UNIVERSIDAD DE CANTABRIA

DEPARTAMENTO DE TECNOLOGÍA ELECTRÓNICA,

INGENIERÍA DE SISTEMAS Y AUTOMÁTICA

TESIS DOCTORAL

METODOLOGÍA PARA EL DESARROLLO DE NUEVAS TÉCNICAS Y ALGORITMOS EN EL RECONOCIMIENTO INTELIGENTE DE IMÁGENES SUBMARINAS

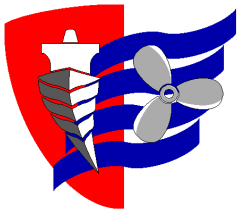
ISABEL ZAMANILLO SAINZ DE LA MAZA

LICENCIADA EN CIENCIAS FÍSICAS

2013

UNIVERSIDAD DE CANTABRIA

ESCUELA TÉCNICA SUPERIOR DE NÁUTICA



DEPARTAMENTO DE TECNOLOGÍA ELECTRÓNICA,
INGENIERÍA DE SISTEMAS Y AUTOMÁTICA

METODOLOGÍA PARA EL DESARROLLO DE NUEVAS TÉCNICAS Y ALGORITMOS EN EL RECONOCIMIENTO INTELIGENTE DE IMÁGENES SUBMARINAS

MEMORIA

presentada para optar al grado de

DOCTORA POR LA UNIVERSIDAD DE CANTABRIA

por la Licenciada en Ciencias Físicas

Isabel Zamanillo Sainz de la Maza

Santander, 2013

DIRECTORES:

Dr. Francisco Velasco González

Dr. José Mª Zamanillo Sainz de la Maza

A mi familia

“Lo que se lee
sin esfuerzo ninguno,
se ha escrito siempre
con un gran esfuerzo”

Enrique Jardiel Poncela

Agradecimientos

Me gustaría expresar mi más sincero agradecimiento a:

- Dr. José Mª Zamanillo y Dr. Francisco Velasco, mis directores de tesis, por su inestimable apoyo, dedicación y consejos a la hora de realizar esta tesis.
- El apoyo de los proyectos de I+D+i, DPI2011-27990: *Navegación de vehículos subacuáticos no tripulados para inspección y asistencia en operaciones y mantenimiento de estructuras sumergidas y tendidos de cables submarinos* (NAVESUBOM) y DN8644: *Navegación de Vehículos Submarinos No Tripulados en Entornos No Estructurados* (NVSNTENE).
- Mis compañeros del grupo de investigación y todas las personas que de forma desinteresada han colaborado con nuestro grupo, sin los cuales sería imposible realizar las salidas con los vehículos submarinos y con los que he vivido todo tipo de aventuras. En especial a Elías, Abel, Emiliano y sobre todo a Ramón, porque sin él no hay expedición.
- El personal de la secretaría del departamento, Sara y Beatriz, porque son fantásticas ayudando con el papeleo.
- El personal de la escuela de Náutica, en especial a Fernando y Gerardo que siempre están dispuestos a echarte una mano y de paso te arrancan una sonrisa cuando más lo necesitas.
- Mi familia y amigos, que siempre han estado dando muestras de su apoyo y cariño a pesar de que en ciertas ocasiones los haya tenido un poco desatendidos por estar trabajando en la tesis. Por último, quiero agradecer a mi padre que siempre me apoyó en todo, y aunque nos dejó hace unos años, siempre está conmigo cada vez que me acuerdo de él. Espero que esté donde esté, se sienta orgulloso de mí.

Índice general

CAPÍTULO 1. INTRODUCCIÓN 1

1.1 Motivación y justificación 1

1.2 Objetivos y metodología 3

1.3 Organización de la tesis 6

CAPÍTULO 2. IMÁGENES ACÚSTICAS SUBMARINAS OBTENIDAS MEDIANTE SONAR DE BARRIDO LATERAL 9

2.1 Justificación del uso de ondas acústicas en el mar 9

2.2 Factores que influyen sobre la propagación de sonido en el mar 10

2.3 Velocidad del sonido en el mar 12

2.4 El sonar 17

2.4.1 Origen y definición del término sonar 18

2.4.2 Historia del sonar 18

2.4.3 Tipos de sonar 24

2.5 El sonar de barrido lateral 24

2.5.1 Historia 25

2.5.2 Montaje y funcionamiento 26

2.5.3 Interpretación de los datos obtenidos 28

2.5.4 Sonar SeaKing 675 kHz 29

2.5.5 Frecuencia versus resolución / alcance 30

2.5.6 Modulación CHIRP 31

2.5.7 Pruebas experimentales para la obtención de imágenes 34
2.6 Ruido acústico 39
2.6.1 Terminología 39
2.6.2 Ruido que afecta al sonar 41

CAPÍTULO 3. PREPROCESADO DE IMÁGENES MEDIANTE FILTRADO DIGITAL 45

3.1 Imágenes digitales 45
3.1.1 Ruido digital 46
3.2 Modelos de color 47
3.2.1 Modelo RGB 48
3.2.2 Modelo YC_bC_r 50
3.3 Filtrado digital 52
3.3.1 Filtrado frecuencial 53
3.3.1.1 Clasificación de los filtros frecuenciales 54
3.3.2 Ecualización del histograma 55
3.3.3 Filtrado espacial 57
3.3.3.1 Clasificación de los filtros espaciales 57
3.3.3.2 Convolución en el filtrado espacial 59
3.3.3.3 Tipos de máscaras 60

CAPÍTULO 4. FILTROS PARA ELIMINAR RUIDO Y REALZAR BORDES 63

4.1 Terminología utilizada en los filtros 63
4.2 Implementación de las máscaras 65
4.3 Filtrado según el tipo de imagen 65
4.4 Filtro mediana 68
4.5 Filtro LUM 69
4.5.1 Algoritmo implementado 71
4.6 Filtro CS 72
4.6.1 Algoritmo implementado 74
4.7 Filtro WMMR-MED 76

4.7.1 Algoritmo implementado....77
4.8 Filtro Volterra....79
4.8.1 Algoritmo implementado....82
4.9 Filtro fractal....83
4.9.1 Algoritmo implementado....85

CAPÍTULO 5. FILTROS QUE UTILIZAN FUNCIONES DE MATLAB....87

5.1 Filtro imadjust....88
5.2 Filtro histeq....89
5.3 Filtro adapthisteq....89
5.4 Filtro average....90
5.5 Filtro disk....90
5.6 Filtro gaussian....91
5.7 Filtro laplacian....92
5.8 Filtro laplacian of gaussian (log)....93
5.9 Filtro motion....94
5.10 Filtro prewitt....95
5.11 Filtro sobel....95
5.12 Filtro unsharp....96

CAPÍTULO 6. FILTROS PARA DETECTAR BORDES Y OBJETOS....99

6.1 Segmentación de la imagen....99
6.2 Detectores de bordes....100
6.2.1 Métodos de detección de bordes en la función edge....102
6.2.1.1 Método Sobel....103
6.2.1.2 Método Prewitt....104
6.2.1.3 Método Roberts....105
6.2.1.4 Método del Laplaciano del Gaussiano (LoG)....105
6.2.1.5 Método Canny....105
6.3 Detección de objetos hechos por el hombre....106

6.3.1 Filtro de potencia adaptativo 108
6.3.1.1 Algoritmo implementado 109
6.3.2 Segmentación y extracción de rasgos 112
6.3.2.1 Algoritmo implementado 114

CAPÍTULO 7. PROGRAMA DESARROLLADO CON MATLAB 127

7.1 Entorno Matlab 127
7.2 Compilación del programa 129
7.3 Programa de filtrado 131
7.3.1 Abrir y guardar una imagen 132
7.3.2 Tratamiento del color 133
7.3.3 Filtro 134
7.3.4 Barra de progreso 135
7.3.5 Comparar 136
7.3.6 Barra de menús 137

CAPÍTULO 8. RESULTADOS OBTENIDOS AL REALIZAR EL FILTRADO DE LAS IMÁGENES 139

8.1 Imágenes utilizadas 139
8.2 Filtrado LUM 141
8.2.1 Filtrado LUM de suavizado 141
8.2.2 Filtrado LUM de realce 144
8.3 Filtrado fractal 148
8.4 Comparación entre diferentes filtrados no lineales 149
8.5 Filtrado de detección de objetos 150
8.6 Aplicación de varios filtros 154
8.6.1 Filtrado CS 155
8.6.2 Filtrado WMMR-MED 155
8.6.3 Filtrado de detección de bordes 156
8.6.4 Filtrado de Prewitt y Sobel 157

CAPÍTULO 9. CONCLUSIONES Y FUTURAS LÍNEAS DE INVESTIGACIÓN 159

9.1 Conclusiones 159

9.2 Publicaciones derivadas de la tesis 160

9.3 Futuras líneas de investigación 163

APÉNDICE A. EL OJO HUMANO COMO INSTRUMENTO DE VISIÓN 165

A.1 Percepción acromática 166

A.1.1 Los bastones 166

A.1.2 Brillo 167

A.1.2.1 Sensibilidad a la intensidad luminosa 167

A.1.2.2 Inhibición lateral 170

A.2 Percepción cromática 171

A.2.1 Los conos 171

A.2.2 Tono, saturación y brillo 172

A.2.3 Variaciones en la visión de los colores 174

A.3 Ilusiones ópticas 176

A.4 Implicaciones del sistema visual humano 178

BIBLIOGRAFÍA 181

Índice de figuras

Figura 1. Sonda de mano 2
Figura 2. Etapas del procesado digital de imágenes 4
Figura 3. Experimento de Colladon y Sturm 12
Figura 4. Traza baticelerimétrica 15
Figura 5. Propagación del sonido en el canal de sonido profundo o canal SOFAR 17
Figura 6. Experimento de Beudant 19
Figura 7. Funcionamiento del sonar activo 24
Figura 8. Funcionamiento del sonar de barrido lateral 27
Figura 9. Composición del fondo marino utilizando el sonar de barrido lateral 28
Figura 10. Gráfico de la frecuencia versus alcance / resolución 31
Figura 11. Longitud del pulso de una sola frecuencia 31
Figura 12. Efecto de red que se produce en los sonares con modulación de un solo tono 32
Figura 13. Longitud del pulso CHIRP 33
Figura 14. Efecto de red que se produce en los sonares con modulación CHIRP 33
Figura 15. Vehículo C'Inspector 34
Figura 16. Salida del C'Inspector en la Bahía de Santander 37
Figura 17. El C'Inspector sumergido en la piscina 38
Figura 18. Operativa desplegada a bordo del barco de aduanas 39
Figura 19. Clasificación del nivel de ruido acústico en el mar 41
Figura 20. Dispersión o scattering 42
Figura 21. Sistema aditivo de color RGB 48
Figura 22. Cubo de color RGB 49
Figura 23. Imagen dividida en las distintas capas de los modelos YC_bC_r y RGB 51
Figura 24. Proceso de filtrado en el dominio de la frecuencia 53
Figura 25. Diferentes histogramas según la claridad y contraste de la imagen 56
Figura 26. Ecualización del histograma de una imagen sonar para mejorar el contraste 57
Figura 27. Funcionamiento del filtrado espacial utilizando una máscara cuadrada 3x3 60

Figura 28. Tipos de máscaras según su forma: (a) Cuadrada (b) Rectangular (c) Circular (d) En cruz (e) Unidimensional 61
Figura 29. Distintos ejemplos de bordes 64
Figura 30. Ejemplos del proceso de implementación de las máscaras 65
Figura 31. Diagrama de flujo del funcionamiento del programa de procesado de imágenes 67
Figura 32. Ejemplo de aplicación del filtro mediana 68
Figura 33. Diagrama de bloques del filtro LUM 69
Figura 34. Filtro LUM operando como: (a) filtro LUM de suavizado (b) filtro LUM de realce (c) filtro LUM de suavizado y de realce 70
Figura 35. Funcionamiento del filtro CS 73
Figura 36. Diagrama de bloques del filtro Volterra 80
Figura 37. El fractal bidimensional XOR 84
Figura 38. Comparación entre los histogramas de los fractales XOR, AND y OR 84
Figura 39. Influencia del parámetro gamma en la función imadjust 88
Figura 40. Detección de bordes 101
Figura 41. Máscaras de convolución del método Sobel y sección de la imagen de entrada 103
Figura 42. Máscaras de convolución del método Prewitt 104
Figura 43. Máscaras de convolución del método Roberts 105
Figura 44. Sombra acústica de los objetos sumergidos 107
Figura 45. Dimensiones del fondo de un objeto 112
Figura 46. Códigos en cadena de 4 y 8 direcciones 113
Figura 47. Imagen binaria resultado de la segmentación de objetos 115
Figura 48. Resultado de la aplicación del filtro de objetos hechos por el hombre 117
Figura 49. Proyecto de implementación del programa 129
Figura 50. La pestaña y el botón build 130
Figura 51. La pestaña y el botón package 131
Figura 52. Resultado del empaquetado del programa 131
Figura 53. Ventana del programa de filtrado 132
Figura 54. Diferentes formas de abrir una imagen 132
Figura 55. Ventana emergente de búsqueda de imágenes 133
Figura 56. Tratamiento del color 134
Figura 57. Menú desplegable que contiene los distintos tipos de filtros 134
Figura 58. Selección del tipo de máscara del filtro 135
Figura 59. Elección de la imagen a la que se va a aplicar el filtro 135
Figura 60. Barra de progreso 135
Figura 61. Detención del proceso 136
Figura 62. Elección de la imágenes que se van a comparar 136
Figura 63. Comparación de dos imágenes utilizando el zoom in 136
Figura 64. Menú Archivo 137

Figura 65. Menú Ayuda 137
Figura 66. Imágenes que se han utilizado para probar los filtros 141
Figura 67. Filtrado LUM de suavizado 143
Figura 68. Detalle del filtrado LUM de suavizado 143
Figura 69. Filtrado LUM de realce 145
Figura 70. Detalle del filtrado LUM de realce 146
Figura 71. Filtrado Canny aplicado a la imagen original y a la realzada con LUM 147
Figura 72. Detalle de filtrado Canny aplicado a la imagen original y a la realzada con LUM 147
Figura 73. Detalle del filtrado fractal 148
Figura 74. Histogramas de la imagen original y de la suavizada con el filtro fractal 148
Figura 75. Panel de control del filtrado de objetos hechos por el hombre 150
Figura 76. Filtrado de detección de objetos en la imagen de los quince objetivos 152
Figura 77. Filtrado de detección de objetos en la imagen del ancla y los dos objetivos 153
Figura 78. Imagen original con rejilla 154
Figura 79. Filtrado CS variando el tamaño de la máscara y su parámetro J 155
Figura 80. Filtrado WMMR-MED variando el tamaño de la máscara y su parámetro p 156
Figura 81. Filtrado de detección de bordes 156
Figura 82. Filtrado de Prewitt y Sobel 157
Figura 83. Principales componentes del ojo humano y de la retina 166
Figura 84. Curva que relaciona la intensidad real y la intensidad percibida por el ojo humano 168
Figura 85. Contraste simultáneo acromático 169
Figura 86. Asimilación acromática 169
Figura 87. Bandas de Mach 170
Figura 88. Rejilla de Hermann 171
Figura 89. Espectro electromagnético visible para el hombre (luz visible) 172
Figura 90. Distintas gamas de tonos del rojo, naranja, amarillo, verde, azul y magenta 173
Figura 91. Distintos grados de saturación y brillo para el mismo tono de verde de partida 174
Figura 92. Ejemplos de efectos que varían la percepción de los colores 175
Figura 93. Ilusiones ópticas geométricas 176
Figura 94. Ilusión óptica de movimiento 177
Figura 95. Ilusión óptica de imagen reversible 178

Índice de tablas

Tabla 1. Densidad, celeridad del sonido e impedancia acústica para el agua salada y el aire.....11
Tabla 2. Rangos de temperatura, profundidad y salinidad para calcular la celeridad del sonido en el mar.....13
Tabla 3. Especificaciones técnicas del sonar de barrido lateral SeaKing 30
Tabla 4. Especificaciones técnicas del vehículo autónomo C'Inspector.....35
Tabla 5. Representación de colores puros en RGB.....49
Tabla 6. Comparación entre los filtrados no lineales CS, WMMR-MED y LUM 149

Resumen

Esta tesis presenta un conjunto de técnicas y algoritmos que en el pasado han sido utilizadas en el filtrado digital de imágenes ópticas y esta vez se han aplicado en imágenes acústicas, más concretamente, en imágenes submarinas procedentes de un sonar de barrido lateral de un solo haz. Para ello, se ha implementado un programa en Matlab con una interfaz de aspecto intuitivo y amigable en el que se pueden probar los distintos filtros y ver el efecto que producen en la imagen. Este proceso se realiza con la finalidad de optimizar la imagen, pero teniendo especial cuidado en la eliminación del ruido y el resaltado de bordes. Una vez realizado el preprocesado de la imagen, es el turno de la segmentación de la imagen aplicando un filtro de reconocimiento de objetos hechos por el hombre, que resulta ser muy útil a la hora de inspeccionar los fondos marinos.

Abstract

This thesis presents a collection of techniques and algorithms that in the past have been used in digital filtering of optical images and this time have been applied in acoustic images, more specifically, in underwater images from a sidescan sonar of single beam. For this, a program has been implemented in Matlab with an intuitive and user-friendly interface where the user can try out different filters and see the effect they have on the image. This process is performed in order to optimize the image, but taking special care in removing noise and finding edges. After the image pre-processing, it is time for the image segmentation applying a recognition filter of man-made objects, which is very useful for inspecting the seabed.

Capítulo 1.
Introducción

1.1 Motivación y justificación

El ser humano siempre ha sentido una gran curiosidad por todo lo que le rodea y esto le ha llevado a realizar grandes descubrimientos a lo largo de los años. Explorar, enfrentarse a lo desconocido, es el reto que se debe afrontar cuando se realiza la primera toma de contacto con un nuevo tema a investigar.

La exploración de los mares y océanos siempre ha interesado al hombre ya que desde hace muchos siglos utiliza el mar como medio de navegación. Las técnicas de orientación y la posibilidad de determinar la longitud y la latitud con una relativa exactitud han proporcionado una navegación más segura. Sin embargo, solamente desde hace unas décadas, se han logrado perfeccionar las técnicas de exploración submarinas para poder establecer de una forma fiable y precisa la topografía y naturaleza de los fondos marinos.

Conocer el fondo del mar ha sido uno de los puntos clave a la hora de reducir los riesgos de accidente que se pueden producir en la navegación marítima. Antiguamente, se utilizaba el *escandallo* para medir la profundidad y la naturaleza del fondo. El escandallo es una pieza de plomo en forma de campana que tiene en su base una cavidad rellena de sebo para tomar muestras del fondo, y a su vez va unido a un cabo, llamado *sondaleza*, marcado con nudos y cintas para marcar la profundidad. De ahí proviene que la velocidad naval se mida en nudos. El escandallo junto con la sondaleza forma lo que se denomina como *sonda de mano*. Pero utilizar

este método presenta grandes problemas a la hora de medir la profundidad del fondo, debido a los errores producidos en las mediciones por las corrientes marinas y el propio movimiento del barco. Además este método no es útil para determinar grandes levantamientos topográficos del fondo, puesto que hay que suponer una continuidad entre la zona que se está midiendo y el resto que lo rodea.

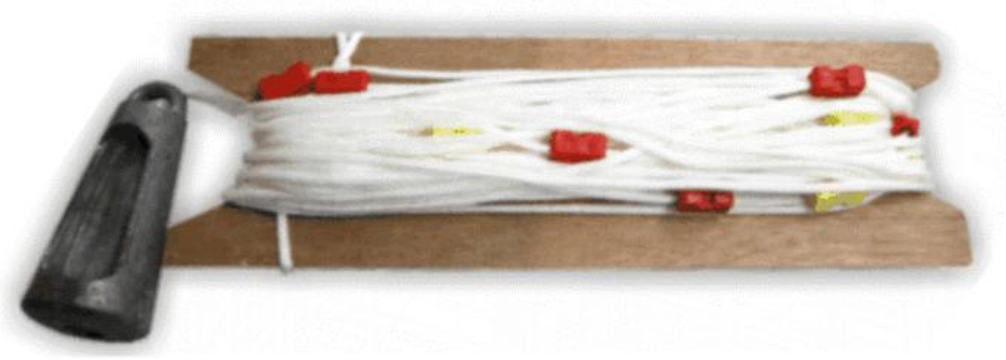


Figura 1. Sonda de mano

Con los medios tecnológicos actuales, la investigación submarina ha alcanzado cotas que serían impensables tiempo atrás. Poder sumergirse a mayores profundidades ha posibilitado mejores y precisas investigaciones en campos tan diversos como la biología, la historia o la zoología. El empleo de técnicas de sonar y dispositivos similares permite obtener desde mapas topográficos del fondo marino hasta localizar pecios hundidos, bancos de peces, determinar la composición del fondo, etc. Los vehículos sumergibles existentes son una fuente muy importante de estudio para los investigadores.

En un principio, se utilizaron medios muy simples para poder tener una perfecta visión cuando se está sumergido en el mar. Al introducir la cabeza en el agua y abrir los ojos, los objetos aparecían borrosos, por lo que se crearon dispositivos para establecer una visión nítida, basados en interponer una capa de aire entre los ojos y el medio marino. Éste sería el caso del visor que utilizan los buceadores, conocido también como careta o *anteojo de calafate,* y el casco que usan los buzos, denominado *escafandra*. Desde el punto de vista individual (buzos y buceadores), existen muchas limitaciones a la hora de realizar las inmersiones, como la velocidad reducida a la que se desplazan, la profundidad y la reserva de aire.

Tales limitaciones dieron origen al diseño de un vehículo con un sistema propulsor. Los vehículos submarinos no tripulados pueden desplazarse remolcados

por una embarcación a velocidades moderadas, o ser propulsados por motores eléctricos autónomos y gobernados de forma remota desde una embarcación en la superficie. Con este sistema, el investigador puede avanzar con relativa velocidad a unos cuantos metros del fondo, siguiendo la topografía del mismo y grabando las imágenes en vídeo al encontrar un lugar o un objeto que le interese. El problema que presenta este tipo de vehículos es que la calidad de la imagen se degrada bastante en función de la profundidad. Este hecho se produce porque los sistemas ópticos de captura de imagen necesitan de potentes sistemas de iluminación y para vehículos autónomos, que funcionan con baterías, resulta inviable. En estos casos, se utilizan imágenes obtenidas a partir de ecos de un sonar de barrido lateral. Pero debido a su coste elevado no se suelen utilizar sonares de alta resolución como son los de tipo multihaz, sino que los más empleados son los de tipo monohaz. Las imágenes que se obtienen con este tipo de sonar suelen tener baja resolución. La baja calidad unida a la aparición de ruido hace que estas imágenes sonar deben ser tratadas de un modo inteligente mediante un procesado digital para obtener resultados comparables a las imágenes obtenidas mediante métodos ópticos de mayor cuantía.

En 2009 la Universidad de Cantabria en colaboración con el Gobierno Regional de Cantabria, adquirió un vehículo submarino autónomo de inspección de altas prestaciones, el modelo C'Inspector de la marca noruega Konsberg. Lleva incorporado un sonar de barrido lateral de simple haz modelo SeaKing 675 kHz de la marca británica Tritech. A raíz de esta operación surgió la necesidad de implementar un software de procesado digital de las imágenes acústicas capturadas con el sonar para tratar de mejorar su calidad.

1.2 Objetivos y metodología

Para eliminar el ruido y mejorar la baja resolución de las imágenes obtenidas con un sonar de barrido lateral, una buena opción es utilizar técnicas de *procesado o procesamiento digital*. El procesado digital trata de imitar al que tiene lugar en el ser humano. El hombre captura la luz con cada ojo (ver Apéndice A) y después envía esta información, a través del nervio óptico, al cerebro para su procesado. Se cree que el primer paso de este procesado consiste en descomponer la imagen en elementos más simples y entonces el cerebro interpreta la escena y actúa. De esta manera, el procesado digital consta normalmente de cuatro fases:

- *Adquisición o captura de imágenes*: se realiza la digitalización de una imagen con un dispositivo adecuado para tal fin. Es decir, se transforma una imagen del mundo real a un formato que entienda el ordenador utilizando un conversor analógico-digital. Por ejemplo: el escáner, la cámara fotográfica, el sonar de barrido lateral, etc. Al capturar las imágenes, se produce una degradación en la imagen digitalizada al compararla con la original en forma de ruido y pérdida de definición. Generalmente la degradación está causada por múltiples factores: ruido generado por los sensores de captura, transmisión defectuosa de la señal captada, perturbaciones aleatorias, movimiento del sistema de captura, mala calibración del dispositivo, etc.

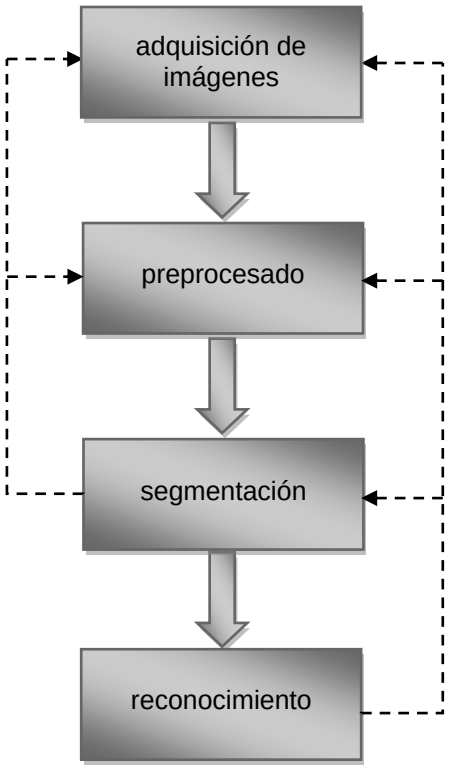


Figura 2. Etapas del procesado digital de imágenes

- *Preprocesado o procesado previo*: consiste en el tratamiento digital de las imágenes. Se efectúa para contrarrestar la degradación introducida en la etapa anterior y para que las siguientes etapas tengan mayor probabilidad de éxito. En esta fase, mediante la aplicación de filtros y algoritmos se elimina ruido de la imagen o se realzan bordes de la misma aumentando para ello el contraste.

- *Segmentación*: se extrae la información que contiene la imagen. Para ello se descompone la imagen en secciones según unas características determinadas. Habitualmente estas características suelen ser de objetos o áreas del mundo real dependiendo de qué aplicación vaya a tener el proceso. Por ejemplo, en un sistema de reconocimiento submarino con fines militares los objetos que serían de interés serían minas, barcos, submarinos, etc. Mientras que si se utiliza para fines civiles, la misma imagen tendría otros puntos de interés como animales marinos, barcos hundidos, prospección del fondo, etc.
- *Reconocimiento o clasificación*: se intenta distinguir los objetos segmentados mediante el análisis de ciertas características establecidas previamente para poder diferenciarlos. Esto es, se etiquetan o clasifican los distintos objetos de la imagen.

Estas cuatro etapas no tienen por qué seguirse de manera secuencial. En caso de que alguna de ellas falle, siempre que sea posible, se puede volver a la etapa anterior repitiendo el proceso cuantas veces sea necesario. Así, si se produce algún fallo en la etapa de reconocimiento, se volvería a la de segmentación y si el fallo es en ésta se volvería a la de preprocesado y así sucesivamente (ver figura 2).

Dentro de estas técnicas de procesado se encuentra la utilización de algoritmos que normalmente provienen de un filtro concreto o de la aplicación de varios de ellos. Con los filtros de realce se puede obtener una imagen con mejor definición y contraste de los bordes y ciertos detalles de interés. Por ejemplo, el filtro de la mediana se utiliza para eliminar ruido. Posteriormente empleando el filtro *manmade* se pueden detectar objetos hechos por el hombre. No es un método infalible ya que se pueden producir falsos positivos por lo que es esencial perfeccionarlo depurando el algoritmo.

El análisis de los datos geoespaciales de imágenes de sonar es crucial para el reconocimiento del fondo marino. Por ello, el objetivo principal de esta tesis es optimizar la calidad de las imágenes adquiridas con un sonar de barrido lateral de simple haz mediante la validación de técnicas de filtrado digital. De forma tradicional, estas técnicas han ofrecido buenos resultados en la limpieza y eliminación de ruido en imágenes ópticas. En este trabajo de tesis dichas técnicas se han aplicado con buenos resultados en las imágenes acústicas.

Como objetivos más específicos de la tesis se pueden indicar los siguientes:

- Desarrollar una aplicación con Matlab en un entorno sencillo y amigable, para aplicar las técnicas de filtrado digital en las imágenes.
- Probar los distintos filtros en las imágenes procedentes de un sonar de barrido lateral monohaz.
- Observar y evaluar cuáles son los que mejor eliminan el ruido y realzan los bordes de las imágenes del sonar de barrido lateral.

1.3 Organización de la tesis

- En el **Capítulo 1** se explican los motivos que justifican esta tesis así como los objetivos que se pretenden alcanzar. También se describe la forma en que está estructurada la tesis mediante un breve resumen de cada uno de los capítulos que la forman.
- En el **Capítulo 2** se muestran los fundamentos teóricos en los que se basa esta tesis, presentando los conceptos importantes para el desarrollo de la misma: ondas acústicas en el mar, sonar de barrido lateral y ruido acústico. Primero se explica la propagación del sonido en el mar, haciendo hincapié en la propagación no uniforme debido a las estratificaciones que hay en el mar. Después se relata una breve historia del sonar para a continuación pasar a estudiar el sonar poniendo especial atención al sonar de barrido lateral, porque es el dispositivo con el que se han capturado las imágenes. Por último se estudia el ruido acústico ya que va a tener mucha relevancia en la degradación de las imágenes obtenidas del sonar de barrido lateral.
- En el **Capítulo 3** se describen las técnicas de filtrado digital que se utilizan a la hora de realizar el preprocesado digital de las imágenes. Se comienza describiendo las imágenes digitales y el ruido digital que también es responsable de la pérdida de calidad que sufren las imágenes capturadas de forma remota por el sonar de barrido lateral. También se explican los modelos de color utilizados, el RGB y el YC_bC_r. A continuación se describen los distintos tipos de filtrado así como su clasificación. Se hace especial hincapié en el filtrado espacial pues es el que se ha empleado en la realización de esta tesis. Por último, se explican los tipos de máscaras de filtrado espacial.

- En el **Capítulo 4** se definen los términos más utilizados en los filtros. También se expone cómo se efectúa el filtrado según el tipo de imagen y la implementación de las máscaras de filtrado. A continuación se explican los filtros implementados en Matlab, así como sus fundamentos teóricos. Además se muestran los algoritmos que se han programado para realizar de forma conjunta la eliminación del ruido y el realce de los bordes en las imágenes submarinas registradas con el sonar de barrido lateral.
- En el **Capítulo 5** se describen los filtros que se han implementado utilizando las funciones que tiene Matlab dentro de su toolbox de procesado de imagen. Una toolbox es caja de herramientas con filtros y funciones pre-integradas para el filtrado de imagen óptica. Así, se han empleado filtros que se encuentran dentro de las rutinas de realce de imagen de Matlab y se han creado filtros empleando funciones especiales de dicho programa.
- En el **Capítulo 6** se presentan técnicas de filtrado para detección. En primer lugar se comenta la importancia de la segmentación de la imagen como base para la detección. Seguidamente se describe la detección de bordes mediante la utilización de diferentes métodos basados en el uso de la derivada. Después se explica la detección de objetos hechos por el hombre y se muestra el algoritmo implementado.
- En el **Capítulo 7** se comenta y explica el programa que se ha desarrollado en Matlab que incluye todos los filtros que se han descrito en los capítulos anteriores para el preprocesado de las imágenes y para la detección de bordes y objetos. Primero se describe cómo se ha compilado el programa. Después se muestra el funcionamiento de las distintas características y herramientas con las que cuenta el programa.
- En el **Capítulo 8** se analizan los resultados que se han obtenido al aplicar los distintos filtros (los de preprocesado y los de detección) sobre las imágenes obtenidas del sonar de barrido lateral. Se incluyen ejemplos con la aplicación de los diferentes filtros sobre varias imágenes de sonar de barrido lateral.
- En el **Capítulo 9** se presentan las conclusiones sobre el desarrollo de la presente tesis doctoral y se comentan las líneas de investigación que se pueden llevar a cabo en el futuro. También se muestran las publicaciones derivadas de la investigación llevada a cabo en la realización de la tesis.

- En el **Apéndice A** se explica el proceso de percepción cromática y acromática del ojo humano. Es decir, cómo el hombre percibe los colores y la escala de grises. Además, se comentan las limitaciones que el ojo presenta en determinadas circunstancias cuando observa los detalles de una imagen y las implicaciones que esto conlleva a la hora de procesar imágenes.

Capítulo 2.
Imágenes acústicas submarinas obtenidas mediante sonar de barrido lateral

En este capítulo se va a presentar el proceso de obtención de las imágenes submarinas con el sonar de barrido lateral de simple haz. Para ello, primero se explicarán unos conceptos básicos de acústica en el mar para lograr entender mejor el funcionamiento del sonar. A continuación se describirá el sonar de barrido lateral y más concretamente el modelo que se ha utilizado para la obtención de las imágenes. Finalmente se comentarán los principales fenómenos que causan degradación en las imágenes de sonar.

2.1 Justificación del uso de ondas acústicas en el mar

Se conocen cuatro métodos generales a la hora de realizar comunicaciones submarinas [33]. A continuación se va a ir viendo la idoneidad de cada uno de ellos:

- *Óptico*: el agua es muy opaca a la luz infrarroja y a la luz ultravioleta. Además, el agua no es demasiado transparente para la luz visible, con lo cual sólo se utiliza en aguas poco profundas.
- *Magnético*: la transmisión y recepción magnética se utiliza para distancias relativamente cortas, inferior a los 2 m, ya que el campo magnético se atenúa con el cuadrado de la distancia.
- *Eléctrico*: las ondas de radio o electromagnéticas se atenúan rápidamente al propagarse a través del agua salada. Esto es debido a que el agua salada es una buena conductora ya que tiene una constante dieléctrica alta y esto hace que las moléculas de agua absorban radiación electromagnética.
- *Acústico*: el agua es un buen medio para la transmisión de las ondas acústicas De hecho, es la forma de comunicación preferida en la naturaleza por algunos animales marinos como los delfines y las ballenas.

Cabe preguntarse por qué en el agua no se utilizan ondas electromagnéticas en lugar de las ondas acústicas, la respuesta está precisamente en el medio de transmisión. Como se ha visto, el agua salada al ser buena conductora hace que las ondas electromagnéticas se amortigüen muy deprisa. Por otra parte, el sonido se propaga a mayor velocidad cuando menos compresible sea el medio de transmisión. Así, el sonido se propaga más rápidamente en el agua que en el aire ya que los líquidos se pueden comprimir menos que los gases.

En conclusión, salvo para aplicaciones muy específicas, el método acústico es el que se utiliza normalmente para transmitir señales bajo el agua

2.2 Factores que influyen sobre la propagación de sonido en el mar

Para poder explicar la transmisión del sonido en el mar hay que conocer los factores que influyen sobre ella, algunos de ellos son:

- *el mar no es un medio ilimitado*: tanto el fondo como la superficie afectan a la transmisión. La superficie del mar produce una reflexión casi perfecta de la onda acústica porque el cambio de impedancia acústica característica del agua salada al aire es muy grande, aproximadamente 3600 como se va demostrar a

continuación. La impedancia acústica característica de un material viene dada por la expresión:

$$Z_0 = \rho * c \tag{2.1}$$

donde Z_0 es la impedancia acústica característica del material en Pa (kg/m*s^2), ρ es la densidad del material en kg/m^3 y c es la celeridad del sonido en ese material en m/s.

A temperatura ambiente, unos 20 ºC, para el agua salada y el aire se tienen los valores que aparecen en la siguiente tabla:

material	*ρ (kg/m^3)*	*c (m/s)*	*Z_0 (Pa)*
agua salada	1000	1500	1,5 * 10^6
aire	1,21	343	415

Tabla 1. Densidad, celeridad del sonido e impedancia acústica para el agua salada y el aire

Entonces, dividiendo la impedancia acústica característica del agua salada entre la impedancia acústica característica del aire se obtiene:

$$\frac{Z_0\ (agua\ salada)}{Z_0(aire)} = 3614 \tag{2.2}$$

Hay que tener en cuenta que el mar nunca está en reposo, siempre hay olas de mayor o menor tamaño, por lo que no es una superficie perfectamente plana. Por ello, la reflexión no puede tomarse como una imagen opuesta al otro lado de la superficie límite. Además, en aguas poco profundas se introduce otra condición de contorno al no conocerse la forma del fondo.

- *el mar no es un medio homogéneo*: existen variaciones verticales que producen refracción de las ondas acústicas por lo que se reduce la eficacia en la transmisión de la señal. Esto se debe a las variaciones de densidad y temperatura que experimenta el agua de mar.
- *el mar no es un medio isótropo*: presenta variaciones de presión y densidad.

2.3 Velocidad del sonido en el mar

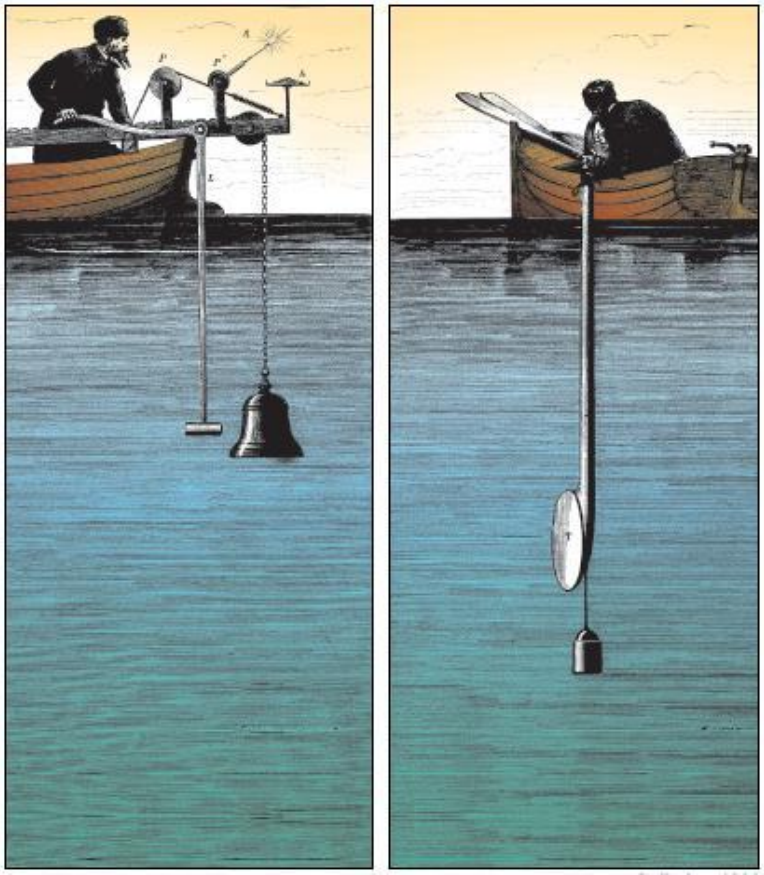


Figura 3. Experimento de Colladon y Sturm

La velocidad del sonido en el agua dulce fue medida por vez primera en 1826 por el físico suizo Jean Daniel Colladon (1802-1893) en colaboración con el matemático francés Jacques Charles François Sturm (1803-1855) en el lago Ginebra, también conocido como lago Lemán. El experimento consistía (ver figura 3) en hacer sonar una campana sumergida y al mismo tiempo explotar una carga de pólvora. El resplandor de la explosión era observado a 16 km de distancia y comparado con la llegada del sonido a la campana submarina, que era oído a través de un tubo largo sumergido en el agua. Las pruebas se realizaron por la noche para evitar interferencias de otros sonidos y para observar mejor los destellos de las explosiones. Se obtuvo un valor de la velocidad del sonido de 1435 m/s a 8,1 ºC, que está muy cerca del valor actual que es de 1439 m/s a dicha temperatura.

Para establecer el comportamiento del sonido en el agua de mar es muy importante conocer la velocidad de propagación del sonido en ese medio. Más

concretamente, lo que interesa saber es la *celeridad*, que es el valor escalar de la velocidad o el módulo del vector velocidad, porque el sonido se propaga mediante ondas esféricas en todas direcciones. La celeridad del sonido se calcula como:

$$c = \sqrt{\frac{K}{\rho}} \tag{2.3}$$

donde *c* es la celeridad del sonido en m/s, *K* es el módulo de compresibilidad del medio en Pa (kg/m*s^2) y *ρ* es la densidad del medio en kg/m^3.

La celeridad con la que el sonido se propaga en el mar no es uniforme. Esto es debido a que el módulo de compresibilidad varía con la profundidad, y la densidad depende tanto de la temperatura como de la salinidad del agua de mar. De este modo, existen tres factores que van a jugar un papel determinante en el cálculo de la celeridad:

- Presión (profundidad)
- Temperatura
- Salinidad

Durante muchos años se ha estudiado el grado de influencia de estos tres factores en el valor de la celeridad. Tanto desde el punto de vista teórico, mediante formulación matemática, como desde el punto de vista práctico. Como resultado, se han obtenido varias fórmulas empíricas a partir de los experimentos realizados, ver [10], [23], [27] y [44]. Las ecuaciones más utilizadas son las que aparecen en la tabla 2:

Autor	*Rango de temperatura (ºC)*	*Rango de profundidad (m)*	*Rango de salinidad (‰)*	*Número de términos en la fórmula*
Wilson (1960)	−4 - 30	10 - 10.000	0 - 37	23
Leroy (1969)	−2 - 34	0 - 8.000	20 - 42	13
Frye and Pugh (1971)	−3 - 30	10,33 - 9.843	33,1 - 36,6	12
Del Grosso (1974)	0 - 35	0 - 10.000	29 - 43	19
Medwin (1975)	0 - 35	0 - 1.000	0 - 45	6
Chen and Millero (1977)	0 - 40	0 - 10.000	5 - 40	15
Lovett (1978)	0 - 30	0 - 10.000	30 - 37	13
Coppens (1981)	−2 - 35	0 - 4.000	0 - 42	6
Mackenzie (1981)	−2 - 30	0 - 8.000	25 - 40	9

Tabla 2. Rangos de temperatura, profundidad y salinidad para calcular la celeridad del sonido en el mar

En la tabla 2 se puede comprobar cómo cada fórmula es válida para un rango determinado de profundidad, temperatura y salinidad [11]. El cálculo de la celeridad fuera de esos límites puede conducir a resultados erróneos. En la tabla no se incluyen las ecuaciones individuales pero sí se indica el número de términos que contiene cada una. Así la ecuación más sencilla tiene 6 términos mientras que la más complicada tiene 23. Por lo tanto, la celeridad con la que el sonido se propaga en el mar no es uniforme y viene dada por la fórmula empírica simple de Medwin [28]:

$$c = 1449{,}2 + 4{,}6\,T - 0{,}055\,T^2 + 0{,}00029\,T^3 + (1{,}34 - 0{,}01\,T)(S - 35) + 0{,}016\,D \qquad (2.4)$$

donde *c* es la celeridad del sonido en m/s, *T* es la temperatura del agua en ºC, *D* es la profundidad en relación a la superficie del mar en m y *S* es la salinidad expresada en tanto por mil (‰).

Debido a la variación de la celeridad con la profundidad, se puede decir que el mar en el plano vertical se encuentra estratificado en varias zonas. Para analizar las estratificaciones, se utilizan *gradientes* para cada una de las variables que intervienen, es decir, para la presión, temperatura y salinidad. Los gradientes se obtienen mediante la relación que existe entre la diferencia de valores de la variable y de la función.

- El *gradiente de presión*, a temperatura y salinidad constantes, es de 0,016 m/s por cada metro de aumento de profundidad.
- El *gradiente de temperatura*, a presión y salinidad constantes, es de 4,6 m/s por cada ºC de aumento de temperatura.
- El *gradiente de salinidad*, a presión y temperatura constantes, es de 1,34 m/s por cada 1‰ de aumento de salinidad.

La traza baticelerimétrica o celerimétrica representa de forma gráfica los valores de la celeridad del sonido con respecto a la profundidad. En la figura 4 puede observarse que el mar está compuesto por varias capas verticales o estratificaciones:

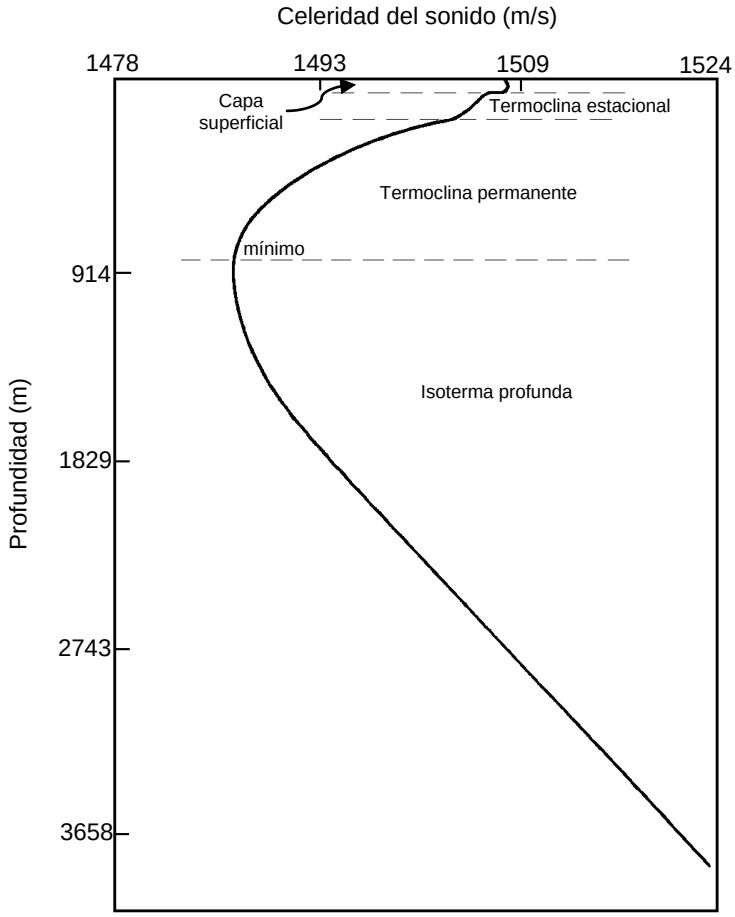


Figura 4. Traza baticelerimétrica

1. *Capa superficial o de mezcla*. Se encuentra justo debajo de la superficie del mar. En esta capa, la celeridad del sonido responde a los cambios diarios y locales de la temperatura y del viento. Puede estar compuesta por una capa de *mezcla* de agua isotérmica producida por el viento al soplar sobre la superficie del mar donde el sonido es canalizado. En días soleados y con el mar en calma, esta capa de mezcla se disipa y es reemplazada por agua cuya temperatura disminuye al aumentar la profundidad.
2. *Termoclina*. Es una región en la cual la temperatura varía con la profundidad. A su vez está dividida en dos capas:
    a. *Termoclina estacional*. Está situada bajo la capa superficial. Se caracteriza por tener un gradiente negativo de temperatura. Esto significa que la temperatura, y por lo tanto la celeridad del sonido, disminuyen al aumentar la profundidad. Durante el verano y el otoño, la

termoclina está bien definida porque las aguas superficiales son muy cálidas. Por el contrario, durante el invierno y la primavera tiende a no distinguirse de la capa superficial.

b. *Termoclina permanente*. Está ubicada bajo la termoclina estacional. Presenta un gradiente negativo de temperatura de forma suave y uniforme. Luego los cambios de estación apenas la afectan. En esta capa es donde la temperatura sufre un mayor cambio, pues la temperatura disminuye rápidamente al aumentar la profundidad.

3. *Isoterma profunda*. Comprende desde el final de la termoclina permanente hasta llegar al fondo marino. Suele tener una temperatura casi constante, entre los 3 y 4 ºC. En esta capa, la celeridad del sonido aumenta con la profundidad debido a que tiene un gradiente positivo de temperatura y al efecto de la presión.

La existencia y tamaño de estas capas varía dependiendo de la latitud, la estación del año, la hora del día y las condiciones meteorológicas [32]. Entre el gradiente negativo de la termoclina permanente y el gradiente positivo de la isoterma profunda, se produce un mínimo en la celeridad del sonido, donde el sonido tiende a concentrarse debido a la refracción. La profundidad a la que se produce esta concentración del sonido es lo que se conoce como *canal de sonido profundo* [43] o *canal SOFAR* (**SO**und **F**ixing **A**nd **R**anging). Esta dualidad en el nombre se debe a que fue descubierto y descrito de forma independiente en los años 40 por el científico ruso Leonid Brekhovskikh y por el geofísico y oceanógrafo americano Maurice Ewing. Para aprovechar este canal, la fuente acústica se coloca cerca del eje del mínimo, de esta forma el sonido queda confinado dentro del canal y puede llegar a recorrer miles de km en dirección horizontal con una pérdida mínima de señal. En la figura 5, se muestra una representación gráfica del haz sonoro de una fuente situada cerca del mínimo de la celeridad del sonido. Debido a que las ondas del sonido se curvan o refractan en dirección a la región de velocidad mínima, los cambios de temperatura o presión provocan que las ondas sonoras reboten de un lado a otro dentro del canal de sonido profundo o SOFAR, propagándose a grandes distancias.

