011). *Fast/Robust Template Matching*. From http://www.mathworks.com/matlabcentral/fileexchange/24925-fast-robust-template-matching

Lehmann, E. L., & Casella, G. (1998). Theory of Point Estimation (2nd ed.). New York: Springer.

Lewis, J. P. (1995). Fast Normalized Cross-Correlation. En *Industrial Light & Magic.*

Lewis, J. P. (1995). Fast template Matching. En *Vision Interface* (págs. 120-123).

Lina, Y., & Chen, C. (2008). Template matching using the parametric template vector with translation, rotation and scale invariance. En *Pattern Recognition 41* (págs. 2413-2421).

Liu, D., Wang, Y., Tang, Z., & Lu, X. (2014). A robust circle detection algorithm based on top-down least-square fitting analysis. En *Computers and Electrical Engineering 40* (págs. 1415–1428).

Liu, G., Ma, Z., Du, Z., & Wen, C. (2011). The calculation method of road travel time based on license plate recognition technology. En *Proc. Adv. Inform. Tech. Educ. Commun. Comput. Inform. Sci.* (Vol. 201, págs. 385-389).

Liu, Y., Wei, D., Zhang, N., & Zhao, M. (2011). Vehicle-License-Plate Recognition Based on Neural Networks. En *IEEE International Conference on Information and Automation* (págs. 363-366).

Massoud, M., Sabee, M., Gergais, M., & Bakhit, R. (2013). Automated new license plate recognition in Egypt. En *Alexandria Engineering Journal 52* (págs. 319-326).

Mathworks. (2014). Obtenido de http://www.mathworks.com/

Mathworks, ejemplo uso imfindcircles. (s.f.). Obtenido de http://es.mathworks.com/help/images/examples/detect-and-measure-circular-objects-in-an-image.html

MathWorks, función imfindcircles. (s.f.). Obtenido de http://es.mathworks.com/help/images/ref/imfindcircles.html

Muammar, H., & Nixon, M. (1989). Approaches to extending the Hough transform. En *Proceedings of the International Conference on Acoustics, Speech and Signal Processing ICASSP-89* (págs. 1556-1559).

Nao, Q., Ye, Y., Mo, C., Wu, Y., & Liu, L. (2010). Method for the Detection of Concentric Circles of Photoelectric Image of Circular Hole in Printed Circuit Board. En *Acta Photonica Sinica,vol.30* (págs. 75-78).

Orrite, C. (2009). *Patter Recognition. Apuntes asignatura Sistemas Biométricos.* Universidad de Zaragoza.

Otsu, N. (1979). A threshold selection method from gray level histogram. En *Transactions on Systems Man and Cybernetics 9, IEEE* (págs. 62-66).

Pajares, G., & Cruz, J. (2007). En *Visión por Computador. Imágenes digitales y aplicaciones.* Madrid.

Peng, T. (2005). *Detect circles with various radii in grayscale image via Hough Transform*. Obtenido de http://www.mathworks.com/matlabcentral/fileexchange/9168-detect-circles-with-various-radii-in-grayscale-image-via-hough-transform/content/CircularHough_Grd.m

Pfragner, M. (2004). *Fast implementation of circular Hough-transform*. Obtenido de http://www.mathworks.com/matlabcentral/fileexchange/6345-fast-implementation-of-circular-hough-transform

Poli, R., Kennedy, J., & Blackwell, T. (2007). Particle Swarm Optimization: An overview. En *Swarm Intelligence* (págs. 33-37).

Rashid, M. M., Musa, A., Ataur Rahman, M., Farahana, N., & Farhana, A. (2012). Automatic Parking Management System and Parking Fee Collection Based on Number Plate Recognition. En *International Journal of Machine Learning and Computing, Vol. 2.*

Roth, G., & Levine, M. (1994). Geometric primitive extraction using a genetic algorithm. En *IEEE Transactions on Pattern Analysis and Machine Intelligence 16* (págs. 901-905).

Shaked, D., Yaron, O., & Kiryati, N. (1996). Deriving stopping rules for the probabilistic Hough transform by sequential analysis. En *Computer Vision Image Understanding 63* (págs. 512-526).

Shoelson, B. (2008). *Detecting Circles in an Image*. Obtenido de http://blogs.mathworks.com/pick/2008/05/23/detecting-circles-in-an-image/

Shoelson, B. (2008). *Fitting a circle, easily*. Obtenido de http://blogs.mathworks.com/pick/2008/03/14/fitting-a-circle-easily/

Silveira, M. (2005). An Algorithm for the Detection of Multiple Concentric Circles. En *IbPRIA 2005* (págs. 271-278).

Steinmetz, R. (2000). Multimedia Technologia. Berlin: Springer.

Topal, C., Ozsen, O., & Akinlar, C. (2011). Real-time Edge Segment Detection with Edge Drawing Algorithm. En *7th International Symposium on Image and Signal Processing and Analysis* (págs. 313-318).

Wackerly, D., Mendenhall, W., & Scheaffer, R. L. (2008). Mathamatical Statistics with Applications (7 ed.). Belmont, CA, USA: Thomson Higher Education.

Xu, L., Oja, E., & Kultanen, P. (1990). A new curve detection method: randomized Hough transform (RHT). En *Pattern Recognition Letters 11* (págs. 331-338).

Young, D. (2014). *Hough transform for circles*. Obtenido de http://www.mathworks.com/matlabcentral/fileexchange/26978-hough-transform-for-circles

Yuen, H., Princen, J., Illingworth, J., & Kittler, J. (1990). Comparative study of Hough transform methods for circle finding. En *Image and Vision Computing 8* (págs. 71-77).

Zhao, T.-t., Zhao, J.-y., Zheng, R.-r., & Zhang, L.-l. (2010). Study on RMB Number Recognition Based on Genetic Algorithm Artificial Neural Network. En *3rd International Congress on Image and Signal Processing* (págs. 1951-1955).

Zhu, L., & Chen, Y. (2007). Unsupervised Learning of a Probabilistic Grammar for Object Detection and Parsing. En *Advances in Neural Information Processing Systems 19* (págs. 827-834).

Ziou, D., & Tabbone, S. (1998). Edge detection techniques: An overview. En *International Journal of Pattern Recognition and Image Analysis 8* (págs. 537-559).