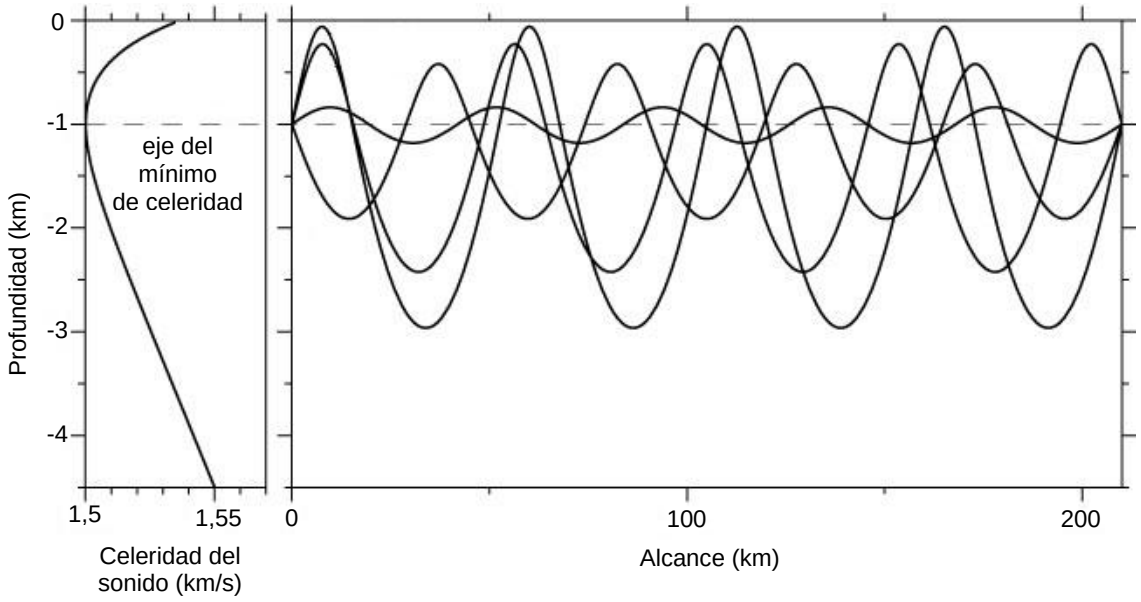


Figura 5. Propagación del sonido en el canal de sonido profundo o canal SOFAR

La posición del mínimo de la celeridad del sonido varía con la temperatura del agua y por lo tanto con la latitud. En latitudes bajas (cerca del ecuador), el mínimo de celeridad se encuentra a profundidades cercanas a 900 m, tal y como ocurre en la figura 4. Por otro lado, en latitudes altas (cerca de los polos), la temperatura fría del agua hace que el mínimo de celeridad se encuentra muy cerca de la superficie, a sólo unos cientos de metros, y tanto la termoclina estacional como la permanente tienden a desaparecer.

En aguas poco profundas, la traza no es clara y tiende a ser impredecible. En este tipo de aguas, la celeridad está muy influenciada por el calor o el frío que haga en la superficie, cambios en la salinidad y la presencia de corrientes de agua.

2.4 El sonar

Las referencias bibliográficas que se han utilizado en la elaboración del apartado referente al sonar y al sonar de barrido lateral son [2], [4], [6], [9], [15], [36], [43] y [45].

2.4.1 Origen y definición del término sonar

Durante la Primera Guerra Mundial nació el término ASDICS en Gran Bretaña para referirse a la tecnología de ecolocalización de alta frecuencia. La expresión *ASDICS* es un acrónimo de **A**nti-**S**ubmarine **D**ivision **ICS**, lo que se podría traducir como *perteneciente a la División Antisubmarina,* ya que la terminación -ics en inglés significa "perteneciente a". Una explicación alternativa para el mismo vocablo (pero sin la s final) fue el acrónimo *ASDIC*: **A**llied **S**ubmarine **D**etection **I**nvestigation **C**ommittee, *Comité aliado de investigación para la detección de submarinos*. Pero se cree que fue una invención creada por el almirantazgo británico en 1939 como respuesta a su significado para la editorial Oxford University Press, que se encontraba redactando su diccionario de inglés, puesto que ningún comité con este nombre ha sido hallado en los archivos de la marina británica.

En la Segunda Guerra Mundial se acuñó el término de *sonar* en Estados Unidos. En un principio, sonar no era un acrónimo sino una analogía fonética de la palabra radar. Más tarde se le añadió su significado como acrónimo. Pero independientemente de su origen, en 1948 con la creación de la OTAN y la estandarización de las señales el nombre sonar acabó sustituyendo al vocablo original.

El término sonar, según la Real Academia de la Lengua (RAE), es el acrónimo de **SO**und **NA**vigation and **R**anging, es decir, navegación y localización por sonido. De este modo, un sonar es el aparato electroacústico que detecta la presencia y situación de objetos sumergidos, mediante ondas producidas por el propio objeto o por la reflexión de las que emite el aparato. Además, la RAE informa que la palabra sonar primero se consideró una palabra llana y tenía tilde (sónar) y actualmente se considera una palabra aguda y por tanto no lleva tilde (sonar).

2.4.2 Historia del sonar

En el mundo animal algunos mamíferos como delfines y ballenas llevan utilizando el sonido para comunicarse y detectar objetos desde hace millones de años. Comparado con estos animales marinos, el ser humano es todavía un aprendiz utilizando el sonar, ya que en comparación su descubrimiento es mucho más reciente.

Las primeras investigaciones que hacen referencia al sonar datan de finales del siglo XV. En 1490, Leonardo da Vinci escribió: "*Si paras tu barco e introduces el extremo de un tubo en el agua, y aplicas el oído al otro extremo, oirás barcos que se encuentran a gran distancia de ti*". Es decir, Leonardo da Vinci utilizó un tubo metido

en el agua con el extremo colocado en su oreja para poder detectar barcos escuchando el sonido que hacían bajo el agua. Además, disminuyó el ruido al parar su propia embarcación. Esto sería un rudimentario ejemplo de lo que hoy conocemos como sonar pasivo.

En 1762, Benjamin Franklin llevó a cabo un experimento en el río Delaware donde golpeando dos piedras observó que el ruido producido podía ser oído a una distancia mayor si se introducía el oído dentro del agua del río. Aunque Franklin sólo oyó el sonido a aproximadamente 1,5 km supuso que podía ser escuchado a una distancia mayor porque el sonido no se oía débil sino fuerte, como si estuviera muy cerca de la oreja. La explicación de estas observaciones se basa en que la velocidad de propagación del sonido es mucho mayor en el agua que en el aire. De esta forma, el sonido bajo el agua recorre distancias mayores que en el aire y por tanto puede ser escuchado a mayores distancias.

Durante el siglo XIX, varios científicos realizaron investigaciones que sentaron las bases para el posterior desarrollo del sonar:

Figura 6. Experimento de Beudant

- A comienzos del siglo XIX, M. Beaudant realizó la primera medida de la velocidad del sonido en agua salada en Marsella, en el Mar Mediterráneo. Realizó su experimento de una manera tosca pero efectiva (ver figura 6). Dos observadores con sus relojes sincronizados se encontraban a bordo de dos barcas a una distancia conocida. Cada uno iba acompañado por un buceador. Una campana era golpeada a intervalos regulares en una de las barcas y en el momento que era oída por el buceador que se encontraba en el otro puesto hacía una señal y el tiempo era anotado por el observador. No era un método muy exacto porque se perdía tiempo en el proceso de anotación de los tiempos.
- En 1826, Daniel Colladon y Charles Sturm miden la velocidad del sonido en agua dulce en el lago Ginebra. Realizaron las pruebas con un método más preciso que el empleado por Beaudant, utilizando una sincronización de luz y sonido (ver figura 3).
- En 1842, Joule descubre el fenómeno de la *magnetoestricción*, es decir, el cambio en la forma de algunos materiales ferromagméticos cuando son atravesados por un campo magnético.
- En 1880, los hermanos Pierre y Jacques Curie descubrieron el efecto *piezoeléctrico*, propiedad que presentan ciertos cristales dieléctricos como el cuarzo y la sal Rochelle. Dicho fenómeno consiste en la respuesta de esos materiales a una presión aplicada, presentando una pequeña diferencia de potencial en varias de sus caras. El efecto inverso, es decir, que un campo eléctrico aplicado al cristal distorsione su forma fue predicho por Gabriel Lippmann y confirmado por los hermanos Curie en 1881.

A finales del siglo XIX y principios del XX se utilizaron campanas subacuáticas conjuntamente con faros para avisar a los barcos que se acercaban a la costa en condiciones peligrosas o de poca visibilidad. En condiciones de buena visibilidad proporcionaban una indicación de la distancia estimando el tiempo de retraso entre la luz y el sonido, al igual que se hace en una tormenta contando los segundos que pasan desde que se ve el relámpago hasta que se oye el trueno.

El siglo XX fue el periodo de tiempo en el que mayor avance tuvo el sonar, motivado en gran medida por la pérdida de vidas humanas tanto en el hundimiento del Titanic como durante las dos grandes guerras mundiales, sin olvidar la carrera armamentística que se produjo entre EEUU y la URSS durante la llamada guerra fría.

- En 1912 se hundió el transatlántico Titanic al chocar contra un iceberg. Este hecho desencadenó una serie investigaciones para evitar la colisión de barcos contra icebergs. Así, unas semanas más tarde del desgraciado accidente del Titanic, el físico inglés Lewis Richardson patentó el primer dispositivo de ecolocalización submarina.
- En 1913, el ingeniero canadiense Reginald Fessenden patentó un transductor electromagnético y demostró su utilidad al año siguiente (1914) al detectar la presencia de un iceberg a una distancia aproximada de unos 3 km. Este dispositivo fue conocido como el *oscilador de Fessenden*.

Las bajas causadas por el uso de los submarinos de combate durante la Primera Guerra Mundial (1914-1918) provocaron que las investigaciones se centraran en el desarrollo de un sistema de detección submarina. Los estudios de franceses y británicos comenzaron de forma simultánea:

- En 1915, el físico francés Paul Langevin junto con el ingeniero ruso Constantin Chilowski investigaban en París sobre la ecolocalización (sonar activo). Al mismo tiempo en Manchester, Albert Beaumont Wood y Harold Gerrard trabajaban en dispositivos de escucha pasivos, es decir, hidrófonos (sonar pasivo). En 1916, el físico canadiense Robert Boyle se unió al equipo de investigación británico para encargarse de la parte de la ecolocalización.
- En 1917, Boyle hizo progresos con el oscilador de Fessenden detectando submarinos a 910 m. Sin embargo esta línea fue abandonada temporalmente porque para localizar submarinos se necesitaba un transductor de alta frecuencia. Mientras tanto ese mismo año en Francia, Langevin había comenzado a experimentar con el cuarzo (material piezoeléctrico). Como al principio las señales obtenidas eran muy débiles, Brillouin y Beauvais diseñaron un amplificador de válvulas que hizo que las señales llegaran hasta una distancia de 6 km.

Pero los grandes avances en el sonar activo llegaron cuando los equipos franceses y británicos empezaron a compartir sus hallazgos durante una serie de reuniones. Como consecuencia de estas puestas en común, Boyle comenzó a trabajar en los transductores de cuarzo y los franceses se encargaron del transductor de alta frecuencia.

- En 1918, Langevin obtuvo ecos de un submarino usando los transductores de cuarzo de alta frecuencia. Un mes después, Boyle consiguió captar ecos de un submarino a una distancia de unos 460 m y construyó el primer sonar activo. Así, en 1918 tanto Gran Bretaña como Francia lograron construir sistemas de sonar activos.
- En 1919, el científico alemán H. Lichte enuncia una teoría según la cual las ondas sonoras cambian de dirección hacia arriba o hacia abajo en el agua cuando se encuentran con pequeñas diferencias de temperatura, salinidad y presión.
- En 1920, los británicos instalaron su ASDIC (sonar activo) en su buque HSM Antrim para poder probarlo. Una vez concluidas las pruebas, comenzó la producción de unidades en 1922. Así, en 1923, la 6ª flotilla británica de destructores, tenía buques equipados con ASDIC. Este desarrollo del sonar activo en Gran Bretaña no se produjo en Estados Unidos hasta 1931, año en el que se construyó el sonar QB, el primer sonar activo americano. Este sonar llevaba un transductor de sal de Rochelle, material piezoeléctrico más eficiente que el cuarzo.
- En 1937, el geofísico afroamericano Athelstan Spilhaus construye el batitermógrafo (BT) dispositivo provisto de un sensor de temperatura y un transductor para detectar los cambios de temperatura que se producen en el agua con la profundidad. En la Segunda Guerra Mundial el BT se instaló en los cascos de los submarinos americanos para ajustar la precisión del sonar y también para realizar tácticas de evasión.
- En 1943, los científicos americanos Maurice Ewing y J. L. Worzel de la Universidad de Columbia descubren el canal de sonido profundo o canal SOFAR. Trabajando de forma independiente, el científico ruso Leonid Brekhovskikh descubre el mismo fenómeno.
- En 1944, los británicos desarrollaron las sonoboyas, una mezcla entre sonar y boya, con finalidad militar.

Al inicio de la Segunda Guerra Mundial (1939-1945), la Armada Británica poseía un completo sistema de ataque antisubmarino instalado tanto en buques de superficie como en submarinos. Pero la efectividad de los primeros ASDIC era limitada ya que se

perdía el contacto con el sonar en los momentos previos al ataque, y por tanto, se debía disparar a ciegas, lo que facilitaba al enemigo la adopción de tácticas evasivas durante ese periodo de tiempo. En el transcurso de la guerra se desarrollaron unos nuevos equipos ASDIC que utilizaban diferentes formas de onda para cubrir los puntos ciegos de forma continua.

Al estallar la Segunda Guerra Mundial, la tecnología del sonar desarrollada por los británicos, fue transferida a los Estados Unidos. De esta forma, la investigación en el campo del sonar y de la acústica submarina creció notablemente en este país. Se mejoraron las sonoboyas, se desarrolló el sonar para detección de minas, etc. Se creó el NDRC (*National Defence Research Commitee*), el comité nacional de investigación y defensa. La sección sexta del NDRC realizó un amplio programa de acústica submarina llevado a cabo por la Universidad de California, el laboratorio de electrónica naval de San Diego y la institución oceanográfica de Woods Hole. Las publicaciones de los estudios realizados al acabar la guerra constituyen la base de la acústica submarina.

Mientras los esfuerzos de los aliados se centraron en la detección de los submarinos alemanes (*U-Boots*) utilizando el sonar activo, los científicos alemanes desarrollaron equipos muy sofisticados de sonar pasivo. Estos equipos eran conocidos como *Gruppenhorchgerät* (GHG). El sistema GHG fue instalado en el crucero Prinz Eugen, para avisar de forma efectiva de los ataques de torpedos dando tiempo a evitarlos.

Durante el final de la Segunda Guerra Mundial y el comienzo de la Guerra Fría (1947-1991), las investigaciones del sonar continúan. Al acabar la guerra, los avances logrados por los alemanes son estudiados por británicos y americanos. Y durante la Guerra Fría los Estados Unidos y la URSS, entablaron una carrera de armamento nuclear hasta la disolución de la Unión Soviética en 1991.

- En 1955, con la aparición de submarinos nucleares, capaces de lanzar misiles de largo alcance, se volvió a la investigación del sonar pasivo. La marina de los Estados Unidos comienza a desarrollar grandes redes de escuchas submarinas tanto fijas, como remolcadas por buques. A este tipo pertenecen los sistemas SOSUS (SOund SUrveillance System), TACTAS (TACtical Towed Array Sonar) y SURTASS (SURveillance Towed Array Sensor System) que se siguen utilizando en la actualidad.

Durante los últimos años, el estudio del sonar no se centra únicamente en su uso militar sino también en su uso civil. De esta forma, se ha intensificado el estudio del sonar activo de bajas frecuencias para poder realizar la detección a grandes distancias. También se ha potenciado la reducción al máximo del ruido emitido por los buques. Además, se utiliza el análisis espectral de las frecuencias que son emitidas en el mar para realizar su clasificación, es lo que se denomina LOFAR (LOw Frequency Analisys and Recording).

2.4.3 Tipos de sonar

Básicamente existen dos tipos de sonar: el *sonar activo* y el *sonar pasivo*. El sonar activo emite una señal sonora mediante un emisor. Dicha señal al encontrar un obstáculo rebota, y es recogida por un receptor. Es decir, emite y recibe ondas acústicas; en cambio, el sonar pasivo sólo recibe (escucha).

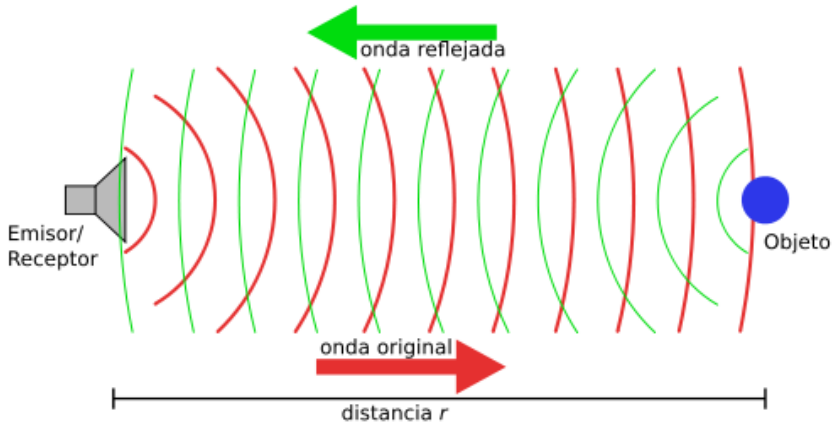


Figura 7. Funcionamiento del sonar activo

2.5 El sonar de barrido lateral

El sistema de sonar de barrido lateral (SBL) es un sonar activo que proporciona imágenes digitales de la superficie del fondo marino. Las aplicaciones más comunes de este tipo de sonar incluyen: cartografía de grandes secciones de los fondos marinos; localización de tuberías o cables, accidentes submarinos, obstrucciones, etc. Más concretamente, la localización de naufragios y minas, así como la búsqueda de aeronaves y de mercancías hundidas, requieren el uso del sonar de barrido lateral.

2.5.1 Historia

Uno de los inventores del sonar de barrido lateral fue el científico alemán Julius Hagemann que fue llevado a los Estrados Unidos tras finalizar la Segunda Guerra Mundial. Trabajó para la Marina Americana desde 1947 hasta su fallecimiento en 1964. Su patente de un sistema de sonar de barrido lateral en 1958, no fue conocido hasta su desclasificación en 1980.

Sistemas experimentales de sonar de barrido lateral fueron desarrollados en los años 50 en la Institución Scripps de oceanografía, los laboratorios Hudson y por Harold Edgerton en el MIT (Massachusetts Institute of Technology). También, durante los años 50, la empresa Westinghouse fabricó sonares de barrido lateral con finalidad militar. En la década de los 90, se encargó de desarrollar y fabricar sistemas más avanzados con propósitos militares determinados, como encontrar bombas H hundidas o un submarino ruso perdido en el mar.

El primer sistema sonar de barrido lateral con finalidad comercial fue el de Kevin Hughes en 1960, que consistía en una ecosonda de un solo canal con un transductor en forma de abanico.

En 1963, Harold Edgerton, Edward Curley y John Yules utilizaron un sonar de barrido lateral de 12 kHz para encontrar el buque faro Vineyard hundido en la costa de Massachusetts.

De 1963-1966, un equipo dirigido por Martin Klein en Edgerton, Germeshausen & Grier (EG&G) desarrolló con éxito el primer sistema comercial de sonar de barrido lateral de arrastre de doble canal. Por ello, Martin Klein es considerado como el padre del sonar de barrido lateral comercial.

En 1967, Edgerton utilizó el sonar de Klein para ayudar a Alexander McKee a encontrar el buque insignia del rey Enrique VIII, el Mary Rose. Ese mismo año, Klein utilizó su sonar para ayudar al arqueólogo George Bass a localizar una nave de 2000 años de antigüedad en la costa de Turquía.

En 1968, Klein fundó la empresa Klein Associates (conocida actualmente como L-3/Klein) y continuó trabajando en realizar mejoras en el sonar de barrido lateral. Así, se desarrolló el primer sistema comercial de alta frecuencia (500 kHz), así como el primer sonar de barrido lateral de doble frecuencia y el primer sonar que combinaba el barrido lateral con la caracterización del suelo marino.

En 1985, Charles Mazel de Klein Associates realizó el primer manual de formación en el sonar de barrido lateral y los primeros video-tutoriales acerca del sonar de barrido lateral.

A principios de los años 80, se implementa el sistema de sonar de barrido lateral llamado GLORIA (Geological Long-Range Inclined Asdic) para determinar la topografía del suelo marino en grandes áreas, para ello operaba a bajas frecuencias. Fue utilizado por el Servicio Geológico de Estados Unidos y por el Instituto Oceanográfico de Reino Unido para la obtención de imágenes de las plataformas continentales en todo el mundo.

Hasta mediados de los 80, las imágenes de los sonares de barrido lateral comerciales se registraban en papel especial mediante impresoras térmicas. A finales de los 80, los sistemas comerciales comenzaron a utilizar los nuevos convertidores de escaneo digital que imitaban de forma más barata a los convertidores de escaneo analógico que utilizaban los sistemas militares para almacenar las imágenes obtenidas con el sonar de barrido lateral en cintas de vídeo. Hoy en día, los datos procedentes del sonar de barrido lateral se almacenan directamente en el disco duro del ordenador.

2.5.2 Montaje y funcionamiento

Los sistemas de sonar de barrido lateral se pueden montar en UUVs (Unmanned Underwater Vehicles) con los transductores colocados en cada lado (babor y estribor) del vehículo o en un sistema de arrastre (towfish). El sistema de arrastre tiene forma hidrodinámica y se asemeja a un torpedo o un misil con un cuerpo largo que contiene los transductores, la electrónica y un conjunto de aletas en la cola para mantenerse en la dirección del arrastre. El sistema towfish normalmente va remolcado bajo el agua, yendo por detrás del barco de investigación que se encuentra en la superficie.

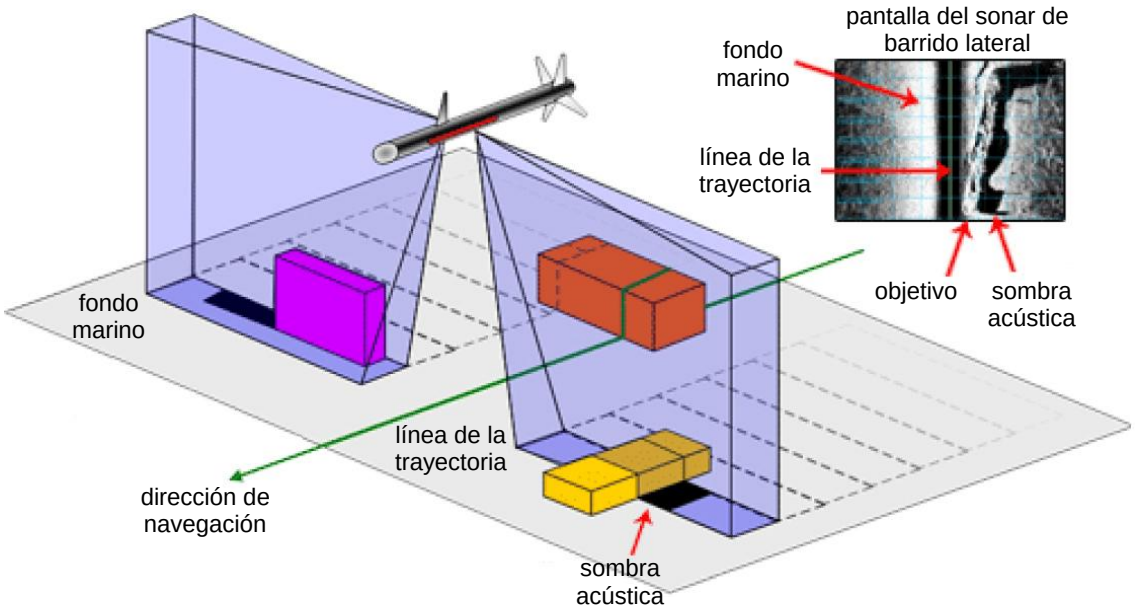


Figura 8. Funcionamiento del sonar de barrido lateral

Los transductores, que se encuentran fijados en los laterales del vehículo submarino, transmiten un estrecho haz acústico cónico o en forma de abanico en ambos lados de la línea de la trayectoria.

El sonar de barrido lateral transmite un haz acústico estrecho hacia los lados de la dirección en la que se propaga a través del medio marino. Como el haz acústico viaja hacia fuera desde el sonar de barrido lateral, el fondo marino refleja algo de la energía del sonido incidente en la dirección del sonar, esto se conoce como retrodispersión (*backscatter*). El tiempo de recorrido de los pulsos acústicos desde el SBL se registra junto con la amplitud de la señal de retorno como una serie de tiempo y se envía a una consola para su visionado e interpretación, tal y como se muestra en la figura 8.

Como con cualquier sonar acústico, el SBL sólo muestra ecos de objetos que reflejan el sonido de nuevo al transductor del sonar, de tal manera que las superficies que son duras y brillantes a veces sólo se ven cuando están a determinados ángulos con respecto al sonar de barrido lateral.

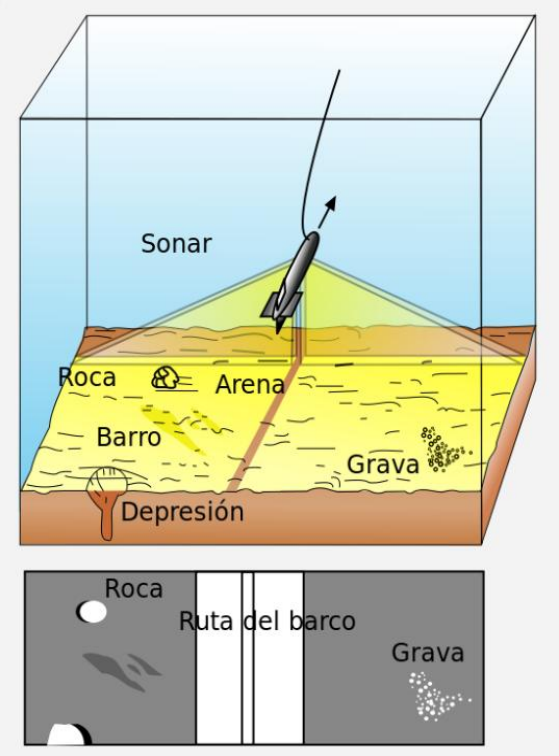


Figura 9. Composición del fondo marino utilizando el sonar de barrido lateral

Además, las texturas rugosas del fondo marino pueden borrar completamente los blancos que son pequeños. Algunos tipos de materiales, como metales, piedras, grava o roca volcánica reciente, son muy eficientes reflejando los pulsos acústicos (alta retrodispersión). Por el contrario, sedimentos más finos como la arcilla y limo, no reflejan bien el sonido (baja retrodispersión). Los reflectores potentes crean ecos fuertes, mientras que los reflectores débiles crean ecos débiles. Conociendo estas características, se utiliza el sonar de barrido lateral para examinar la composición del fondo del mar, como puede apreciarse en la figura 9.

2.5.3 Interpretación de los datos obtenidos

La interpretación de los datos de sonar de barrido lateral se desarrolla con la experiencia. Las reflexiones de objetos pequeños y aislados no dan ninguna indicación de su forma o su posición. Estructuras realizadas por el hombre, tales como plataformas o paredes de piedras tienden a tener patrones regulares que son más fáciles de identificar. Ver con un sonar de barrido lateral es ver el mundo como si estuviera hecho de plástico negro y brillante, en la oscuridad y con sólo una linterna de haz muy estrecho para iluminarlo. Conviene recordar que cuando se está cerca de

objetos grandes, o en una depresión marina, el campo de visión del sonar de barrido lateral puede estar muy limitado. Los reflectores potentes pueden ocasionar múltiples ecos a lo largo de la línea de propagación, y se identifican por ser equidistantes en el alcance. La vista en planta proporcionada por el sonar de barrido lateral tampoco muestra la altura de un objeto, a menos que aparezca la sombra acústica, en cuyo caso la longitud de la sombra acústica se relaciona con la altura del objeto, su alcance y la altura del sonar de barrido lateral.

La experiencia con el sonar de barrido lateral permite al operador ser capaz de poner a punto controles con rapidez y eficacia, tales como la ganancia del receptor y el alcance dinámico para obtener el mejor fondo posible y maximizar las capacidades de rendimiento del sonar de barrido lateral. Los controles independientes están disponibles para los transductores de babor y estribor. Aunque normalmente la configuración sería la misma, bajo ciertas condiciones (por ejemplo, fondos marinos en pendiente) diferentes ajustes pueden ser necesarios.

2.5.4 Sonar SeaKing 675 kHz

El sonar de barrido lateral que está instalado en el vehículo autónomo C'Inspector es el modelo SeaKing de la empresa británica Tritech [41], que opera a una frecuencia de 675 kHz. Esto significa que aunque tiene mayor resolución (que otro de baja frecuencia) también tiene menor alcance. Es decir, este tipo de sonar está pensado para la detección de objetos cercanos. Es un sonar de un sólo haz (monohaz) y por lo tanto no va tener la resolución que proporciona un sonar de barrido lateral multihaz. Pero para intentar contrarrestar esta desventaja es por lo que se ha desarrollado el software de procesado digital. En la tabla 3 se pueden observar las especificaciones del sonar de barrido lateral utilizado para la obtención de las imágenes que posteriormente van a ser procesadas digitalmente con el programa desarrollado.

Especificaciones técnicas del sonar de barrido lateral SeaKing de Tritech	
Frecuencia de operación	675 kHz (modulación chirp)
Ancho de haz horizontal (-3dB)	0.5°
Ancho de haz vertical (-3dB)	30°
Peso en aire (electrónica)	500 g
Peso en aire/agua (transductor)	300/120 g
Profundidad máxima del transductor	300 m
Consumo eléctrico	18 - 36V@12VA
Nivel de la señal del transmisor	200 dB re 1µP @ 1 m
Longitud del pulso del transmisor	200 µs
Sensibilidad del receptor	> 2µV rms
Rango de control de ganancia	80 dB
Rango dinámico de visualización	40 dB (configurables por el usuario)
Rango de muestreo de datos	5 - 200 µs
Resolución de los datos	4 - 8 bits (configurables por el usuario)
Protocolos de Comunicación	Arcnet, RS-232
Velocidad de transmisión de datos	RS232 hasta 115.2 kbaud, Arcnet 156 ó 78 kbaud

Tabla 3. Especificaciones técnicas del sonar de barrido lateral SeaKing

2.5.5 Frecuencia versus resolución / alcance

La transmisión del sonido bajo el agua es limitada. Esto se nota más en los alcances que normalmente se utilizan. El alcance de la energía acústica de alta frecuencia se reduce considerablemente, debido al agua de mar, de 50 a 100 m. La energía acústica de baja frecuencia se reduce a una velocidad mucho menor, pudiendo tener un alcance de más de 200 m. Por lo tanto, existe una solución de compromiso entre las imágenes de mayor resolución producidas por un sonar de barrido lateral de alta frecuencia y el mayor alcance proporcionado por un sonar de barrido lateral de baja frecuencia (pero menor resolución). Los sistemas sonar de barrido lateral de la empresa Tritech se suministran normalmente con una de las dos frecuencias de operación, por lo general 325 kHz y 675 kHz. Con el sonar de menor frecuencia (325 kHz) se pueden detectar objetivos a distancias grandes de más de 200

metros. El sonar de mayor frecuencia (675 kHz) tiene un haz más estrecho y corto (100 m de alcance) para obtener imágenes más detalladas de objetivos más cercanos.

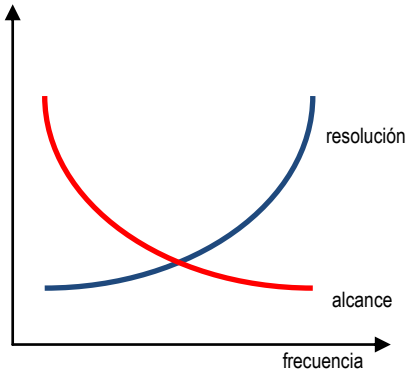


Figura 10. Gráfico de la frecuencia versus alcance / resolución

2.5.6 Modulación CHIRP

La modulación CHIRP (Compressed High Intensity Radar Pulse), es un pulso comprimido de alta intensidad de radar. Las técnicas CHIRP se han utilizado durante muchos años en múltiples sistemas de radar comerciales y militares. Actualmente, estas técnicas se han modificado y adaptado a los sistemas comerciales acústicos de sonar de barrido lateral. La empresa Tritech utiliza la modulación CHIRP como base acústica de toda su gama de sonares digitales.

Para entender los beneficios de la utilización de las técnicas acústicas CHIRP, es necesario analizar las limitaciones que se presentan cuando se utilizan las técnicas convencionales que utilizan un solo tono. Un pulso acústico consta de un interruptor con dos posiciones on / off que modula la amplitud de la única frecuencia de la portadora.

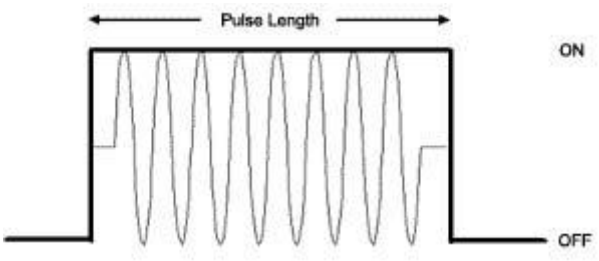


Figura 11. Longitud del pulso de una sola frecuencia

La capacidad del sistema acústico para distinguir o resolver objetivos viene determinada por la longitud del pulso. Para obtener en el agua la energía acústica suficiente para una correcta identificación del objetivo y con una amplia variedad de alcances, la longitud de transmisión del pulso tiene que ser relativamente larga. La ecuación para determinar la resolución en alcance de un sistema acústico de un solo tono viene dada por la siguiente expresión:

$$Resolución\ en\ alcance = (longitud\ del\ pulso * velocidad\ del\ sonido)/\ 2 \qquad (2.5)$$

En un sistema sonar de Tritech de un solo tono la longitud de pulso más pequeña es de 50 µs. El valor típico de la velocidad del sonido en el agua es de 1500 m/s. Por tanto, la resolución sería de 37,5 mm. Este resultado determina la *resolución en alcance* (capacidad para distinguir objetivos separados) del sistema sonar de un solo tono.

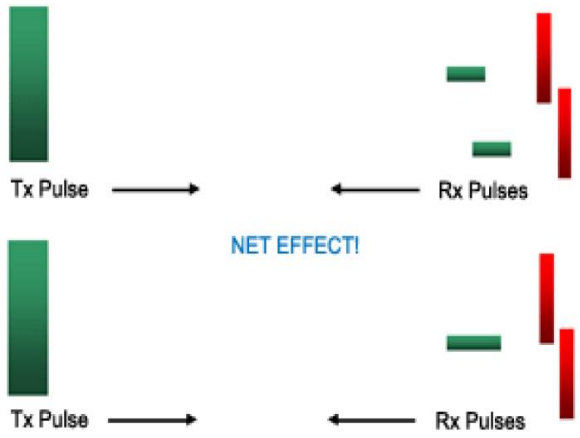


Figura 12. Efecto de red que se produce en los sonares con modulación de un solo tono

Utilizando el ejemplo anterior, si dos objetivos se encuentran a una distancia menor que 37,5 mm, significa que el sonar no puede distinguir uno de otro y sólo mostrará un único objetivo de mayor tamaño que los originales. En la figura 12, se ilustra este fenómeno también conocido como efecto de red.

Con la utilización de la modulación CHIRP se pueden superar estas limitaciones. En lugar de utilizar un pulso con una sola frecuencia de la portadora, la frecuencia dentro del pulso es barrida en un amplio intervalo durante el tiempo que dura la transmisión del pulso. Esto crea una firma del pulso acústico, y así el sonar

sabe qué se transmite y cuándo. Mediante técnicas de reconocimiento de patrones, el sonar puede localizar su propia firma a través del eco de los objetivos.

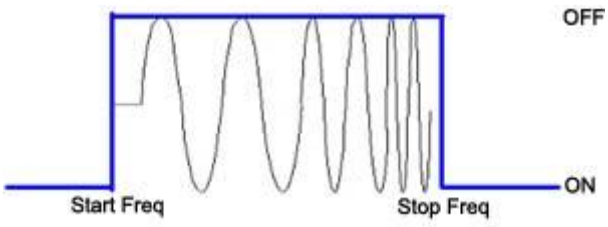


Figura 13. Longitud del pulso CHIRP

En un sistema CHIRP, el factor que determina la resolución en alcance es el ancho de banda del pulso. Ahora, la resolución en alcance viene dada por la ecuación:

$$Resolución\ en\ alcance = (velocidad\ del\ sonido)/(ancho\ de\ banda\ x\ 2) \qquad (2.6)$$

El ancho de banda típico de un sistema de Tritech con modulación CHIRP es 100kHz. El valor estándar de la velocidad del sonido en el agua es 1500 m/s. Ahora, la resolución sería de 7,5 mm, valor que si lo comparamos con el anterior de 37,5 mm se observa que se ha producido una mejora en la resolución en un factor de 5. De esta manera, cuando dos ecos acústicos se superponen, los pulsos CHIRP no se funden en uno solo. La frecuencia en cada punto del pulso es diferente, y el sonar es capaz de resolver los dos objetivos de forma independiente como se observa en la figura 14.

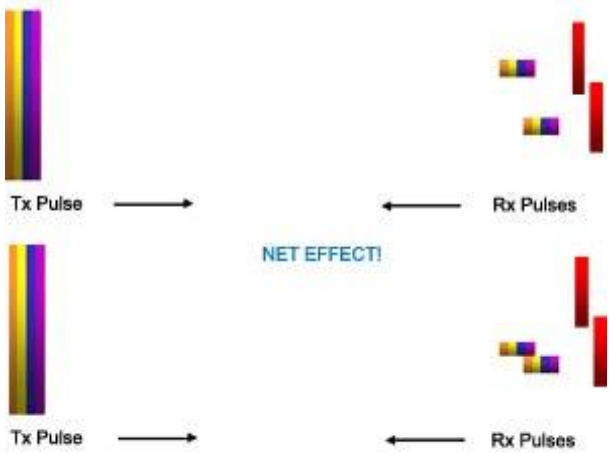


Figura 14. Efecto de red que se produce en los sonares con modulación CHIRP

En resumen, con la modulación CHIRP se logra una mejora en la resolución, mayor longitud en los pulsos transmitidos (mayor rango de operación) y una mejora en el rechazo del ruido de fondo.

2.5.7 Pruebas experimentales para la obtención de imágenes

Las pruebas se han realizado con el vehículo autónomo, propiedad de la Universidad de Cantabria, en el que está instalado el sonar de barrido lateral, como se muestra en la figura 15. Se trata de un vehículo autónomo submarino AUV (*Autonomous Underwater Vehicle*) modelo C'Inspector de la empresa noruega Kongsberg [19]. Cuenta con una conexión de datos de alta velocidad a través de cable de fibra óptica y dispone de la autonomía que le proporcionan dos baterías.

Figura 15. Vehículo C'Inspector

El C'Inspector puede utilizarse para tareas de inspección de fondo y detección de artefactos sumergidos. El vehículo lleva incorporados, entre otros:

- sonar de barrido lateral
- sonar de navegación
- compás electrónico
- sensor de inclinación
- sensor de profundidad

- sensor de fugas
- cámara submarina

En la tabla 4, se muestran las especificaciones técnicas del C'Inspector dadas por el fabricante.

Especificaciones técnicas del vehículo autónomo C'Inspector de Konsberg	
Longitud	1650 mm
Anchura (máxima)	459 mm
Altura	212 mm
Peso en aire	35 kg
Iluminación	2 luces halógenas
Velocidad máxima de avance frontal	6 nudos
Velocidad máxima marcha atrás	3 nudos
Velocidad máxima vertical	4,2 nudos
Inclinación	+/− 45º
Duración de baterías en funcionamiento	2 horas a 2 nudos 1 hora a 3 nudos
Profundidad máxima de trabajo	300 m

Tabla 4. Especificaciones técnicas del vehículo autónomo C'Inspector

El lugar en el que se han realizado las pruebas experimentales ha sido en la Bahía de Santander. En la figura 16 se muestran varias fotografías de una campaña de medidas realizada con el C'Inspector desde la dársena del muelle de Gamazo.

En la figura 16 (a) se muestra el centro de control y navegación del vehículo (*operator station, power station y winch*). En la parte superior izquierda se encuentra el winch o máquina que automáticamente despliega o recoge la fibra óptica. A través de la fibra óptica se controlan y se reciben los datos tanto del sonar de navegación como del sonar de barrido lateral, así como de los demás sensores (posicionamiento, balanceo, profundidad, etc.) con los que cuenta el vehículo. La longitud de esta fibra óptica que actúa de interfaz entre el puesto de control y el vehículo es de 1 km, pudiéndose ampliar hasta los 10 km.

(a) Centro de control y navegación del C'Inspector

(b) Estación de operación

(c) Despliegue del sistema de hidrófonos de posicionamiento acústico

(d) El C'Inspector en superficie

Figura 16. Salida del C'Inspector en la Bahía de Santander

En la figura 16 (b) se muestra en detalle la estación de operación del C'Inspector. Se puede observar una pantalla en las que se muestra las imágenes ópticas captadas por la cámara de vídeo instalada en el vehículo. También se ve la

pantalla del ordenador de control. En la zona izquierda de dicha pantalla se puede visualizar el programa que controla todos los aspectos relacionados con la navegación del vehículo. En la zona derecha, se observa el programa que sirve para almacenar las imágenes acústicas obtenidas con el sonar de barrido lateral.

En la figura 16 (c) se muestra el despliegue del sistema de posicionamiento acústico, que para funcione correctamente debe sumergirse aproximadamente 1 m. Este sistema está formado por un arreglo de tres hidrófonos, un transductor y el sistema de GPS.

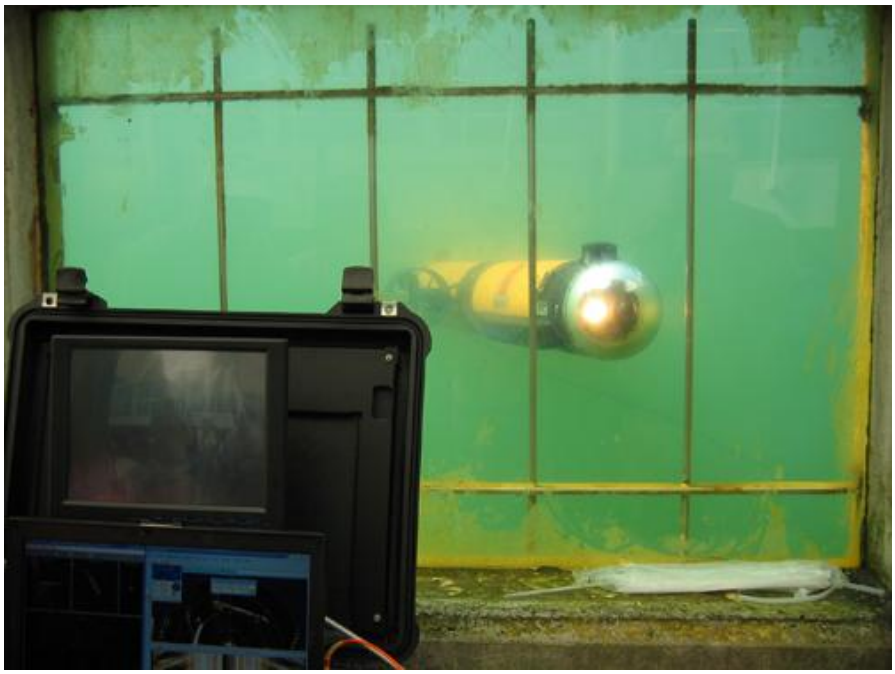


Figura 17. El C'Inspector sumergido en la piscina

También se han realizado pruebas para puesta a punto del equipo en una piscina con agua de mar (figura 17) y una salida en el barco del servicio de aduanas (figura 18), en la que se obtuvo la imagen del remolcador hundido que se utilizará posteriormente para comprobar las técnicas de filtrado.

Hay que explicar que para realizar estas pruebas y experiencias, con el propósito de obtener imágenes del fondo marino con el sonar de barrido lateral, es una ardua tarea porque se necesitan que se den una serie de circunstancias que en alguno de los casos no se puede controlar. Una persona sola no puede ni montar, ni manejar todo el equipo. Para poder realizar salidas con este equipo es necesario como mínimo cuatro personas y a veces no es fácil conseguir que dicho número de personas

puedan quedar el mismo día. También es muy importante la previsión del tiempo, ya que al ser experiencias al aire libre en caso de que haga mal tiempo, la prueba se tiene que cancelar. Además, no hay que olvidar que en este caso el laboratorio de pruebas es el mar con todos los inconvenientes que éste puede originar. Por ejemplo, no se puede dejar todo el equipo montado como se podría hacer en un laboratorio normal sino que cada vez que se vayan a realizar unas nuevas medidas hay que volver a montar todo el equipo de nuevo.

Figura 18. Operativa desplegada a bordo del barco de aduanas

2.6 Ruido acústico

Las señales acústicas en el mar pueden ser originadas por una gran cantidad de fuentes. Este tipo de señales pueden ser generadas por actividades humanas, organismos marinos, fenómenos naturales, etc. Un mismo sonido puede ser considerado como señal o como ruido dependiendo del receptor y de si obstaculiza o no la recepción.

2.6.1 Terminología

Seguidamente se van a definir una serie de conceptos que son importantes a la hora de comprender el concepto de ruido acústico.

- *Ruido*: toda señal no deseada que se mezcla con la señal que se quiere transmitir. Dicha señal está formada por la suma de diversas componentes que varían en amplitud y fase, en muchos casos de forma aleatoria.
- *Interferencia*: cualquier señal que obstaculiza la observación de otra.
- *Nivel de ruido*: ruido total de todas las fuentes que interfieren en la recepción del sonido. Es la suma del ruido propio y del ruido de fondo
    - *Ruido propio*: ruido generado desde la plataforma de recepción de la señal sonora y que interfiere en su recepción. Ejemplos: ruido de la maquinaria (motores), ruido de flujo (suciedad en el casco de la nave), cavitaciones de la hélice, etc.
    - *Ruido ambiente o de fondo*: ruido en segundo plano que puede causar interferencias en la recepción de la señal. Se origina en numerosas fuentes, las cuales no se pueden identificar fácilmente. De esta forma, el ruido ambiente se refiere al ruido que queda tras haber identificado todas las fuentes conocidas. Tipos de ruido de fondo:
        - Hidrodinámico: causado por el movimiento del agua, como son las mareas, las corrientes, el viento, la lluvia, etc.
        - Sísmico: causado por el movimiento de la tierra, un claro ejemplo son los terremotos.
        - Biológico: producido por la fauna marina. Por ejemplo: ballenas, delfines, etc.
        - Tráfico marítimo: originados por vehículos marinos, como barcos, submarinos, yates, motos acuáticas, etc.
- *Relación señal/ruido*: es la relación que hay entre la potencia de la señal que se transmite y la potencia de ruido que la contamina. Se mide en dB.

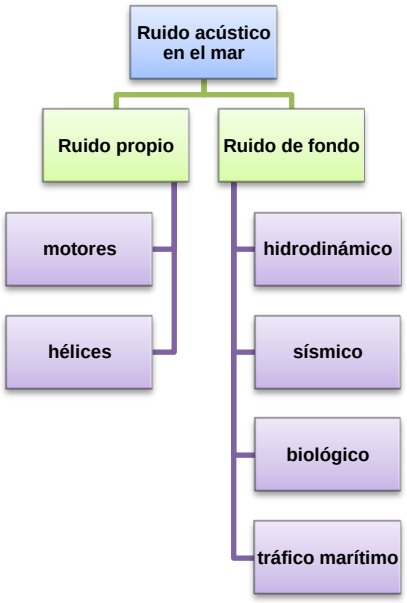


Figura 19. Clasificación del nivel de ruido acústico en el mar

2.6.2 Ruido que afecta al sonar

Además de los ecos devueltos por los blancos submarinos, el sonar está sujeto a señales de ruido. Las señales de ruido enmascaran los ecos recibidos y esto interfiere con la detección de objetos sumergidos. Cuanto más grande es el nivel de ruido, más difícil se vuelve la localización. El ruido que afecta al sonar es una combinación de reverberación, ruido ambiental y ruido propio del equipo sonar.

- *Reverberación*: se produce debido a que el agua de mar no es un líquido puro sino que contiene impurezas que van desde pequeñas partículas como sedimentos y arena hasta organismos de fitoplancton y zooplancton. Después de una tormenta, el agua que se encuentra cerca de la superficie contiene gran cantidad de pequeñas burbujas de aire. Todas estas impurezas interceptan y vuelven a radiar una parte de la energía acústica transmitida. La radiación de sonidos por estas impurezas es lo que se denomina *dispersión (scattering)*. Esto significa que si las ondas sonoras golpean un objeto cuyas dimensiones son pequeñas en comparación con la longitud de onda del sonido, se produce este fenómeno que falsea los resultados al no producirse el retorno. El efecto total de la dispersión es lo que se denomina *reverberación*. Puede ser tratada como parte del ruido de fondo que afecta al sonar.

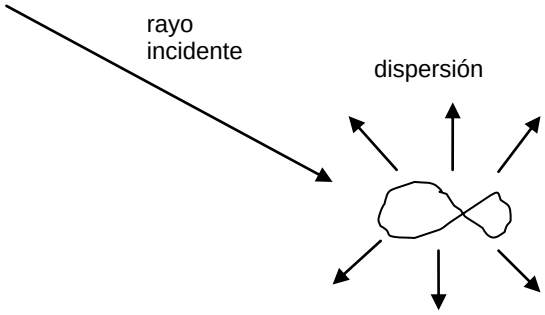


Figura 20. Dispersión o scattering

- *Ruido ambiental*: está formado por los sonidos naturales del mar que son generados por olas, animales marinos, etc. Se ha visto con más detalle en el anterior apartado de terminología.
- *Ruido propio*: es el provocado por el propio sonar junto con la maquinaria y movimientos del vehículo en el que está instalado. El sonar normalmente se monta en un barco o en un vehículo submarino autónomo (UUV). El ruido electrónico del propio sonar se considera despreciable frente a los demás ruidos. Las principales fuentes de ruido propio que afectan al funcionamiento del sonar son:
    - Ruido producido por la vibración del motor y de la hélice del vehículo.
    - Ruido de flujo: es un ruido hidrodinámico generado por el flujo de agua que hay alrededor del transductor y del casco del vehículo.

  La *cavitación* es un efecto hidrodinámico que se produce cuando la hélice gira en el agua a gran velocidad. Entonces el agua pasa rápidamente entre las palas de la hélice, generándose una descompresión debido a la conservación del principio de Bernoulli. Si se alcanza la presión de vapor, el agua pasa del estado líquido al gaseoso, formándose burbujas o cavidades. Éstas se mueven a zonas de mayor presión donde implosionan, es decir, repentinamente regresan al estado líquido. La implosión causa ondas de presión que se desplazan en el agua a una velocidad cercana a la del sonido. Esto significa que cuando las burbujas se rompen, originan ruido. Durante la navegación se puede disminuir el ruido reduciendo las revoluciones de la hélice o el ángulo de sus palas.

El nivel de ruido propio se eleva a medida que aumenta la velocidad del vehículo. El aumento de velocidad hace que la maquinaria se mueva más deprisa, la hélice gire más rápido y el flujo de agua alrededor del transductor sea mayor. Por otra parte, el nivel de ruido también depende del estado en que se encuentre el mar. Así, el nivel de ruido propio será bajo cuando el mar esté en calma mientras que será alto cuando el mar esté agitado o con fuerte oleaje.

Capítulo 3.
Preprocesado de imágenes mediante filtrado digital

En este capítulo se van a revisar las técnicas de filtrado que se han empleado en el software desarrollado. Para ello primero se comenzará explicando las características de las imágenes digitales así como los modelos de color utilizados. Después se comentarán los filtros empleados en el filtrado digital.

3.1 Imágenes digitales

Una imagen continua f(x,y) se describe de manera aproximada por una serie de muestras igualmente espaciadas que se encuentran organizadas en una matriz MxN, donde cada elemento de la matriz es una cantidad discreta:

$$f(x,y) = \begin{pmatrix} f(0,0) & f(0,1) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & \dots & f(1,N-1) \\ \vdots & \vdots & \ddots & \vdots \\ f(M-1,0) & f(M-1,1) & \dots & f(M-1,N-1) \end{pmatrix} \quad (3.1)$$

La matriz f(x,y) representa lo que comúnmente se denomina una *imagen digital*. Cada uno de sus elementos es un componente de la imagen, o también llamado *píxel*.

3.1.1 Ruido digital

El ruido se debe a las variaciones aleatorias que se producen en los valores de los píxeles de la imagen. De este modo, se entiende por *ruido en imágenes digitales* cualquier valor de un píxel que no se corresponda exactamente con la realidad. Generalmente el ruido digital es producido por el dispositivo utilizado en la adquisición de las imágenes y el que se añade en su transmisión. También se puede añadir ruido en el procesado de la imagen si no se tiene cuidado en la elección de los filtros. Los principales tipos de ruido en imágenes digitales son:

- *Ruido impulsivo*: también llamado *ruido sal y pimienta*, se caracteriza por la aparición de píxeles aleatorios con valores de intensidad extremos. Tiene mucha influencia en la calidad de la imagen debido a la gran diferencia entre el valor de la intensidad del píxel con ruido y la de sus píxeles vecinos. Una imagen con esta clase de ruido presenta píxeles oscuros en regiones claras, y píxeles claros en regiones oscuras. La distribución de ruido impulsivo P(x) puede expresarse como [46]:

$$P(x) = \begin{cases} P_a & x = a \\ P_b & x = b \\ 0 & resto \end{cases} \tag{3.2}$$

  donde, P_a y P_b son funciones de densidad de probabilidad, *a* y *b* son valores de intensidad de píxeles de la imagen.

  - Si $b > a$ ➔ el valor de *b* será un punto claro en la imagen y el de *a* será un punto oscuro en la imagen.
  - Si P_a o $P_b = 0$ ➔ el ruido impulsivo se llama pulso unipolar.
  - Si P_a y $P_b \neq 0$, especialmente cuando $P_a \sim P_b$ ➔ el ruido impulsivo se llama ruido sal y pimienta y se encuentra distribuido por la imagen.

  Normalmente los píxeles con ruido son puntos blancos y negros. El efecto de este ruido en la imagen se traduce en el deterioro de bordes y detalles, así como en la distorsión de ciertas características de la imagen. Se puede eliminar aplicando el *filtro mediana*, pero tiene efectos secundarios negativos en la resolución, la textura y el contorno de la imagen.

- *Ruido blanco gaussiano:* señal arbitraria con una densidad espectral constante y cuya amplitud sigue una distribución normal o de Gauss, por lo que su espectro de energía es constante para todas las frecuencias [22]. Su resultado es la suma o resta de un determinado valor al nivel de intensidad real,

independientemente de su valor original. Esto hace que los píxeles que forman la imagen cambien un pequeño valor siguiendo la distribución gaussiana:

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}}\, e^{\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{3.3}$$

donde, *P(x)* es la distribución gaussiana del ruido, μ es la media y σ es la desviación estándar.

Suele ser debido a componentes electrónicos como sensores, digitalizadores, etc. Aunque produce pequeñas variaciones en la imagen, la afecta de manera global, es decir, la intensidad de todos los píxeles se ve alterada. El ruido blanco gaussiano se consigue reducir con el *filtro de la media*, pero provoca que los contornos y otros detalles de la forma de la imagen se difuminen.

- *Ruido estructural*: también llamado *ruido grano o ruido speckle*. Es una variación de la señal reflejada en una región no homogénea produciendo una apariencia granulada de la imagen. Causa una reducción del contraste lo que hace más difícil el posterior análisis de las texturas que aparecen en las imágenes adquiridas de forma remota, como es el caso de imágenes procedentes de un sonar de barrido lateral. Generalmente se produce por el eco que se forma en ciertos materiales, muchos de ellos metálicos, que poseen una estructura interna formada por pequeños reflectores que no son resolubles por la longitud de onda del haz ultrasónico que incide sobre ellos.

3.2 Modelos de color

Un modelo de color tiene como finalidad la determinación de colores de una forma estándar. Así pues, se define *modelo de color* como la especificación de un sistema de coordenadas y un subespacio dentro de él donde cada color puede ser representado por un sólo punto. Hoy en día existen distintos modelos de color, según estén orientados al hardware o al usuario. Seguidamente se van a explicar los modelos que se han utilizado a la hora de implementar el software.

3.2.1 Modelo RGB

Recibe su nombre por las iniciales en inglés de los tres colores primarios: **R**ed, **G**reen, **B**lue (rojo, verde, azul). Es un modelo de color orientado al hardware y está basado en la fisiología de la retina del ojo humano (ver Apéndice A). Es de los modelos más utilizados y se emplea principalmente en la mayoría de dispositivos de hardware, como por ejemplo en televisores y monitores de ordenador. Los colores primarios son los correspondientes a la respuesta máxima de los conos del ojo a estímulos monocromáticos puros, que para este modelo son: el rojo, el verde y el azul. Es un sistema aditivo de color ya que toda la gama de colores se obtiene mediante la suma de los colores primarios a distintas intensidades. En este sistema colorimétrico los colores secundarios son: el amarillo, el cian (azul verdoso) y el magenta (rosa fucsia). Son producidos a partir de la combinación de los colores primarios a intensidades iguales.

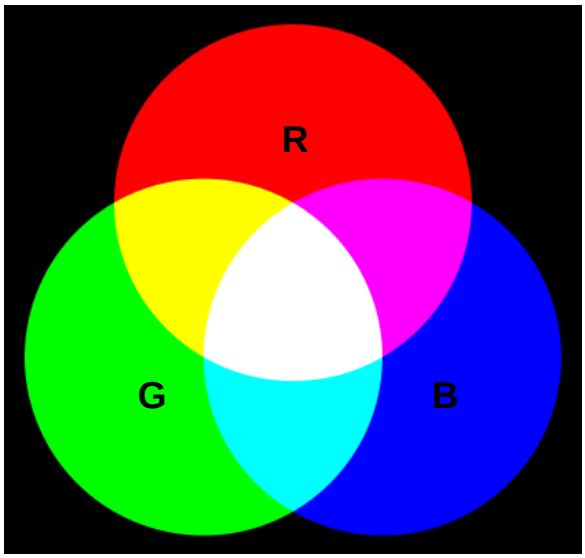


Figura 21. Sistema aditivo de color RGB

Cada color primario se codifica con 1 byte (8 bits) ➔ 2^8 = 256 tonos de ese color, que van desde el valor mínimo de intensidad: 0, hasta el valor máximo de intensidad: 255.

En la tabla 5 se muestran los valores de intensidad de los colores primarios y secundarios puros, así como los del blanco y el negro.

Colores	(R, G, B)
Blanco	(255, 255, 255)
Negro	(0, 0, 0)
Primarios	
Rojo	(255, 0, 0)
Verde	(0, 255, 0)
Azul	(0, 0, 255)
Secundarios	
Amarillo	(255, 255, 0)
Cian	(0, 255, 255)
Magenta	(255, 0, 255)

Tabla 5. Representación de colores puros en RGB

Como se puede observar en la figura 22, el modelo RGB se basa en un sistema de coordenadas cartesianas que forma un cubo en el que los valores de rojo, verde y azul se encuentran en los tres vértices de los ejes. Los valores de amarillo, cian y magenta están situados en los otros tres vértices exteriores. El negro se halla en el origen y el blanco queda en el vértice opuesto. Entre el color negro y el blanco se localiza la escala de grises, en la que se encuentran todos los tonos de gris.

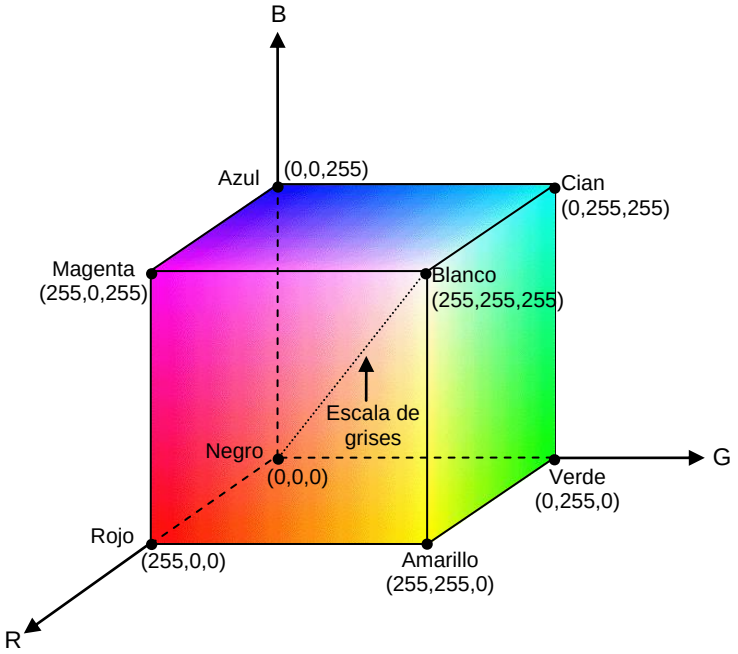


Figura 22. Cubo de color RGB

Utilizar este modelo de color para realizar el procesado de imágenes por computador, hace que la digitalización se realice en 3 capas. Cada una de ellas representa a un color primario (rojo, verde, azul). Por lo tanto, la suma de todas las capas da lugar a la imagen. Este hecho debe tenerse en cuenta cuando se van a procesar imágenes en color, ya que en este caso se utiliza una matriz tridimensional de la forma MxNx3. Por lo tanto, el coste computacional se triplica si es comparado con el procesado de una imagen en escala de grises donde se emplea una matriz bidimensional del tipo MxNx1.

La *profundidad de color* de una imagen es el número de colores distintos que pueden contener cada uno de los píxeles o puntos que la forman. Depende del número de bits que puede almacenar un píxel. A mayor profundidad de color, mayor número de bits y por tanto, mayor número de colores que puede almacenar una imagen. Para imágenes en RGB se utiliza una profundidad de color de 8 bits en cada capa, teniendo en cuenta las 3 capas de este modelo, se tiene un total de 24 bits por píxel ➔ 2^{24} = 16.777.216 millones de colores distintos para cada píxel. Esto es lo que comúnmente se conoce como *color verdadero* porque es aproximadamente el número de colores que el ojo humano puede detectar. Para imágenes en escala de grises se emplea una profundidad de píxel de 8 bits por píxel ➔ 2^8 = 256 tonos de gris diferentes para cada píxel.

3.2.2 Modelo YC_bC_r

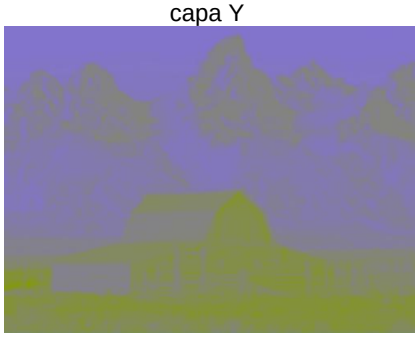
Es un modelo de color orientado al usuario, lo que quiere decir que está basado en la premisa de que el ojo humano es más sensible a la luminosidad que a los colores. Además es un modelo no uniforme, ya que separa la luminancia del resto de componentes. Este modelo es muy utilizado tanto en fotografía como en vídeo digital. Consta de tres capas, cada una de las cuales es una combinación lineal del modelo RGB [34]:

$$
\begin{aligned}
&Y = 0{,}299\,R + 0{,}587\,G + 0{,}114\,B \\
&C_b = 0{,}564\,(B - Y) = 0{,}564\,(-0{,}299\,R - 0{,}587\,G + 0{,}886\,B\,) \\
&C_r = 0{,}713\,(R - Y) = 0{,}713\,(\,0{,}701\,R - 0.587\,G - 0.114\,B\,)
\end{aligned}
\qquad (3.4)
$$

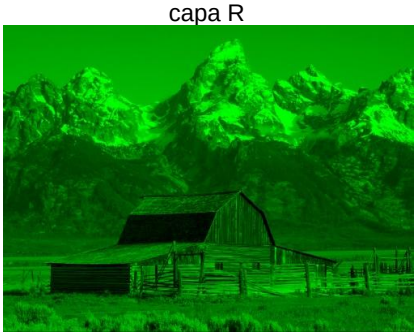
imagen original

capa Y

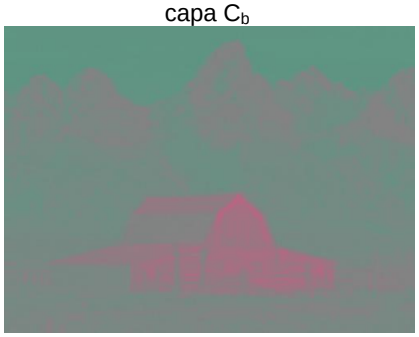
capa R



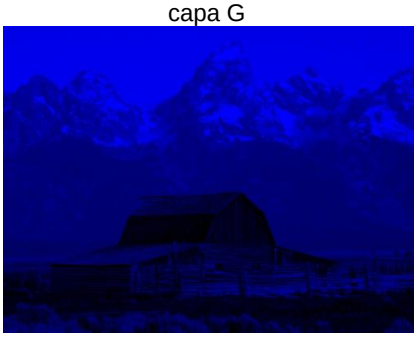
capa C_b



capa G



capa C_r



capa B

Figura 23. Imagen dividida en las distintas capas de los modelos YC_bC_r y RGB

La capa Y es la componente de *luminancia*. Representa la imagen original en escala de grises. Contiene la información más relevante de la imagen porque el ojo es más sensible a esta información que a los colores.

Las otras dos capas contienen la información sobre el color, es decir, son las capas cromáticas o de *crominancia*. La capa C_b es la componente de crominancia que contiene la diferencia entre el canal azul y la luminancia. La capa C_r es la componente de crominancia que contiene la diferencia entre el canal rojo y la luminancia.

La capa Y es una copia de la imagen original en escala de grises, mientras que las capas de color C_b y C_r no aportan detalles sino color a la imagen, tal y como se puede apreciar en la

figura 23.

Así pues, en el modelo de color RGB las tres capas contienen información sobre los elementos y el color de la imagen, mientras que en el modelo YC_bC_r sólo la capa Y es la portadora de los detalles de la imagen. Este hecho será muy útil más adelante, a la hora de procesar las imágenes procedentes del sonar de barrido lateral.

3.3 Filtrado digital

El filtrado digital es el conjunto de técnicas que se encuentran englobadas dentro del preprocesado digital de imágenes, como ya se explicó en la introducción de la tesis. Su objetivo principal es obtener a partir de una imagen original otra imagen final que resulte más adecuada para una aplicación específica. Para ello, se mejoran ciertas características de la imagen realizando operaciones de procesado sobre ella. Luego, las metas que se pretenden alcanzar con la aplicación de filtros sobre una imagen son:

- Eliminar ruido: suprimir aquellos píxeles cuyo nivel de intensidad es muy distinto al de sus vecinos.
- Suavizar la imagen: reducir la cantidad de variaciones de intensidad entre píxeles vecinos.
- Detectar bordes: localizar los píxeles donde se produce un cambio brusco en el nivel de intensidad.
- Realzar bordes: destacar los bordes que se encuentran en una imagen.

Por lo tanto, los filtros son considerados como operaciones que se aplican en los píxeles de una imagen digital para optimizarla. Estas operaciones engloban técnicas globales y locales. Las técnicas globales actúan sobre toda la imagen en conjunto,

como el filtrado frecuencial y la ecualización del histograma. Las técnicas locales operan de forma individual sobre cada píxel de la imagen, como ocurre en el filtrado espacial. Así pues, el filtrado digital se puede llevar a cabo tanto en el dominio de la frecuencia como en el dominio del espacio.

3.3.1 Filtrado frecuencial

Este tipo de filtrado utiliza técnicas en el dominio de la frecuencia, lo que significa que la imagen original de partida se pasa a dicho dominio haciendo su transformada de Fourier, ver [13] y [18]. Una vez en el dominio de la frecuencia, mediante el teorema de convolución, se aplica un filtro frecuencial para suprimir las frecuencias que no interesan. Por último se calcula la transformada inversa de Fourier para regresar al dominio espacial de partida (ver figura 24).

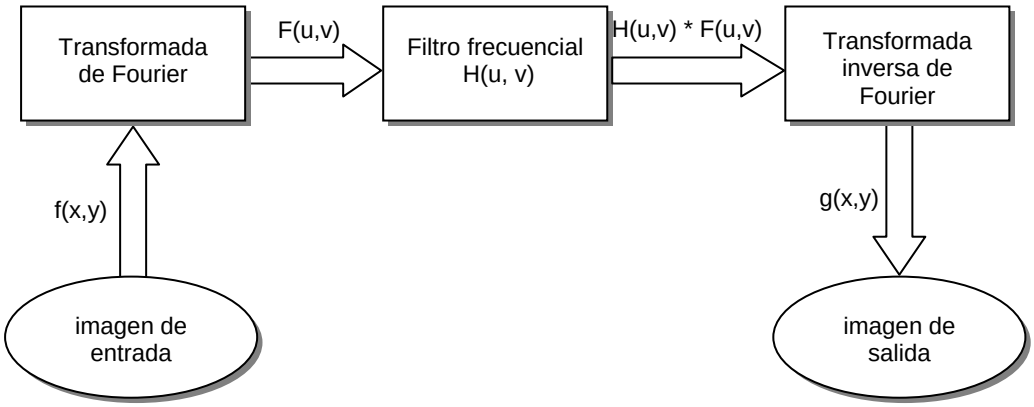


Figura 24. Proceso de filtrado en el dominio de la frecuencia

Teorema de convolución en el domino espacial: dadas dos funciones *f(x)* y *g(x)*, la convolución de ambas viene dada por *f(x)*g(x),* y se define mediante la integral:

$$f(x) * g(x) \ = \int_{-\infty}^{\infty} f(\alpha) g(x-\alpha) d\alpha \qquad (3.5)$$

donde α es una variable ficticia que se utiliza para la integración.

Si se pasa al dominio frecuencial y se aplica la transformada de Fourier a la ecuación (3.3), queda más sencilla:

$$F[f(x) * g(x)] = F[f(x)] * F[g(x)] \quad (3.6)$$

Sea *g(x,y)* una imagen formada por la convolución de una imagen original *f(x,y)* y *h(x,y)* un operador lineal invariante de posición:

$$g(x,y) = h(x,y) * f(x,y) \quad (3.7)$$

Entonces, por el teorema de convolución, se cumple la siguiente relación en el dominio de la frecuencia:

$$G(u,v) = H(u,v) * F(u,v) \quad (3.8)$$

donde *G*, *H* y *F* son respectivamente las transformadas de Fourier de *g*, *h* y *f*.

Por último, se vuelve al dominio espacial realizando la transformada inversa de Fourier:

$$g(x,y) = F^{-1}[H(u,v) * F(u,v)] \quad (3.9)$$

donde g(x,y) es la imagen de salida y F^{-1} es la transformada inversa de Fourier.

3.3.1.1 Clasificación de los filtros frecuenciales

Los filtros frecuenciales basan su funcionamiento en la eliminación de bandas de frecuencia que no contienen información útil, dejando pasar únicamente las que convengan para un determinado propósito. De esta forma, existen varios tipos de filtros frecuenciales:

- *Filtro paso alto*: atenúa o elimina las frecuencias bajas manteniendo sin cambios las frecuencias altas. Como las frecuencias altas corresponden a cambios bruscos de intensidad, este tipo de filtro consigue mejoras en el realce de bordes y refuerza el contraste de la imagen. Pero como desventaja, aumenta el ruido existente en la imagen.
- *Filtro paso bajo*: atenúa o elimina las frecuencias altas y mantiene sin variaciones las frecuencias bajas. Como las bajas frecuencias son las responsables de las pequeñas variaciones de intensidad, la aplicación de este

tipo de filtro es equivalente a la de de un filtro de suavizado ya que consigue reducir el ruido suavizando las transiciones de la imagen. Pero como inconveniente, diluye los bordes provocando que la imagen quede difuminada.

- *Filtro paso banda*: atenúa o elimina las frecuencias muy altas o muy bajas manteniendo invariantes una banda de frecuencias intermedias.

El filtrado frecuencial no es aconsejable para el tratamiento de imágenes sonar puesto que para este tipo de imágenes se quiere obtener de forma simultánea una reducción del ruido y un aumento de los detalles. Además, se prefiere utilizar los filtros espaciales ya que en general tienen un mejor rendimiento con un menor coste en memoria y en tiempo de ejecución. Sin embargo no existe un único filtro espacial que sea óptimo para todas las imágenes [24].

3.3.2 Ecualización del histograma

El *histograma* es una representación gráfica normalmente en forma de barras verticales, en la que la superficie de cada una de las barras es proporcional a la frecuencia de los valores representados.

Para representar el histograma de una imagen, en el eje horizontal se representan los niveles de luminosidad y en el eje vertical se representan las frecuencias. De este modo, aquél nivel que más se repita será el que tenga una barra más alta.

En la parte izquierda del histograma es donde se acumulan los tonos más oscuros mientras que en la parte derecha se acumulan los tonos más claros. Si al observar el histograma se comprueba que las barras son más altas en la parte izquierda, disminuyendo a medida que van hacia la derecha hasta desaparecer completamente, se puede afirmar que es el histograma de una *imagen subexpuesta* o demasiado oscura. Si ocurre el efecto contrario, es decir, no hay barras en la parte izquierda y comienzan a aumentar hacia la derecha hasta llegar a su valor máximo, se puede decir que el histograma es el de una *imagen sobreexpuesta* o demasiada clara.

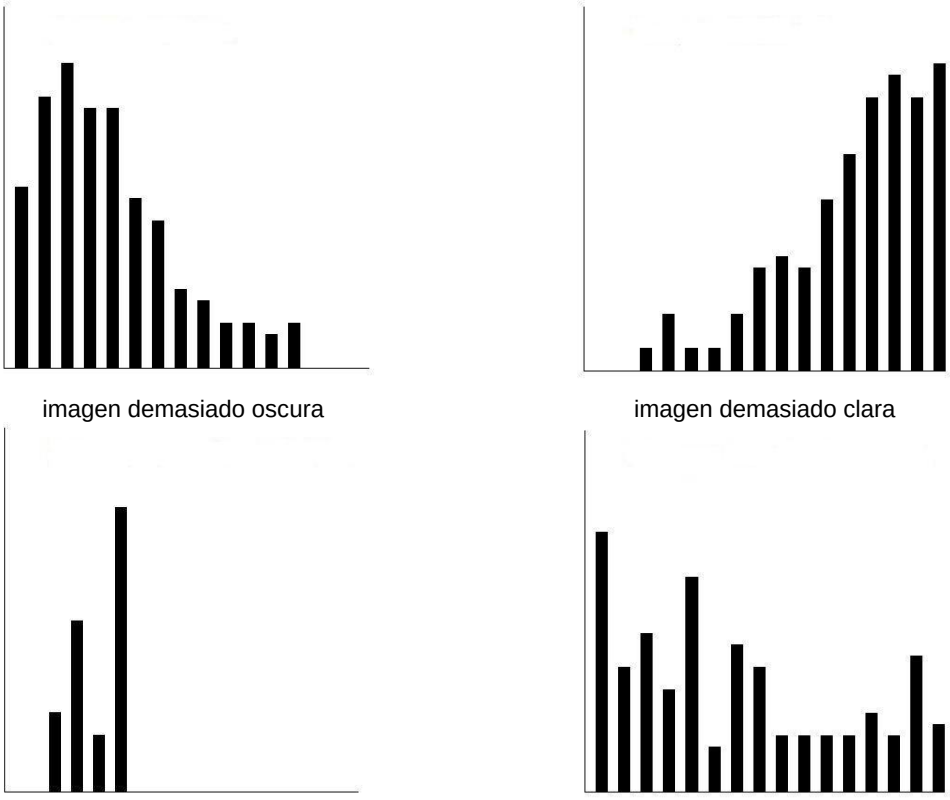


Figura 25. Diferentes histogramas según la claridad y contraste de la imagen

La *ecualización del histograma* de una imagen es una transformación que pretende obtener una distribución uniforme, es decir, que exista la misma frecuencia de píxeles para cada nivel de intensidad. Se busca así que la imagen final tenga una apariencia de alto contraste y presente sus niveles de luminosidad con una distribución uniforme. En la figura 26 se puede comprobar cómo se ha mejorado el contraste de la imagen de las ruedas capturada con el sonar de barrido lateral, al realizar la ecualización de su histograma. En este caso, al ser una imagen en escala de grises, la imagen con el histograma ecualizado posee una distribución más uniforme que la imagen original.

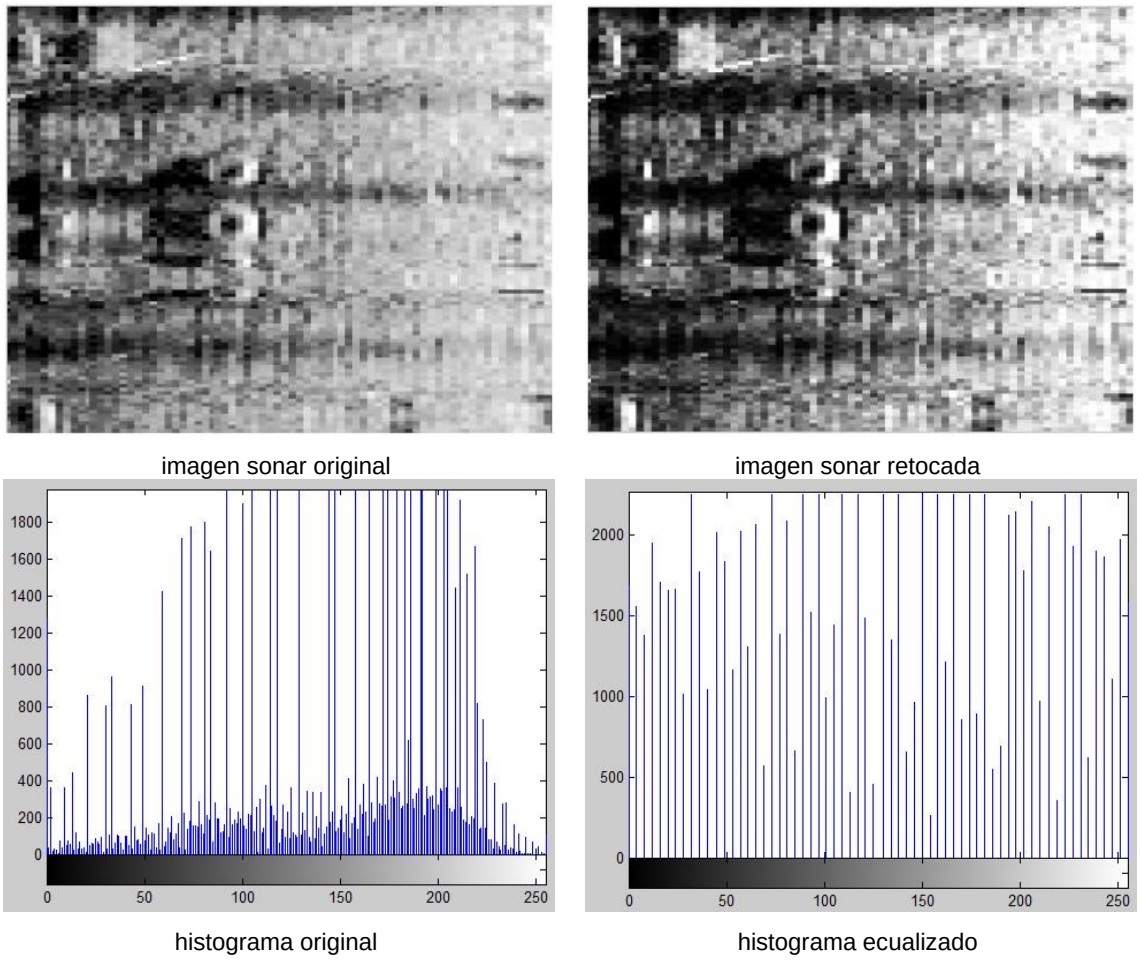


Figura 26. Ecualización del histograma de una imagen sonar para mejorar el contraste

3.3.3 Filtrado espacial

El filtrado espacial es una operación local puesto que modifica el valor de intensidad que tiene cada píxel de la imagen en función de los valores de los píxeles vecinos. Se trata de transformar los niveles de intensidad originales de forma que se parezcan o diferencien más de los píxeles que los rodean, según se quiera obtener una imagen final con unos detalles atenuados o realzados.

3.3.3.1 Clasificación de los filtros espaciales

En este tipo de filtrado se utilizan filtros espaciales, los cuales se pueden clasificar según su linealidad o según su finalidad. Así, atendiendo a su linealidad los filtros espaciales se pueden dividir en:

- *Filtros lineales*: se basan en el uso de máscaras que recorren toda la imagen. Las operaciones se llevan a cabo sobre los píxeles de la imagen original que coinciden con la máscara. Los resultados son unas sumas de convolución entre los píxeles originales y los coeficientes de las máscaras, que se denominan *máscaras de convolución*. Al igual que en los filtros frecuenciales, los filtros lineales se pueden catalogar según la frecuencia que dejen pasar.
    - *Filtros paso alto*: atenúan o eliminan las componentes de baja frecuencia, realzando zonas de altas frecuencias como son los bordes.
    - *Filtros paso bajo*: atenúan o eliminan las componentes de alta frecuencia, suavizando la imagen.
    - *Filtros paso banda*: atenúan o eliminan zonas de frecuencias intermedias.
- *Filtros no lineales*: operan directamente sobre los valores de los píxeles vecinos del píxel central de la máscara. Un ejemplo típico es el *filtro mediana*, que tiene como resultado el difuminado de la imagen y la eliminación de gran cantidad de componentes de ruido.

Si se realiza la clasificación en base a su finalidad, los filtros espaciales se dividen en:

- *Filtros de realce*: permiten eliminar zonas borrosas y definir zonas de alta frecuencia, es decir, los bordes. A su vez engloba dos tipos de filtros:
    - *Filtros de suavizado*: sirven para difuminar la imagen.
    - *Filtros diferenciales*: sustraen de la imagen original una copia desplazada según sea la máscara (horizontal, vertical o diagonal). Se usan para la detección de bordes.

Los filtros de realce también se pueden clasificar según su linealidad:

- *Filtros de realce lineales*: utilizan máscaras de convolución, siendo la salida del filtro el resultado de una operación directa entre los píxeles que haya en ese momento en la máscara del filtro. El más habitual es el *filtro paso alto*, que realza los bordes pero como desventaja aumenta el ruido.
- *Filtros de realce no lineales*: también utilizan máscaras de convolución pero ahora la salida del filtro es el resultado de una operación no lineal entre los píxeles que están alrededor del píxel central de la máscara. Suprimen gran cantidad de ruido a la vez que realzan los bordes. Pertenecen a este grupo los

filtros LUM, CS, WMMR-MED y Volterra que serán explicados en el siguiente capítulo. Todos ellos son conocidos como filtros de rango.

Las características de los filtros de realce lineales no permiten obtener los resultados deseados en el procesamiento de imágenes adquiridas de forma remota, como es el caso de las imágenes capturadas con el sonar de barrido lateral. Debido a la gran cantidad de ruido existente en ellas, la aplicación de estos filtros hace que el nivel de ruido todavía aumente más, aunque realcen los bordes.

Los filtros de realce no lineales son adecuados si la imagen a procesar es borrosa y tiene bordes poco marcados ya que este tipo de filtros consiguen eliminar o suavizar el nivel de ruido a la vez que realzan y delimitan los bordes. No resaltan falsos bordes creados a partir de ruido, sino que acotan las zonas que tienen un mismo gradiente de forma suave y uniforme. Para ello, eliminan el ruido impulsivo que se encuentra en determinados píxeles que tienen valores extremos.

3.3.3.2 Convolución en el filtrado espacial

En el filtrado espacial, la convolución se realiza trasladando una matriz bidimensional, también conocida como *máscara, ventana, kernel o núcleo*. Esta matriz contiene los pesos o ponderaciones que se realizan sobre cada píxel de la imagen. De esta forma, se evalúa el píxel central de la máscara de acuerdo con los valores de los píxeles vecinos y sus valores ponderados. De acuerdo con este mecanismo, cuando un nuevo valor es calculado, la máscara se desplaza sobre el siguiente píxel, volviéndose a repetir la misma operación. Al proceso de evaluar la vecindad ponderada del píxel se le denomina *convolución bidimensional* y a la matriz del filtro se le conoce como *máscara de convolución*.

En general, la convolución de una imagen *f* de MxN con una máscara cuadrada *ω* de mxn, viene dada por la siguiente expresión:

$$g(x,y) = \sum_{i=-a}^{a} \sum_{j=-b}^{b} f(x+i, y+j)\ w(i,j) \tag{3.10}$$

donde:

$$a = \frac{m-1}{2};\ b = \frac{n-1}{2} \tag{3.11}$$

Se ha escogido la ecuación de la convolución de una imagen con una máscara cuadrada porque es el tipo de máscara más utilizada. Si se particulariza para una máscara cuadrada 3x3, se obtiene el esquema que se muestra en la figura 27.

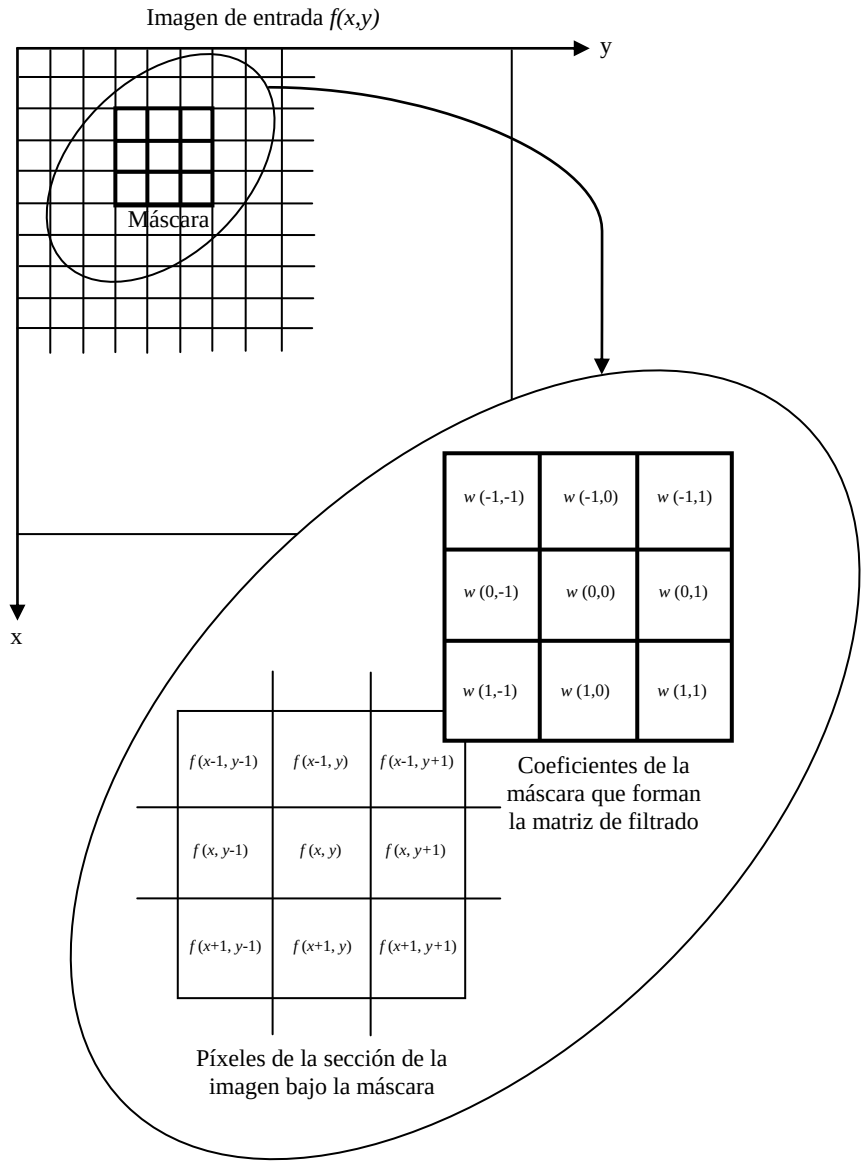


Figura 27. Funcionamiento del filtrado espacial utilizando una máscara cuadrada 3x3

3.3.3.3 Tipos de máscaras

Las *máscaras* también llamadas *kernels (núcleos) o ventanas* pueden ser de distintas formas y tamaños. Dependiendo de qué tipo de máscara se utilice el resultado del procesado varía significativamente. Debido a la naturaleza del filtrado en

el dominio del espacio, las máscaras tienen que ser de tamaño impar para que se tenga el mismo número de muestras a cada lado del píxel central. En esta tesis se ha utilizado como tamaño de las máscaras los impares desde 3 hasta 15. La clasificación de las máscaras se realiza según sea la forma de la máscara:

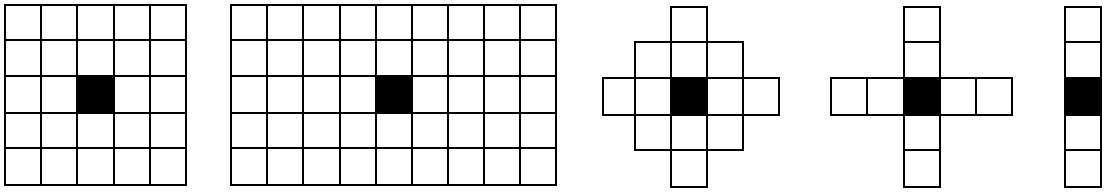


Figura 28. Tipos de máscaras según su forma: (a) Cuadrada (b) Rectangular (c) Circular (d) En cruz (e) Unidimensional

- *Cuadrada*: se caracteriza por ser la que tiene mayor capacidad de realce ya que también es la que mayor número de valores posee para un tamaño absoluto. Esto implica que su gasto computacional también sea el mayor, al ser mayor la superficie que se evalúa. Es el tipo de máscara que generalmente más se utiliza en la implementación de filtros.
- *Rectangular*: se caracteriza por tener una capacidad de realce diferente según sea la dirección (horizontal o vertical). El mayor realce se consigue en la dirección en la que la máscara tiene mayor longitud puesto que en esa dirección hay mayor cantidad de muestras.
- *Circular*: tiene un comportamiento similar a la máscara cuadrada pero al poseer menor número de muestras, también tiene menor capacidad de realce. Su ventaja es generar un menor gasto computacional al ser menor la superficie evaluada.
- *En cruz*: se caracteriza por realzar los contornos de igual forma tanto en dirección horizontal como en dirección vertical. Da un resultado muy parecido al que se consigue con una ventana cuadrada pero con un menor realce ya la cantidad de muestras es mucho menor. En condiciones de poco ruido es una mejor opción que la ventana cuadrada ya que el gato computacional será mucho menor mientras que el realce será similar.
- *Unidimensional*: produce realces en la única dirección de la máscara, es decir, en dirección horizontal o en dirección vertical. Es la que menor gasto

computacional genera ya que es la que posee menor número de muestras pero esto conlleva que sea la que menor realce tenga.

Como ya se ha comentado anteriormente, las imágenes obtenidas por un sonar de barrido lateral contienen una gran cantidad de ruido. Por lo tanto, es muy recomendable el uso de una máscara cuadrada para el filtro porque al contener un mayor número de muestras que el resto se atenúan de forma más eficaz las componentes de ruido. Además, la máscara cuadrada posee la misma capacidad de realce en ambas direcciones, de modo que se mantiene una proporcionalidad en el realce. La desventaja que tiene la máscara cuadrada es su alto coste computacional, que además aumenta a medida que se incrementa el número de muestras.

Capítulo 4.
Filtros para eliminar ruido y realzar bordes

En este capítulo se explicarán los filtros y algoritmos que se han utilizado a la hora de programar el software de filtrado en Matlab, para realizar el preprocesado de las imágenes obtenidas mediante el sonar de barrido lateral. Siempre con el objetivo de lograr una mejora en la imagen que posteriormente podrá ser visualizada por el ojo humano o ser procesada por computador.

4.1 Terminología utilizada en los filtros

A continuación, se presentan una serie de términos que resultan fundamentales para comprender los filtros implementados:

- *Secuencia*: serie de datos que provienen de una imagen.
- *Segmento*: número finito de valores consecutivos en una secuencia.
    - *Segmento constante*: segmento de N+1 o más valores iguales.
- *Impulso*: segmento no constante entre dos segmentos de la misma amplitud.
- *Borde rampa*: segmento creciente o decreciente entre dos segmentos de la misma amplitud.

- *Borde perfecto*: formado por dos segmentos constantes y distintos, separados como máximo por una muestra de valor contenido entre los dos segmentos.
- *Borde imperfecto*: formado por dos segmentos constantes y distintos, separados por un segmento de longitud menor a N+1 y limitada por los valores de las regiones constantes.
- *Raíz de un filtro*: es una secuencia que no cambia con el filtrado. Por ejemplo: si el filtro pretende llegar al borde perfecto, una vez que lo consiga, éste será la raíz del filtro.
- *Filtro OS (Order Statistic)*: filtro cuya salida es la suma algebraica ordenada por pesos de los valores de la máscara.
- *Filtro GOS (Generalized Order Statistic)*: filtro cuya salida es la suma ordenada por pesos de valores ordenados, que son seleccionados en un proceso de ordenación diferente a una simple ordenación algebraica.
- *Gradiente*: vector de dirección perpendicular a las curvas de nivel de la superficie y sentido el de crecimiento de la función.

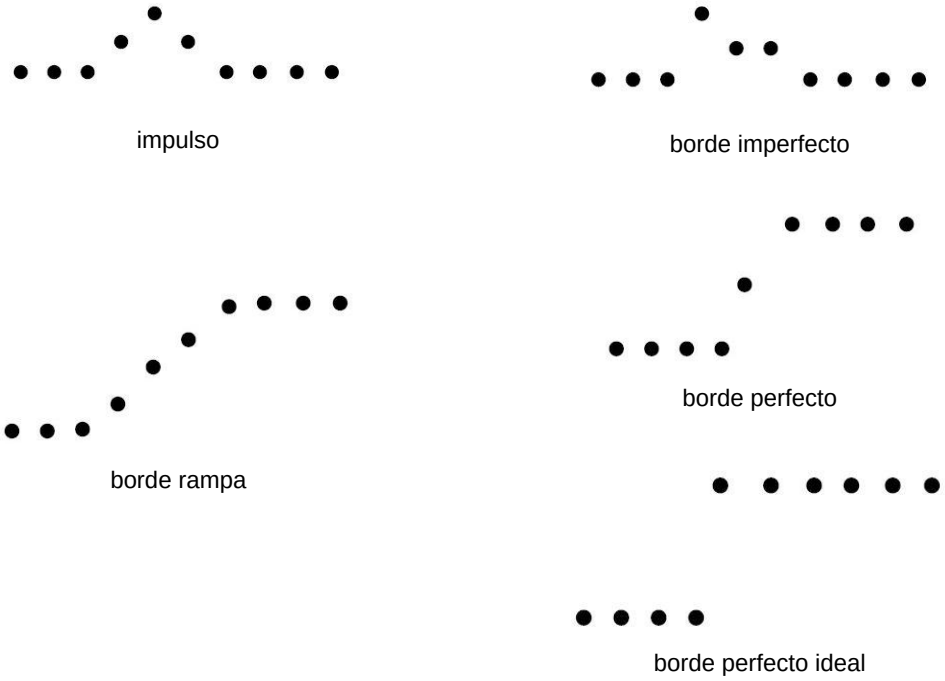


Figura 29. Distintos ejemplos de bordes

4.2 Implementación de las máscaras

Para que los filtros puedan evaluar del mismo modo las muestras obtenidas de cada máscara, independientemente de su forma, se utiliza una función que extrae los datos de la máscara y los introduce en un vector fila. Como se puede observar en la figura 30, el vector fila se va formando con las muestras que se van extrayendo de la máscara por columnas, es decir, primero se obtienen las muestras de la 1ª columna, luego las de la 2ª columna y así sucesivamente. Este proceso es igual para todas las máscaras excepto para la máscara en cruz, en la que primero se obtienen las muestras de la columna central y a partir de ahí se sigue el orden normal.

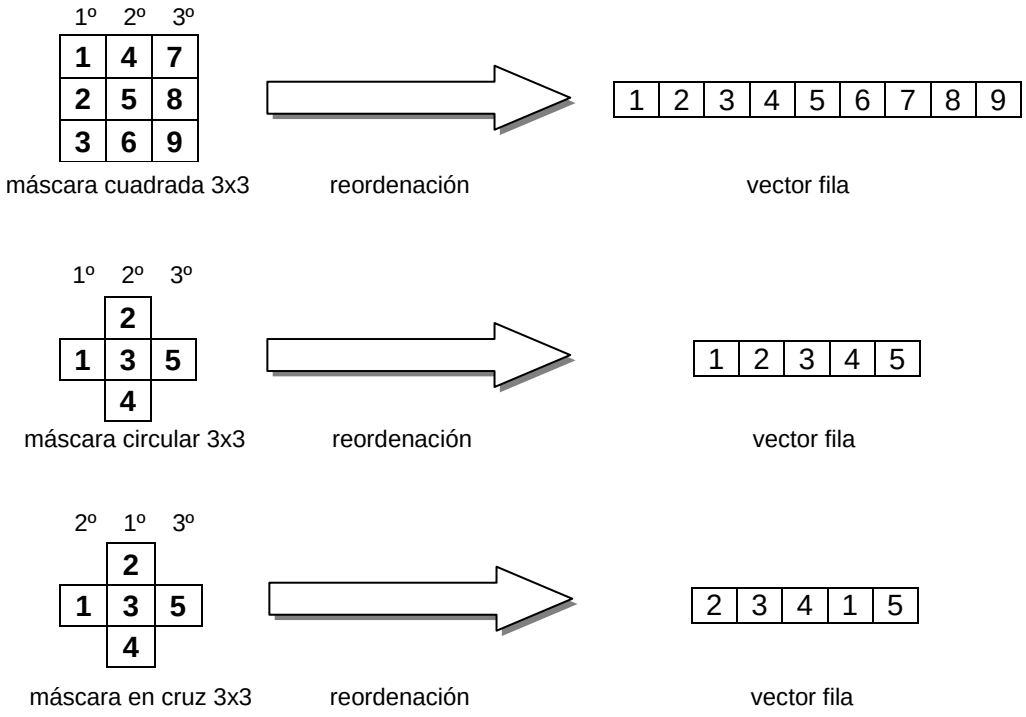


Figura 30. Ejemplos del proceso de implementación de las máscaras

4.3 Filtrado según el tipo de imagen

En la implementación de los filtros que se van a ver en los siguientes apartados, se ha tenido en cuenta que el software soporte tanto imágenes en color como en escala de grises.

Normalmente el filtrado de imágenes de sonar se realiza con imágenes en escala de grises. Aunque la información que proporciona el color en imágenes

obtenidas por un sonar de barrido lateral es muy pobre y normalmente es añadido por el programa informático que se utiliza como receptor de las imágenes, está demostrado que las imágenes en color hacen que el ojo humano visualice mejor ciertos detalles. Además, el coste computacional apenas se incrementa filtrando imágenes en color que filtrando imágenes en escala de grises. Para ello, se han utilizado dos tipos de modelos de color: el RGB y el YC_bC_r. Se ha elegido para realizar el filtrado por defecto el YC_bC_r porque es el que menos gasto computacional tiene. Ambos espacios de color han sido explicados en el capítulo 3 de esta tesis.

A continuación se va explicar en qué consiste cada filtrado según sea el tipo de imagen:

- *Filtrado de imagen en escala de grises*: se aplica el filtro escogido sobre la imagen en escala de grises, que está formada únicamente por una capa.
- *Filtrado de imagen en color RGB*: consiste en aplicar el filtro elegido sobre cada una de las tres capas que componen la imagen, es decir, la capa roja, la verde y la azul. La imagen de salida está compuesta por la suma del filtrado de estas tres capas. Con este procesado no se pierde información pero el gasto computacional se triplica si lo comparamos con el filtrado de una imagen en escala de grises.
- *Filtrado de imagen en color YC_bC_r*: lo primero que se hace es convertir la imagen RGB de partida a este modelo de color. La capa Y es una copia de la imagen original en escala de grises y las otras dos capas restantes sólo contienen información acerca del color. De esta manera, se aplica el filtro elegido solamente sobre la capa Y, y después se recompone la imagen añadiendo la información del color que poseen las capas C_b y C_r. Por último, se convierte la imagen a RGB. Con este tipo de filtrado el gasto computacional es similar al que se tiene en el filtrado de una imagen en escala de grises, pero obteniendo una imagen de salida en color.

En el diagrama de flujo de la figura 31, se puede observar cómo funciona el software desarrollado dependiendo del tipo de imagen que se vaya a filtrar, es decir, si la imagen de entrada es en escala de grises o en color.

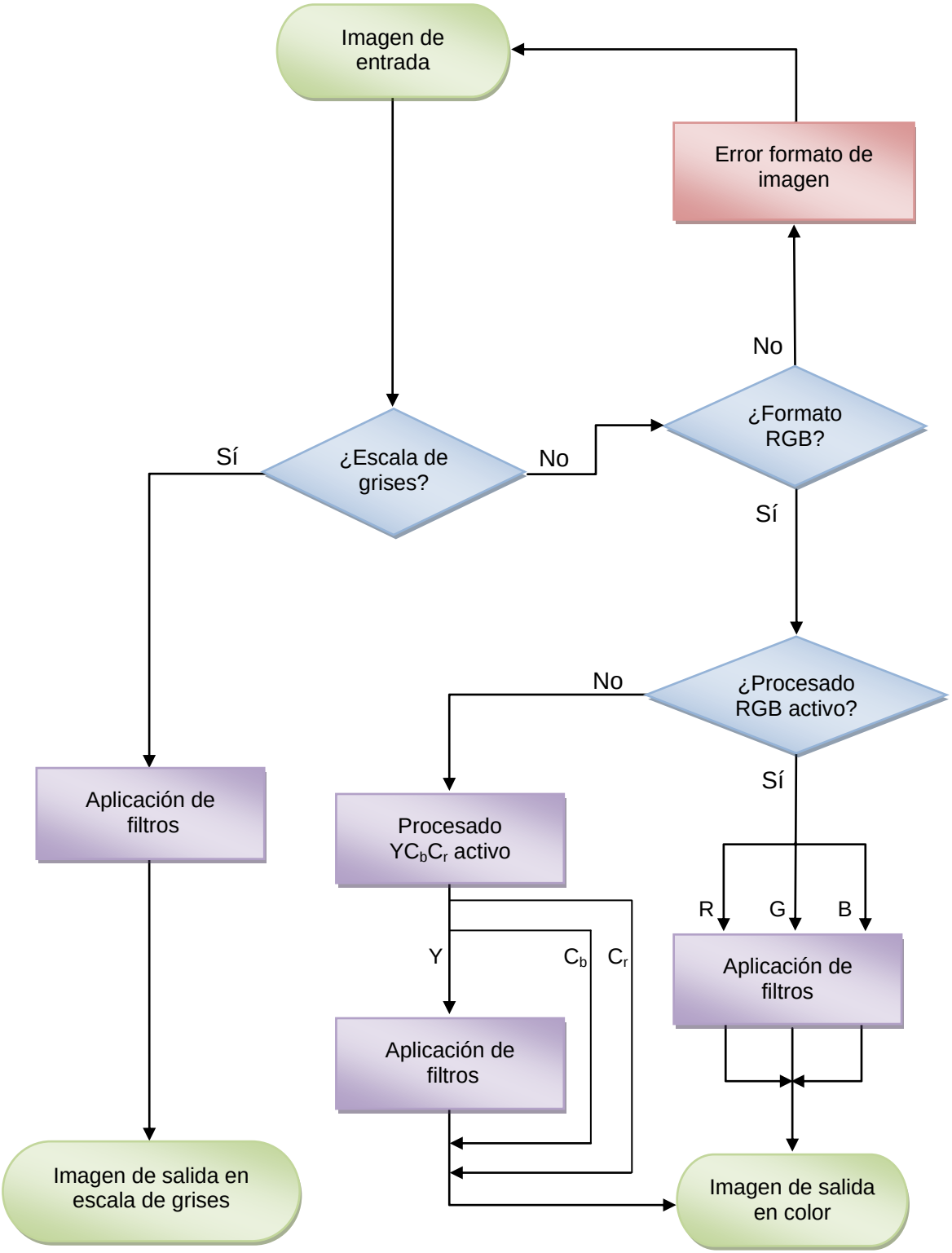


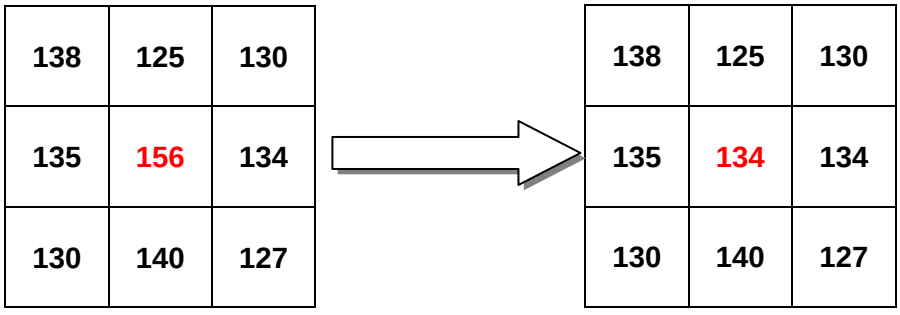
Figura 31. Diagrama de flujo del funcionamiento del programa de procesado de imágenes

4.4 Filtro mediana

El filtro mediana no ha sido implementado directamente en el programa desarrollado, pero se va a explicar porque diversas modificaciones de él han dado lugar a la familia de filtros OS (*Order Statistic*) [5], entre los que se encuentran los filtros LUM, CS y WMMR que se verán en los siguientes apartados.

Es un filtro espacial no lineal de gran utilidad a la hora de eliminar ruido en una imagen. Además, preserva los rasgos y bordes de la imagen original. Consta de una máscara de convolución de tamaño impar en la que la intensidad del píxel central es reemplazada por la *mediana* de los niveles de intensidad de los píxeles vecinos. La mediana *m* de un conjunto de valores es aquél tal que la mitad de los valores del conjunto son menores que *m* y la otra mitad son mayores que *m*.

Por lo tanto, con la aplicación de este filtro se tiende a eliminar los valores extremos de intensidad en comparación con los niveles de intensidad de los vecinos. De este modo, se suprimen los picos de intensidad que aparecen en áreas aisladas. Un ejemplo de este filtro se puede ver en la figura 32, donde se ha tomado una máscara cuadrada 3x3 en la que aplicar el filtro mediana. Nótese que para este caso concreto la mediana es el valor central de la serie de muestras ordenadas: 134, mientras que la media de los valores es 135. Hay que tener especial cuidado de no confundir el término mediana con la media, ya que son conceptos distintos. Cuando el número de elementos de la máscara es impar, la mediana corresponde al valor central de la serie ordenada numéricamente. Por otro lado, cuando el número de elementos de la máscara es par, la mediana se calcula como la media de los dos valores centrales de la series ordenada de menor a mayor.



Serie ordenada de los valores de las muestras: 125, 127, 130, 130, **134**, 135, 138, 140, 156

Figura 32. Ejemplo de aplicación del filtro mediana

4.5 Filtro LUM

El filtro LUM (Lower-Upper-Middle) o *filtro Inferior-Superior-Intermedio* es un filtro no lineal que permite conseguir tanto un suavizado de la imagen mediante la eliminación de ruido, como una mejora en los bordes resaltando así los detalles. Todo esto se logra variando los dos parámetros del filtro, ya que influyen de forma directa en el rango de salida. El filtro LUM, como ya se ha comentado en el apartado anterior, es una modificación del filtro mediana. En la figura 33, se puede observar el diagrama de bloques del filtro.

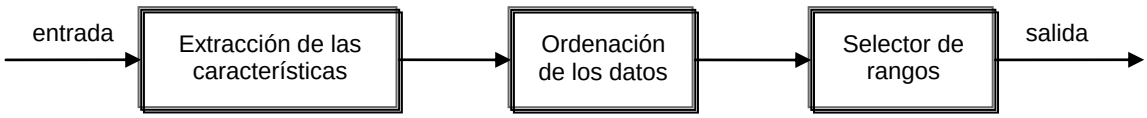


Figura 33. Diagrama de bloques del filtro LUM

El filtro LUM está formado por tres bloques diferentes. El primer bloque extrae información de las muestras: X = ($X_{(1)}$, $X_{(2)}$,...,$X_{(N)}$), es decir, crea un vector sin ordenar con las muestras extraídas de la máscara. El segundo bloque ordena numéricamente el vector de datos que se ha obtenido en el bloque anterior. Por último, el tercer bloque es el selector de rangos, su función es la de escoger el rango más apropiado de la muestras que forman el vector ordenado para ser la salida del filtro.

La ecuación de la salida del filtro LUM viene dada por:

$$Y_k = \begin{cases} X^L \ si\ X_k \leq (X^L + X^U)/2 \\ X^U \ resto \end{cases} \quad (4.1)$$

donde Y_k es la salida del filtro, X_k es el valor de la muestra central del vector ordenado y por último, X^L y X^U vienen definidos por:

$$X^L = med\{X_{(K)},\ X_k,\ X_{(L)}\} \quad (4.2)$$

$$X^U = med\{\ X_{(N-L+1)},\ X_k, X_{(N-K+1)}\} \quad (4.3)$$

K y L son los parámetros del filtro tal que 1 ≤ K ≤ L ≤ (N+1)/2 siendo N el tamaño del vector de las muestras.

Las características del filtro LUM se pueden cambiar variando los valores de los parámetros K y L [17]. En la figura 34, se ha representado el proceso de aplicar el filtro LUM para tres casos. Las muestras están ordenadas de menor a mayor yendo de izquierda a derecha. Las muestras sombreadas representan el rango de valores que la muestra central puede tener sin tener que ser modificada. Las muestras $X_{(K)}$ y $X_{(N-K+1)}$ forman el límite inferior y superior respectivamente de la salida del filtro LUM.

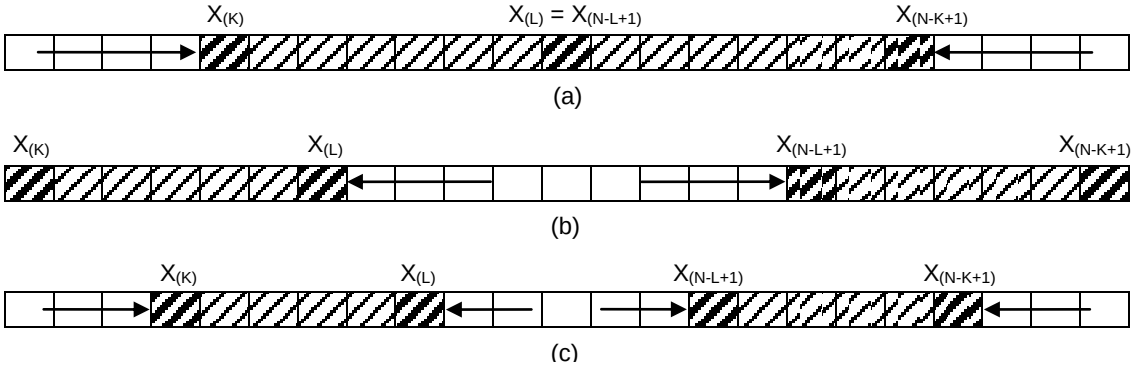


Figura 34. Filtro LUM operando como: (a) filtro LUM de suavizado (b) filtro LUM de realce (c) filtro LUM de suavizado y de realce

- Figura 34(a): se mantiene el valor de L máximo, es decir, L = (N+1)/2 y K se va variando dentro de su rango $1 < K \leq L$. Como resultado, se obtiene un filtro LUM de suavizado. A medida que aumenta el valor de K se obtiene un mayor suavizado en la imagen.
    - En el caso particular de K = L, se obtiene como resultado el filtro mediana, en el que el suavizado es máximo.
- Figura 34(b): se mantiene el valor de K máximo, es decir, K = 1 y L se va variando dentro de su rango $K < L \leq (N+1)/2$. En este caso, se obtiene como resultado un filtro LUM de realce. A medida que disminuye el valor de L, $X_{(L)}$ se va desplazando hacia los valores del extremo inferior de las muestras mientras que $X_{(N-L+1)}$ se va desplazando hacia los valores del extremo superior de las muestras, obteniéndose así un mayor realce en los bordes de la imagen.
- Figura 34(c): se van variando los valores de K y L dentro del rango $1 < K \leq L < (N+1)/2$. El resultado es un filtro de suavizado y de realce al mismo tiempo. El parámetro K controla el grado de suavizado, se aumenta su valor para incrementar el suavizado y se disminuye su valor para disminuir el

suavizado. Por otra parte, el parámetro L controla el grado de realce, se disminuye su valor para incrementar el realce y se aumenta su valor para reducir el realce.

4.5.1 Algoritmo implementado

A continuación se muestra el código del algoritmo para el filtro LUM, que se ha implementado como una función en el software que se ha desarrollado bajo el entorno de Matlab:

```
function [im_new,error]=lum(im,nker1,nker2,l,k,tipo_ven,color)

%im: imagen de entrada original
%im_new: imagen de salida tras el filtrado
%nker1: tamaño horizontal de la máscara
%nker2: tamaño vertical de la máscara
%l,k: parámetros del filtro LUM
%tipo_ven: tipo de máscara utilizada
%color=0 procesado YCbCr
%color=1 procesado RGB
%error: función de error

error=0;
[h w c]=size(im);
if ((color == 1) && (c==3))
    cont=3;
else
    cont=1;
    if c==3
        imcbcr=rgb2ycbcr(im);
        im=imcbcr(:,:,1);
    end
end

temp = ((nker1-1)/2)+1; %sólo para la barra de progreso
progressbar(0);
tic;

for i=1:cont;
    im_new(:,:,i)=uint8(zeros(h,w));
    im_new(1:(nker1-1)/2,:,i)=im(1:(nker1-1)/2,:,i);
    im_new(h-((nker1-1)/2-1):h,:,i)=im(h-((nker1-1)/2-1):h,:,i);
    im_new(:,1:(nker2-1)/2,i)=im(:,1:(nker2-1)/2,i);
    im_new(:,w-((nker2-1)/2-1):w,i)=im(:,w-((nker2-1)/2-1):w,i);
    for y=((nker1-1)/2)+1:h-((nker1-1)/2); %mirar límites
        for z=((nker2-1)/2)+1:w-((nker2-1)/2);
            [ker,error]=ventana(nker1,nker2,tipo_ven,im,y,z,i);
            if error
                progressbar(1)
                error=1;
                return;
            end
            N=(length(ker));
            med=mean(ker); %media de las muestras de la máscara
```

```
            ker=sort(ker); %se ordenan las muestras de menor a mayor
            p=(ker((N+1)/2));
            xl=(ker(k)+p+ker(l))/3;
            xu=(ker(N-k+1)+p+ker(N-l+1))/3;
            limite=(xl+xu)/2;
            if p <= limite
                pixelout=xl;
            else
                pixelout=xu;
            end
            im_new(y,z,i)=pixelout;
        end
        stopbar=progressbar( temp /(((h-((nker1-1)/2)) )*cont) );
        temp=temp+1;
        if stopbar
            return
        end
    end
end

if (color == 0.0)
    if c==3
        im_new(:,:,2)=imcbcr(:,:,2);
        im_new(:,:,3)=imcbcr(:,:,3);
        im_new=ycbcr2rgb(im_new);

    end
end

progressbar(1);
toc
end
```

4.6 Filtro CS

El filtro CS (Comparison and Selection) o *filtro de Comparación y Selección*, es un filtro no lineal que se caracteriza por mejorar los bordes en rampa a la vez que suprime componentes de ruido impulsivo y de ruido blanco gaussiano. Además, se puede utilizar bajo circunstancias de falta de conocimiento del grado de borrosidad de la imagen. Por todas estas características resulta un filtro muy adecuado para realzar imágenes de sonar, ya que éstas suelen contener gran cantidad de ruido, produciendo distorsiones en los contornos de las formas y creando un efecto de imagen borrosa.

Está basado en el uso combinado de los filtros media y mediana. Si el valor de la media de las muestras de la máscara es mayor o igual que el valor de la mediana, la salida del filtro será un valor menor que la mediana (valores mínimos). Para el resto de casos, la salida del filtro será un valor mayor que la mediana (valores máximos). Se

utilizan máscaras impares con un tamaño total de muestras 2N+1, siendo N un número positivo. El algoritmo para calcular la salida del filtro viene dado por [20]:

$$Y_k = \begin{cases} X_k^{(N+1-J)} & si\ \mu_k \geq M_k \\ X_k^{(N+1+J)} & resto \end{cases} \tag{4.4}$$

donde Y_k es la salida del filtro, μ_k es la media de las muestras, M_k es la mediana de las muestras y J es el parámetro del filtro tal que 1 ≤ J ≤ N.

En la figura 35, se puede ver un ejemplo de cómo funciona el filtro CS a medida que se va variando el parámetro J. Las muestras están ordenadas de menor a mayor, encontrándose el valor mínimo de las muestras en el extremo izquierdo y el valor máximo de las muestras en el extremo derecho. En el centro se encuentra la mediana de las muestras.

- Cuando J = 0, el filtro CS se convierte en el filtro mediana, por este motivo existe la restricción de que J ≥ 1.
- A medida que aumenta el valor del parámetro J, la salida del filtro se va desplazando hacia los valores extremos de las muestras.
- Cuando J = N, el filtro CS selecciona el valor máximo o el valor mínimo de las muestras, dependiendo del valor que tenga la mediana comparado con el valor de la media. Si el valor de la media es mayor o igual que el de la mediana, la salida del filtro se encontrará en el valor extremo inferior, es decir, en el valor mínimo de las muestras. En cambio, si el valor de la media es menor que el de la mediana, la salida del filtro estará localizada en el valor extremo superior, es decir, en el valor máximo de las muestras.

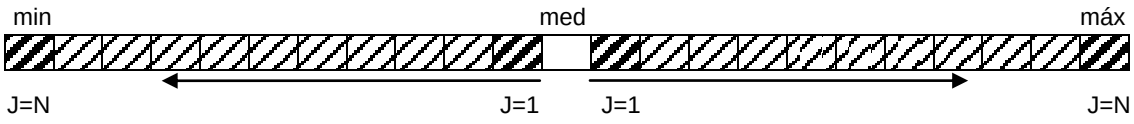


Figura 35. Funcionamiento del filtro CS

Normalmente el ruido, sobre todo el impulsivo, se suele localizar en los valores extremos de las muestras. Por lo tanto, cuanto menor sea el valor de J, más ruido se elimina, ya que la salida del filtro estará más lejos de los valores extremos de las muestras. Por otra parte, cuanto mayor es el valor de J, mayor es el realce que se

obtiene en los bordes pero se suprime menos ruido. De esta forma, se debe buscar un equilibrio a la hora de elegir el valor del parámetro J para eliminar la mayor cantidad de ruido a la vez que se mejoran los bordes.

También la elección del tamaño de la máscara es muy importante a la hora de que el resultado de la aplicación del filtro sea óptimo. A mayor tamaño de la máscara, mayor será la definición de los bordes, pero también mayor será la pérdida de los pequeños detalles de la imagen. Esto es debido a que al tratarse de un número mayor de muestras, los valores que caracterizan a zonas pequeñas son eliminados, resultando un suavizado de la imagen. El tamaño de la máscara también influye en el tiempo de filtrado, ya que cuanto más grande sea la máscara mayor será el coste computacional con lo que se ralentizará el proceso de filtrado.

Para solucionar todos estos inconvenientes, es aconsejable utilizar una máscara de tamaño intermedio, con un valor pequeño del parámetro J y repetir varias veces el filtrado CS. Pero se debe tener en cuenta que el uso repetido del filtro CS puede llegar a deformar los bordes, al convertirlos en bordes perfectos ideales.

4.6.1 Algoritmo implementado

A continuación se muestra el código del algoritmo para el filtro CS, que se ha implementado como una función en el software que se ha desarrollado bajo el entorno de Matlab:

```
function [im_new,error]=cs(im,nker1,nker2, param, tipo_ven, color)

%im: imagen de entrada original
%im_new: imagen de salida tras el filtrado
%nker1: tamaño horizontal de la máscara
%nker2: tamaño vertical de la máscara
%param: J parámetro del filtro CS tal que 1<=J<=N
%tipo_ven: tipo de máscara utilizada
%color=0 procesado YCbCr
%color=1 procesado RGB
%error: función de error

error=0;
[h w c]=size(im);
if ((color == 1) && (c==3))
    cont=3;
else
    cont=1;
    if c==3
        imcbcr=rgb2ycbcr(im);
        im=imcbcr(:,:,1);
    end
end
```

```
temp = ((nker1-1)/2)+1; %sólo para la barra de progreso
progressbar(0);
tic;

for i=1:cont;
    im_new(:,:,i)=uint8(zeros(h,w));
    im_new(1:(nker1-1)/2,:,i)=im(1:(nker1-1)/2,:,i);
    im_new(h-((nker1-1)/2-1):h,:,i)=im(h-((nker1-1)/2-1):h,:,i);
    im_new(:,1:(nker2-1)/2,i)=im(:,1:(nker2-1)/2,i);
    im_new(:,w-((nker2-1)/2-1):w,i)=im(:,w-((nker2-1)/2-1):w,i);

    for y=((nker1-1)/2)+1:h-((nker1-1)/2); %mirar límites
        for z=((nker2-1)/2)+1:w-((nker2-1)/2);
            [ker,error]=ventana(nker1,nker2,tipo_ven,im,y,z,i);
            if error
              progressbar(1)
              error=1;
              return;
            end
            N=(length(ker)-1)/2;
            med=mean(ker); %media de las muestras de la máscara
            ker=sort(ker); % se ordenan las muestras de menor a mayor
            min=ker(N+1-param);
            max=ker(N+1+param);
            %(lenght(ker)+1)/2 es la mediana
            if ker((length(ker)+1)/2)<=med
                pixelout=min;
            else
                pixelout=max;
            end
            im_new(y,z,i)=pixelout;
        end
        stopbar=progressbar( temp /   (((h- ((nker1-1)/2)) )*cont) );
        temp=temp+1;
        if stopbar
          return
        end
    end
end

if (color == 0.0)
    if c==3
        im_new(:,:,2)=imcbcr(:,:,2);
        im_new(:,:,3)=imcbcr(:,:,3);
        im_new=ycbcr2rgb(im_new);

    end
end

progressbar(1);
toc
end
```

4.7 Filtro WMMR-MED

El filtro WMMR-MED (Weighted Majority of m values with Minimum Range-Median), o *filtro de mayor peso a los m valores de menor rango-mediana*, es un filtro no lineal que se caracteriza por eliminar ruido impulsivo y ruido blanco gaussiano al mismo tiempo que transforma bordes distorsionados en bordes perfectos y realza los bordes imperfectos. Debido a estas características resulta un filtro muy útil en el procesado de imágenes con mucho ruido.

Este filtro pertenece a la familia de filtros *GOS (Generalized Order Statistic)*, en los que los valores se ordenan por un procedimiento diferente a una simple ordenación algebraica [21]. Lo primero que hace un filtro WMMR es clasificar los valores de la máscara de filtrado en orden numérico. Después toma un conjunto de *m* muestras de rango mínimo, es decir, que tengan valores muy próximos. La salida del filtro WMMR viene dada por la expresión [16]:

$$Y_k = \sum_{j=i}^{\frac{(N-1)}{2}+i} w_j \, x_{(j)} \tag{4.5}$$

donde w_j es:

$$w_j = \begin{cases} 1 & si \;\; j = \dfrac{N-1}{4} + i \\ 0 & resto\ de\ casos \end{cases} \tag{4.6}$$

y el rango mínimo viene dado por:

$$rango\ mínimo = min\left\{x_{((N-1)/2+i)} - x_i\right\} \tag{4.7}$$

El filtro WMMR-MED es un filtro WMMR que reemplaza el píxel central de la máscara de filtrado de tamaño 2N+1 por la mediana de N+1 valores de rango mínimo. El valor de salida del filtro es la suma ponderada del valor de entrada y del valor de la mediana de las muestras de menor rango o rango mínimo.

De esta manera, la salida del filtro WMMR-MED viene dada por:

$$Y_k = p \; * x_i + (1-p) * M_K \tag{4.8}$$

donde p es el parámetro de ponderación, x_i es el valor de la muestra de entrada y M_k es la mediana de las muestras de menor rango.

La ponderación se controla a través del parámetro p, cuyo valor varía entre 0 y 1 de la siguiente forma: $0 \leq p \leq 1$.

- Cuando el parámetro p = 0, la salida del filtro es M_k, la mediana de las muestras de rango mínimo.
- Cuando el parámetro p = 1, la salida del filtro es x_i, el valor de entrada.

El filtro WMMR-MED puede transformar bordes imperfectos y con ruido en bordes perfectos ideales en un solo filtrado dependiendo del tamaño de la máscara y del valor del parámetro p.

El tamaño de la máscara del filtro va a determinar el realce de los bordes así como el nivel de pérdida de detalles de la imagen. Cuanto más grande es el tamaño de la máscara, mayor es la definición de los bordes pero también es mayor el nivel de pérdidas de detalles debido al incremento que se produce en el número de muestras de la máscara. La cantidad de muestras influye tanto en el número de aplicaciones como en el tiempo de ejecución del filtro. Así, al aumentar el número de muestras el tiempo de filtrado es mayor, mientras que el número de veces que hay que aplicar el filtro para obtener un resultado similar es menor.

El parámetro p va a establecer el grado de realce y definición que se pretende lograr en cada filtrado WMMR-MED. Como ya se ha visto, el valor del parámetro p oscila entre 0 y 1. De esta forma, cuanto más pequeño es su valor, es decir, está más próximo a 0, elimina más ruido y realza mejor los bordes aunque se produzca una pérdida de definición. Seleccionando un valor pequeño de p, se pueden realzar los bordes de forma efectiva con un solo filtrado.

4.7.1 Algoritmo implementado

A continuación se muestra el código del algoritmo para el filtro WMMR-MED, que se ha implementado como una función en el software que se ha desarrollado bajo el entorno de Matlab:

```
function [im_new,error]=wmmr(im, nker1, nker2, param, tipo_ven, color)

%im: imagen de entrada original
```

```
%im_new: imagen de salida tras el filtrado
%nker1: tamaño horizontal de la máscara
%nker2: tamaño vertical de la máscara
%param: p parámetro del filtro WMMR-MED tal que 0<=p<=1
%tipo_ven: tipo de máscara utilizada
%color=0 procesado YCbCr
%color=1 procesado RGB
%error: función de error

error=0;
progressbar (0); %sólo para la barra de progreso
temp = ((nker1-1)/2)+1;
tic;
[h w c]=size(im);
if ((color == 1) && (c==3))
    cont=3;
else
    cont=1;
    if c==3
        imcbcr=rgb2ycbcr(im);
        im=imcbcr(:,:,1);
    end
end

for p=1:cont;
    im_new(:,:,p)=uint8(zeros(h,w));
    im_new(1:(nker1-1)/2,:,p)=im(1:(nker1-1)/2,:,p);
    im_new(h-((nker1-1)/2-1):h,:,p)=im(h-((nker1-1)/2-1):h,:,p);
    im_new(:,1:(nker2-1)/2,p)=im(:,1:(nker2-1)/2,p);
    im_new(:,w-((nker2-1)/2-1):w,p)=im(:,w-((nker2-1)/2-1):w,p);
    for y=((nker1-1)/2)+1:h-((nker1-1)/2);
        for z=((nker2-1)/2)+1:w-((nker2-1)/2);
            [ker,error]=ventana(nker1,nker2,tipo_ven,im,y,z,p);
            if error
                 progressbar(1)
                 error=1;
                 im_new=[];
                 return;
            end
            N=(length(ker)-1)/2;
            ker=sort(ker);

            %cálculo del algoritmo WMMR-MED
            min=0;  %valor de rango mínimo
            JJ=[];  %vector de J's
            kerm=[];%vector que tendrá los N+1 valores de rango mínimo
            %cada fila corresponde a un J si hay más de uno
            pixelout=uint8(0);
            for j= 1:(N+1)
                r=ker(j+N)-ker(j);
                if (j==1 || r<=min)
                    if (r==min && j~=1)
                        JJ=[JJ j];
                    else
                        min=r;
                        JJ=[j];
                    %se resetea J a un valor por si tenía más de uno
                    end
                end
```

```
            end
            %se sacan ahora los N+1 valores de rango mínimo
            if length(JJ)==1
                kerm=ker(JJ:JJ+N);
            else
                for i=1:length(JJ)
                    kerm(i,:)=ker(JJ(i):JJ(i)+N);
                end
            end

            if rem(N,2) %Si N es impar -> N+1= par
              pixelout=round(mean(mean(kerm(:,(N+1)/2:(N+1)/2+1),2)));
              %media de los dos valores centrales de cada fila de Kerm
              %se promedian los dos valores anteriores y se redondea
            else
              med=(N+1+1)/2; %posición de la mediana si N par
                pixelout=round(mean(kerm(:,med)));
            end
            %fin del algoritmo WMMR-MED
            im_new(y,z,p)=param*im(y,z)+(1-param)*pixelout;
        end

        stopbar=(progressbar( temp / (((h- ((nker1-1)/2)) )*cont) ));
        temp=temp+1;
        if stopbar
          return
        end
    end

end

if (color == 0.0)
    if c==3
        im_new(:,:,2)=imcbcr(:,:,2);
        im_new(:,:,3)=imcbcr(:,:,3);
        im_new=ycbcr2rgb(im_new);
    end
end

progressbar(1);
toc
end
```

4.8 Filtro Volterra

El filtro implementado es una clase especial de filtro cuadrático de Volterra para el realce de bordes y la reducción del ruido [40]. El realce se realiza según la media de intensidad que poseen los píxeles locales. De esta forma, se consigue un realce más exhaustivo en las zonas donde los detalles son menos perceptibles por el ojo humano y un realce más reducido en las zonas donde al realzar también se aumenta el ruido, afectando a la calidad de la imagen.

El diagrama de bloques del filtro, ver figura 36, está formado por el filtro cuadrático no lineal que extrae la información de los bordes de la imagen, después se multiplica por el parámetro de ponderación ρ, y finalmente se añade a la imagen original para obtener la salida del filtro.

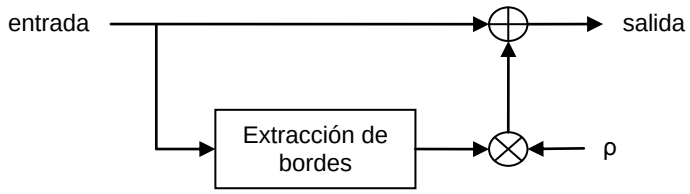


Figura 36. Diagrama de bloques del filtro Volterra

El realce que se obtiene al aplicar el filtro de Volterra es equivalente al producto de la media local de los píxeles y un filtro paso alto lineal. Pero con la diferencia de que el filtro Volterra distingue entre el ruido y la información de la imagen, con lo que evita realzar detalles que están formados por ruido, cosa que sí haría un filtro lineal. Además, el filtro Volterra comparado con un filtro lineal de realce de similares características tiene un menor gasto computacional, lo cual es muy importante a la hora de procesar imágenes captadas con un sonar de barrido lateral.

Para entender las principales características del filtro cuadrático Volterra que se ha implementado se debe explicar la ley de Weber-Fechner. Dicha ley, caracteriza la percepción de los distintos niveles de grises dependiendo de la intensidad de fondo de la imagen. Por lo tanto, la percepción de los detalles de la imagen es proporcional a la media de la intensidad de los píxeles vecinos. La media será mayor cuando mayor sea el brillo o claridad de la imagen. Esto hace que se puedan apreciar los detalles más fácilmente en las regiones oscuras, mientras que en las regiones claras los detalles tienden a enmascararse. Por lo tanto, se deben realzar las zonas oscuras en menor grado que las zonas claras. Así, el filtro Volterra realza las distintas zonas de la imagen en mayor o menor grado basándose en la percepción visual del ojo humano con lo que se consigue mejorar la imagen.

Como ya se ha comentado, el filtro que se ha implementado es una clase de los filtros cuadráticos de Volterra, en el que se ha utilizado la aproximación 2D del algoritmo de Teager:

$$y(i,j) = 3x^2(i,j) - \frac{1}{2}x(i+1,j+1)\ \cdot x(i-1,j-1)$$
$$-\frac{1}{2}x(i+1,j-1)\ \cdot x(i-1,j+1)$$
$$-x(i+1,j)\ \cdot x(i-1,j)$$
$$-x(i,j+1)\cdot\ x(i,j-1) \qquad (4.9)$$

donde i y j indican la fila y columna respectivamente de la máscara.

Se ha elegido este tipo de algoritmo porque presenta unos resultados muy similares a los que se tiene utilizando un filtro ideal pero con un menor gasto computacional.

La salida del filtro Volterra viene dada por:

$$Y_k = \rho * y(i,j) + x_i \qquad (4.10)$$

donde ρ es el parámetro de ponderación, *y(i, j)* es el resultado de aplicar el algoritmo de Teager y x_i es el valor de la muestra de entrada.

El parámetro de ponderación ρ se encuentra en un rango de valores entre 0.001 y 0.1 ($0.001 \leq \rho \leq 0.1$), ya que es recomendable utilizar valores pequeños de ρ. Al aumentar el valor del parámetro, se aumenta el contraste de la imagen pero también se pueden llegar a realzar demasiado los bordes y componentes de ruido impulsivo. De este modo, si se toman valores próximos al valor mínimo del parámetro se obtendrá como resultado una imagen realista de gran nitidez, mientras que si se escogen valores altos del parámetro se obtendrá una imagen con un alto contraste y con los bordes muy acentuados. El valor de $\rho = 0$, se ha omitido deliberadamente ya que la salida del filtro sería la imagen original, es decir, sería como no aplicar el filtro.

El tamaño de la máscara del filtro determina el realce que se va a lograr así como el tiempo de ejecución del filtro. A mayor tamaño de la máscara, mayor realce pero también mayor es el tiempo de filtrado. Para el filtro de Volterra el tamaño de la máscara es crucial, ya que al ser un filtro cuadrático, para máscaras de gran tamaño se necesitan realizar un elevado número de operaciones lo que hace que el filtrado llegue a ser muy lento. Para evitar este problema, se ha reducido el tamaño de la máscara únicamente a una máscara cuadrada 3x3 para reducir el gasto computacional al mínimo.

4.8.1 Algoritmo implementado

A continuación se muestra el código del algoritmo para el filtro Volterra, que se ha implementado como una función en el software que se ha desarrollado bajo el entorno de Matlab:

```
function [im_new,error]=cs(im,nker1,nker2,param,tipo_ven, color)
%im: imagen de entrada original
%im_new: imagen de salida tras el filtrado
%nker1: tamaño horizontal de la máscara
%nker2: tamaño vertical de la máscara
%param: p parámetro del filtro Volterra tal que 0.001<=p<=0.1
%tipo_ven: tipo de máscara utilizada
%color=0 procesado YCbCr
%color=1 procesado RGB
%error: función de error

error=0;
[h w c]=size(im);

if ((color == 1) && (c==3))
    cont=3;
else
    cont=1;
    if c==3
        imcbcr=rgb2ycbcr(im);
        im=imcbcr(:,:,1);
    end
end

temp = ((nker1-1)/2)+1; %sólo para la barra de progreso
progressbar(0);
tic;

for i=1:cont;
    im_new(:,:,i)=uint8(zeros(h,w));
    im_new(1:(nker1-1)/2,:,i)=im(1:(nker1-1)/2,:,i);
    im_new(h-((nker1-1)/2-1):h,:,i)=im(h-((nker1-1)/2-1):h,:,i);
    im_new(:,1:(nker2-1)/2,i)=im(:,1:(nker2-1)/2,i);
    im_new(:,w-((nker2-1)/2-1):w,i)=im(:,w-((nker2-1)/2-1):w,i);
    for y=((nker1-1)/2)+1:h-((nker1-1)/2);
        for z=((nker2-1)/2)+1:w-((nker2-1)/2);
            [ker,error]=ventana(nker1,nker2,tipo_ven,im,y,z,i);
            if error
                progressbar(1)
                error=1;
                im_new=[];
                return;
            end

            ker=double(ker);
            p1=3*ker(5)^2-0.5*ker(9)*ker(1);
            p2=0.5*ker(3)*ker(7);
            p3=ker(6)*ker(4);
            p4=ker(8)*ker(2);
            masc=p1-p2-p3-p4;
            pixelout=masc*param+double(im(y,z,i));
            pixelout=uint8(pixelout);
```

```
            im_new(y,z,i)=pixelout;
        end
        stopbar=progressbar( temp /(((h- ((nker1-1)/2)) )*cont) );
        temp=temp+1;
        if stopbar
          return
        end
    end
end

if (color == 0.0)
    if c==3
        im_new(:,:,2)=imcbcr(:,:,2);
        im_new(:,:,3)=imcbcr(:,:,3);
        im_new=ycbcr2rgb(im_new);

    end
end

progressbar(1);
toc
end
```

4.9 Filtro fractal

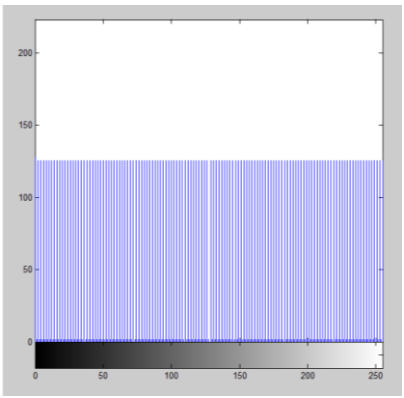
El ruido grano o speckle es uno de las principales clases de ruido que aparecen en las imágenes sonar de barrido lateral. Mediante técnicas de convolución con una serie de patrones fractales, se ha conseguido una reducción sustancial de este tipo de ruido que está presente en las imágenes de sonar de barrido lateral. Hay que tener en cuenta que se trata de un filtro lineal, lo que significa que se va a reducir el ruido pero no se van a realzar los bordes.

El uso de este algoritmo para la mejora de imágenes está inspirado en el funcionamiento de los microscopios ópticos, en los que la lente emplea estos patrones [26]. Los resultados experimentales demuestran que, en la mayoría de los casos, la reducción de estos fractales a una matriz de unos pocos píxeles es suficiente para generar una máscara de convolución óptima y obtener así un algoritmo eficiente.

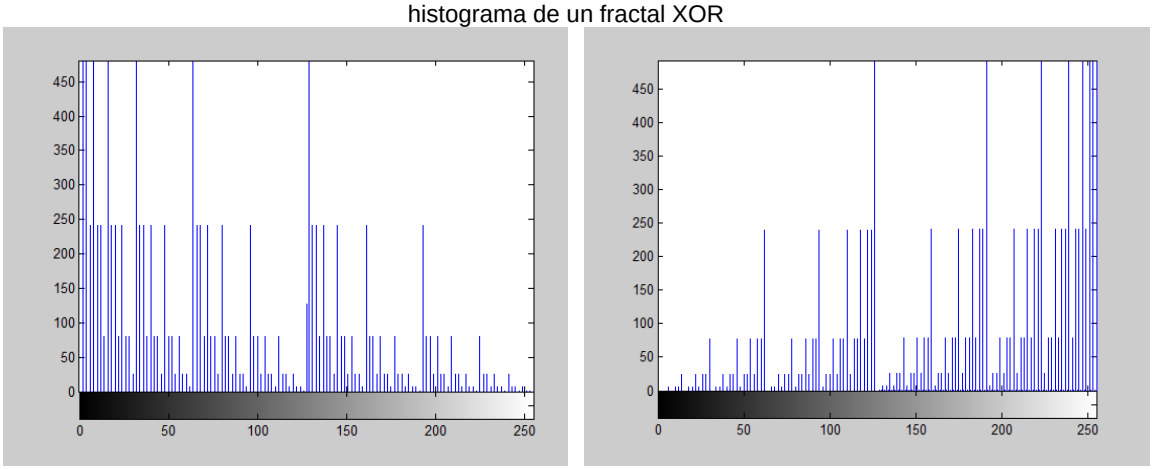
Un fractal bidimensional de dimensión impar, se genera al aplicar la operación booleana XOR a un par de imágenes ortogonales que tienen degradado en escala de grises, ver figura 37. El fractal obtenido es matemáticamente equivalente a un conjunto bidimensional de Cantor [29], con lo que sus propiedades pueden ser utilizadas posteriormente para extraer características relevantes de la imagen.

Figura 37. El fractal bidimensional XOR

El fractal XOR no es único, ya que los fractales AND y OR también poseen las mismas propiedades del conjunto de Cantor. Sin embargo, no presentan un histograma plano como el que tiene el fractal XOR. En la figura 38, se comparan los histogramas de los fractales bidimensionales XOR, AND y OR de un tamaño de 128x128 píxeles. Como se puede observar, el histograma del fractal XOR es completamente plano mientras que los histogramas de los fractales AND y OR presentan un histograma tipo Cantor.



histograma de un fractal XOR



histograma de un fractal AND

histograma de un fractal OR

Figura 38. Comparación entre los histogramas de los fractales XOR, AND y OR

4.9.1 Algoritmo implementado

Para evitar emborronar la imagen y preservar sus detalles, es necesario realizar una ampliación de la misma por un factor igual a la dimensión del fractal utilizado. Para ello, se ha implementado una pequeña función (*ampliacion*), ya que las que proporcionaba Matlab para este cometido incluían una interpolación no deseada. Una vez realizada la convolución del fractal XOR de tamaño MxM y la imagen de entrada ampliada M veces, la imagen de salida es devuelta a su tamaño original.

El grado de reducción de ruido y el suavizado resultante se puede controlar mediante el ajuste de las dimensiones de los fractales. Normalmente, un fractal de tamaño 3x3 resulta suficiente. Se debe tener en cuenta que a mayor tamaño del fractal, mayor es la ampliación de la imagen necesaria, y por lo tanto se produce un aumento en el tiempo de procesado.

A continuación se muestra el código del algoritmo para el filtro fractal, que se ha implementado como una función en el software que se ha desarrollado bajo el entorno de Matlab:

```
function im_new=fractal(im, tamano)
%im: imagen de entrada original
%im_new: imagen de salida tras el filtrado
%fractal: realiza la convolución de un fractal XOR de tamaño MxM
(impar) con la imagen de entrada ampliada M veces
%La imagen de salida se devuelve a su tamaño original

%% Cálculo del fractal, guardado en Cm
tic
progressbar(0);
imSize=tamano;
X = 1:imSize;
X0 = (X / imSize);
[Xm Ym] = meshgrid(X0);             % matrices 2D
Xm=uint8(255*Xm);
Ym=uint8(255*Ym);
Cm=bitxor(Xm,Ym);

stopbar=progressbar(1/4);
if stopbar
    return
end
%% Convolución de la imagen (en escala de grises) con el fractal
[h w c]=size(im);
if(c==3)

    imcbcr=rgb2ycbcr(im);
    im=imcbcr(:,:,1);

end
Cm=single(Cm);
```

```
a=sum(Cm);
b=sum(a);
Cm=Cm/b;
im_es=ampliacion(im, imSize);
%se amplía la imagen de entrada al tamaño del fractal
stopbar=progressbar(2/4);
if stopbar
    return
end
Ap=filter2(Cm,im_es,'full');
stopbar=progressbar(3/4);
if stopbar
    return
end
im_new=ampliacion(Ap, 1/imSize);
 if c==3
    im_new(:,:,2)=imcbcr(:,:,2);
    im_new(:,:,3)=imcbcr(:,:,3);
    im_new=ycbcr2rgb(im_new);
end
toc
progressbar(1);
```

Capítulo 5.
Filtros que utilizan funciones de MATLAB

Los siguientes filtros se han implementado utilizando funciones internas que posee el programa Matlab dentro de su *toolbox* de mejora de imagen. La documentación que se ha utilizado en la elaboración de este capítulo se encuentra en las referencias [3], [14], [25], [31], [35] y [39].

Los filtros imadjust, histeq y adapthisteq se han implementado utilizando funciones que se encuentran dentro de las rutinas de realce de contraste que tiene incorporadas Matlab.

Los filtros average, disk, gaussian, laplacian, log, motion, prewitt, sobel y unsharp son filtros lineales que se han implementado utilizando las funciones fspecial e imfilter de Matlab. La función fspecial crea filtros bidimensionales según el tipo que se haya especificado. Se pueden variar los valores de distintos parámetros según el tipo de filtro o utilizar los valores que tienen por defecto. Por otra parte, la función imfilter procesa una imagen según el filtro que se haya elegido, creando una imagen del mismo tamaño y clase que la imagen original.

5.1 Filtro imadjust

El filtro imadjust aumenta el contraste de la imagen transformando los valores de intensidad de la imagen de entrada en nuevos valores de tal manera que el 1% de los datos se satura a bajas y altas intensidades de la imagen de entrada.

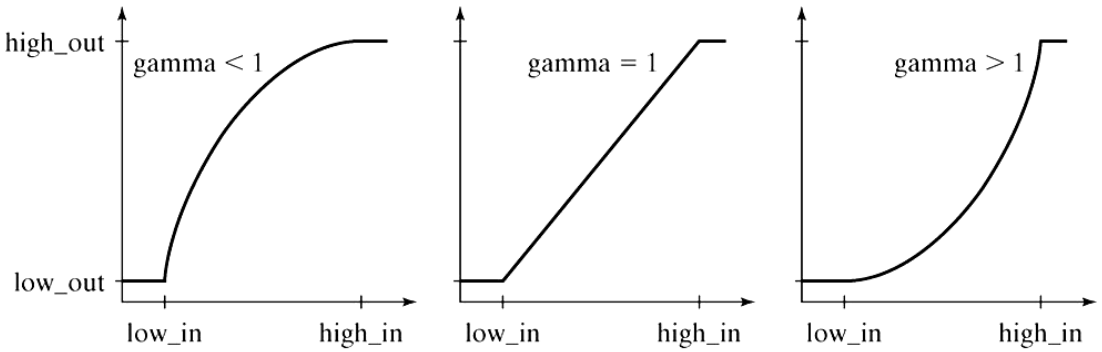


Figura 39. Influencia del parámetro gamma en la función imadjust

La función imadjust puede ampliar, reducir y, en general, cambiar los rangos de intensidad de la imagen de entrada a unos nuevos rangos en la imagen de salida, tal y como se muestra en la figura 39. Las intensidades de entrada más baja y más alta son low_in y high_in respectivamente, mientras que de forma análoga, las intensidades de salida más baja y más alta son low_out y high_out. Obviamente, los valores que en la entrada se encuentran entre low_in y high_in, a la salida deben encontrarse entre low_out y high_out. Todos los valores entre low_in y high_in y entre low_out y high_out están comprendidos entre 0 y 1. Así, los valores inferiores a low_in y los valores superiores a high_in se recortan, es decir, se asignan respectivamente a low_out y high_out.

El parámetro gamma especifica la forma de la curva que describe la relación entre los valores de la imagen de entrada y la imagen de salida.

- Si gamma < 1, la asignación de valores se inclina hacia los valores de salida superiores (píxeles brillantes).
- Si gamma = 1, la asignación de valores se realiza de forma lineal.
- Si gamma > 1, la asignación de valores se inclina hacia los valores de salida inferiores (píxeles oscuros).

Por defecto, los parámetros del filtro se definen con los siguientes valores: low_in = 0, high_in = 1, low_out = 0, high_out = 1 y gamma = 1.

5.2 Filtro histeq

El filtro histeq aumenta el contraste de la imagen mediante la transformación de los valores de intensidad de la imagen de entrada, de forma que el histograma de la imagen de salida coincida aproximadamente con un histograma especificado, que por defecto es una distribución uniforme. Así pues, el filtro histeq realiza la ecualización del histograma de la imagen.

Como ya se explicó en el apartado 3.3.2, la ecualización del histograma es un proceso en el que se redistribuyen las intensidades de la imagen, haciendo que el análisis de la imagen sea más sencillo. La distribución de los valores de intensidad se realiza de manera que el histograma acumulativo de la imagen sea aproximadamente lineal. Esto puede hacer que a veces la imagen parezca poco natural, pero también permite distinguir detalles fácilmente.

La función histeq implementa la ecualización del histograma. Aplicando la ecualización del histograma a las intensidades de la imagen de entrada, se crea otra imagen de salida con un histograma aproximadamente uniforme. Cuantos menos niveles de intensidad de entrada se utilizan, más plano es el histograma de salida que se obtiene.

5.3 Filtro adapthisteq

El filtro adapthisteq aumenta el contraste de la imagen transformando los valores de intensidad de la imagen de entrada utilizando CLAHE (*Contrast-Limited Adaptive Histogram Equalization*), es decir, realiza un contraste limitado mediante la ecualización adaptativa del histograma de la imagen.

La función adapthisteq implementa la ecualización adaptativa del histograma de la imagen. Para ello, opera en pequeñas regiones de la imagen en lugar de en toda la imagen como hace la función histeq. El contraste de cada región se mejora, de modo que el histograma de la región de salida coincida aproximadamente con el histograma especificado, que por defecto es un histograma plano de distribución

uniforme. Las regiones vecinas de la imagen se combinan utilizando interpolación bilineal para eliminar bordes producidos artificialmente. El contraste, especialmente en áreas homogéneas, puede ser limitado para evitar amplificar el ruido que pueda estar presente en la imagen.

5.4 Filtro average

El filtro average es un filtro promedio de un determinado tamaño según sea el parámetro *hsize*. Este parámetro puede ser un vector que especifica el número de filas y columnas del filtro, o puede ser un escalar en caso de que el filtro sea una matriz cuadrada. El valor por defecto de hsize es una matriz cuadrada 3x3, que es el tamaño que se ha utilizado en la implementación del filtro.

La sintaxis del filtro average es:

```
h = fspecial('average', hsize)
```

Ejecución del filtro average:

```
h = fspecial('average')

h =   0.1111    0.1111    0.1111
      0.1111    0.1111    0.1111
      0.1111    0.1111    0.1111
```

5.5 Filtro disk

El filtro disk es un filtro promedio circular donde se da más importancia al píxel central. La salida del filtro es una matriz cuadrada de tamaño 2*radius+1. El radio predeterminado es radius = 5, por lo que la matriz cuadrada de salida es de 11x11.

La sintaxis del filtro disk es:

```
h = fspecial('disk', radius)
```

Ejecución del filtro disk:

```
h = fspecial('disk')
```

```
h =
       0         0         0    0.0012    0.0050    0.0063    0.0050    0.0012         0         0         0
       0    0.0000    0.0062    0.0124    0.0127    0.0127    0.0127    0.0124    0.0062    0.0000         0
       0    0.0062    0.0127    0.0127    0.0127    0.0127    0.0127    0.0127    0.0127    0.0062         0
  0.0012    0.0124    0.0127    0.0127    0.0127    0.0127    0.0127    0.0127    0.0127    0.0124    0.0012
  0.0050    0.0127    0.0127    0.0127    0.0127    0.0127    0.0127    0.0127    0.0127    0.0127    0.0050
  0.0063    0.0127    0.0127    0.0127    0.0127    0.0127    0.0127    0.0127    0.0127    0.0127    0.0063
  0.0050    0.0127    0.0127    0.0127    0.0127    0.0127    0.0127    0.0127    0.0127    0.0127    0.0050
  0.0012    0.0124    0.0127    0.0127    0.0127    0.0127    0.0127    0.0127    0.0127    0.0124    0.0012
       0    0.0062    0.0127    0.0127    0.0127    0.0127    0.0127    0.0127    0.0127    0.0062         0
       0    0.0000    0.0062    0.0124    0.0127    0.0127    0.0127    0.0124    0.0062    0.0000         0
       0         0         0    0.0012    0.0050    0.0063    0.0050    0.0012         0         0         0
```

5.6 Filtro gaussian

El filtro gaussiano es un filtro paso bajo gaussiano rotacionalmente simétrico con desviación estándar sigma positiva, en el que el valor de cada píxel es el resultado de realizar la media con distintos pesos de los valores vecinos a ambos lados de dicho píxel. Este tipo de filtro reduce especialmente el ruido gaussiano, el cual produce pequeñas variaciones en la imagen (ver 3.1.1). Tiene su origen en diferencias de ganancias del sensor, ruido en la digitalización, etc. Pero como desventaja presenta el problema del difuminado de los bordes, aunque no lo tiene de manera tan acusada como en el caso de la media simple.

El filtro gaussiano viene dado por las siguientes ecuaciones matemáticas:

$$h_g(i,j) = e^{-\left(\frac{i^2+j^2}{2\sigma^2}\right)} \tag{5.1}$$

$$h(i,j) = \frac{h_g(i,j)}{\sum_i \sum_j h_g(i,j)} \tag{5.2}$$

donde *i* y *j* indican la fila y columna de la máscara, σ es la desviación estándar, h_g es el filtro paso bajo gaussiano y *h* es la salida del filtro.

Al igual que el filtro average, el tamaño de salida del filtro gaussiano depende del parámetro *hsize*, que puede ser un vector que especifica el número de filas y columnas del filtro, o un escalar en caso de que el filtro sea una matriz cuadrada. El parámetro *sigma* (σ) es la desviación estándar y controla la forma de la curva gaussiana, cuanto mayor es su valor, más plana es la curva. El valor predeterminado

de la salida del filtro gaussiano es una matriz cuadrada de tamaño 3x3 y con una desviación estándar de valor sigma = 0.5.

La sintaxis del filtro gaussiano es:

```
h = fspecial('gaussian', hsize, sigma)
```

Ejecución del filtro gaussiano:

```
h = fspecial('gaussian')

h =   0.0113    0.0838    0.0113
      0.0838    0.6193    0.0838
      0.0113    0.0838    0.0113
```

5.7 Filtro laplacian

El filtro laplaciano es un filtro paso alto en el que su salida es una matriz cuadrada 3x3 cuya forma se aproxima a la del operador laplaciano bidimensional. Como se basa en un operador derivativo, va a acentuar las zonas que tienen gran discontinuidad en la imagen, es decir, los bordes. La segunda derivada es más sensible que la primera derivada ante cambios bruscos en la imagen, por lo que el filtro también será más sensible al ruido. Por esta razón, se suele aplicar primero un filtro gaussiano para eliminar ruido de la imagen y después un filtro laplaciano para realzar bordes, esto es lo que viene a ser el filtro log (*laplacian of gaussian*) que se verá en el siguiente apartado.

El laplaciano de una imagen con valores de intensidad *I(x,y)* viene dado por la siguiente expresión:

$$\nabla^2 I(x,y) = \frac{\partial^2 I(x,y)}{\partial x^2} + \frac{\partial^2 I(x,y)}{\partial y^2} \tag{5.3}$$

donde ∇^2 es el operador laplaciano y ∂ es la derivada parcial.

La máscara del filtro laplaciano que se ha utilizado para implementar la ecuación (5.3) viene dada por:

$$\frac{4}{\alpha+1}\begin{bmatrix} \frac{\alpha}{4} & \frac{1-\alpha}{4} & \frac{\alpha}{4} \\ \frac{1-\alpha}{4} & -1 & \frac{1-\alpha}{4} \\ \frac{\alpha}{4} & \frac{1-\alpha}{4} & \frac{\alpha}{4} \end{bmatrix} \tag{5.4}$$

donde α es un parámetro del filtro laplaciano. El parámetro *alfa* (*α*) controla la forma del laplaciano y debe estar dentro del rango de 0 a 1. Su valor predeterminado es de alfa = 0.2.

La sintaxis del filtro laplaciano es:

```
h = fspecial('laplacian', alpha)
```

Ejecución del filtro laplaciano:

```
h = fspecial('laplacian')

h =   0.1667    0.6667    0.1667
      0.6667   -3.3333    0.6667
      0.1667    0.6667    0.1667
```

5.8 Filtro laplacian of gaussian (log)

El filtro laplaciano del gaussiano es un filtro en el que su salida es la aplicación del operador laplaciano a un filtro gaussiano rotacionalmente simétrico con una desviación estándar sigma positiva.

El filtro log está formado por la composición del filtro gaussiano y del laplaciano, y viene dado por la siguiente expresión:

$$h(i,j) = \frac{(i^2 + j^2 - 2\sigma^2) \cdot h_g(i,j)}{2\pi\sigma^6 \sum_i \sum_j h_g(i,j)} \tag{5.5}$$

donde *i* y *j* indican la fila y columna de la máscara, *σ* es la desviación estándar, h_g es el filtro gaussiano y *h* es la salida del filtro log.

Como ya se explicado anteriormente, el parámetro *hsize* determina el tamaño de salida del filtro, mientras que el parámetro *sigma* (σ) se encarga de controlar la forma de la curva gaussiana. El valor predeterminado de la salida del filtro log es una matriz cuadrada de tamaño 5x5 y con un valor de sigma = 0.5.

La sintaxis del filtro log es:

```
h = fspecial('log', hsize, sigma)
```

Ejecución del filtro log:

```
h = fspecial('log')
```

```
h =   0.0448    0.0468    0.0564    0.0468    0.0448
      0.0468    0.3167    0.7146    0.3167    0.0468
      0.0564    0.7146   -4.9048    0.7146    0.0564
      0.0468    0.3167    0.7146    0.3167    0.0468
      0.0448    0.0468    0.0564    0.0468    0.0448
```

5.9 Filtro motion

El filtro motion es un filtro que una vez que la imagen está convolucionada, crea una sensación de movimiento en ella que se aproxima al movimiento lineal de una cámara de longitud *len* en píxeles y con un ángulo de *theta* grados en el sentido contrario a las agujas del reloj. De esta forma, el filtro se convierte en un vector de movimientos horizontales y verticales. El valor predeterminado para la longitud es len = 9 y el del ángulo es theta = 0, los cuales corresponden a un movimiento horizontal de 9 píxeles.

La sintaxis del filtro motion es:

```
h = fspecial('motion', len, theta)
```

Para calcular el filtro motion:

- Se construye un segmento de recta ideal, con la longitud y el ángulo deseados, centrado en el centro del coeficiente h del filtro.
- En cada posición (i, j) del coeficiente h, se calcula la distancia más cercana entre esa ubicación y el segmento de la recta ideal:

  ```
  h = max (1 - nearest_distance, 0)
  ```

- Por último, se normaliza el filtro:

  ```
  h = h/(sum(h(:))
  ```

Ejecución del filtro motion:

```
h = fspecial('motion')
```

```
h = 0.1111 0.1111 0.1111 0.1111 0.1111 0.1111 0.1111 0.1111 0.1111
```

5.10 Filtro prewitt

El filtro prewitt es un filtro en el que su salida es una matriz cuadrada 3x3 que resalta los bordes horizontales mediante la aproximación del gradiente vertical, es decir, el gradiente de los bordes verticales. En caso de que se necesite resaltar los bordes verticales se realiza la trasposición de la matriz de salida del filtro, o lo que es lo mismo, se calcula h'.

La sintaxis del filtro prewitt es:

```
h = fspecial('prewitt')
```

Ejecución del filtro prewitt de gradiente vertical (h) y de gradiente horizontal (h'):

```
h = fspecial('prewitt')

h =   1     1     1
      0     0     0
     -1    -1    -1

h'=   1     0    -1
      1     0    -1
      1     0    -1
```

Para el software desarrollado se ha utilizado el filtro prewitt de gradiente vertical que acentúa los bordes horizontales de la imagen.

5.11 Filtro sobel

El filtro sobel es un filtro en el que su salida es una matriz cuadrada 3x3 que resalta los bordes horizontales mediante la aproximación del gradiente vertical, es decir, el gradiente de los bordes verticales. En caso de que se necesite resaltar los bordes verticales se realiza la trasposición de la matriz de salida del filtro, o lo que es lo mismo, se calcula h'.

La sintaxis del filtro sobel es:

```
h = fspecial('sobel')
```

Ejecución del filtro sobel de gradiente vertical (h) y de gradiente horizontal (h'):

```
h = fspecial('sobel')

h =   1     2     1
      0     0     0
     -1    -2    -1
```

```
h'=   1     0    -1
      2     0    -2
      1     0    -1
```

Como se ha podido comprobar el filtro sobel es muy similar al filtro prewitt, pues si bien los dos filtros realizan la misma función de resalte de bordes mediante la utilización de gradientes, hasta las máscaras de los gradientes son casi idénticas. Esto hace que la utilización de estos dos filtros tenga unos resultados muy parecidos aunque la respuesta del filtro prewitt no es tan isotrópica como la del filtro sobel.

Para el software desarrollado se ha utilizado el filtro sobel de gradiente vertical que acentúa los bordes horizontales de la imagen.

5.12 Filtro unsharp

El filtro unsharp es un filtro en el que la salida del filtro es una matriz cuadrada 3x3. Es un filtro que se crea a partir del negativo del filtro laplaciano. El filtro unsharp sustrae una versión desenfocada o suavizada de la imagen original para obtener una imagen de salida con bordes más realzados. Es decir, suaviza la imagen en las zonas con poca transición entre los píxeles y resalta las zonas que tienen una rápida transición entre los píxeles como ocurre en los bordes.

La máscara del filtro unsharp que se ha utilizado viene dada por la siguiente expresión:

$$\frac{1}{\alpha+1}\begin{bmatrix} -\alpha & \alpha-1 & -\alpha \\ \alpha-1 & \alpha+5 & \alpha-1 \\ -\alpha & \alpha-1 & -\alpha \end{bmatrix} \tag{5.6}$$

donde α es un parámetro del filtro unsharp. Al igual que en el filtro laplaciano, el parámetro *alfa* (*α*) controla la forma del laplaciano y debe encontrarse dentro del rango de 0 a 1. Su valor predeterminado es de alfa = 0.2.

La sintaxis del filtro unsharp es:

```
h = fspecial('unsharp', alpha)
```

Ejecución del filtro unsharp:

```
h = fspecial('unsharp')

h =   -0.1667   -0.6667   -0.1667
      -0.6667    4.3333   -0.6667
      -0.1667   -0.6667   -0.1667
```

Capítulo 6.
Filtros para detectar bordes y objetos

La finalidad de los filtros vistos en capítulos anteriores era preparar las imágenes para un posterior análisis automatizado, consistente en la detección de los elementos con información de interés, es decir, los objetos. Para poder distinguir un objeto de un fondo hay que fijarse en su contorno o borde más característico. Por ello, en este capítulo se presentan varios métodos y algoritmos para la detección de bordes y objetos hechos por el hombre.

6.1 Segmentación de la imagen

Los filtros para detectar bordes y objetos utilizan la *segmentación* como base fundamental [1]. La segmentación divide una imagen en los distintos objetos o regiones que la forman. El proceso es iterativo y debe finalizar cuando los objetos de interés hayan sido detectados. La precisión de la segmentación puede ser decisiva para determinar el acierto o el fallo de los posteriores análisis computacionales.

La mayoría de algoritmos para segmentar imágenes se basan en una de las siguientes propiedades de los valores de intensidad de la imagen:

- *Discontinuidad*: se divide la imagen en regiones tomando como referencia los cambios bruscos de intensidad, como ocurre en los bordes. En esta propiedad se fundamentan los algoritmos para detectar bordes, líneas y puntos aislados.
- *Similitud*: se divide la imagen en regiones similares de acuerdo a un criterio preestablecido. Se apoyan en esta propiedad la elección de umbrales, la separación de regiones y las uniones.

También se pueden combinar ambas propiedades, como ocurre en los métodos de detección de bordes presentados en el siguiente apartado.

6.2 Detectores de bordes

Los detectores de bordes son métodos locales de procesado de imagen, diseñados para detectar los píxeles que forman parte de los bordes. Una línea se considera que es un borde cuando la intensidad del fondo a cada lado de ella es mucho mayor o mucho menor que su propia intensidad.

Los cambios abruptos de intensidad pueden ser detectados de forma efectiva utilizando derivadas [13], especialmente la primera y segunda derivada. Las derivadas de funciones digitales están basadas en diferencias puesto que se emplean aproximaciones. De esta forma, dada una función unidimensional f(x), la primera y segunda derivada en un punto x, vienen dadas por las siguientes aproximaciones:

$$\frac{\partial f(x)}{\partial x} = f(x+1) - f(x) \tag{6.1}$$

$$\frac{\partial^2 f(x)}{\partial x^2} = f(x+1) + f(x-1) - 2f(x) \tag{6.2}$$

Las ecuaciones (6.1) y (6.2), cumplen las siguientes propiedades:

- *La primera derivada* debe ser cero en áreas de intensidad constante y distinta de cero en el comienzo y final de un salto o variación de intensidad así como a lo largo de una rampa de intensidad.
- *La segunda derivada* debe ser cero en áreas de intensidad constante, distinta de cero en el comienzo y final de un salto o variación de intensidad y cero a lo largo de una rampa de intensidad.

Por lo tanto, el uso de las derivadas en la detección de bordes hace que se obtengan las siguientes conclusiones:

- La primera derivada generalmente detecta bordes muy gruesos.
- La segunda derivada detecta con mayor facilidad detalles finos como son puntos aislados y líneas pero también es muy sensible al ruido.
- La segunda derivada muestra una respuesta de borde doble en las transiciones de intensidad.
- El signo de la segunda derivada indica si la transición en un borde va de claro a oscuro o viceversa.

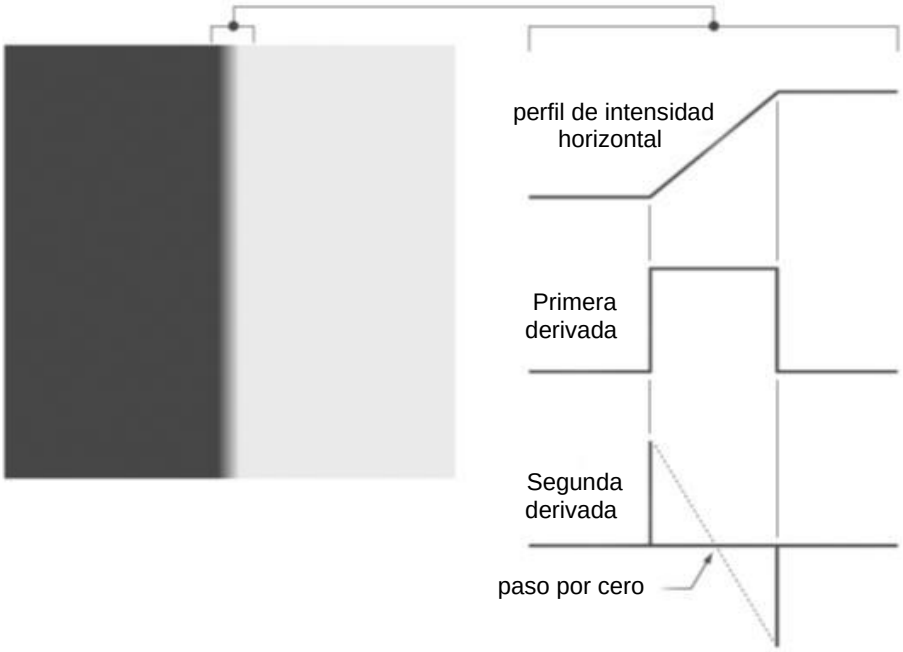


Figura 40. Detección de bordes

En la figura 40, se muestra un ejemplo de cómo se detecta un borde en una transición de intensidad de gris oscuro a gris claro. La primera derivada detecta un borde (la zona en que la derivada es distinta de cero) y la segunda derivada confirma el borde añadiendo la información de que la transición se ha producido del oscuro al claro (al pasar la segunda derivada de tener signo positivo al comienzo de la transición a tener signo negativo al final de la transición).

La aplicación de las derivadas al procesado de imágenes, se puede realizar a través del uso de máscaras y del filtrado espacial. Dependiendo del tipo de método

utilizado para detectar bordes los coeficientes de la máscara de filtrado tendrán valores distintos y se utilizarán distintos tipos de filtros.

Para el programa que se ha desarrollado se ha utilizado la función edge de Matlab que reúne varios métodos de detección de bordes, todos ellos basados en el uso de la derivada y que se explicarán brevemente en el siguiente apartado utilizando las referencias bibliográficas [13], [14], [31] y [39].

6.2.1 Métodos de detección de bordes en la función edge

La función edge tiene como entrada una imagen en escala de grises o binaria (`I`) y como salida produce una imagen binaria (BW) del mismo tamaño que la original con 1's donde la función ha encontrado bordes, y 0's en el resto. Una imagen en escala de grises se puede convertir en una imagen binaria umbralizando los niveles de gris a sólo 2 niveles: blanco y negro. Así en las imágenes binarias, un borde corresponde a un cambio del nivel de intensidad de 1 a 0 y viceversa.

La sintaxis general de la función edge es la siguiente:

```
[BW,thresh]=edge (I)

[BW,thresh]=edge (I,'método',parámetros)
```

Con la primera notación se utiliza el método por defecto, que es el de Sobel, y como salida se obtiene una imagen binaria con los bordes según el umbral utilizado (thresh). Con la segunda notación, se especifica el método que se quiere emplear para la detección y los valores de los parámetros adicionales. Entre los parámetros que se pueden especificar con estos métodos se encuentran:

- el umbral: por debajo de su valor se ignoran los bordes obtenidos.
- la dirección de detección: se puede escoger entre detección horizontal, vertical y ambas direcciones.

Si no se especifican estos parámetros, se toman los valores por defecto. De esta forma, el valor del umbral se elige de forma automática y la detección se realiza en dirección horizontal y vertical.

Los distintos métodos de detección de bordes que vienen incluidos en la función edge se van a describir en los próximos apartados.

6.2.1.1 Método Sobel

Este método encuentra bordes empleando la aproximación de Sobel para la derivada. Su imagen de salida presenta bordes en los puntos donde el gradiente de la imagen de entrada es máximo. Las máscaras de convolución del método Sobel pueden verse en la figura 41. Como se puede apreciar, se utilizan dos máscaras de tamaño 3x3. La máscara G_x se utiliza para calcular el gradiente en la dirección horizontal y la G_y para calcular el gradiente en la dirección vertical. Una máscara es la otra rotada 90º.

1	**0**	**-1**
2	**0**	**-2**
1	**0**	**-1**

G_x

1	**2**	**1**
0	**0**	**0**
-1	**-2**	**-1**

G_y

f_1	f_2	f_3
f_4	f_5	f_6
f_7	f_8	f_9

Sección de la imagen de entrada

Figura 41. Máscaras de convolución del método Sobel y sección de la imagen de entrada

Las máscaras se pueden aplicar de forma separada en la imagen de entrada, para obtener las componentes del gradiente en cada orientación. Una vez obtenidos los valores de G_x y G_y, se pueden combinar para calcular la magnitud absoluta del gradiente.

Tomando una sección de la imagen de entrada de las mismas dimensiones que la máscara del método Sobel, ver figura 41, el resultado de la convolución viene dado por:

$$|G| = |(f_1 + 2f_2 + f_3) - (f_7 + 2f_8 + f_9)| + |(f_1 + 2f_4 + f_7) - (f_3 + 2f_6 + f_9)| \qquad (6.3)$$

donde |G| es el módulo del gradiente y los f_1,…, f_9 son las posiciones de los píxeles de la imagen de entrada.

Por otra parte, el módulo de una magnitud, en este caso el gradiente, viene dado por:

$$|G| = \sqrt{{G_x}^2 + {G_y}^2} \qquad (6.4)$$

Y para realizar los cálculos de manera más rápida se suele emplear la aproximación:

$$|G| \approx |G_x| + |G_y| \tag{6.5}$$

La dirección del gradiente se calcula como:

$$\theta = arctan^{-1}\left(\frac{G_y}{G_x}\right) \tag{6.6}$$

donde θ es 0 cuando se produce la dirección de máximo contraste en la imagen, es decir, de negro a blanco yendo de izquierda a derecha.

6.2.1.2 Método Prewitt

Este método encuentra bordes utilizando la aproximación de Prewitt de la derivada. Al igual que con el método Sobel, su imagen de salida presenta bordes en los puntos donde el gradiente de la imagen de entrada es máximo. Funciona de manera muy similar al método Sobel, pero utiliza unas máscaras con valores ligeramente diferentes como puede verse en la figura 42. Además, la aplicación de las máscaras de Prewitt produce resultados similares a las de Sobel pero con una respuesta menos isotrópica.

1	**0**	**-1**
1	**0**	**-1**
1	**0**	**-1**

G_x

1	**1**	**1**
0	**0**	**0**
-1	**-1**	**-1**

G_y

Figura 42. Máscaras de convolución del método Prewitt

6.2.1.3 Método Roberts

Este método encuentra bordes usando la aproximación de Roberts de la derivada. Al igual que los dos métodos anteriores, Sobel y Prewitt, su imagen de salida presenta bordes en los puntos donde el gradiente de la imagen de entrada es máximo. Las máscaras del método Roberts son más sencillas que las máscaras anteriores pero su respuesta es más sensible al ruido. En la figura 43, se muestran las máscaras de convolución del método Roberts.

Figura 43. Máscaras de convolución del método Roberts

6.2.1.4 Método del Laplaciano del Gaussiano (LoG)

Este método encuentra bordes localizando los cruces por cero tras filtrar la imagen de entrada con un filtro Laplaciano del Gaussiano (LoG), ver 5.8. Una vez que la imagen se ha filtrado es cuando se detectan los cruces por cero. Este paso se puede realizar de varias maneras, siendo la más sencilla poner el valor del umbral a 0 para obtener una imagen binaria de salida con bordes cerrados, al incluir todos los pasos por cero de la imagen de entrada.

6.2.1.5 Método Canny

Este método encuentra bordes buscando los puntos donde el gradiente de la imagen de entrada es un máximo local. El gradiente es calculado usando la derivada de un filtro Gaussiano, ver 5.6. Esto significa que primero se realiza una convolución de la imagen de entrada con un filtro gaussiano para suavizar y eliminar ruidos. Y después se calcula el gradiente de la imagen suavizada, para determinar los píxeles donde se produce una variación máxima, es decir, un mayor módulo del vector gradiente. También se determina la dirección del vector gradiente.

La matriz del módulo del gradiente de la función gaussiana tendrá valores grandes donde la variación de la intensidad sea grande. Se eliminan (igualan a cero) aquellos píxeles que no son máximos locales en la dirección del gradiente (dirección perpendicular al borde).

El método Canny [7] emplea dos umbrales para el gradiente, uno para detectar bordes finos y otro para detectar bordes gruesos. Los bordes finos se incluyen en la imagen de salida sólo si están conectados con los bordes gruesos o principales. Esta doble umbralización elimina la aparición de bordes dobles o falsos. En comparación con los métodos anteriores, éste es el más robusto al ruido y el más adecuado a la hora de detectar bordes finos.

En resumen:

- En todos los métodos el valor del umbral es elegido heurísticamente de una forma que depende de los datos de entrada. La mejor forma de modificar el umbral es obtener su valor de forma automática y en función del resultado obtenido, ajustar su valor de forma manual, introduciendo un valor más alto (menos píxeles en los bordes) o un valor más bajo (más píxeles en los bordes).
- La detección de bordes utilizando métodos que se basan en aproximaciones del gradiente suelen funcionar bien en casos en los que se tienen imágenes con transiciones de intensidad claramente definidas y ruidos relativamente bajos. Para los casos en que los bordes están emborronados o cuando hay un alto contenido de ruido en la imagen se utilizan los cruces por cero. El cruce por cero ofrece fiabilidad en las localizaciones de bordes y reduce los efectos del ruido gracias a la propiedad de suavizado de la convolución gaussiana. Sin embargo, este método hace que aumente el tiempo y la complejidad de cálculo. El método de Canny es el que ofrece mejores resultados para bordes de imágenes con ruido gaussiano, pero también aumenta el tiempo de cómputo

6.3 Detección de objetos hechos por el hombre

Continuando con la idea de segmentación que se explicó en el apartado 6.1, una vez que se tengan los objetos localizados se pueden determinar ciertas características que permitan clasificarlos y por consiguiente conocer si son objetos,

simples rocas o formas del fondo marino. Existen varios métodos para detectar objetos en imágenes pero los principales son:

- *Los métodos basados en propiedades de las texturas de los objetos*. Primero se crea una base de datos de texturas de objetos conocidos que estén presentes en las imágenes de sonar. En dicha base de datos se tiene información acerca de la homogeneidad, la entropía o el valor medio de los píxeles que forman el objeto. Posteriormente se estudia la imagen por regiones, se extrae la información de la textura en cada región y se compara con la base de datos de las texturas de objetos conocidos. Cuando se produce una diferencia mínima al realizar la comparación, se puede concluir que en esa región de la imagen se ha encontrado un objeto de similares características al de la base de datos. Las imágenes obtenidas con un sonar de barrido lateral suelen presentar mucho ruido así como grandes diferencias en su iluminación dependiendo del tipo de sonar que se haya utilizado. Lo que se traduce en la aparición de falsos positivos y por ello este método no es aconsejable para las imágenes de sonar de barrido lateral.
- *Los métodos basados en una segmentación de la imagen en zonas de similares características*. Posteriormente se puede realizar una búsqueda de objetos atendiendo a características como la forma, el tamaño, etc. Este método es el más efectivo para imágenes de sonar de barrido lateral. La técnica que se va a explicar en este apartado pertenece a este tipo de método y se puede encontrar en la referencia [8]. El método propuesto está diseñado para ser aplicado en imágenes de sonar de barrido lateral, pero con pequeñas modificaciones resultaría igualmente válido para otros tipos de imágenes.

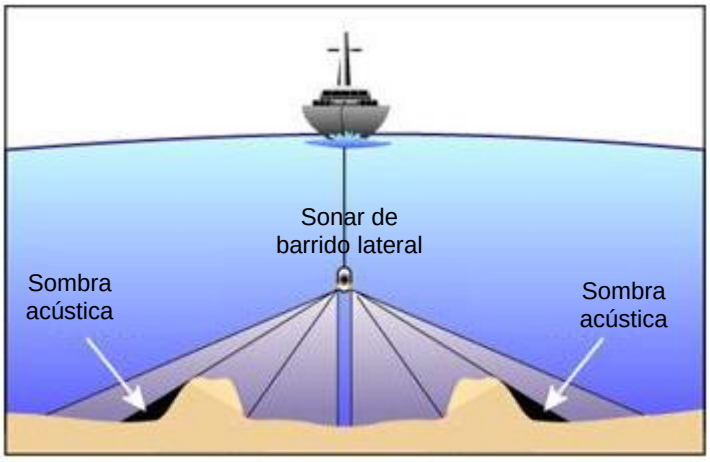


Figura 44. Sombra acústica de los objetos sumergidos

En las imágenes de sonar, un objeto o una perturbación significativa del terreno, aparece reflejado como una zona de mayor luminosidad en comparación con el fondo que está más oscuro. A las regiones brillantes se las denomina *zonas iluminadas* y a las regiones del fondo se las denomina *zonas oscuras o de sombra*. Además, los objetos presentan lo que se conoce como una *zona de sombra acústica*, que es un lugar de la imagen que es más oscuro que el fondo. La sombra acústica es la sombra del objeto sumergido. Así, en una imagen de sonar de barrido lateral un objeto localizado en la parte izquierda de la imagen tiene su sombra acústica a su izquierda y el que se encuentre en la parte derecha tiene su sombra acústica a su derecha tal y como puede apreciarse en la figura 44.

Para realizar la detección automática de objetos, primero hay que realizar una segmentación separando las zonas iluminadas de las zonas oscuras, para posteriormente clasificar los resultados según los rasgos extraídos. Para llevar a cabo la segmentación, hay que tener en cuenta las variaciones de iluminación que se producen en las imágenes de sonar de barrido lateral. Este problema se puede corregir mediante un filtrado que distribuya las intensidades de forma uniforme, es decir, una mejora del histograma aumentando así el contraste de la imagen como se muestra en el apartado 6.3.1. La segmentación y extracción de rasgos se presentan en el apartado 6.3.2.

6.3.1 Filtro de potencia adaptativo

Como ya se ha comentado, la detección de los objetos se basa en la segmentación, separando las zonas de interés (zonas iluminadas) de las que no interesan (zonas oscuras). Para que la segmentación ofrezca unos resultados aceptables, se debe maximizar el número de zonas que no son iluminadas ni oscuras (zonas intermedias), con el propósito de obtener un menor número de resultados positivos y evitar en la medida de lo posible falsos positivos.

En este tipo de procesado se trabaja con una capa de la imagen que tiene una profundidad de 8 bits, es decir, 256 valores posibles para cada píxel. Con el fin de discernir si una región es considerada como zona luminosa, zona oscura o ninguna de las dos, a partir de las imágenes de sonar de barrido lateral se hacen las siguientes consideraciones:

- La mayoría de los píxeles de las imágenes de sonar de barrido lateral forman parte del fondo.

- Los píxeles de las zonas iluminadas serán aquéllos cuyo valor sea >192.
- Los píxeles de las zonas oscuras serán aquéllos cuyo valor sea <64.

Estos umbrales tienen estos valores por defecto, pero en la aplicación desarrollada el usuario puede personalizar sus valores mediante una barra de desplazamiento, teniendo en cuenta los cambios de iluminación.

El filtrado de potencia es una técnica utilizada muy a menudo en el procesado de la imagen para aumentar el contraste de la imagen. El filtro de potencia utilizado viene dado por la siguiente expresión:

$$salida\ (i,j) = entrada\ (i,j)^r \tag{6.7}$$

donde i, j son las coordenadas de cada píxel y r es el factor gamma.

- Si r > 1, se incrementan los píxeles de alta intensidad, es decir, los píxeles iluminados, a costa de reducir los oscuros.
- Si r < 1, se incrementan los píxeles de baja intensidad, es decir, los píxeles oscuros, a costa de reducir los iluminados.

Para maximizar el número de píxeles con valores de intensidad comprendidos entre 64 y 192 (zonas intermedias), el algoritmo debe buscar el valor de r que lo haga posible.

Cuando se procesan imágenes de barrido lateral completas existe un problema añadido puesto que toda la superficie no aparece iluminada de manera homogénea. Así la iluminación se distribuye de forma homogénea en columnas (verticalmente), mientras que horizontalmente se producen en ocasiones variaciones importantes. La solución a este problema es aplicar el algoritmo a cada columna por separado para después juntar los resultados, aumentando de forma considerable el cálculo computacional. En esta tesis no se ha implementado este proceso, ya que las imágenes que se han utilizado están recortadas y no presentan este problema.

6.3.1.1 Algoritmo implementado

Cumpliendo con las consideraciones descritas en el apartado anterior, se ha implementado un algoritmo para encontrar el valor óptimo de r que maximiza el número de píxeles con valores de intensidad comprendidos entre 64 y 192. Para ello

se realiza un estudio previo del histograma de la imagen y a continuación se efectúan una serie de iteraciones para obtener el valor final del valor central r.

Una vez que se tiene el histograma de la imagen:

- si el nº de píxeles con valores de intensidad < 64 (oscuros) es ≥ que el nº de píxeles con valores de intensidad > 192 (brillantes), entonces se optará por un r < 1 (se incrementan los píxeles oscuros).
- si el nº de píxeles con valores de intensidad < 64 (oscuros) es < que el nº de píxeles con valores de intensidad > 192 (brillantes), entonces se optará por un r > 1 (se incrementan los píxeles brillantes).

El valor de r se hace variar con un paso de 0,01 en un rango de $\pm$ 0,2 para un valor central. Este valor central de r se calcula en función del % (p) sobre el total de píxeles con valores de intensidad menor que 64 o mayor que 192, según sea el caso. De esta forma:

- Cuando r > 1, (mayoría de píxeles brillantes, de valor >192) el valor central se calcula como 1+ p/100. Por ejemplo, si el 30% de píxeles son mayores que 192, entonces el valor central vale r = 1,30.
- Cuando r < 1 (mayoría de píxeles oscuros, de valor < que 64), el valor central se calcula como 1− p/100. Por ejemplo si el 30% de píxeles son menores que 64, entonces el valor central vale r = 0,7.

Variando el valor de r dentro del rango definido de $\pm$ 0,2, el valor final del valor central será aquél que maximice el número de píxeles comprendidos entre 64 y 192.

A continuación se muestra el código del algoritmo para el filtro de potencia adaptativo, que se ha implementado como una función en el software que se ha desarrollado bajo el entorno de Matlab:

```
function [iout error]=histoganma(i, um2, um1)
%Se introduce una imagen de entrada en escala de grises, y como imagen
%de salida se obtiene una cuyo histograma se ha modificado según la
%expresión out(x,y)=in(x,y)^r, siendo r elegido de forma que se
%maximicen el número de píxeles cuya intensidad está entre 64 y 192.
um1=int16(um1);
um2=int16(um2);
um2=um2+1;
error=0;
if isequal(class(i),'logical')
    msgbox(['La imagen debe estar en escala de grises, no
    binaria'],'Error imagen','error','modal');
    error=1;
    iout=i;
```

```
    return;
end
[y,x]=imhist(i);    %y:número de píxeles, x: valores de 1 a 255
tot=numel(i);
y64=sum(y(1:um1));   %número de píxeles menores que 64
y192=sum(y(um2:256));
m1=tot-y64-y192;    %m1: número de píxeles entre 64 y 192

if y64<=y192        %r será < 1

    g=(100-round(y64*100/tot))/100; %valor de g de partida varía en
                                     pasos de 0.01
    g1=double(uint8(g*100-20))/100; %por si es negativo que valga 0
    g2=g+0.020;                     %aunque valga más de 1 no importa
    paso=0.01;
    rango=g1:paso:g2;
    i2=imadjust(i,[0 1],[0 1],g);
    [yy,xx]=imhist(i2);
    yy64=sum(yy(1:um1)); % número de píxeles menores que 64
    yy192=sum(yy(um2:256));
    m2=tot-yy64-yy192;
    m2_max=m2;          %valor previo de m2 para comparar con el nuevo
    for r=1:length(rango)
        i2=imadjust(i,[0 1],[0 1],rango(r));
        [yy,xx]=imhist(i2);
        yy64=sum(yy(1:um1));
        yy192=sum(yy(um2:256));
        m2=tot-yy64-yy192;
        if m2>m2_max
            m2_max=m2;
            g=rango(r);
        end
    end

else                %r será >1

    g=(100+round(y192*100/tot))/100; %valor de g de partida ,varia en
                                      pasos de 0.01
    g1=(g*100-20)/100;
    g2=g+0.020;
    paso=0.01;
    rango=g1:paso:g2;
    i2=imadjust(i,[0 1],[0 1],g);
    [yy,xx]=imhist(i2);
    yy64=sum(yy(1:um1));
    yy192=sum(yy(um2:256));
    m2=tot-yy64-yy192;
    m2_max=m2;          %valor previo de m2 para comparar con el nuevo
    for r=1:length(rango)
        i2=imadjust(i,[0 1],[0 1],rango(r));
        [yy,xx]=imhist(i2);
        yy64=sum(yy(1:um1));
        yy192=sum(yy(um2:256));
        m2=tot-yy64-yy192;
        if m2>m2_max
            m2_max=m2;
            g=rango(r);
        end
```

```
    end
end
iout=imadjust(i,[0 1],[0 1],g);
```

6.3.2 Segmentación y extracción de rasgos

Después del filtrado con el que se consigue una iluminación uniforme en la imagen, se realiza la segmentación de la imagen en zonas iluminadas y zonas oscuras, es decir, en objetos y sus respectivas sombras, ambos por separado. Para lograrlo, se emplean métodos basados en umbrales. Se han utilizado los umbrales de 192 para localizar los objetos y de 64 para encontrar las sombras, lo que en imágenes en escala de grises equivale a (3/4)*256 y (1/4)*256 respectivamente en los valores de intensidad.

Posteriormente, se extraen los rasgos y características que indican las componentes de los posibles objetos de interés. De la observación se deduce que los objetos hechos por el hombre reúnen las siguientes características [8]:

- *Tamaño*: los objetos hechos por el hombre se encuentran dentro de un cierto rango de tamaño. El tamaño de cada objeto se define como su número total de píxeles. Se sigue el criterio de que si un objeto es muy grande o muy pequeño, probablemente se trate de ruido o de rocas. También se pueden encontrar defectos en la imagen, pero éstos tienen unas dimensiones características, por lo que se descartarán aquellos componentes que cumplan que su *ancho* < *0.15*alto* o su *alto* < *0.15*ancho*.
- *Intensidad media del objeto:* es significativamente diferente de la intensidad media de su fondo. El fondo de un objeto se define como todos los píxeles de fondo que se encuentran dentro del recuadro punteado de la figura 45, exceptuando el rectángulo que contiene los píxeles del objeto.

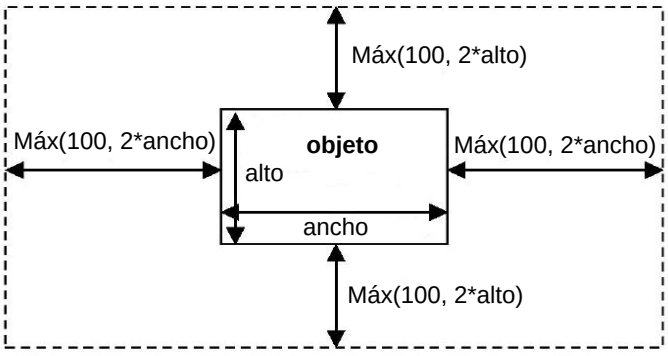


Figura 45. Dimensiones del fondo de un objeto

- *Número de agujeros*: los objetos de interés deberán tener un número pequeño de agujeros, como máximo dos. Los componentes que tengan más de dos agujeros serán descartados, pues se considerarán que son ruido o rocas.
- *Contorno regular*: los objetos hechos por el hombre tienden a tener contornos regulares caracterizados por cambios mínimos en su dirección.

Para conseguir detectar los cambios mínimos de dirección, primero se utiliza un algoritmo que se encarga de trazar los contornos y después un código en cadena para delimitar la dirección de los contornos. Los algoritmos de código en cadena, sustituyen el contorno por una serie de líneas rectas caracterizadas por su dirección y tamaño. Las líneas rectas se codifican con un código específico dependiendo de su dirección. En la figura 46 se muestran los códigos en cadena de 4 y 8 direcciones, siendo éste último el que se ha utilizado para la aproximación de los contornos.

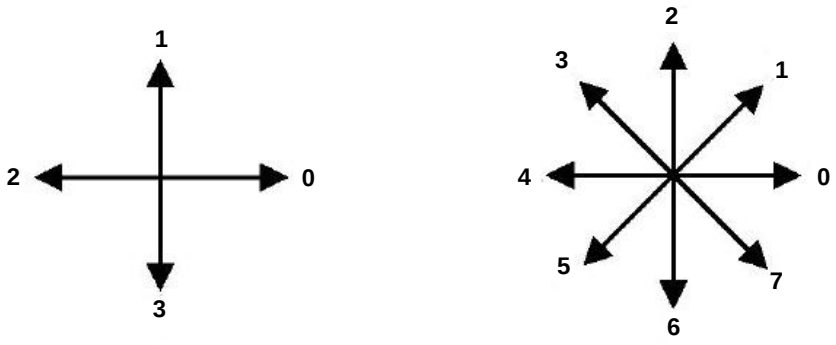


Figura 46. Códigos en cadena de 4 y 8 direcciones

Después, se localizan los fallos del código en cadena que ocurren cuando:

$$c[i-1] = c[i+1] \quad tal\ que\ \ c[i-1] \neq c[i] \tag{6.8}$$

donde c[i] es el código en cadena del píxel de contorno i.
Para eliminar el fallo se hace que c[i] = c[i-1] logrando así unos contornos más suaves.

A continuación, se emplean dos parámetros para cuantificar la regularidad de los contornos:

- *nº cambios totales en el código cadena / longitud del contorno*.

- *longitud del contorno / perímetro del rectángulo del objeto*.

Hasta ahora, las zonas iluminadas y las zonas de sombra se han procesado de forma independiente. Sin embargo, los objetos de interés están compuestos por una zona iluminada y una zona de sombra. Por ello, para reducir el número de falsos positivos, se debe:

- Eliminar las zonas de sombra que no vayan acompañadas de zonas iluminadas. Se pueden hacer excepciones para el caso de sombras pequeñas con contornos muy regulares y sin agujeros.
- Eliminar las zonas iluminadas que no tengan zona de sombra asociada. También se pueden hacer excepciones en casos similares a los del caso anterior.
- Verificar la posición de la zona iluminada con respecto a su sombra, para identificar si se trata de un objeto o un agujero. Hay que recordar que los objetos siempre tienen su sombra en el mismo lado en el que se encuentran en la imagen de sonar de barrido lateral.

En el programa que se ha desarrollado, se permite al usuario la posibilidad de eliminar los objetos sin sombra asociada mediante la activación de una casilla de verificación.

6.3.2.1 Algoritmo implementado

En el apartado anterior se han mostrado una serie de características que se deben extraer para realizar la clasificación de las imágenes, pero no será necesario implementarlas todas si se realizan algunas transformaciones previas en la imagen. Por ejemplo, se puede limpiar la imagen de pequeñas muestras que corresponden a ruido, y de esta forma no hace falta descartarlas posteriormente. Además, los parámetros que se van a establecer como umbrales se pueden mejorar con la experiencia a medida que se van procesando una mayor número de imágenes de sonar de barrido lateral y en el programa implementado se pueden variar sus valores de forma manual.

El resultado de la segmentación de los objetos es una imagen binaria en blanco y negro que tiene valor 1 en los objetos de interés (regiones en blanco), y valor 0 en el fondo (regiones en negro), como se ilustra en la figura 47. Se han utilizado como umbrales los valores 192 para los objetos y 64 para las sombras. Para localizar las

sombras la segmentación se realiza de forma inversa, es decir, las sombras tienen valor 1 (regiones en blanco) y los objetos tienen valor 0 (regiones en negro).

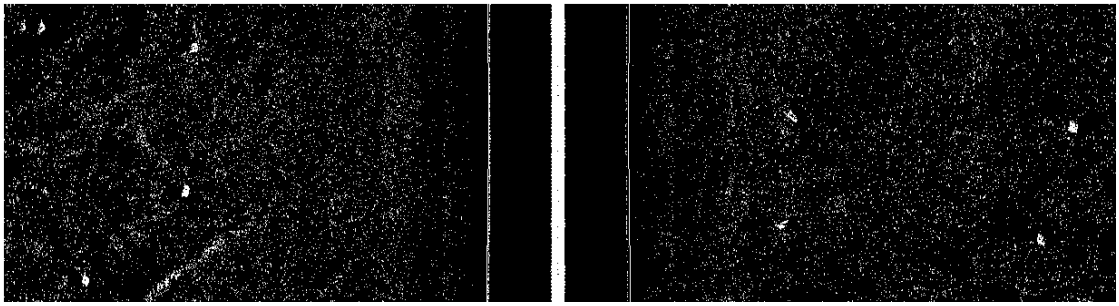


Figura 47. Imagen binaria resultado de la segmentación de objetos

Sólo se va a explicar el proceso llevado a cabo para realizar la segmentación de los objetos, ya que para las sombras se realizaría de forma análoga. Al realizar la segmentación de los objetos aparecen tres problemas:

- los objetos no están completamente rellenos.
- hay gran variedad de regiones con ruido.
- aparecen franjas verticales que se deben eliminar, una franja central ancha y dos líneas finas a cada lado. Dichas franjas corresponden a la zona muerta del sonar de barrido lateral.

Para solventar todos estos problemas, primero se elimina el ruido y los pequeños agujeros de los objetos segmentados. Para ello, se emplean métodos de dilatación y erosión de la imagen, por medio de las funciones *imopen* e *imclose.* A continuación, se procede a la delimitación de todas las regiones presentes en la imagen, siguiendo los límites definidos en el apartado anterior, de este modo se irán descartando las regiones que no sean válidas. Por medio de la función *bwlabel* se consigue el etiquetado de todos los píxeles conectados, de forma que las regiones con píxeles adyacentes tengan el mismo valor. Las diferentes propiedades de las regiones se obtienen con la función *regionprops*. Para una mayor información acerca de estas funciones de Matlab se pueden consultar las referencias [14] y [39]. Seguidamente se pasa a comentar las elecciones que se han tomado a la hora de programar el algoritmo:

- Para eliminar los objetos de tamaño muy grande y muy pequeño, se ha elegido un tamaño de área inferior: *Amin* = 10 píxeles y un tamaño de área superior: *Amáx* = 1500 píxeles.
- Para eliminar las franjas verticales que aparecen en la parte central de la imagen del sonar de barrido lateral se han suprimido las regiones cuya dimensión vertical excede de 0.95*altura de la imagen (variable *Raya*), y cuyo área sea mayor que el 90% del área del rectángulo que encierra las franjas verticales. Además se ha eliminado un área extra a ambos lados de la franja central de 20 píxeles de ancho (variable *extra*).
- Se han eliminado las zonas con *ancho* < *0.15*alto* y con *alto* < *0.15*ancho*.

Para determinar qué regiones son de interés se han creado tres vectores: *imedia, agujeros y cadena*. Están formados por tantos elementos como regiones haya. Además, si el valor de un elemento del vector es 1 indica que esa región cumple las condiciones que se presentan a continuación, y en caso contrario su valor es 0.

- La intensidad media. Se compara la intensidad media del objeto con la intensidad media de su fondo. Así se obtienen las posiciones de todos los píxeles que forman el rectángulo o caja que delimita el objeto (*BoundingBox)* y se hace lo mismo para el fondo. Al poner las dimensiones del fondo, se ha reducido el límite inferior de la anchura a cada lado de 100 píxeles a 50 píxeles, ver figura 45, ya que era un número muy elevado para las imágenes que se han utilizado. Después, se promedian las intensidades de ambas regiones y se almacenan en las variables *im1reg* e *im1fon* para las regiones y fondo de los objetos, y en *im2reg* e *im2fon* para las regiones y fondo de las sombras. De la experimentación con el algoritmo, se puede deducir que una diferencia de 60 píxeles (variable *ui*) es adecuada para discriminar las regiones.
- El número de agujeros. Se tiene que determinar el número de agujeros de cada región. Para ello se usará el número de Euler, que indica el número de objetos conectados en la región (será siempre 1 puesto que ya se han detectado las regiones conectadas) menos el número de agujeros en esos objetos. Por lo tanto, si este número es < -1 para una determinada región, indicará que tiene más de 2 agujeros por lo que dicha región se descartará.

- El contorno. Por medio del uso de la función *fchcode* [14] se obtiene el código en cadena del contorno introducido, por lo que previamente se deben calcular los contornos de todas las regiones por medio de la función *bwboundaries* de Matlab. Una vez obtenido el código en cadena, se cuentan los cambios de dirección producidos (*cdireccion)*. A continuación se calculan los siguientes parámetros:

  paramcca = nº cambios de dirección en los códigos en cadena / *longitud del contorno.*

  paramccb = longitud del contorno / dimensiones BoundingBox.

  Se descartarán las regiones cuyo *paramcca* > 0.8 y cuyo *paramccb* < 0.6.

Para terminar, se construye un vector con unos en las posiciones donde se han seleccionado regiones (*region)*. Después, se muestran todas las regiones que están seleccionadas (objetos y sombras). De esta forma, en la imagen de salida aparecen recuadrados los objetos y sombras. Las dimensiones del rectángulo que delimita cada objeto (BoundingBox) se han agrandado para que abarque toda la región. Para seleccionar únicamente aquellas sombras que posean un objeto cercano, descartando así agujeros o perturbaciones del terreno, se utiliza la función *Centroid* y un diámetro de búsqueda de 1.5 veces el eje máximo de la región que forma la sombra.

Una vez finalizado todo el proceso, se puede comprobar en la figura 48 cómo han sido detectados los objetos hechos por el hombre y sus sombras. Los posibles objetos están recuadrados en rojo y sus sombras en verde. También aparecen falsos positivos que se pueden reducir si se ajustan manualmente los valores de los umbrales.

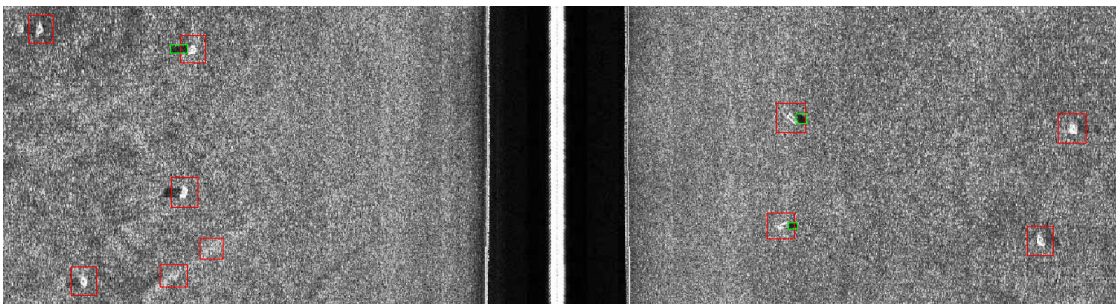


Figura 48. Resultado de la aplicación del filtro de objetos hechos por el hombre

A continuación se muestra el código del algoritmo para el filtro de objetos hechos por el hombre (manmade), que se ha implementado como una función en el software que se ha desarrollado bajo el entorno de Matlab:

```
function [imRGB]=manmade (im, u1, u2, obj)
% Detecta objetos hechos por el hombre
% im: imagen de entrada
% u1 y u2: umbrales
% BoundingBox: rectángulo que delimita el tamaño del objeto
tic;
[h w c]=size(im);

    imRGB=(0);

if c==3
    imcbcr=rgb2ycbcr(im);
    im=imcbcr(:,:,1);
end

tic;
progressbar(0);

%Parámetros
Amin=10;        %Área mínima en píxeles
Amax=1500;      %Área máxima en píxeles
Raya=0.95;      %fracción sobre la cual se detectará la línea vertical
                %del sonar. Raya*height sobre altura de la imagen
extra=20;       %píxeles que se quitan a ambos lados de la línea
                %vertical
uireg=60;       %umbral para la diferencia de intensidades media entre
                %fondo y región
uifon=60;
aumento=10;     % píxeles que se aumentarán cada lado del BoundingBox
                % al representar el resultado

for i=1:w
    im(:,i)=histoganma(im(:,i),u1,u2);
    stopbar=progressbar((i/w)*0.8);
    if stopbar
        return
    end
end

u1=u1/255;
u2=u2/255;
b1=im2bw(im,u1);         %objetos
b2=not(im2bw(im,u2));    %sombras
%%-----------------------------------------------------------------
% Eliminación de ruido
se=strel('disk',1);
b1a=imopen(b1,se);
b1a=imclose(b1a,se);
b2a=imopen(b2,se);
b2a=imclose(b2a,se);
b1a=bwmorph(b1a,'bridge');  %para conectar pequeños huecos
b2a=bwmorph(b2a,'bridge');
b1a=bwmorph(b1a,'fill');  %para rellenar píxeles rodeados de 1's
```

```
b2a=bwmorph(b2a,'fill');
%%-------------------------------------------------------------------
% Etiquetado de las regiones
[L1,num1]=bwlabel(b1a,8);
[L2,num2]=bwlabel(b2a,8);
%%-------------------------------------------------------------------
% Eliminación de regiones según Área
P1=regionprops(L1,'Area','BoundingBox');
P2=regionprops(L2,'Area','BoundingBox');
A1=[P1.Area];
A2=[P2.Area];

   %Eliminar regiones demasiado pequeñas Área < de Amin píxeles
f1=find(A1<Amin);
f2=find(A2<Amin);
        %Se eliminan esas regiones de la imagen segmentada
        for i=1:length(f1)
           L1(find(L1==f1(i)))=0;
        end
        for i=1:length(f2)
           L2(find(L2==f2(i)))=0;
        end
%   figure;imshow(L1)
%   figure;imshow(L2)

    %Eliminar regiones demasiado grandes Área > Amax
f1=[];f2=[];
f1=find(A1>Amax);
f2=find(A2>Amax);
        %Se eliminan esas regiones de la imagen segmentada
        for i=1:length(f1)
           L1(find(L1==f1(i)))=0;
        end
        for i=1:length(f2)
           L2(find(L2==f2(i)))=0;
        end
% figure;imshow(L1)
% figure;imshow(L2)

     %Eliminar franjas verticales
BBox1=[P1.BoundingBox]; %cada 4 elementos es una región
BBox2=[P2.BoundingBox]; %están todos los datos seguidos
f1=[];f2=[];
for i=1:length(BBox1)/4
    Area_caja=BBox1(i*4-1)*BBox1(i*4);
    Area_region=A1(i);
    if find(BBox1(i*4)>=Raya*h) & (Area_region > 0.85*Area_caja)
        f1=[f1 i];
    end
end
for i=1:length(BBox2)/4
    Area_caja=BBox2(i*4-1)*BBox2(i*4);
    Area_region=A2(i);
    if find(BBox2(i*4)>=Raya*h) & (Area_region > 0.85*Area_caja)
        f2=[f2 i];
    end
end
        %Se eliminan píxeles extra a ambos lados de la línea vertical
        %Se eliminan esas regiones y el boundingbox que las encierra
```

```
        for i=1:length(f1)
           L1(find(L1==f1(i)))=0;
               %BoundingBox de la franja
           L1(:,round(BBox1(f1(i)*4-3)):round((BBox1(f1(i)*4-
3)+BBox1(f1(i)*4-1))))=0;
        end
        for i=1:length(f2) %también se eliminan los píxeles extra
           columna=round(BBox2(f2(i)*4-3));
           ancho=round(BBox2(f2(i)*4-1));
           if (columna-extra)<1;columna=columna+extra+1;end
           L2(:,(columna-extra):columna+ancho+extra)=0;
           L1(:,(columna-extra):columna+ancho+extra)=0;
           L2(:,round(BBox2(f2(i)*4-3)):round((BBox2(f2(i)*4-
3)+BBox2(f2(i)*4-1))))=0;
           L2(find(L2==f2(i)))=0;
        end

      % Eliminar regiones delgadas o estrechas
      % ancho < 0.15*alto o alto < 0.15*ancho
f1=[];f2=[];
for i=1:length(BBox1)/4
    if (find(BBox1(i*4-1)<=0.15*BBox1(i*4)) |
find(BBox1(i*4)<=0.15*BBox1(i*4-1)))
        f1=[f1 i];
    end
end
for i=1:length(BBox2)/4
    if (find(BBox2(i*4-1)<=0.15*BBox2(i*4)) |
find(BBox2(i*4)<=0.15*BBox2(i*4-1)))
        f2=[f2 i];
    end
end
        %Se eliminan esas regiones de la imagen segmentada
        for i=1:length(f1)
           L1(find(L1==f1(i)))=0;
        end
        for i=1:length(f2)
           L2(find(L2==f2(i)))=0;
        end
%%-----------------------------------------------------------------
% Se crean y extraen las propiedades de las imágenes anteriores
% ahora sin las regiones que se han descartado
L1(find(L1~=0))=1;
L2(find(L2~=0))=1;
b1b=L1;
b2b=L2;
[Bo1,L1,N1]=bwboundaries(b1b,8);   %se extraen contornos y matriz de
etiquetas
[Bo2,L2,N2]=bwboundaries(b2b,8);   %N indica el numero de regiones sin
los agujeros
num1=length(Bo1);
num2=length(Bo2);

    %Variables con las regiones válidas, tendrán un 1
    %dependiendo de la característica que se haya utilizado
imedia1=ones(1,num1);
agujeros1=ones(1,num1);
cadena1=ones(1,num1);
```

```
imedia2=ones(1,num2);
agujeros2=ones(1,num2);
cadena2=ones(1,num2);

region1=ones(1,num1);
region2=ones(1,num2);

P1=regionprops(L1,'BoundingBox','EulerNumber','Centroid','MajorAxisLen
gth');
P2=regionprops(L2,'BoundingBox','EulerNumber','Centroid','MajorAxisLen
gth');

BBox1=[P1.BoundingBox]; %cada 4 elementos es una región
BBox2=[P2.BoundingBox]; %están todos los datos seguidos
%%-----------------------------------------------------------------
% Comparar intensidad media del objeto con la de su fondo
im1reg=[];  %intensidad media de cada región del objeto
im1fon=[];  %intensidad media de cada fondo del objeto
im2reg=[];  %intensidad media de cada región de las sombras
im2fon=[];  %intensidad media de cada fondo de las sombras

longBbox1=zeros(1,num1);    %Perímetro del BoundingBox
longBbox2=zeros(1,num2);

for i=1:length(BBox1)/4
        % primero se saca del objeto
    longBbox1(i)=2*BBox1(i*4-1)+2*(BBox1(i*4)) ; %2*ancho+2*alto
    x=uint16(BBox1(i*4-3));
    deltax=uint16(x+BBox1(i*4-1)-1);
    y=uint16(BBox1(i*4-2));
    deltay=uint16(y+BBox1(i*4)-1);
    if x==0;x=1;end
    if y==0;y=1;end
    if deltax>w;deltax=w;end
    if deltay>h;deltay=h;end
    cajareg=im(y:deltay, x:deltax);
    im1reg=[im1reg sum(sum(cajareg))/numel(cajareg)];
        % ahora del fondo
    ancho=uint16(max(70,2*BBox1(i*4-1)));
    alto=uint16(max(70,2*BBox1(i*4)));
    x1=uint16(x-ancho);
    x2=x;
    x3=deltax+1;
    x4=uint16(x3+ancho-1);
    if x1==0;x1=1;end
    if x2==0;x2=1;end
    if x3>w;x2=w;end
    if x4>w;x4=w;end

    y1=uint16(y-alto);
    y2=y;
    y3=deltay+1;
    y4=uint16(y3+alto-1);
    if y1==0;y1=1;end
    if y2==0;y2=1;end
    if y3>h;y2=h;end
    if y4>h;y4=h;end

    cajafon1=im(y1:y4,x1:x2-1);
```

```
    cajafon2=im(y1:y2-1,x2:x3-1);
    cajafon3=im(y3+1:y4,x2:x3-1);
    cajafon4=im(y1:y4,x3:x4);
    cajafon=[sum(cajafon1),sum(cajafon2),sum(cajafon3),sum(cajafon4)];

longitud=numel(cajafon1)+numel(cajafon2)+numel(cajafon3)+numel(cajafon
4);
    im1fon=[im1fon sum(cajafon)/longitud];
end

for i=1:length(BBox2)/4
        % primero se saca del objeto
    longBbox2(i)=2*BBox2(i*4-1)+2*(BBox2(i*4)) ; %2*ancho+2*alto
    x=uint16(BBox2(i*4-3));
    deltax=uint16(x+BBox2(i*4-1)-1);
    y=uint16(BBox2(i*4-2));
    deltay=uint16(y+BBox2(i*4)-1);
    if x==0;x=1;end
    if y==0;y=1;end
    if deltax>w;deltax=w;end
    if deltay>h;deltay=h;end
    cajareg=im(y:deltay, x:deltax);

    im2reg=[im2reg sum(sum(cajareg))/numel(cajareg)];
        % ahora del fondo
    ancho=uint16(max(70,2*BBox2(i*4-1)));
    alto=uint16(max(70,2*BBox2(i*4)));
    x1=uint16(x-ancho);
    x2=x;
    x3=deltax+1;
    x4=uint16(x3+ancho-1);
    if x1==0;x1=1;end
    if x2==0;x2=1;end
    if x3>w;x2=w;end
    if x4>w;x4=w;end

    y1=uint16(y-alto);
    y2=y;
    y3=deltay+1;
    y4=uint16(y3+alto-1);
    if y1==0;y1=1;end
    if y2==0;y2=1;end
    if y3>h;y2=h;end
    if y4>h;y4=h;end

    cajafon1=im(y1:y4,x1:x2-1);
    cajafon2=im(y1:y2-1,x2:x3-1);
    cajafon3=im(y3+1:y4,x2:x3-1);
    cajafon4=im(y1:y4,x3:x4);
    cajafon=[sum(cajafon1),sum(cajafon2),sum(cajafon3),sum(cajafon4)];

longitud=numel(cajafon1)+numel(cajafon2)+numel(cajafon3)+numel(cajafon
4);
    im2fon=[im2fon sum(cajafon)/longitud];
end

clear x; clear deltax; clear y; clear deltay; clear cajareg; clear
ancho;
clear alto; clear x1; clear x2; clear x3;clear x4; clear y1; clear y2;
```

```
clear y3; clear y4; clear cajafon1; clear cajafon2; clear cajafon3;
clear cajafon4; clear cajafon;

dif1=im1reg-im1fon;
dif2=im2fon-im2reg; %aquí la región es más oscura al ser sombras
imedia1(find(dif1<uireg))=0;
imedia2(find(dif2<uifon))=0;
%%-------------------------------------------------------------------
% Encontrar número de agujeros
Eun1=[P1.EulerNumber];
Eun2=[P2.EulerNumber];
agujeros1(find(Eun1<(-1)))=0;
agujeros2(find(Eun2<(-1)))=0;

% Características del contorno
cdireccion1=zeros(1,num1);
cdireccion2=zeros(1,num2);
longcon1=zeros(1,num1);
longcon2=zeros(1,num2);

c1=cell(length(Bo1),1);
c2=cell(length(Bo2),1);
for i=1:length(Bo1)
    c1{i}=fchcode(Bo1{i});
    c1a=circshift(c1{i}.fcc, [0 1]);    %se desplaza 1 hacia derecha
    cdireccion1(i)=length(find(c1{i}.fcc ~= c1a));
    longcon1(i)=length(Bo1{i})-1;
end
for i=1:length(Bo2)
    c2{i}=fchcode(Bo2{i});
    c2a=circshift(c2{i}.fcc, [0 1]);    %se desplaza 1 hacia derecha
    cdireccion2(i)=length(find(c2{i}.fcc ~= c2a));
    longcon2(i)=length(Bo2{i})-1;
end
clear c1a; clear c2a;
paramcca1=cdireccion1./longcon1;
paramccb1=longcon1./longBbox1;
paramcca2=cdireccion2./longcon2;
paramccb2=longcon2./longBbox2;

region1(find(paramcca1 > 0.8))=0;  %serán las descartadas
region1(find(paramccb1 >= 0.6))=1; %si ya estaba a 1 bien, y si había
                            %sido descartada se pone como seleccionada
region2(find(paramcca2 > 0.8))=0;
region2(find(paramccb2 >= 0.6))=1;

stopbar=progressbar(7/8);
if stopbar
    return
end
%%-------------------------------------------------------------------
% Eliminar sombras sin objeto cercano
for i=1:num2

    Centroid2=[P2(i).Centroid];
    MajorAxisLength2=[P2(i).MajorAxisLength];
    if(Centroid2(1) < (w/2)) % dependiendo del lado del objeto,
                            % sombra válida o no
        x1=int16(Centroid2(1));
```

```
        x2=int16(Centroid2(1)+1.5*MajorAxisLength2);
    else
        x1=int16(Centroid2(1)-1.5*MajorAxisLength2);
        x2=int16(Centroid2(1));
    end
    if x1<=0; x1=1; end
    if x2>w; x2=w; end
    y1=int16(Centroid2(2)-1.5*MajorAxisLength2);
    y2=int16(Centroid2(2)+1.5*MajorAxisLength2);
    if y1<=0; y1=1; end
    if y2>h; y2=h; end
    % se crean dos vectores uno con las coordenadas x y otro con las y
    %dentro de las cuales se encontrará el objeto buscado
    x=x1:x2;
    y=y1:y2;
        %Ahora se busca entre todos los objetos
    cond=0;    %valdrá 1 si se encuentra un objeto cercano a la sombra
    for j=1:num1
        if region1(j)
            Centroid1=[P1(j).Centroid];
            cond1=ismember(uint16(Centroid1(1)),uint16(x));
            %coordenada x
            cond2=ismember(uint16(Centroid1(2)),uint16(y));
            %coordenada y
            if cond1 && cond2
            % Se habrá encontrado el objeto
                cond=1;
            end
        end
    end
    if cond
        % ha encontrado el objeto por lo que se deja esa sombra
    else
        %no le ha encontrado el objeto por lo que se quita
        region2(i)=0;
    end
end

if obj==1

    for i=1:num1

        Centroid1=[P1(i).Centroid];

        cond=0;
        for j=1:num2
            if region2(j)

                Centroid2=[P2(j).Centroid];
                MajorAxisLength2=[P2(j).MajorAxisLength];
                x1=int16(Centroid2(1)-1.5*MajorAxisLength2);
                x2=int16(Centroid2(1)+1.5*MajorAxisLength2);
                if x1<=0; x1=1; end
                if x2>w; x2=w; end
                y1=int16(Centroid2(2)-1.5*MajorAxisLength2);
                y2=int16(Centroid2(2)+1.5*MajorAxisLength2);
                if y1<=0; y1=1; end
                if y2>h; y2=h; end
                x=x1:x2;
```

```
                y=y1:y2;


                cond1=ismember(uint16(Centroid1(1)),uint16(x));
%coordenada x
                cond2=ismember(uint16(Centroid1(2)),uint16(y));
%coordenada y
                if cond1 && cond2
            % Se habra encontrado el objeto
                    cond=1;
                end
            end
        end
        if cond
        % ha encontrado la sombra por lo que se deja el objeto
        else
        %no le ha encontrado por lo que se quita
            region1(i)=0;
        end

    end

end

% Se dibujan las regiones selecionadas
% aumentando la región del Bounding Box

%axes (figura);imshow(im); axis tight;
%hold on

if c==3
    %imRGB(:,:,1)=im(:,:,1)
    %imRGB(:,:,2)=imcbcr(:,:,2);
    %imRGB(:,:,3)=imcbcr(:,:,3);
    imRGB=ycbcr2rgb(imcbcr);
else
    imRGB=cat(3,im,im,im);
end

for i=1:num1
    if region1(i)   %sólo si está seleccionada
    r= uint16(P1(i).BoundingBox); %el rectángulo o boundingbox
    r(1)=r(1)-aumento;
    r(2)=r(2)-aumento;
    if r(1)==0; r(1)=1; end
    if r(2)==0; r(2)=1; end
    r(3)=r(3)+2*aumento;
    r(4)=r(4)+2*aumento;
    if r(3)>w-r(1)
        r(3)=w-r(1);
    end
    if r(4)>h-r(2)
        r(4)=h-r(2);
    end
    % se dibujan los recuadros de los objetos en la imagen de salida
    imRGB(r(2):r(2)+r(4),r(1)          ,1)=255;  % lado superior
    imRGB(r(2):r(2)+r(4),r(1)+r(3)     ,1)=255;  % lado inferior
    imRGB(r(2)          ,r(1):r(1)+r(3),1)=255;  % lado izquierdo
    imRGB(r(2)+r(4)     ,r(1):r(1)+r(3),1)=255;  % lado derecho
```

```
    imRGB(r(2):r(2)+r(4),r(1)          ,2)=0;  % lado superior
    imRGB(r(2):r(2)+r(4),r(1)+r(3)     ,2)=0;  % lado inferior
    imRGB(r(2)          ,r(1):r(1)+r(3),2)=0;  % lado izquierdo
    imRGB(r(2)+r(4)     ,r(1):r(1)+r(3),2)=0;  % lado derecho

    imRGB(r(2):r(2)+r(4),r(1)          ,3)=0;  % lado superior
    imRGB(r(2):r(2)+r(4),r(1)+r(3)     ,3)=0;  % lado inferior
    imRGB(r(2)          ,r(1):r(1)+r(3),3)=0;  % lado izquierdo
    imRGB(r(2)+r(4)     ,r(1):r(1)+r(3),3)=0;  % lado derecho

%    rectangle ('Position',r,'EdgeColor','r')
%    string=num2str(i);
%     text(r(1)+5,r(2)-8,string,'Color','r',...
%            'FontSize',10,'FontWeight','bold');
    end
end

% hold on
for i=1:num2
    if region2(i)
    r=uint16(P2(i).BoundingBox);
    if r(3)>w-r(1)
        r(3)=w-r(1);
    end
    if r(4)>h-r(2)
        r(4)=h-r(2);
    end
    % se dibujan los recuadros de las sombras en la imagen de salida
    imRGB(r(2):r(2)+r(4),r(1)          ,1)=0; % lado superior
    imRGB(r(2):r(2)+r(4),r(1)+r(3)     ,1)=0;  % lado inferior
    imRGB(r(2)          ,r(1):r(1)+r(3),1)=0;  % lado izquierdo
    imRGB(r(2)+r(4)     ,r(1):r(1)+r(3),1)=0;  % lado derecho

    imRGB(r(2):r(2)+r(4),r(1)          ,2)=255;  % lado superior
    imRGB(r(2):r(2)+r(4),r(1)+r(3)     ,2)=255;  % lado inferior
    imRGB(r(2)          ,r(1):r(1)+r(3),2)=255;  % lado izquierdo
    imRGB(r(2)+r(4)     ,r(1):r(1)+r(3),2)=255;  % lado derecho

    imRGB(r(2):r(2)+r(4),r(1)          ,3)=0;  % lado superior
    imRGB(r(2):r(2)+r(4),r(1)+r(3)     ,3)=0;  % lado inferior
    imRGB(r(2)          ,r(1):r(1)+r(3),3)=0;  % lado izquierdo
    imRGB(r(2)+r(4)     ,r(1):r(1)+r(3),3)=0;  % lado derecho

%     rectangle ('Position',r,'EdgeColor','g')
%     string=num2str(i);
%     text(r(1)+5,r(2)-8,string,'Color','g',...
%            'FontSize',10,'FontWeight','bold');
    end
end

stopbar=progressbar(8/8);
toc;
```

Capítulo 7.
Programa desarrollado con Matlab

Como ya se ha comentado en capítulos anteriores, el programa en el que se han implementado los algoritmos para el realizar el filtrado digital se ha creado bajo el entorno del conocido software Matlab. En este capítulo, primero se van a exponer las razones de la elección de Matlab, después el proceso de compilación del programa y por último se va a describir el funcionamiento del programa que se ha desarrollado.

7.1 Entorno Matlab

Generalmente el procesado digital de imágenes requiere de un extenso trabajo experimental con el software de simulación, y multitud de pruebas con conjuntos de imágenes de muestra. Por esta razón, elegir un software flexible, completo y bien documentado es fundamental ya que va a repercutir tanto en el tiempo de desarrollo como en la portabilidad del programa que se realice.

En esta tesis se ha empleado MATLAB (MATrix LABoratory), un software matemático que ofrece un entorno de desarrollo integrado (IDE: Integrated Development Environment) con un lenguaje de programación propio. Se ha escogido este entorno como herramienta de programación debido a las múltiples ventajas que ofrece frente a otras soluciones. Algunas de estas ventajas son:

- La toolbox de Matlab basada en el procesado de imagen (Image Processing Toolbox). Cuenta con funciones basadas en los algoritmos más conocidos para trabajar con imágenes binarias, trasformaciones geométricas, morfología y manipulación de color, etc. Estas funciones facilitan el diseño de los pasos intermedios hacia las soluciones finales que se pretenden implementar, lo que se traduce en un ahorro de tiempo y esfuerzo al no tener que programar esas rutinas intermedias.
- La herramienta GUIDE (Graphical User Interface Development Environment) con la que cuenta MATLAB para crear interfaces gráficas de usuario. La interfaz gráfica de usuario (GUI) tiene como finalidad proporcionar un entorno visual sencillo a la hora de utilizar el programa desarrollado, sin tener unos conocimientos previos de MATLAB ni de su lenguaje. De esta forma, el usuario final, no necesita poseer conocimientos de programación para utilizar las herramientas desarrolladas. Por otro lado, GUIDE [38], a diferencia de otros lenguajes orientados a GUIs como puede ser lenguaje java o el C++, ha permitido aprovechar las funciones nativas de Matlab relacionadas con procesado de imágenes y de esta forma, reducir el tiempo de desarrollo del software y la velocidad de cálculo del programa.
- La documentación que hay sobre el Matlab. Al tratarse de una herramienta empleada principalmente en universidades y centros de investigación y desarrollo, se dispone de una gran cantidad de manuales y referencias que facilitan su uso de un modo significativo. Así, la actualización y mejora de los programas implementados es más fácil.
- La portabilidad que ofrece. El lenguaje de programación de MATLAB permite utilizar funciones de otros lenguajes de programación de gran difusión como son C, C++ o Fortran. Además en sus últimas versiones permite tanto la exportación de las aplicaciones realizadas a estos lenguajes, como la creación de ejecutables siempre que el equipo disponga de la instalación de MCR (MATLAB Compiler Runtime) un software de libre distribución que se explicará con más detalle en el siguiente apartado.

En resumen, se ha elegido Matlab por su gran versatilidad, potencia y su fácil uso a la hora de realizar un programa. Además, Matlab posee una toolbox dedicada en exclusiva al procesado de imágenes, y un gran número de funciones para la

manipulación de imágenes lo que lo convierten en una herramienta muy interesante a la hora de preprocesar y segmentar imágenes

7.2 Compilación del programa

Una vez programados los algoritmos de los filtros que se van a utilizar en las imágenes de sonar de barrido lateral y creada la interfaz gráfica del programa, se ha realizado una compilación del código generado. Para realizar la compilación del programa, se crea un proyecto de implementación (*Deployment Project*) con la versión R2012a de Matlab, como se puede ver en la figura 49.

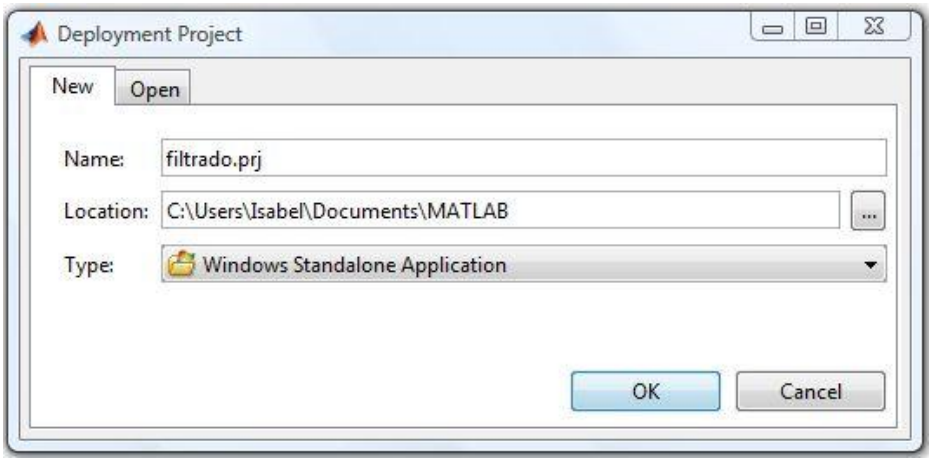


Figura 49. Proyecto de implementación del programa

Una vez creado el proyecto, aparece una ventana que contiene dos pestañas: build y package. Lo primero que se lleva a cabo es la construcción del programa, como se puede observar en la ver figura 50. Para ello, se utiliza la pestaña *build*, en la que se añaden los componentes necesarios para el correcto funcionamiento del programa. Por una parte se agrega la función principal del programa y por otra los recursos compartidos donde se pueden agregar tanto la interfaz gráfica como funciones secundarias, imágenes, archivos de ayuda, etc. Para que comience la construcción del proyecto se pulsa el botón build. Al finalizar el proceso se genera una carpeta con el nombre del proyecto que a su vez contiene dos subcarpetas, una llamada *distrib* y otra llamada *src*. Ambas carpetas contienen una copia del fichero ejecutable (*filtrado.exe*) y un fichero *readme.txt* que explica cómo hay que realizar la instalación del programa. En la carpeta src también se encuentra el fichero *.log* de la compilación.

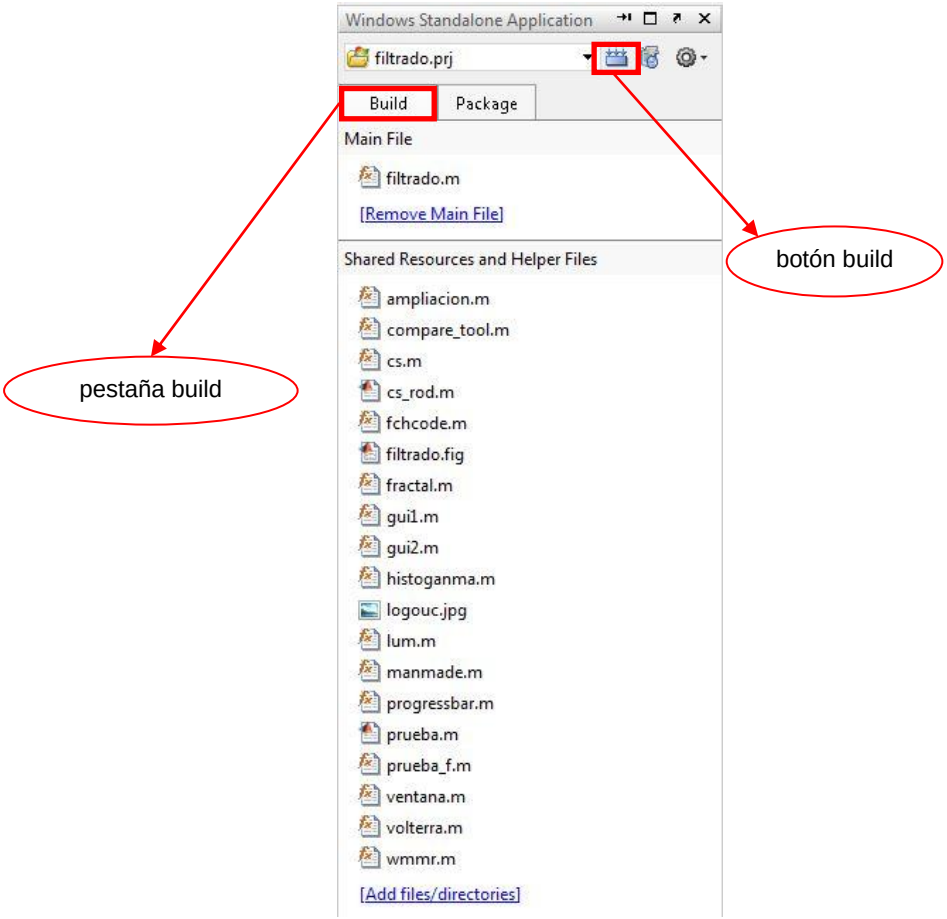


Figura 50. La pestaña y el botón build

El último paso consiste en el empaquetado del programa. En este caso, se utiliza la pestaña *package*, ver figura 51. Por defecto se incluye en el empaquetado, el ejecutable (*filtrado.exe*) y el fichero readme.txt pero si se desea se pueden añadir más ficheros. Suele ser muy conveniente incluir el ejecutable del compilador de Matlab (MCR: Matlab Compiler Runtime). El MCR no es más que un software de libre distribución, diseñado para apoyar la ejecución de programas escritos con Matlab. Contiene las implementaciones de los comandos básicos de bajo nivel. Con cada versión de Matlab viene incluida una versión distinta del MCR. Entonces, al empaquetarlo junto al programa, el usuario puede instalarlo en su ordenador y no necesita siquiera tener instalado ni Matlab ni la misma versión de Matlab con la que se ha creado el programa. En este caso, la versión del MCR es la 7.17 pues es la que viene con Matlab R2012a. Para realizar el empaquetado del programa únicamente hay que pulsar en el botón package.

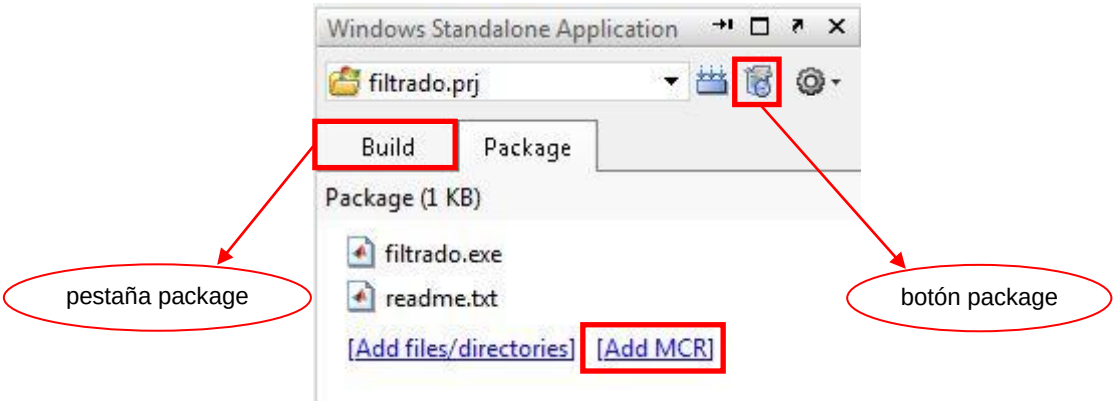


Figura 51. La pestaña y el botón package

En definitiva, al terminar el empaquetado del programa se obtiene un fichero comprimido (autoejecutable o en zip), que una vez descomprimido contiene el ejecutable del programa, el compilador de Matlab (MCR) y un fichero de texto con las recomendaciones para la instalación, ver figura 52. Es decir, el resultado de la compilación del programa es un ejecutable que se puede instalar en cualquier plataforma con sistema operativo Windows con lo que se mejora la portabilidad del programa.

Figura 52. Resultado del empaquetado del programa

7.3 Programa de filtrado

Una vez que se ejecuta el programa de filtrado, aparece una ventana que se puede maximizar para adaptarla al tamaño de pantalla. Como se ilustra en la figura 53, la ventana del programa se puede dividir en tres partes:

(1). *Original*. En este recuadro se carga la imagen original o de entrada que se va a procesar, es decir, es la imagen de partida para realizar el filtrado.

(2). *Filtrado 1*. En este recuadro se muestra la imagen de salida o procesada, que es el resultado de la aplicación del filtro elegido en la imagen original.

(3). *Filtrado 2*. En este recuadro se muestra la imagen de salida o procesada, que es el resultado de la aplicación del filtro elegido en la imagen del filtrado 1 o en la imagen original.

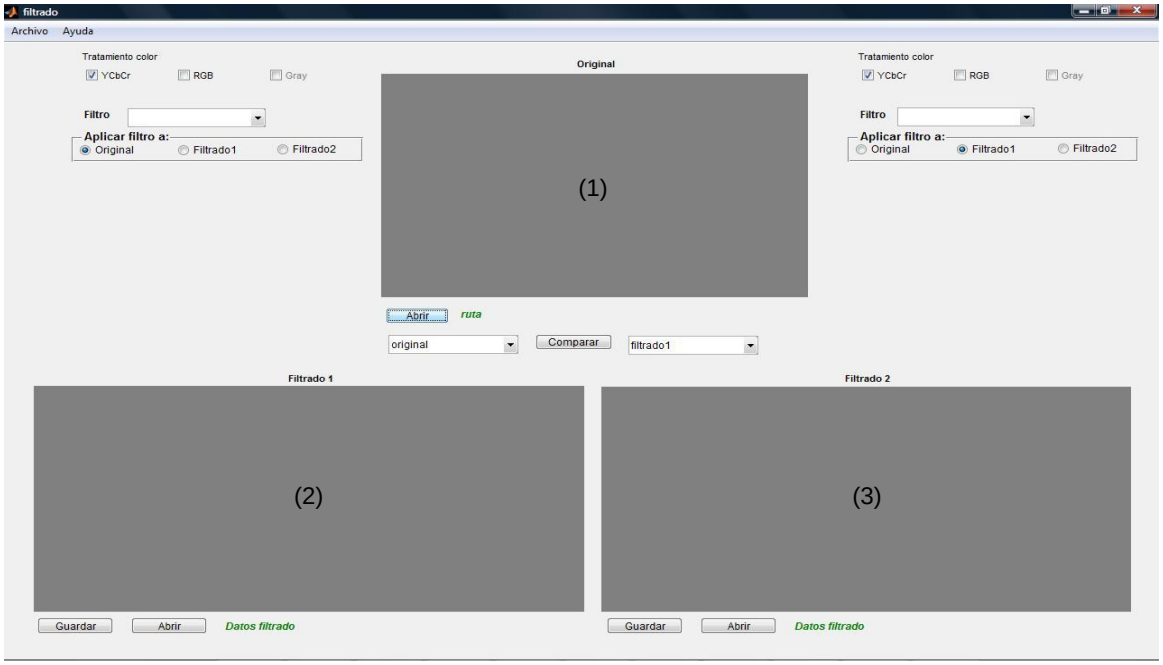


Figura 53. Ventana del programa de filtrado

7.3.1 Abrir y guardar una imagen

Se puede cargar una imagen desplegando el menú *Archivo y pulsando Abrir*, utilizar un atajo de teclado con la combinación de teclas Ctrl + A, pulsando el botón *Abrir* que se está situado debajo del recuadro *Original* y pulsando el botón *Abrir* que se está situado debajo de los recuadros de *Filtrado 1* y *Filtrado 2*.

Figura 54. Diferentes formas de abrir una imagen

A continuación aparece una ventana emergente, ver figura 55, en la que se busca y se elige la imagen que se quiere procesar.

Bajo los recuadros de *Filtrado 1* y *2*, se encuentra el botón *Guardar*, que sirve para guardar la imagen una vez procesada en el formato y lugar que se elija gracias a otra ventana emergente análoga a la que surge al abrir una imagen.

Los formatos de imagen que soporta el programa para abrir imágenes son: bmp, cur, fits, gif, hdf, ico, jpeg, pbm, pcx, pgm, png, pnm, ras, tiff, xwd, pcm, ntf, nsf y rset. Y para guardar imágenes: jpeg, pbm, pgm, png, ppm, ras y tiff.

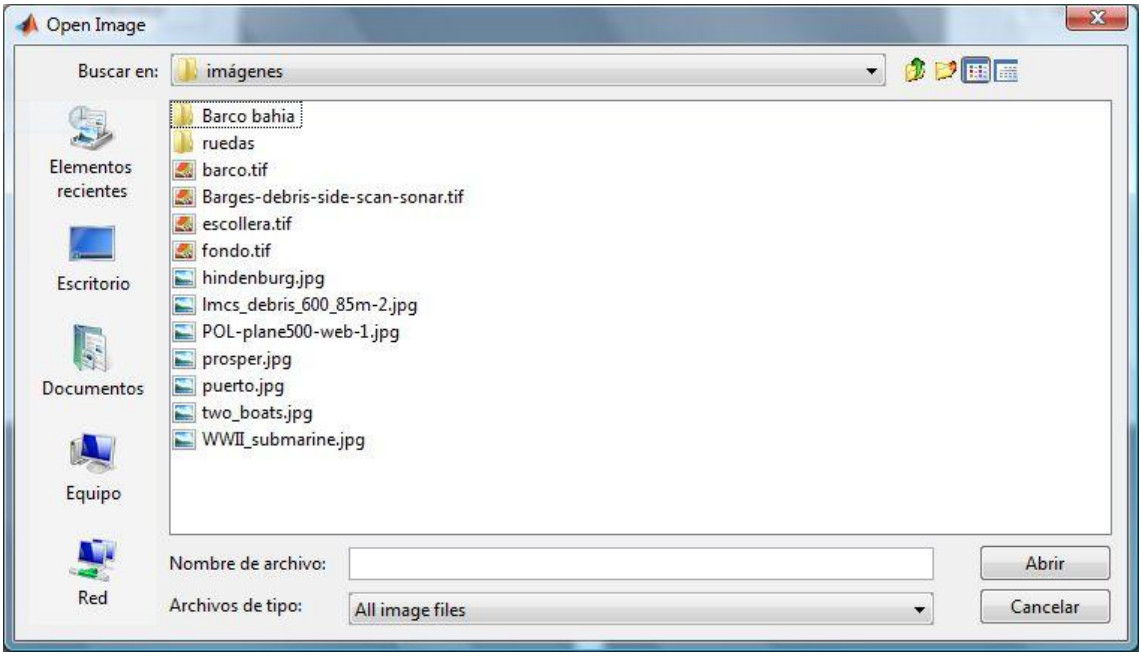


Figura 55. Ventana emergente de búsqueda de imágenes

Ahora se van a estudiar los distintos elementos que componen las funciones del programa. Están duplicadas para poder realizar dos filtrados simultáneos que pueden ser de la misma imagen original para comparar resultados.

7.3.2 Tratamiento del color

Se puede elegir entre realizar el procesado de imágenes en color en RGB o en YC_bC_r, simplemente marcando una casilla u otra, ver figura 56 . Como ya se explicó en capítulos anteriores, utilizar el procesado YC_bC_r reduce considerablemente los tiempos filtrado, al hacerlo sólo sobre una capa (Y) y no sobre las tres como ocurre en el RGB.

Figura 56. Tratamiento del color

Cuando la imagen es en escala de grises se marca automáticamente la casilla *Grey*. En las imágenes en escala de grises el filtrado se realiza sobre su única capa. En la figura 56 se muestran las casillas para el tratamiento del color.

7.3.3 Filtro

Al pulsar sobre esta sección aparece un menú desplegable que permite seleccionar el filtro que se va a aplicar a la imagen original, ver figura 57.

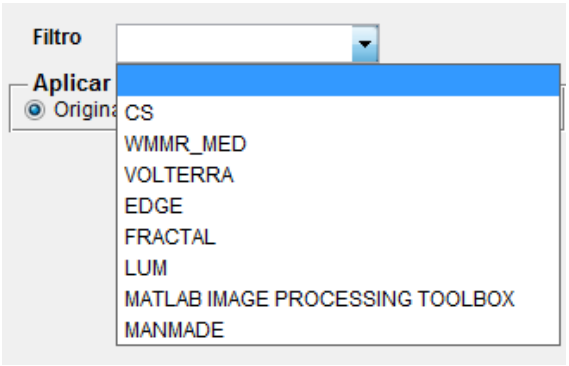


Figura 57. Menú desplegable que contiene los distintos tipos de filtros

En el momento que se elige el filtro, aparecen los controles de los parámetros específicos para dicho filtro. Todos los controles de los parámetros se han programado de forma dinámica, es decir, sólo permiten rangos de valores válidos y en el caso de que haya dependencia entre ellos la modificación de uno varía el rango del otro y viceversa. De esta forma, según el tipo de filtro se pueden elegir y/o variar distintos parámetros. Como ejemplo se ilustra en la figura 58 el filtro CS en el que se puede escoger el tipo de máscara de convolución, así como su tamaño. En este caso también se puede variar el valor de su parámetro J.

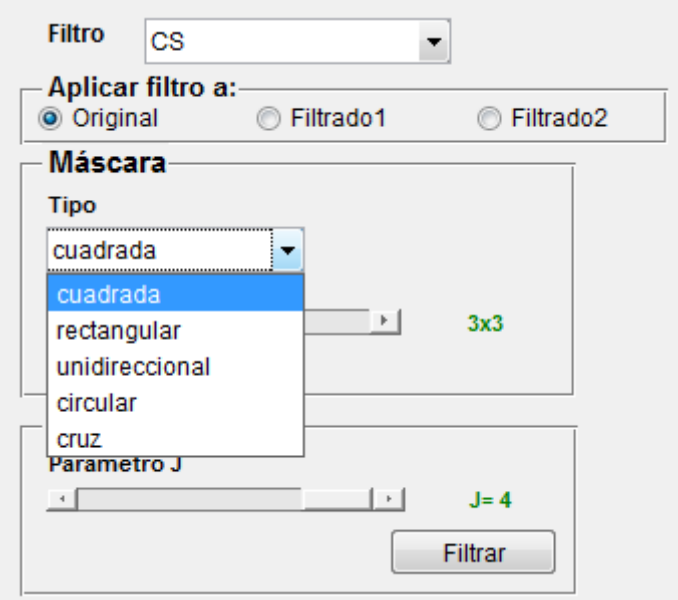


Figura 58. Selección del tipo de máscara del filtro

Además, se puede elegir sobre qué imagen se realiza el filtrado, si sobre la imagen original, la del filtrado 1 o la del filtrado 2, como se puede ver en la figura 59. Por defecto, la aplicación del filtro se realiza sobre la imagen original.

Figura 59. Elección de la imagen a la que se va a aplicar el filtro

7.3.4 Barra de progreso

Mientras se está realizando el filtrado que se ha elegido sobre la imagen, se muestra una ventana con una barra de progreso. Dicha barra de progreso indica de forma aproximada el tiempo que falta para que finalice el proceso que se está realizando en ese momento como se aprecia en la figura 60.

Figura 60. Barra de progreso

En el caso de pulsar sobre el botón cancelar (x) de la ventana de la barra de progreso aparece un cuadro de diálogo preguntando si realmente se desea detener el

proceso, ver figura 61. En caso afirmativo, el filtrado que se esté realizando en ese momento se detendrá inmediatamente.

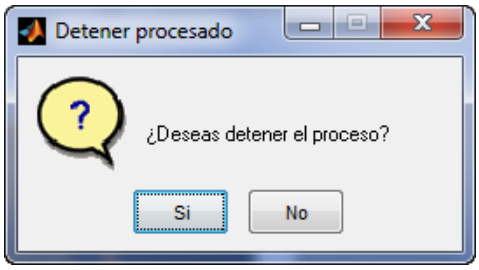


Figura 61. Detención del proceso

7.3.5 Comparar

Con esta herramienta se puede observar de forma más detallada la comparación entre dos imágenes que se han elegido de forma previa, ver figura 62.

Figura 62. Elección de la imágenes que se van a comparar

Tras la elección de las imágenes que se van a comparar se pulsa el botón *Comparar* y se abren dos ventanas, ver figura 63.

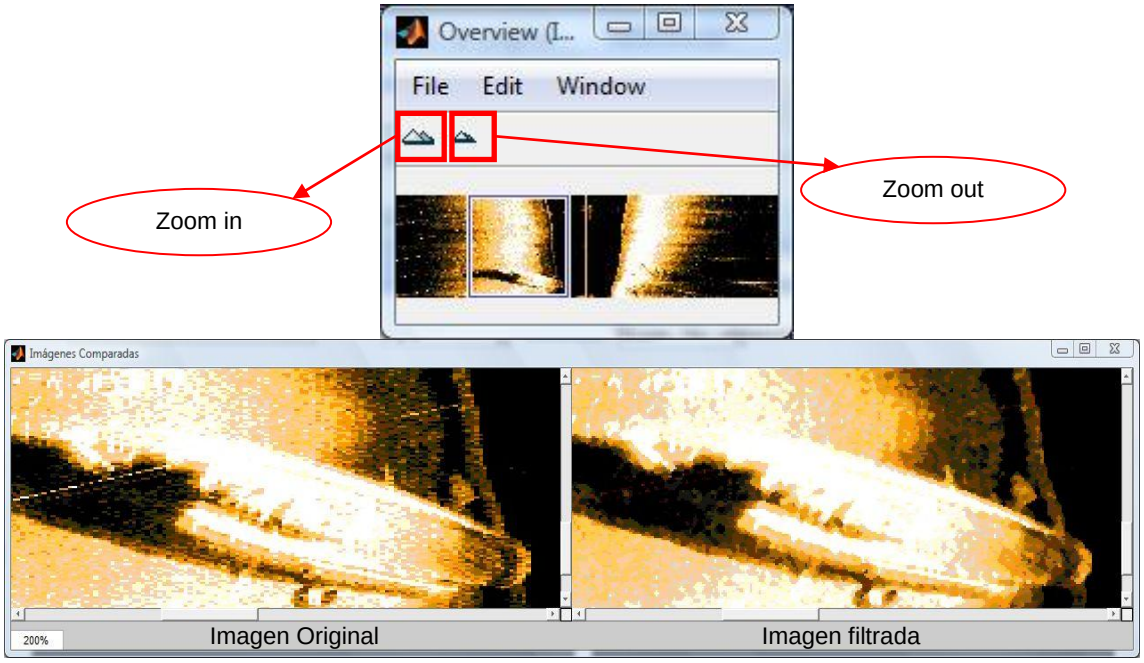


Figura 63. Comparación de dos imágenes utilizando el zoom in

En la ventana más grande se aprecia la comparación de las imágenes con más detalle. Con el rectángulo de la ventana pequeña se controla la zona de la imagen en la que se desea observar la comparación de las imágenes. Y pulsando en los botones de *zoom in* (aumento de la ampliación) y *zoom out* (reducción de la ampliación) se varía el grado de ampliación que se quiere aplicar en las imágenes comparadas.

7.3.6 Barra de menús

En esta barra se encuentran los menús y las opciones que corresponden a cada uno de ellos. Para este programa se ha realizado unos menús muy sencillos ya que lo que interesa es la aplicación de los filtros sobre las imágenes. Así, la barra de menús está formada por el menú *Archivo* y el menú *Ayuda*.

Una vez desplegado el menú archivo contiene las opciones *Abrir* y *Cerrar*. Abrir carga una imagen en el recuadro de la imagen *original*, mientras que Cerrar finaliza el programa.

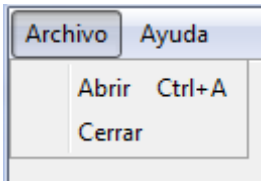


Figura 64. Menú Archivo

El menú *Ayuda* contiene la opción *Acerca de*, que muestra información sobre la versión del programa.

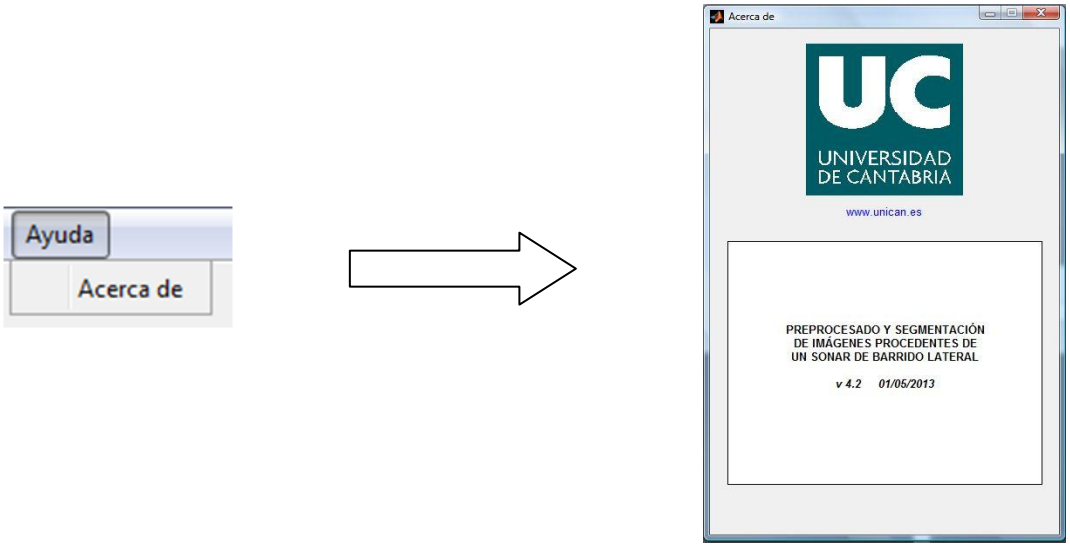


Figura 65. Menú Ayuda

Capítulo 8.
Resultados obtenidos al realizar el filtrado de las imágenes

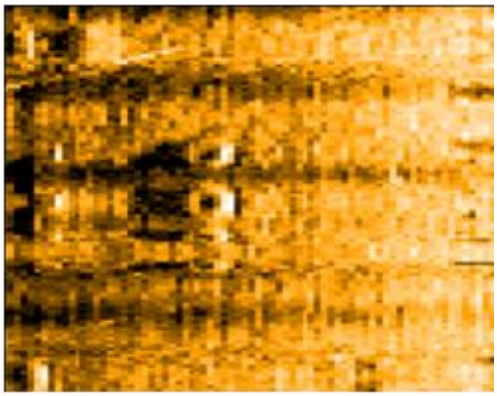
En este capítulo se muestran los resultados de la aplicación de los filtros implementados a una serie de imágenes obtenidas por el sonar de barrido lateral propiedad de la Universidad de Cantabria en sus salidas por la bahía de Santander, así como a imágenes públicas procedentes de sonares de barrido lateral. Debido a que la combinación que se puede hacer con los filtros es muy grande, sólo se van a presentar las aplicaciones de filtros más relevantes o con los que se han logrado mejores resultados.

8.1 Imágenes utilizadas

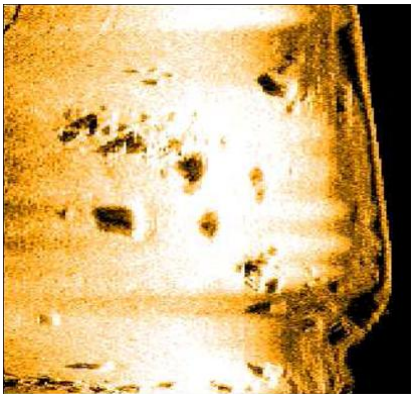
Las imágenes que se han utilizado para probar los filtros se muestran en la figura 66. Como se puede ver, estas imágenes no presentan la forma característica que poseen las imágenes captadas por un sonar de barrido lateral, ya que debido a su gran tamaño, se han reducido para que se pueda observar sin problemas la zona de interés. Las imágenes (1), (2) y (3) se han obtenido con el sonar de barrido lateral en

la Bahía de Santander y las imágenes (4) y (5) son cortesía de la compañía Klein Associates [37].

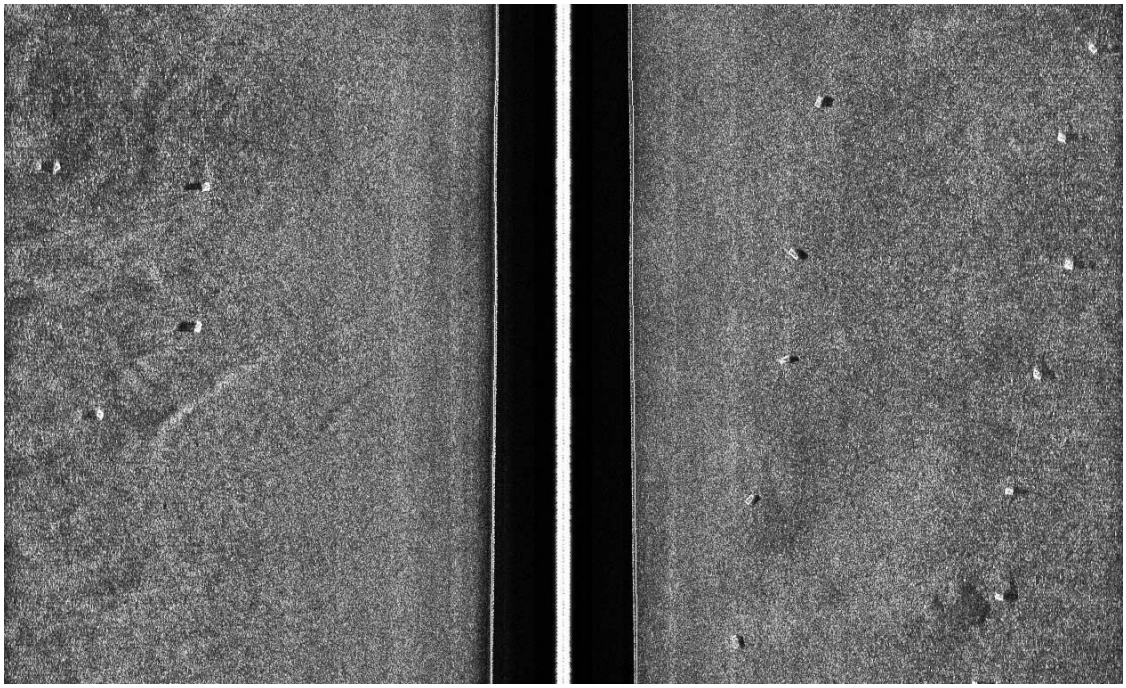
(1) remolcador hundido



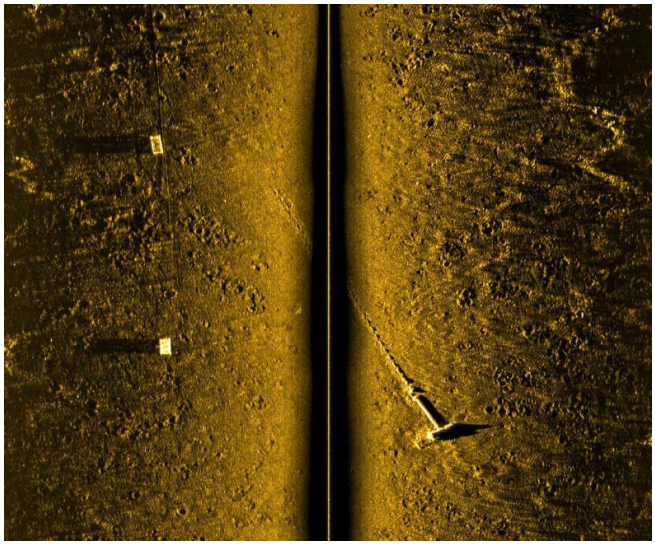
(2) ruedas



(3) bloques de hormigón



(4) quince objetivos



(5) ancla y dos objetivos

Figura 66. Imágenes que se han utilizado para probar los filtros

8.2 Filtrado LUM

Se ha aplicado el filtro LUM con la finalidad de ver cómo se comporta en el suavizado y en el realce de los detalles de la imagen. Para ello, se han ido variando los valores de los parámetros del filtro de dos formas distintas.

8.2.1 Filtrado LUM de suavizado

Para un tamaño de máscara dado, se parte del valor máximo del parámetro L que se deja fijo y se va variando el valor de K. De esta forma, el filtro realiza un suavizado de la imagen como se va a mostrar en la figura 67. Se han empleado los siguientes valores de partida: máscara cuadrada 3x3, L=5 y K=1.

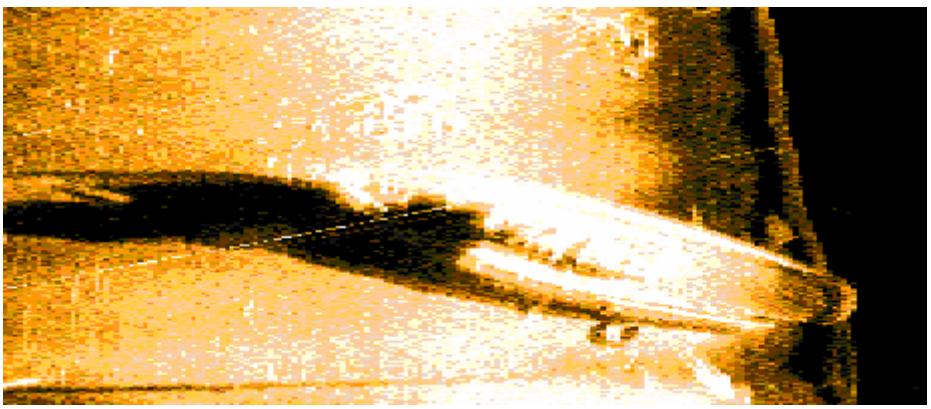
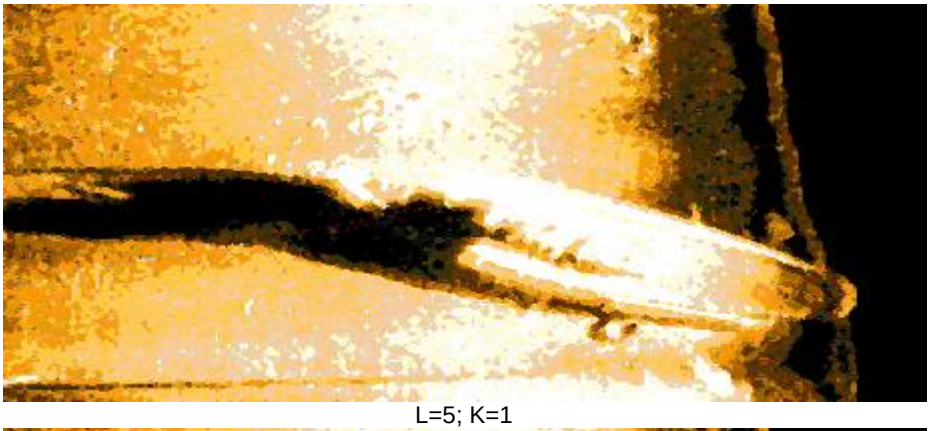
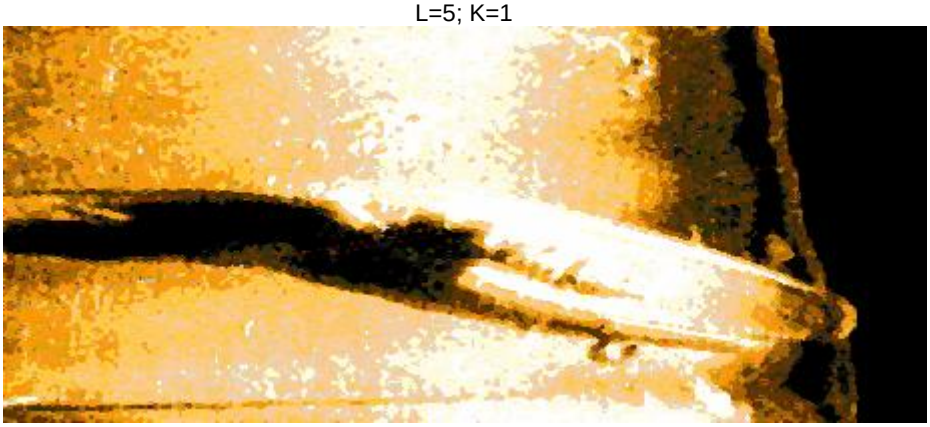


Imagen original



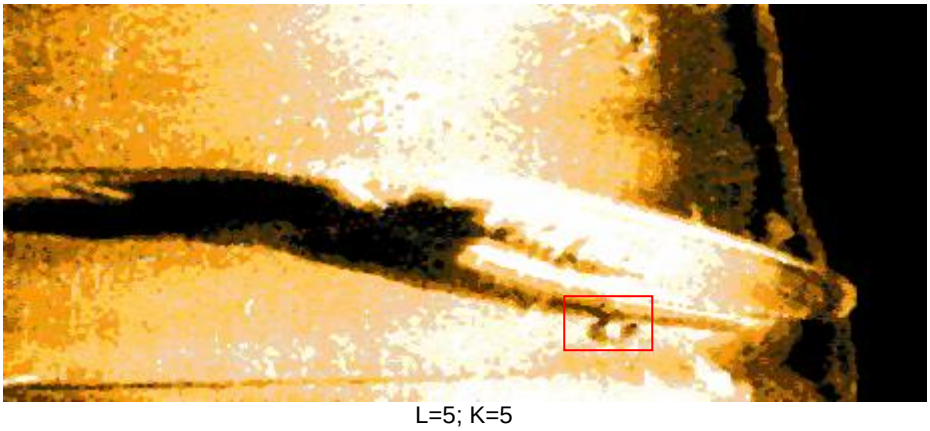
L=5; K=1



L=5; K=2

L=5; K=3

L=5; K=4



L=5; K=5

Figura 67. Filtrado LUM de suavizado

Haciendo zoom en una pequeña zona del remolcador se puede ver en detalle el filtrado LUM de suavizado, tal y como se muestra en la figura 68.

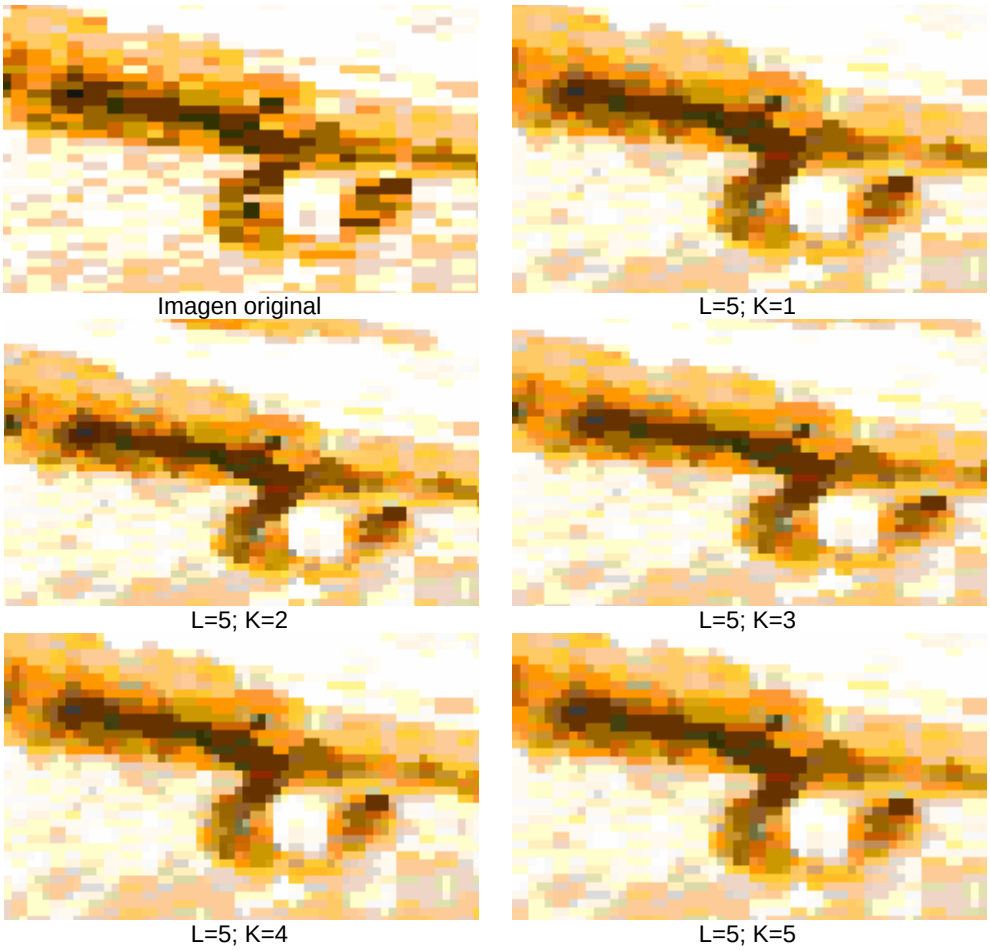


Imagen original | L=5; K=1

L=5; K=2 | L=5; K=3

L=5; K=4 | L=5; K=5

Figura 68. Detalle del filtrado LUM de suavizado

En la vista ampliada de la sección, se ve perfectamente cómo se consigue un suavizado de la zona, aumentando ligeramente al aumentar el valor de K. Este comportamiento responde a las características teóricas del filtro LUM vistas en el capítulo 4.

8.2.2 Filtrado LUM de realce

Ahora se parte del valor mínimo del parámetro K que se va a dejar fijo y se va variando el valor de L. Así, el filtro realiza un realce de la imagen como se puede observar en la figura 69. Se han empleado los siguientes valores de partida: máscara cuadrada 3x3, L=1 y K=1.

De esta forma, se va a conseguir un filtrado que realza los detalles a costa de aumentar ligeramente el ruido presente en la imagen.

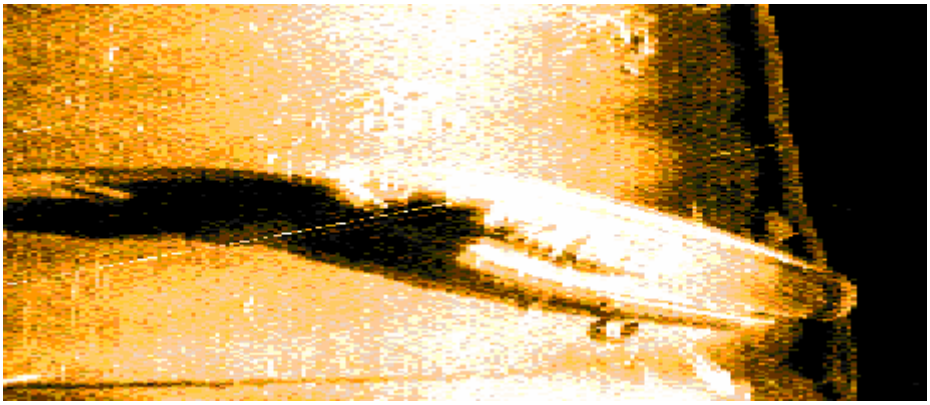


Imagen original

L=1; K=1

L=2; K=1

L=3; K=1

L=4; K=1

L=5; K=1

Figura 69. Filtrado LUM de realce

Ampliando la misma región del remolcador que en el caso anterior, se puede ver en detalle el filtrado LUM de realce, tal y como se muestra en la figura 70.

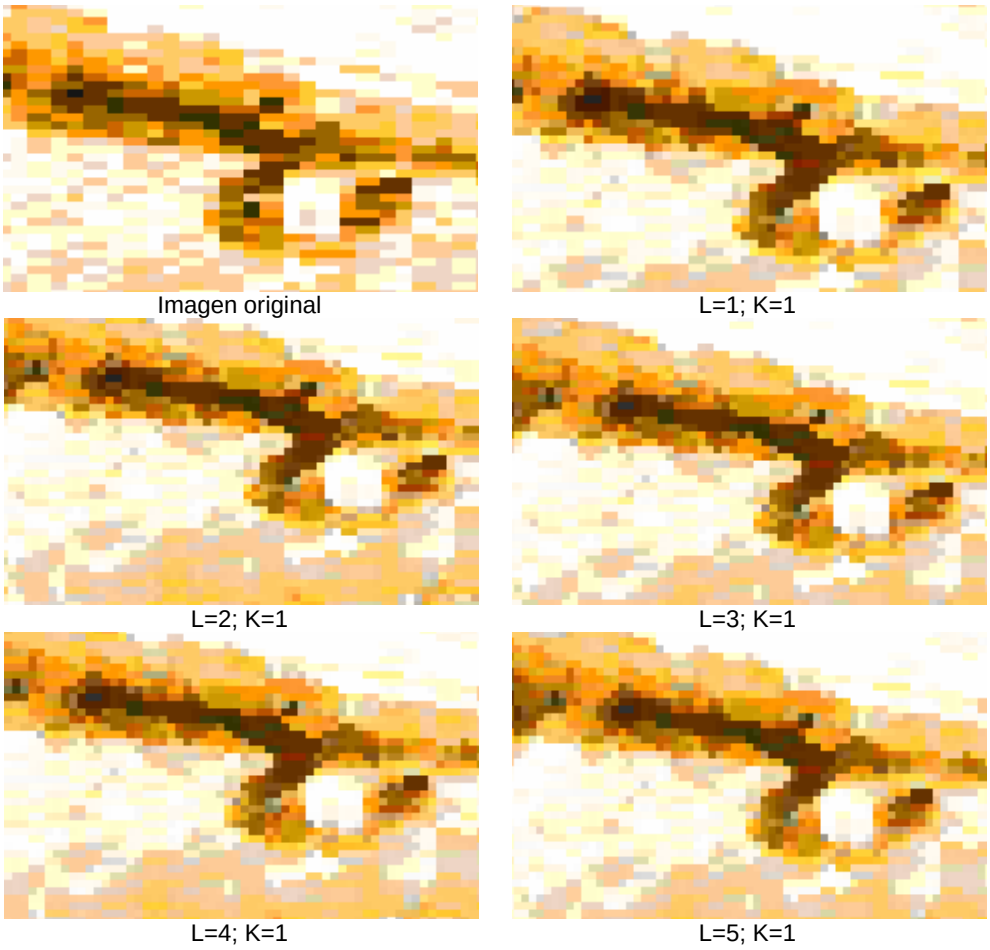


Figura 70. Detalle del filtrado LUM de realce

Se observa cómo con el valor fijo de K=1 y para cualquier valor de L se produce una mejora en los bordes de la imagen, siendo ésta ligeramente superior para un L mínimo, es decir, para L = 1.

Para comprobar la mejora que se ha producido en los bordes del remolcador, se aplica ahora un filtro detector de bordes, en este caso el que utiliza el método de Canny con parámetros U_1 = 0.3 (umbral inferior) y U_2 = 0.5 (umbral superior). En la figura 71 se muestra la imagen completa del remolcador hundido y se aprecia una ligera mejora en la detección de bordes en la imagen previamente filtrada con LUM que en la imagen original.

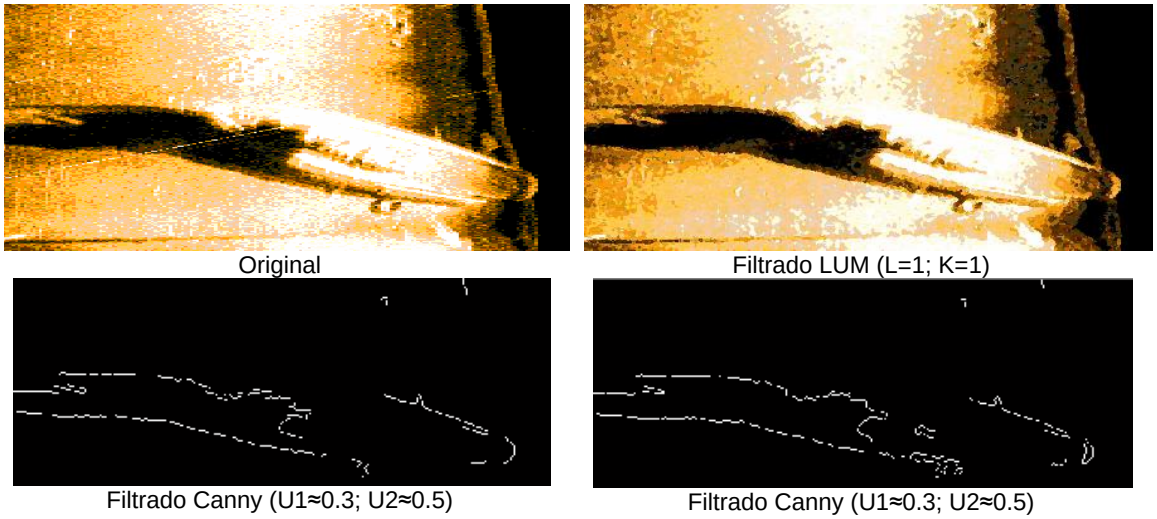


Figura 71. Filtrado Canny aplicado a la imagen original y a la realzada con LUM

En la figura 72 se observa en detalle cómo se detectan los bordes del remolcador de un modo más preciso para la imagen que se ha realzado con el filtro LUM que para la imagen original.

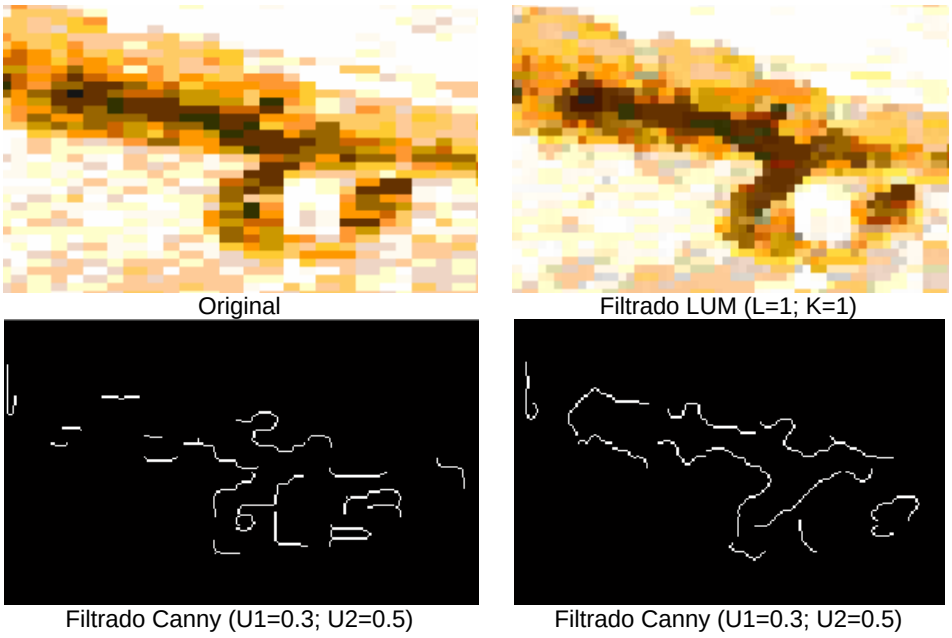


Figura 72. Detalle de filtrado Canny aplicado a la imagen original y a la realzada con LUM

8.3 Filtrado fractal

Se ha aplicado el filtro fractal en la imagen de las ruedas sumergidas con una máscara de tamaño 3x3.

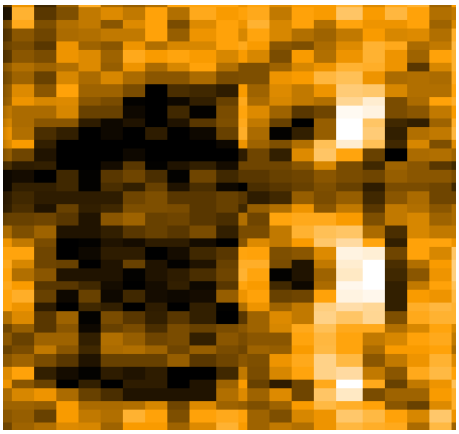


Imagen original

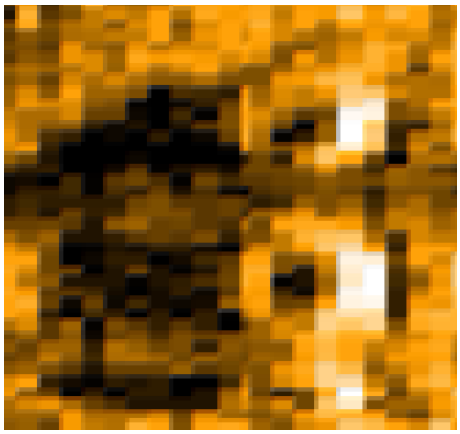
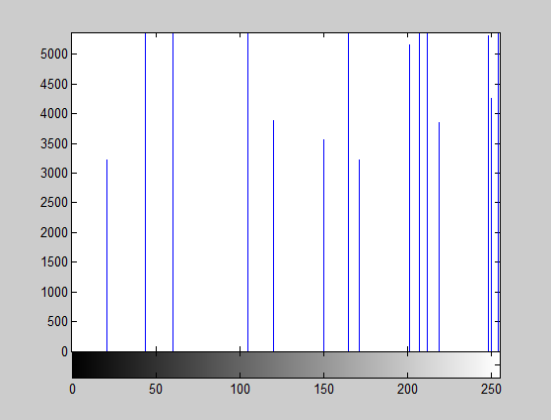


Imagen filtrada

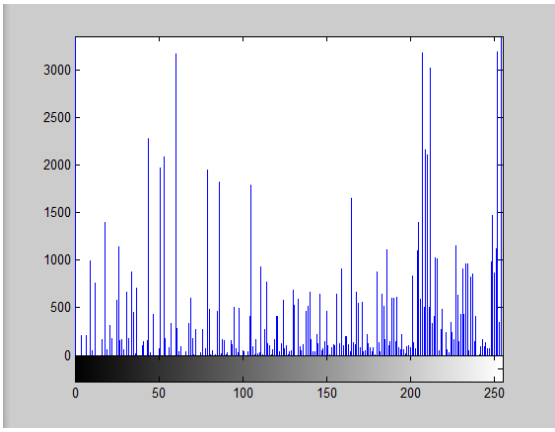
Figura 73. Detalle del filtrado fractal

En la figura 73 se puede ver de forma ampliada la comparación entre la imagen de partida y la imagen filtrada. Se puede observar una ligera mejora del ruido, al suavizarse la imagen. Cabe destacar que al tratarse de un filtro lineal, el tiempo de procesado es menor que cuando se aplica un filtro no lineal.

Al observar el histograma para la capa Y (capa en blanco y negro) de la imagen original y el de la imagen de salida tras el filtrado fractal, ver figura 74, se puede concluir que el filtro fractal aumenta el número de niveles de intensidad, realizando una ecualización de la imagen.



histograma de la imagen original



histograma de la imagen tras el filtrado

Figura 74. Histogramas de la imagen original y de la suavizada con el filtro fractal

8.4 Comparación entre diferentes filtrados no lineales

Tipo de Filtrado		Tamaño de máscara cuadrada		
		3x3	5x5	7x7
	CS	J=1	J=3	J=5
	WMMR-MED	p=1	p=1	p=1
	LUM	L=K=3	L=K=7	L=K=13

Tabla 6. Comparación entre los filtrados no lineales CS, WMMR-MED y LUM

Para una sección de la imagen de los bloques de hormigón se han realizado diferentes filtrados variando el tamaño de la máscara de filtrado de forma cuadrada para intentar encontrar un tamaño de máscara que sea óptima.

En la tabla 6, se observa cómo se acentúa el suavizado de la imagen al aumentar el tamaño de la máscara (independientemente del filtro que se esté utilizando), ya que se procesan más muestras en el algoritmo, con lo que se pueden perder algunos detalles. Por esta razón, el tamaño idóneo de las máscaras de filtrado es el de 3x3.

8.5 Filtrado de detección de objetos

Para realizar el filtrado de objetos hechos por el hombre con el programa desarrollado se utiliza el panel que se muestra en la figura 75. El usuario mediante las barras de desplazamiento puede ajustar los valores que vienen por defecto para los umbrales de detección. También se observa que viene predeterminada la opción de eliminar los objetos sin sombra.

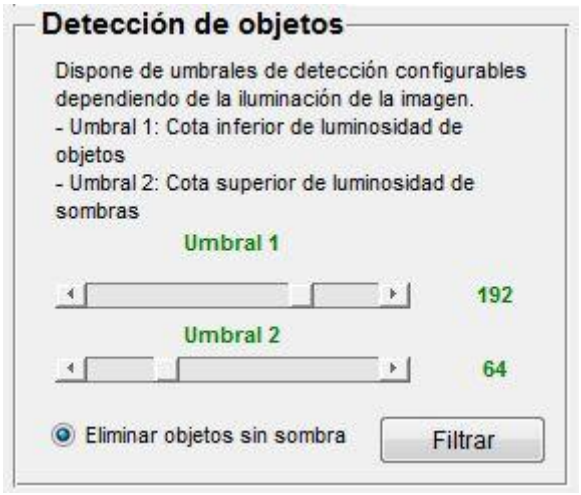
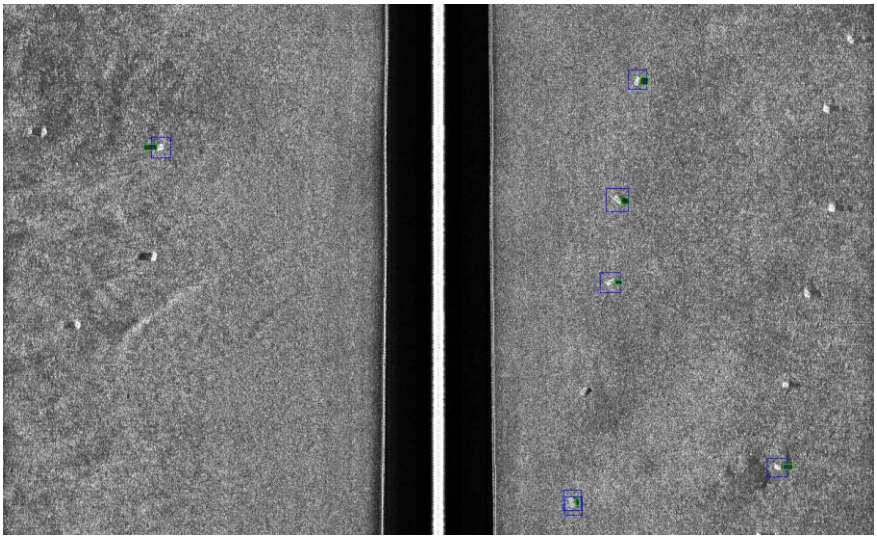
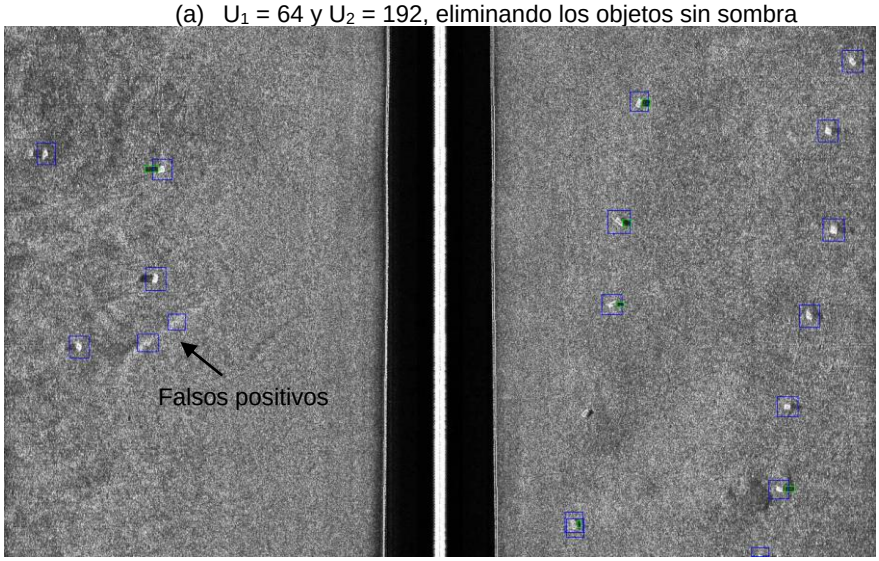


Figura 75. Panel de control del filtrado de objetos hechos por el hombre

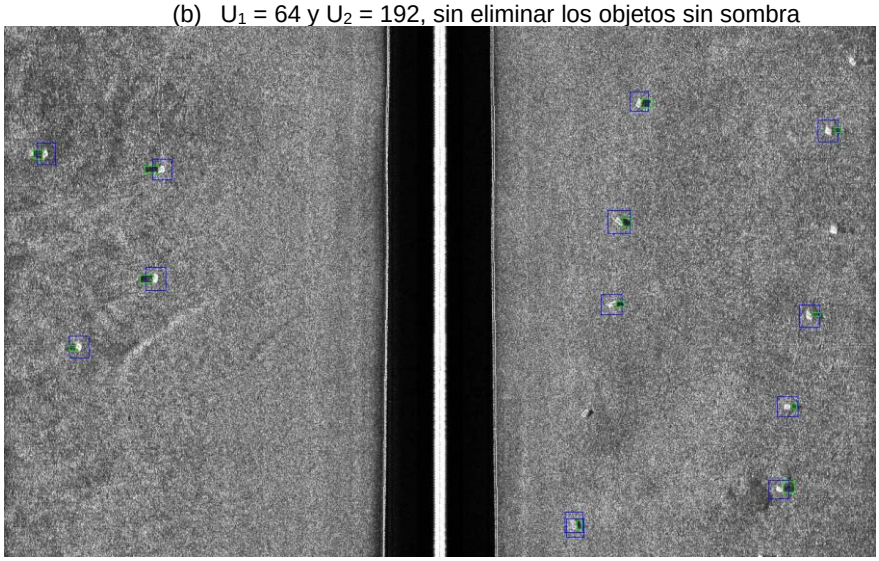
Se ha empleado la imagen de quince objetos para comprobar el funcionamiento del algoritmo de detección. Estos 15 objetivos se corresponden con objetos que pueden estar realizados por el hombre. El filtrado de detección de objetos hechos por el hombre devuelve una imagen en la que los objetos detectados están recuadrados en azul y las sombras en verde como se ve en la figura 76.



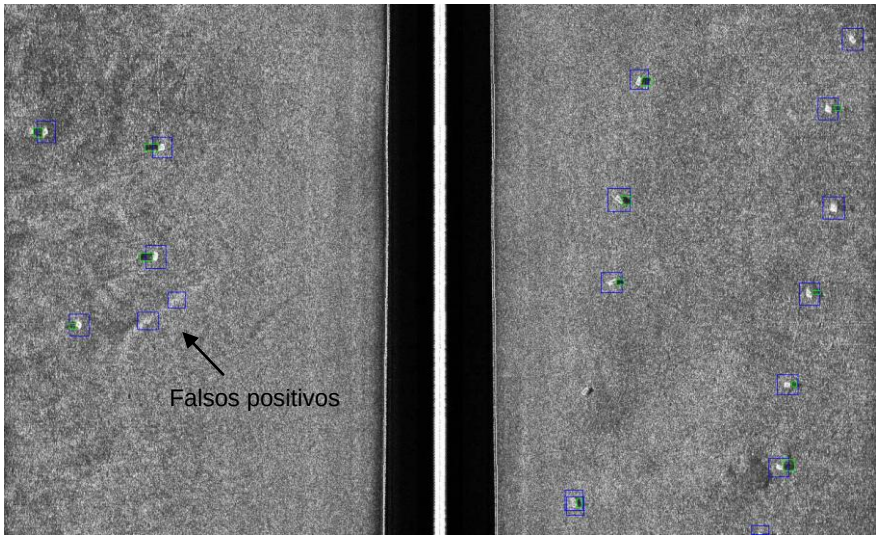
(a) $U_1 = 64$ y $U_2 = 192$, eliminando los objetos sin sombra



(b) $U_1 = 64$ y $U_2 = 192$, sin eliminar los objetos sin sombra



(c) $U_1 = 79$ y $U_2 = 192$, eliminando los objetos sin sombra



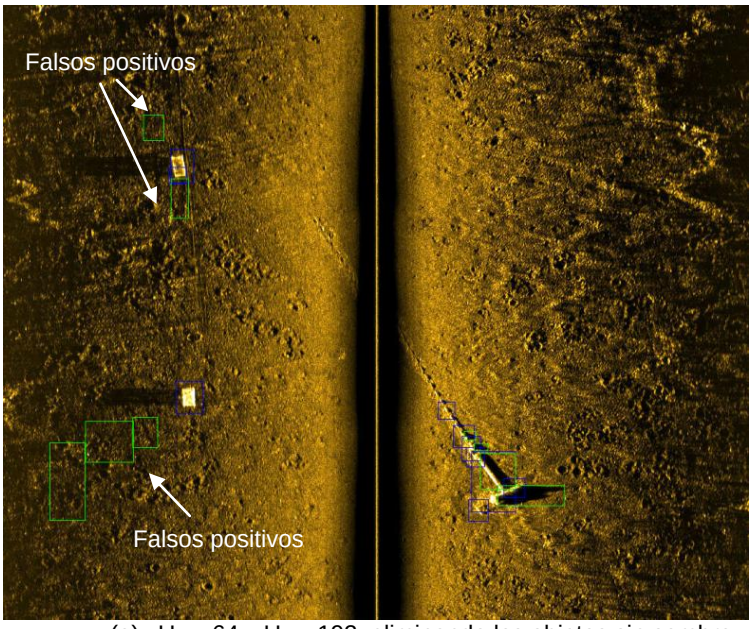
(d) $U_1 = 79$ y $U_2 = 192$, sin eliminar los objetos sin sombra

Figura 76. Filtrado de detección de objetos en la imagen de los quince objetivos

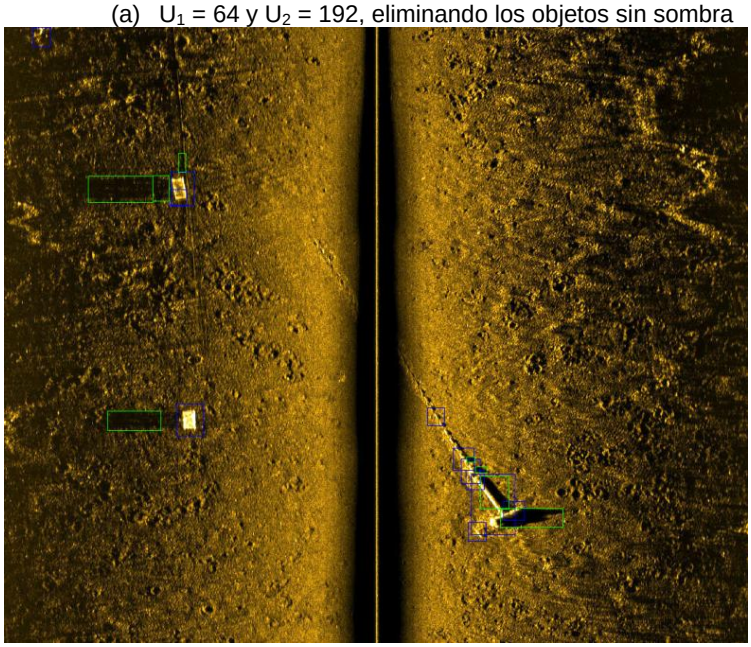
A continuación se van a analizar los resultados obtenidos:

- En la figura 76 (a), se han utilizado los valores de los umbrales por defecto y se han eliminado los objetos sin sombra. Se obtienen unos resultados muy pobres, ya que sólo se han detectado 6 objetos.
- En la figura 76 (b), se ha repetido el filtrado con los valores por defecto pero sin eliminar los objetos sin sombra. Se produce una notable mejora en la detección ya que ahora se detectan 14 objetos, pero aparecen falsos positivos y las sombras más claras no se detectan. Será necesario modificar el umbral de las sombras a un valor mayor.
- En la figura 76 (c), se ha realizado el filtrado con valor umbral para la sombras de 79, el umbral por defecto de los objetos y se han eliminado los objetos sin sombra. Se observa una mejora puesto que ahora se han detectado 12 objetos y no se ha producido ningún falso positivo.
- En la figura 76 (d), se ha repetido el filtrado con los valores umbrales del apartado anterior pero sin eliminar los objetos sin sombra. Se comprueba que se obtiene el mismo resultado que en la figura 76 (b), es decir, aunque se detectan 14 objetos siguen apareciendo falsos positivos.

Ahora se va a probar la detección de objetos con la imagen del ancla y los dos objetivos.



(a) $U_1 = 64$ y $U_2 = 192$, eliminando los objetos sin sombra



(b) $U_1 = 47$ y $U_2 = 192$, sin eliminar los objetos sin sombra

Figura 77. Filtrado de detección de objetos en la imagen del ancla y los dos objetivos

Si se analizan los resultados que se han obtenido:

- En la figura 77 (a), se han utilizado los valores de los umbrales por defecto y se han eliminado los objetos sin sombra. Se han identificado los dos objetos y el ancla, pero aparecen falsos positivos en la detección de las sombras de los objetos. Esto es debido a que en esta imagen concreta las sombras presentan

un contraste más oscuro de lo habitual. Será necesario ajustar el umbral de las sombras a un valor menor.

- En la figura 77 (b), se ha realizado el filtrado con valor umbral para la sombras de 47, el umbral por defecto de los objetos y no se han eliminado los objetos sin sombra. Como en el caso anterior, los dos objetos y el ancla se detectan sin problemas y se produce una mejora considerable en la detección de las sombras de los objetos. Aunque se obtiene un falso positivo pequeño en la parte superior de la imagen, se puede eliminar si se selecciona la opción de eliminar los objetos sin sombra.

La conclusión que se deduce del filtrado de detección de objetos hechos por el hombre es que no se producen falsos positivos si se eliminan los objetos sin sombra. Por otra parte, el número de objetos detectados aumenta si no se eliminan los objetos sin sombra pero a costa de que aparezcan falsos positivos. El ajuste manual de los valores de los umbrales para detectar objetos y sombras variará según la luminosidad u oscuridad que éstos presenten en la imagen. Así, si la sombra de algún objeto es demasiado clara y no se logra detectar, se puede ajustar manualmente el valor umbral de las sombras a un valor mayor del que viene predeterminado. Y para el caso de que la sombra de algún objeto sea demasiado oscura y no se detecta, se puede ajustar de forma manual el valor umbral de las sombras a un valor menor del que viene por defecto.

8.6 Aplicación de varios filtros

En este caso se van a probar varios filtros sobre una misma imagen del fondo marino a la que se le han dejado la rejilla para comprobar con qué filtros se va a mejorar la imagen.

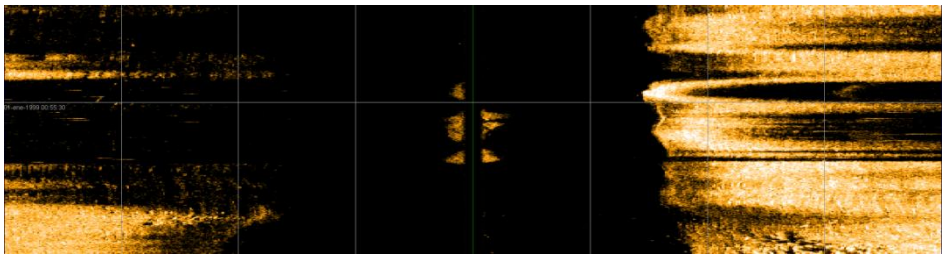
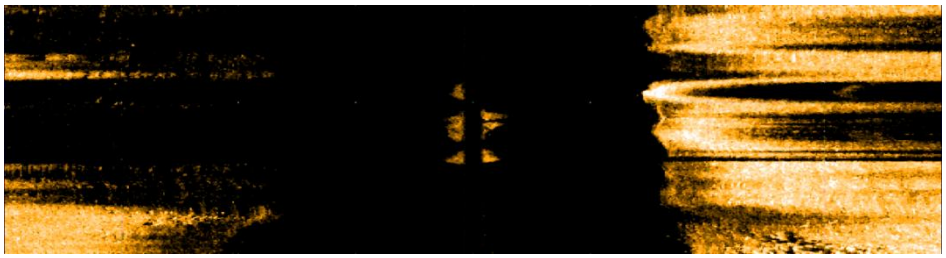
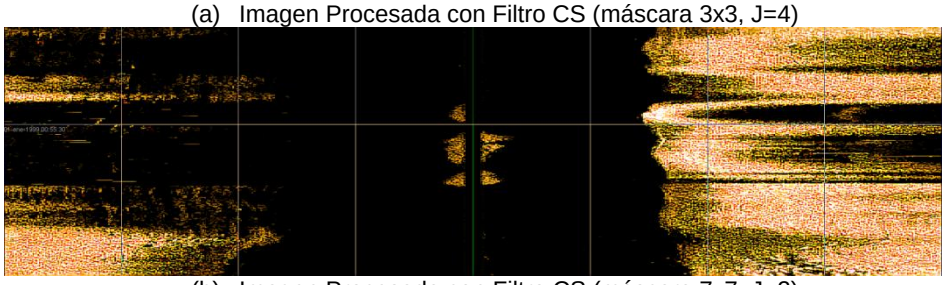


Figura 78. Imagen original con rejilla

8.6.1 Filtrado CS



(a) Imagen Procesada con Filtro CS (máscara 3x3, J=4)



(b) Imagen Procesada con Filtro CS (máscara 7x7, J=2)

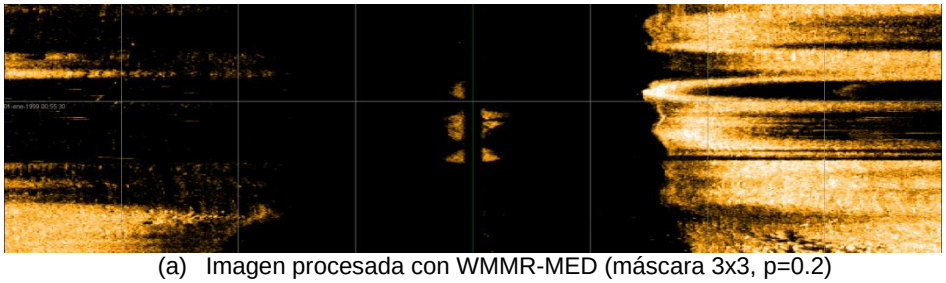
Figura 79. Filtrado CS variando el tamaño de la máscara y su parámetro J

Si se analizan los resultados obtenidos con el filtrado CS:

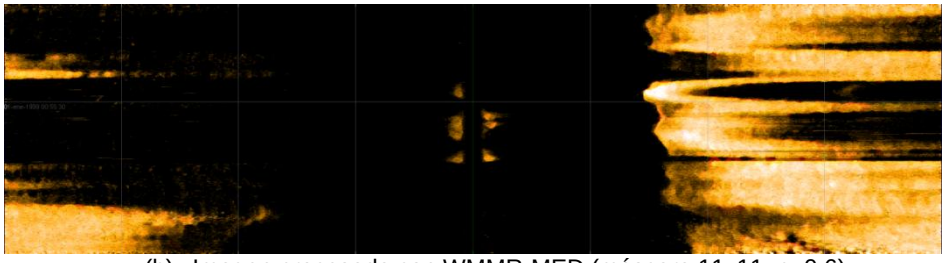
- En la figura 79 (a), con un tamaño de máscara pequeño (3x3) y un valor alto del parámetro J, la rejilla desaparece al producirse un suavizado en la imagen.
- En la figura 79 (b), con un tamaño de máscara mayor (7x7) que el caso anterior y un valor bajo del parámetro J, la rejilla no desaparece, sino que se perfila más ya que en este caso se ha producido un realce de bordes en la imagen.

Se puede concluir que concluir para este caso, los mejores resultados del filtrado CS se obtienen utilizando ventanas pequeñas y un parámetro J elevado, ya que la rejilla desaparece completamente.

8.6.2 Filtrado WMMR-MED



(a) Imagen procesada con WMMR-MED (máscara 3x3, p=0.2)



(b) Imagen procesada con WMMR-MED (máscara 11x11, p=0.6)

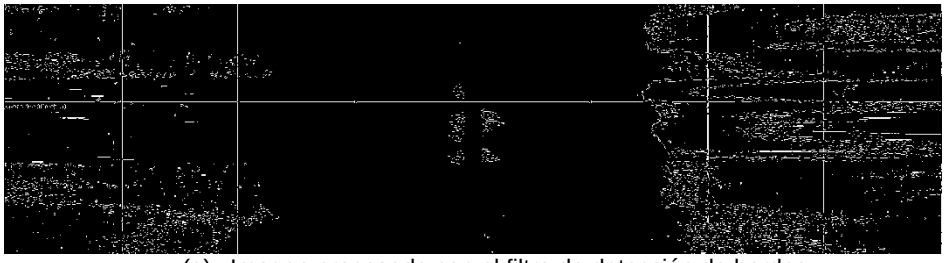
Figura 80. Filtrado WMMR-MED variando el tamaño de la máscara y su parámetro p

Si se analizan los resultados obtenidos con el filtrado WMMR-MED:

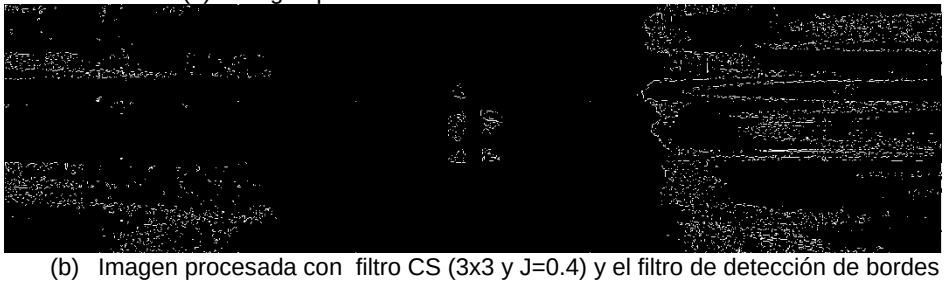
- En la figura 80 (a), con un tamaño de máscara pequeño (3x3) y un valor pequeño del parámetro p, la rejilla no desaparece al producirse un realce de los bordes de la imagen.
- En la figura 80 (b), con un tamaño de máscara grande (11x11) y un valor mayor del parámetro p, la rejilla no desaparece del todo a pesar de haberse producido un suavizado en la imagen

Se puede concluir que concluir para este caso, los mejores resultados del filtrado WMMR-MED se obtienen utilizando ventanas grandes y un parámetro p elevado, aunque la rejilla no desaparece completamente.

8.6.3 Filtrado de detección de bordes



(a) Imagen procesada con el filtro de detección de bordes



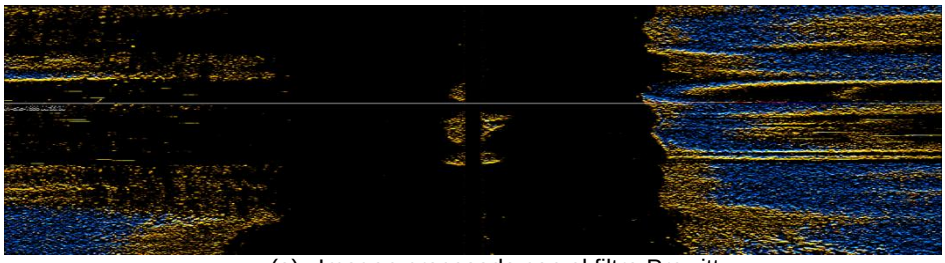
(b) Imagen procesada con filtro CS (3x3 y J=0.4) y el filtro de detección de bordes

Figura 81. Filtrado de detección de bordes

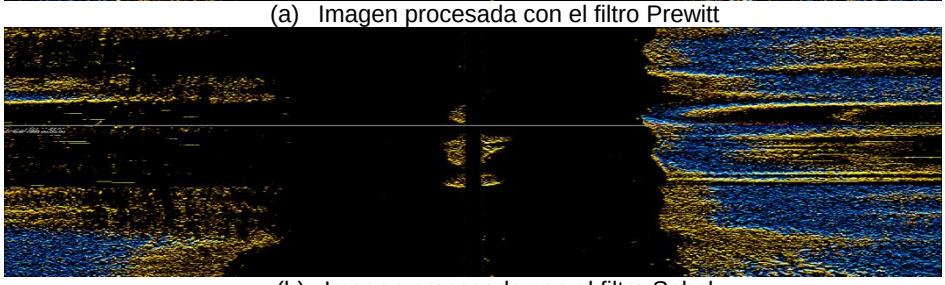
En la figura 81(a) se ha realizado el filtrado de detección de bordes sobre la imagen original. Como se puede apreciar se han detectado los bordes de la imagen incluida la rejilla.

En la figura 81(b) se ha realizado el filtrado de la imagen combinando dos filtros, el CS y el de detección de bordes. Se ha utilizado el filtro CS con los parámetros que hacen desaparecer la rejilla. De esta forma, al aplicar el filtro de detección de bordes, éstos se detectan sin la rejilla.

8.6.4 Filtrado de Prewitt y Sobel



(a) Imagen procesada con el filtro Prewitt



(b) Imagen procesada con el filtro Sobel

Figura 82. Filtrado de Prewitt y Sobel

En la figura 82 se pueden observar el filtrado de la imagen original con los filtros de Prewitt y Sobel. Como se puede ver, tanto la aplicación del filtro Prewitt, como la del filtro Sobel tienen como resultado una imagen casi idéntica, pues sus algoritmos utilizan una máscara con valores muy similares. Su aplicación transforma la imagen realizando un falso 3D que aporta sensación de volumen, lo que puede ser interesante en la prospección de fondos. También se puede observar que la rejilla prácticamente ha desaparecido.

Capítulo 9.
Conclusiones y futuras líneas de investigación

9.1 Conclusiones

Se ha conseguido desarrollar una aplicación software para el procesado de imágenes procedentes de un sonar de barrido lateral. Dicha aplicación realiza un preprocesado de la imagen utilizando para ello filtros espaciales lineales y no lineales, mejorando así las características de la imagen. En este sentido, se reduce el ruido presente en este tipo de imágenes, debido generalmente a fenómenos que se producen durante la captura de las imágenes, y se realzan los bordes, que son un factor clave a la hora de realizar una posterior detección de bordes y objetos.

Cabe destacar que el programa implementado cuenta con una utilidad que permite comparar dos imágenes y utilizar una herramienta de zoom para apreciar mejor los detalles, y comprobar así la mejora producida en la imagen tras el filtrado. El programa también permite aplicar varias veces los filtros a la misma imagen y guardar las imágenes procesadas en varios formatos de imagen.

Una vez que se ha realizado el preprocesado de la imagen, se puede realizar la segmentación de la imagen, con el propósito de realizar una detección. Con el programa que se ha desarrollado se puede hacer tanto la detección de bordes como la

detección de objetos en la imagen. Pero quizá, el aspecto más interesante de este programa sea la detección de objetos cuya forma indica que pueden haber sido realizados por el hombre, ya que es un punto muy importante en la exploración de los fondos marinos. Además el número de falsos positivos en la detección de objetos se reduce de manera considerable al poder ajustar manualmente los valores umbrales y poder eliminar los objetos que no tienen sombra.

Se puede realizar el filtrado de las imágenes tanto en color como en escala de grises. Pero para el filtrado en color se puede elegir hacer el filtrado en YC_bC_r que reduce notablemente el tiempo de cómputo, ya que sólo se aplica el filtrado a una sola capa de la imagen (como si sería en escala de grises) en lugar de a las tres capas como ocurre en RGB.

La interfaz gráfica del programa es de aspecto agradable y de uso intuitivo, ya que se ha programado bajo la filosofía de que resulte fácil de usar. Esto hace que pueda ser utilizado por usuarios de cualquier nivel sin que necesiten tener conocimientos previos en programación.

La portabilidad del programa desarrollado es un factor muy importante ya que puede ejecutarse en diferentes PCs con sólo instalar un compilador que garantiza la perfecta ejecución del programa implementado sin tener instalado el software original con el que fue creado.

9.2 Publicaciones derivadas de la tesis

Durante la elaboración del trabajo de tesis doctoral se han producido diversos artículos para la difusión de los resultados conseguidos, a la par que se investiga en dos proyectos de investigación. Los proyectos de investigación en los que se está colaborando son:

- Navegación de vehículos subacuáticos no tripulados para inspección y asistencia en operaciones y mantenimiento de estructuras sumergidas y tendidos de cables marinos (NAVESUBOM). Este proyecto está financiado por el Ministerio de Economía y Competitividad - Secretaría de estado de investigación, desarrollo e innovación. Proyecto DPI2011-27990.

- Navegación de vehículos submarinos no tripulados en entornos no estructurados (NVSNTENE). Este proyecto está financiado por la Dirección General de Armamento y Material. Proyecto coincidente DN8644.

Los capítulos de libro y revistas en las que se ha publicado como fruto de la actual tesis son:

- Isabel Zamanillo Sainz de la Maza; José María Zamanillo Sainz de la Maza; Elías Revestido Herrero; Francisco Jesús Velasco González. Capítulo 14: Image Easy Sonar: software para el procesado digital de imágenes sonar con aplicaciones docentes. *TIC's para el Aprendizaje de la Ingeniería (TICAI 2010)*, pp. 101-106. Edición promovida por el IEEE de Portugal. Ediciones Politema - Fundación Instituto Politécnico de Porto, noviembre 2011.
- Isabel Zamanillo Sainz de la Maza; José María Zamanillo Sainz de la Maza; Elías Revestido Herrero; Francisco Jesús Velasco González. Sidescan sonar imagery processing software for underwater research and education purposes. *Journal of Maritime Research (JMR)*, vol. 8, no. 1, pp. 87-100, abril 2011.

Los artículos relacionados con la presente tesis que se han publicado y presentado en diferentes congresos son:

- Isabel Zamanillo Sainz de la Maza; José María Zamanillo Sainz de la Maza; Elías Revestido Herrero; Francisco Jesús Velasco González. Software docente para el reconocimiento de objetos sumergidos fabricados por el hombre mediante el procesado de imagen de un sonar de barrido lateral se simple de haz. *XXVII Simposium Nacional de la Unión Científica Internacional de Radio (URSI 2012)*. Libro de resúmenes XXVII Edición de la URSI, Elche, septiembre 2012.
- José María Zamanillo Sainz de la Maza; Isabel Zamanillo Sainz de la Maza; Manuel Solana Salas; Elías Revestido Herrero; Francisco Jesús Velasco González. Software para el reconocimiento de objetos sumergidos fabricados por el hombre basado en el post-procesado digital de imágenes procedentes de un sonar de barrido lateral de simple haz. *VIII Jornadas para el Desarrollo de Grandes Aplicaciones de Red 2011 (JDARE 2011)*. Actas VIII Jornadas, JDARE 2011, pp. 49-61, Alicante, noviembre 2011.

- José María Zamanillo Sainz de la Maza; Manuel Solana Salas; Isabel Zamanillo Sainz de la Maza; Elías Revestido Herrero; Francisco Jesús Velasco González. Software para la detección de objetos sumergidos fabricados por el hombre basado en el post-procesado digital de imágenes procedentes de un sonar de barrido lateral de simple haz. *XXXII Jornadas de Automática*. CD de Actas XXXII Jornadas de Automática, Sevilla, septiembre 2011.
- José María Zamanillo Sainz de la Maza; Manuel Solana Salas; Jonathan González Calderón; Isabel Zamanillo Sainz de la Maza; Elías Revestido Herrero; Francisco Jesús Velasco González. Software para el reconocimiento de objetos sumergidos fabricados por el hombre basado en el procesado de imagen procedente de un sonar de barrido lateral de simple haz. *XXVI Simposium Nacional de la Unión Científica Internacional de Radio (URSI 2011)*. Libro de Actas XXVI Edición de la URSI, Leganés, septiembre 2011.
- José María Zamanillo Sainz de la Maza; Isabel Zamanillo Sainz de la Maza; Elías Revestido Herrero; Francisco Jesús Velasco González. Sidescan sonar imagery processing software for underwater research. *OCEANS 2011 IEEE Santander Conference*. Proceedings of the OCEANS 2011, Santander, junio 2011.
- José Mª Zamanillo, Isabel Zamanillo, Francisco J. Velasco. Software para procesado de imágenes procedentes de un sonar de barrido lateral en exploración submarina. *VII Jornadas para el Desarrollo de Grandes Aplicaciones de Red (JDARE 2010)*. Actas VII Jornadas, JDARE 2010, pp. 155-174, Alicante, octubre 2010.
- Isabel Zamanillo, José Mª Zamanillo, Elías Revestido, Francisco J. Velasco. Image Easy Sonar: Software para el procesado digital de imágenes sonar con aplicaciones docentes. *XXV Simposium Nacional de la Unión Científica Internacional de Radio (URSI 2010)*. CD de actas del Simposium Nacional de la Unión Científica Internacional de Radio (URSI 2010). Bilbao, septiembre 2010.
- José Mª Zamanillo, Rodrigo González Saiz, Isabel Zamanillo, Francisco J. Velasco. Software de procesado de imágenes procedentes de un sonar de barrido lateral en exploración submarina. *XXV Simposium Nacional de la Unión Científica Internacional de Radio (URSI 2010)*. CD de actas del Simposium Nacional de la Unión Científica Internacional de Radio (URSI 2010). Bilbao, septiembre 2010.

9.3 Futuras líneas de investigación

El presente trabajo de tesis ha sentado las bases de la investigación en el procesado de imágenes de sonar pero todavía queda mucho trabajo por hacer. Entre las tareas que se podrían llevar a cabo en un futuro cercano están las siguientes:

- Aumentar la base de datos de las imágenes capturadas con el sonar de barrido lateral para encontrar unos parámetros óptimos en el filtrado. Sobre todo sería interesante optimizar los parámetros de la detección de objetos fabricados por el hombre. De esta forma, la detección se podría llevar de forma más precisa. Al tener una mayor cantidad de imágenes con la que trabajar, se podría probar el uso de redes neuronales para intentar conseguir una detección automatizada.
- Implementar nuevos algoritmos y refinar los existentes para reducir el tiempo de procesado. Se podrían mezclar varios algoritmos para crear un filtro que disminuya el ruido y realce los bordes pero sin distorsionar la imagen. Por ejemplo, se podría desarrollar un algoritmo que sea mezcla del filtro de Kalman y de la media difusa ponderada. También se podrían probar algoritmos de detección que se basen en las características de las texturas de la imagen.
- Montar un sonar de barrido lateral en un vehículo operado de forma remota (ROV). Este vehículo es más robusto y sencillo de manejar que el vehículo autónomo (AUV) con el que se trabaja actualmente. Además, con el ROV se necesita menos personal a la hora realizar su traslado y ponerlo en funcionamiento que con el AUV. Además, como el ROV lleva incorporada una potente cámara de vídeo se podría realizar una comparativa entre las imágenes de sonar obtenidas y las reales, que serviría para comprobar la correcta detección de objetos y su posible identificación.
- Optimizar el procesado de la imagen para que se realice en tiempo cuasi real. Para ello, el programa debería obtener las imágenes casi al mismo tiempo en el que son capturadas por el sonar de barrido lateral. Esta tarea es bastante difícil de implementar puesto que el mismo computador que recoge los datos del sonar, también tiene instalado el programa que se encarga de manejar el vehículo autónomo en el que va montado el sonar de barrido lateral. Si además le añadimos el software de filtrado, el resultado será presumiblemente una

ralentización del funcionamiento de alguno de los programas lo que tendría unas consecuencias nefastas. Parece más viable intentar que el programa de filtrado consiga importar los datos de las imágenes sonar directamente del fichero generado por el programa de sonar de barrido lateral.

Apéndice A.
El ojo humano como instrumento de visión

El sistema óptico utilizado por el ser humano es el ojo, pero como veremos en este apéndice no es un sistema de visión perfecto sino que presenta ciertas limitaciones. En algunos casos nuestro sistema de percepción ocular encuentra diferencias entre elementos completamente iguales y en otras ocasiones tiene las llamadas ilusiones ópticas. Todo ello hace que se deban conocer las características del sistema visual humano ya que es parte esencial en el procesado de imagen ya que es el instrumento que el hombre posee para poder ver y por lo tanto, para poder comprobar si se ha mejorado la imagen. Este capítulo se ha realizado utilizando las siguientes referencias bibliográficas: [25], [30], [31], y [42].

El ojo es el órgano del cuerpo humano que captura la luz y la transforma en un impulso neuronal que manda hasta el cerebro para su procesado. Inicialmente la luz atraviesa la lente del *cristalino* y después incide sobre las células fotorreceptoras de la *retina*, la cual se encuentra situada en la parte posterior del ojo. Dichas células sensibles a la luz son los denominados *conos* y *bastones*. Cuando reciben luz producen impulsos neuronales que transmiten al cerebro a través del *nervio óptico*. Una vez que el cerebro recibe la información comienza a procesarla y a generar sensaciones. Este proceso se conoce como *percepción visual*. En el apartado A.1 se va a explicar en qué consiste la *percepción acromática* en la que sólo se tiene en cuenta el brillo, esto es, la cantidad de energía. Y después, en el apartado A.2 se va a

describir la *percepción cromática* en la que se tiene en cuenta el color, o lo que es lo mismo, la forma de en que la energía se distribuye en distintas longitudes de onda.

En la figura 83, se puede observar un esquema del ojo en el que se aprecia con detalle cómo son los conos (de color rojo, verde y azul) y los bastones (monocromáticos y más delgados que los conos) presentes en la retina.

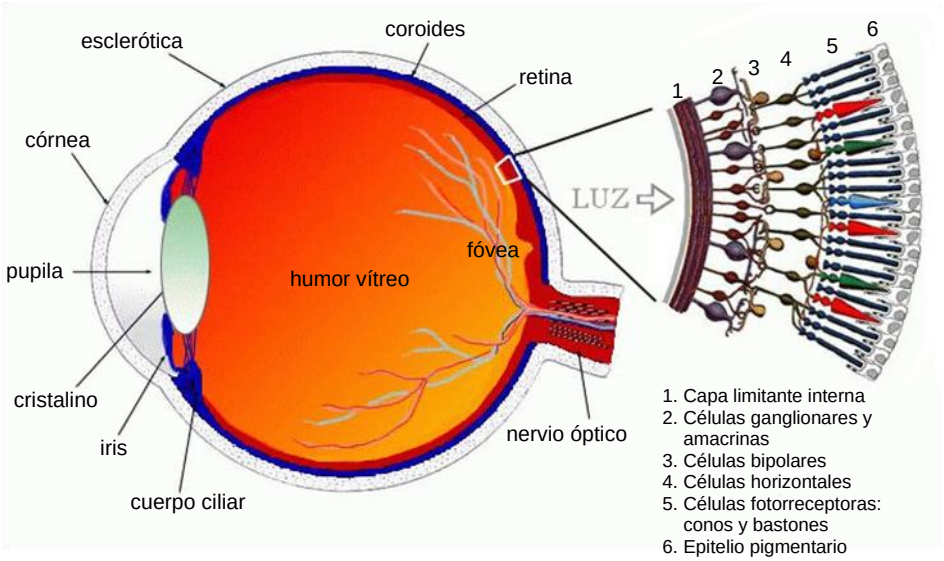


Figura 83. Principales componentes del ojo humano y de la retina

Existe una región de la retina, donde surge el nervio óptico, en la que no hay receptores. Esta zona se denomina *punto ciego*. Como normalmente se utilizan los dos ojos de forma simultánea no se percibe su existencia, ya que el punto ciego de un ojo se suple con la información visual que proporciona el otro ojo.

A.1 Percepción acromática

A.1.1 Los bastones

Los *bastones* se encargan de percibir el brillo de una imagen. Son unas células que se encuentran situadas en la zona periférica de la retina y son capaces de detectar y medir el brillo de los rayos de luz que les llegan. Suele haber del orden de 75 a 100 millones de bastones en cada ojo humano. Tienen forma de bastón y se

encuentran conectados en paralelo con las células nerviosas. Esto hace que la transmisión al cerebro no sea tan directa por lo que no se alcanza una alta resolución. Engloban varios tipos de visión:

- La *visión acromática*: los bastones al no detectar los colores proporcionan una visión en blanco y negro.
- La *visión escotópica*: los bastones reaccionan a bajos niveles de iluminación, por eso los utilizamos de forma casi exclusiva cuando hay poca luz, es decir, durante la visión crepuscular y nocturna. Cuando hay niveles de luz excesivos, los bastones se saturan y dejan de funcionar.
- La *visión periférica*: los bastones se localizan en la zona de la periferia de la retina, siendo ésta una zona de baja *agudeza visual*, o capacidad para percibir los detalles, pero de gran sensibilidad a la luz.

A.1.2 Brillo

La sensación de *brillo o luminosidad* es un concepto que está relacionado con la *sensibilidad a la intensidad luminosa* y con la *inhibición lateral*, que a continuación se van a ver con más detalle.

A.1.2.1 Sensibilidad a la intensidad luminosa

La capacidad de discernir entre un nivel de intensidad de otro se produce gracias a la sensibilidad a la intensidad. El *contraste* se define como la diferencia de intensidad. De este modo, se dice que una imagen tiene un gran contraste si las diferencias de intensidad que tiene son acusadas.

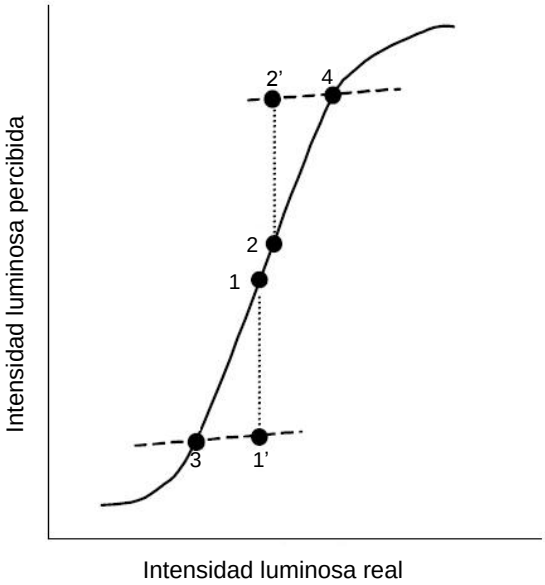


Figura 84. Curva que relaciona la intensidad real y la intensidad percibida por el ojo humano

El hombre puede llegar a diferenciar un rango muy grande de intensidades. Pero la relación entre la intensidad real de la luz reflejada por un pigmento y la intensidad percibida por el ser humano no varía de forma lineal. La variación es una curva como se puede ver en la figura 84. El ojo humano puede distinguir pigmentos con intensidades similares (como los puntos 1 y 2), cuando los objetos que tienen dichos pigmentos se encuentran cerca. Los contrastes muy marcados hacen que esta sensibilidad disminuya. De este modo, cuando se observan pigmentos con intensidades extremas (como los puntos 3 y 4), la capacidad de distinguir entre intensidades próximas decrece. En este caso, las intensidades cercanas 1 y 2 se perciben ahora como lejanas en 1' y 2'.

En el ser humano, la sensibilidad a la intensidad es alta cuando el número de elementos que se comparan es pequeño. Por lo tanto, la sensibilidad a la intensidad es baja cuando aumenta el número de elementos a comparar. Así, cuando el número de intensidades es superior a 24 tonos se pierde la mayor parte de la sensibilidad. Este hecho implica que con 32 o 64 niveles de intensidad sea más que suficiente para representar una imagen en la mayoría de los casos.

Un ejemplo de la sensibilidad a la intensidad luminosa es lo que se conoce como *contraste simultáneo*. Se refiere a la forma en la que los pigmentos de objetos distintos se afectan mutuamente cuando se comparan visualmente. Es un efecto que se produce tanto para pigmentos acromáticos (niveles de gris, blanco o negro) como

para pigmentos cromáticos (los colores). Los cuadrados interiores de la figura 85 tienen la misma intensidad, pero se perciben más oscuros a medida que el fondo se hace más claro.

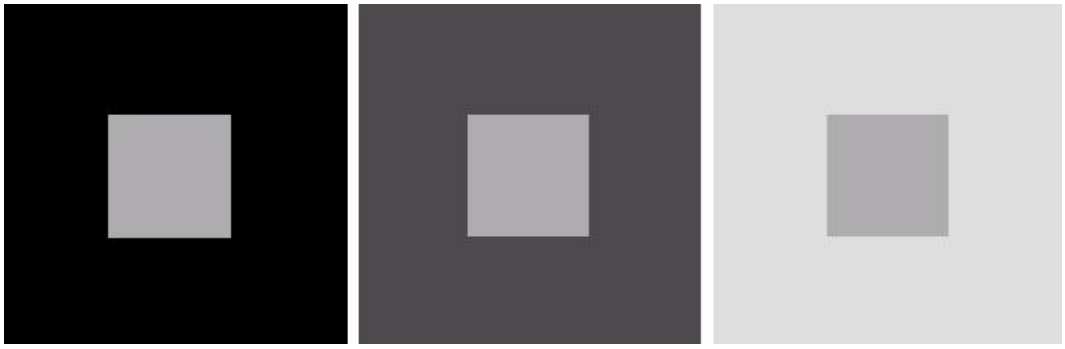


Figura 85. Contraste simultáneo acromático

Asimilación: es el efecto contrario al contraste simultáneo, pigmentos que tienen la misma intensidad se ven más oscuros si el fondo es oscuro y se aprecian más claros si el fondo es más claro. Se da en pigmentos acromáticos y cromáticos.

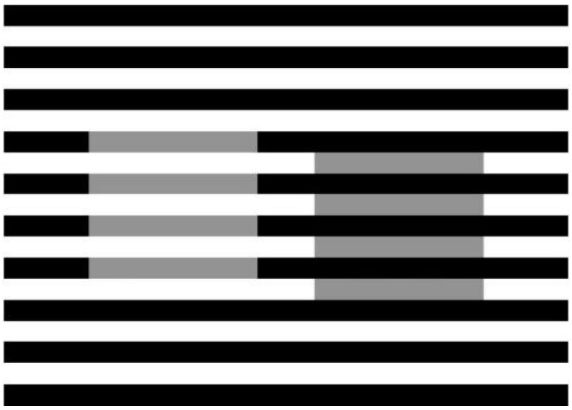


Figura 86. Asimilación acromática

En la figura 86, las barras de gris de la izquierda se ven más claras sobre fondo claro (líneas blancas), que las de la derecha que se perciben más oscuras sobre fondo oscuro (líneas negras). En ambos casos es el mismo gris, aunque el ojo nos engañe.

Como conclusión se puede decir que el ojo humano puede percibir y distinguir entre un amplio rango de niveles de brillo, pero cuando hay muchos niveles involucrados al mismo tiempo se necesita un periodo de adaptación ya que disminuye la precisión.

A.1.2.2 Inhibición lateral

Cuando se detecta un nivel de intensidad las células fotorreceptoras de la retina generan estímulos que recorren un conjunto de neuronas conectadas entre sí formando un circuito neuronal. Como ya se ha mencionado anteriormente, en la retina periférica los bastones se encuentran conectados a las células nerviosas de forma convergente. De este modo se forma un *campo receptor*, en el que puede haber células excitadoras y células inhibidoras. Si el estímulo recae sobre la zona central del campo se produce como respuesta la excitación o activación de las células vecinas. Si por el contrario, el estímulo recae en la zona periférica del campo se tiene por respuesta la inhibición de las células vecinas. Y por último, si el estímulo recae en ambas zonas, la respuesta será mínima. Así pues, la inhibición de las células vecinas produce como resultado perturbaciones en las fronteras de cambio de intensidad.

Las *bandas de Mach* son un claro ejemplo de inhibición lateral. La intensidad de cada una de las franjas verticales de la figura 87 es uniforme. No obstante, se perciben más oscuras por la derecha y más claras por la izquierda. De esta forma, las líneas divisorias de las franjas no parecen estar nítidamente definidas sino que van acompañadas de una banda oscura y una banda clara. Pero tales bandas no existen ya que son una ilusión óptica creada como consecuencia de la inhibición lateral.

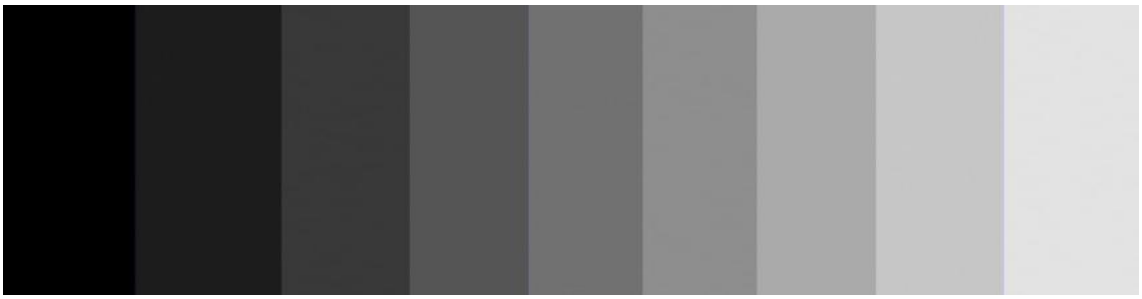


Figura 87. Bandas de Mach

La *rejilla de Hermann* constituye otro ejemplo de inhibición lateral. Al mirar la rejilla en su conjunto en la figura 88, se aprecian puntos grises dispuestos en las intersecciones. Dichos puntos desaparecen si se mira directamente a la intersección, demostrando de esta forma que no son reales. En las intersecciones se produce una inhibición del procesamiento del color blanco, por eso se ven los puntos grises.

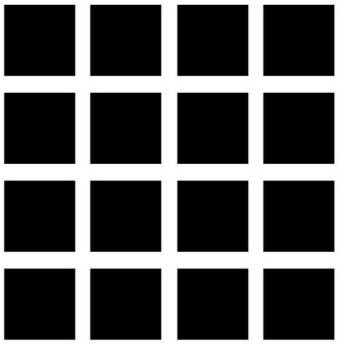


Figura 88. Rejilla de Hermann

La inhibición lateral es un fenómeno que al igual que la sensibilidad a la intensidad tiene gran influencia sobre el hecho de que el brillo percibido no coincida de forma exacta con el brillo físico. Además, es esencial en la detección de contrastes facilitando la posterior extracción de límites de diferente intensidad y de contornos o bordes.

A.2 Percepción cromática

A.2.1 Los conos

Los *conos* son los encargados de percibir los colores. Reciben su nombre por la forma conoidal que tienen en su segmento externo. Existen tres tipos de conos distintos, cada uno según su sensibilidad a la luz de una longitud de onda determinada: roja, verde y azul. Se les denomina de la siguiente forma:

- *Conos L (Large)*: son más sensibles a las radiaciones con longitud de onda larga (rojo ~ de 620 a 750 nm).
- *Conos M (Medium)*: son más sensibles a las radiaciones con longitud de onda media (verde ~ de 495 a 570 nm).
- *Conos S (Short)*: son más sensibles a las radiaciones con longitud de onda corta (azul ~ de 450 a 495 nm).

En la figura 89 se muestra la parte del espectro electromagnético que es visible para el ojo humano, también conocido como luz visible, con sus correspondientes longitudes de onda expresadas en nm.

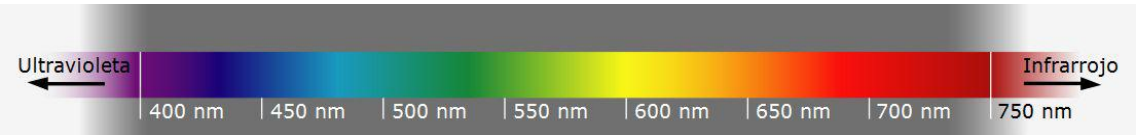


Figura 89. Espectro electromagnético visible para el hombre (luz visible)

Según la estimulación que tengan estos tres tipos de conos envían la información al cerebro que se encarga de interpretar los colores. La mayoría de los conos se encuentran localizados de forma exclusiva en la *fóvea*, que es una pequeña depresión en la zona central de la retina (ver figura 83). En la fóvea suele haber del orden de 6 a 7 millones de conos. Están conectados con las células nerviosas de forma 1:1, esto hace que la transmisión con el cerebro sea más directa por lo que se alcanza mayor resolución, o lo que es lo mismo, mayor nitidez visual. Se caracterizan por varios tipos de visión:

- La *visión cromática*: los conos al detectar los colores proporcionan una visión en color.
- La *visión fotópica*: los conos reaccionan a altos niveles de iluminación, por eso los utilizamos de forma mayoritaria cuando hay mucha luz, es decir, durante la visión diurna.
- La *visión central*: los conos, en su mayoría, se localizan en la zona central de la retina, siendo ésta una región de gran agudeza visual, es decir, de gran captación de detalles.

La *sensación de color* que el hombre percibe está directamente relacionada con la energía que una onda electromagnética tiene a diferentes longitudes de onda. Para explicar esta relación se definen los conceptos de tono y saturación.

A.2.2 Tono, saturación y brillo

Todos los colores pueden describirse en función de tres características principales: el tono, la saturación y el brillo. Cuando se define un color utilizando estos tres atributos se identifica un color de manera específica, pudiéndose distinguir de cualquier otro.

Los colores que el ser humano ve habitualmente no proceden de fuentes monocromáticas. Esto significa, que no se corresponden con una única longitud de

onda. De este modo, la luz blanca se obtiene con la suma de múltiples longitudes de onda y los tonos magenta se obtienen como una mezcla de las longitudes de onda de los colores rojo y azul.

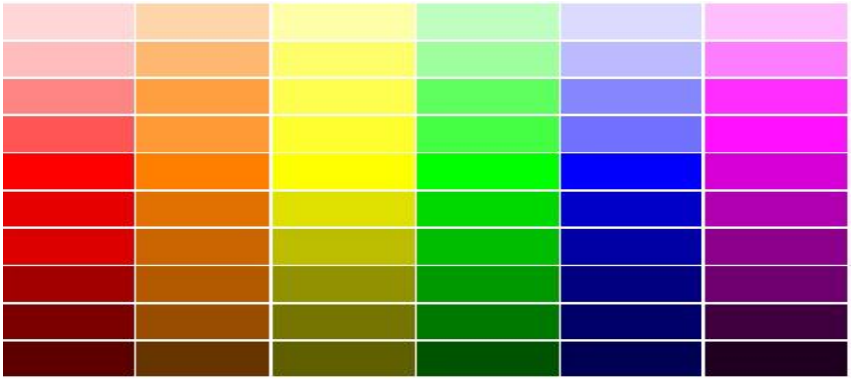


Figura 90. Distintas gamas de tonos del rojo, naranja, amarillo, verde, azul y magenta

El *tono* es un concepto que proviene de las activaciones de los tres tipos de conos al incidir sobre ellos la luz. La determinación del tono depende de la longitud de onda dominante que se encuentra en la muestra de color que se está observando. Así, para un mismo color existen multitud de tonos de ese mismo color dependiendo de las proporciones en las que se mezclen los colores, ver figura 90. El ojo humano es capaz de distinguir entre 125 y 150 tonos distintos cuando están próximos entre sí. Pero dicha capacidad se pierde si están separados espacialmente.

La *saturación* mide la proporción entre la longitud de onda dominante y el resto de las longitudes de onda. Cuando un tono está muy saturado, o lo que es lo mismo, tiene una alta intensidad, presenta un color muy intenso y fuerte. Por el contrario cuando un tono está poco saturado, tiene baja intensidad, y presenta un color muy apagado. Se suele medir en %, siendo el 100% el valor máximo de saturación (color más intenso) y el 0% el valor mínimo de saturación (gris, carencia de tono). Por ejemplo, como se ve en la figura 91, un tono verde saturado al máximo es percibido como un tono verde muy vivo. Con saturación intermedia se percibe como un tono de verde normal, con una saturación baja se observa como un tono verde apagado y con la saturación mínima se ve el gris, es decir, no hay tono de verde.

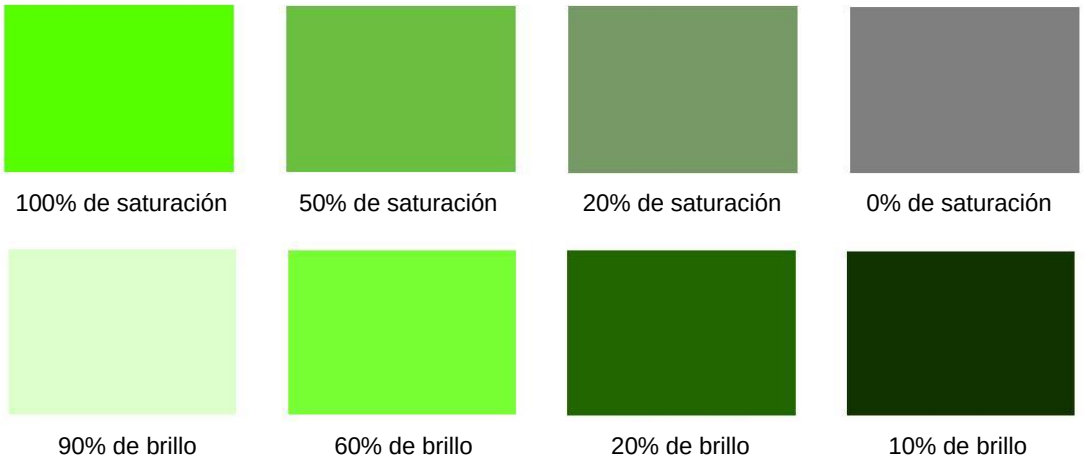
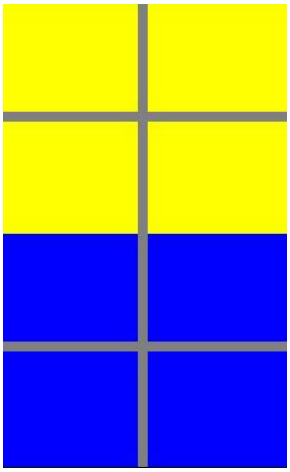


Figura 91. Distintos grados de saturación y brillo para el mismo tono de verde de partida

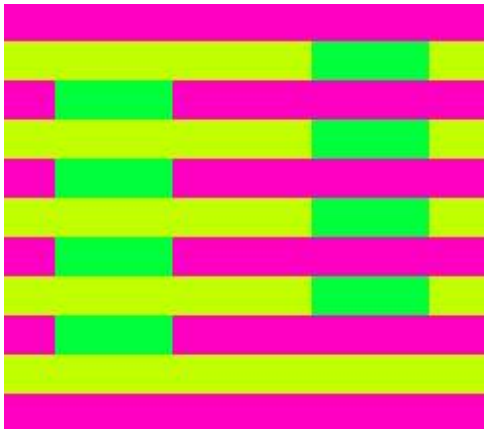
El *brillo* define la intensidad luminosa o la cantidad de luz que tiene un color. Diferencia los colores claros de los oscuros, independientemente de su tono y saturación. Al igual que la saturación, el brillo se mide en %, donde el 0% es el brillo mínimo (negro) y el 100% es el brillo máximo (blanco). A medida que se aumenta el brillo de un tono, éste se va aclarando (va aproximándose al blanco). Y cuando se disminuye el brillo, el tono se va oscureciendo (va acercándose al negro). De nuevo en la figura 91 se puede comprobar cómo el tono de verde de partida con un alto nivel de brillo se ve como un verde muy claro y a medida que el nivel de brillo disminuye el tono percibido es cada vez un verde más oscuro.

A.2.3 Variaciones en la visión de los colores

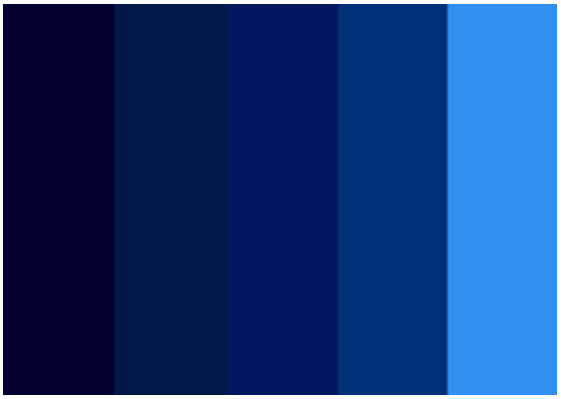
Al igual que ocurre en la visión acromática, en la visión cromática se vuelven a producir una serie de fenómenos que hacen que el ojo perciba distintos colores cuando en realidad es siempre el mismo. Como ya se han explicado para la percepción acromática sólo se va a poner una breve descripción de cada efecto.



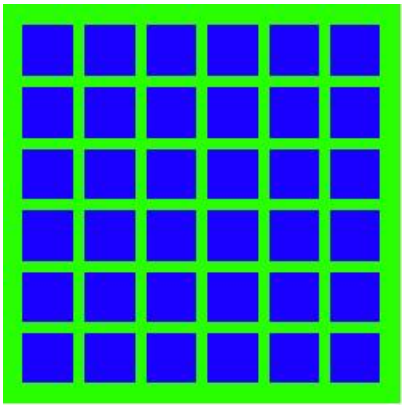
(a) Contraste simultáneo cromático



(b) Asimilación cromática



(c) Bandas de Mach



(d) Rejilla de Hermann

Figura 92. Ejemplos de efectos que varían la percepción de los colores

Contraste simultáneo cromático: la percepción de un color puede cambiar según el color de las áreas que lo rodean. Cuando el fondo es más claro, el objeto se percibe más oscuro y cuando el fondo es más oscuro el objeto se percibe más claro de lo que en realidad es. El efecto es más notable cuando se comparan *colores complementarios*, es decir, aquéllos que al mezclarse en una determinada proporción dan como resultado un pigmento acromático

Asimilación cromática: al revés que en el contraste simultáneo. Si el fondo es claro se aprecia el objeto más claro y si el fondo es más oscuro se aprecia el objeto más oscuro de su color real.

Inhibición lateral: debido a la inhibición de células vecinas, el ojo humano no distingue bien las fronteras de intensidad. Como claros ejemplos están las *bandas de Mach* y la *rejilla de Hermann*. En las bandas de Mach aparecen unas bandas

inexistentes a ambos lados del límite de intensidad de un tono al siguiente. Y en la rejilla de Hermann se ven unos puntos irreales en las intersecciones de la rejilla que desaparecen al mirar directamente a cada intersección.

A.3 Ilusiones ópticas

Aparte de los efectos que se han visto tanto en la percepción acromática como en la cromática existen otros muchos ejemplos en los que la percepción visual humana nos engaña, son las llamadas *ilusiones ópticas*: el ojo proporciona información que no existe o percibe erróneamente propiedades geométricas.

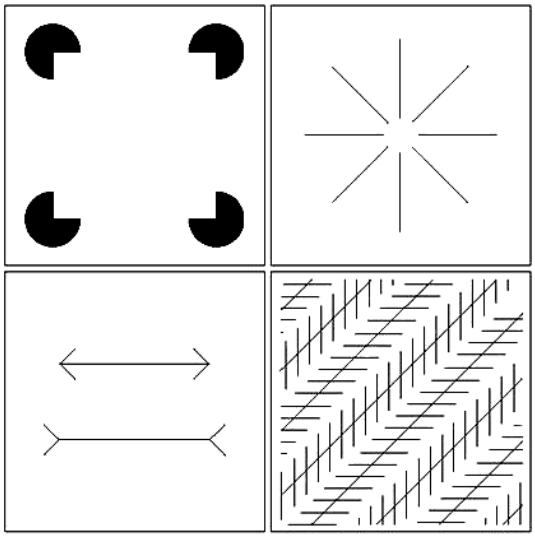


Figura 93. Ilusiones ópticas geométricas

En la figura 93, se puede ver claramente un cuadrado y un círculo, pero sus contornos no están definidos por línea alguna. También se puede observar dos líneas paralelas que parecen de distinto tamaño, cuando son exactamente del mismo tamaño. Y por último, las líneas que están a 45º son perfectamente paralelas aunque no lo aparenten. La teoría del "desplazamiento ocular" explica esta ilusión asegurando que se tiende a percibir los ángulos agudos mayores de lo que en realidad son. Por eso, las dos líneas oblicuas no parecen estar en la misma dirección.

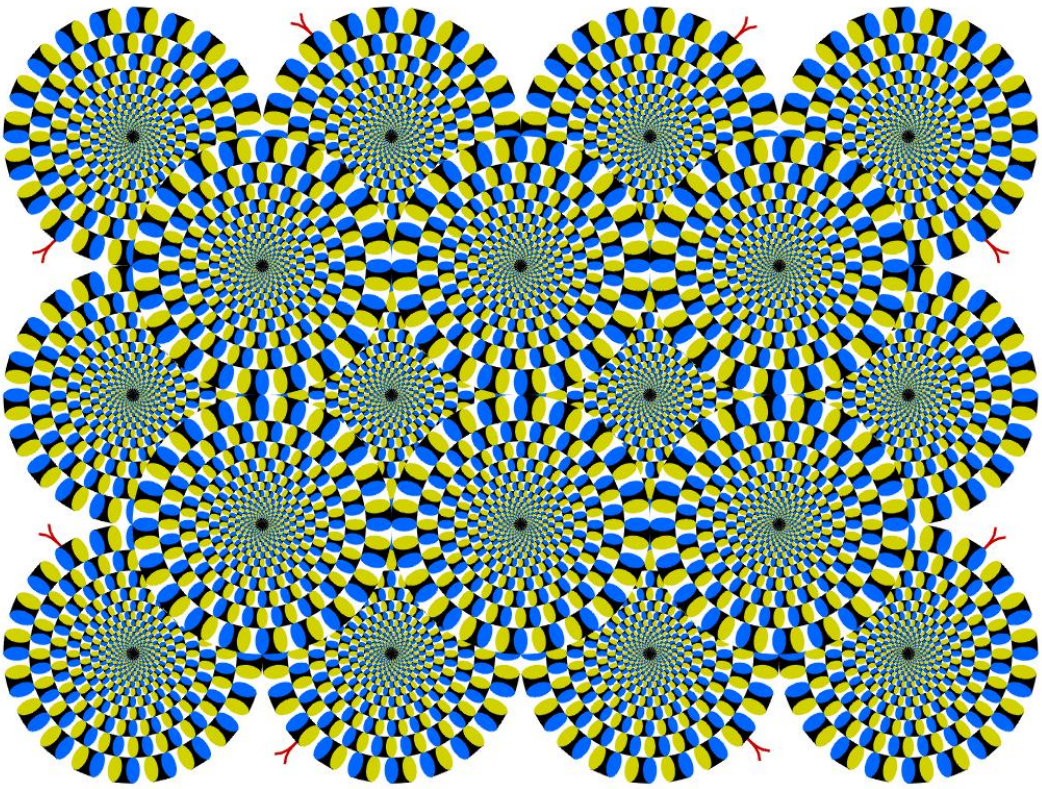


Figura 94. Ilusión óptica de movimiento

En la figura 94, aparece una imagen de Akiyoshi Kitaoka que no es más que una versión optimizada de la ilusión de Fraser y Wilcox [12]. Aunque parezca una imagen en movimiento, realmente es una imagen fija. La sensación de movimiento se debe a las características de nuestra percepción visual. El ser humano ve de manera discontinua, en la retina se captan unas 20 imágenes por segundo, por eso a una frecuencia mayor las distintas imágenes se fusionan, como ocurre en el cine, y se perciben en movimiento. En este ejemplo también influyen los contrastes de color y luminosidad, ya que la dirección del movimiento viene determinada por la gradación del brillo (del blanco al negro). Si se observa fijamente el punto central de cada círculo se podrá ver cómo la imagen se detiene en ese círculo mientras que en los círculos de la periferia continúan moviéndose.

Las *imágenes ambiguas o reversibles* son aquéllas en las que en una misma imagen existen varias posibilidades de interpretación. Así, cuando el ojo percibe una interpretación es incapaz de captar simultáneamente la otra, es decir, o se ve una o se ve la otra, pero nunca se percibirán las dos al mismo tiempo. Este hecho está relacionado con las leyes de la Gestalt. En concreto con la ley de fondo y figura, que

se refiere a que en una imagen no se puede percibir al mismo tiempo el fondo y la figura. En la figura 95, se puede ver el perfil de dos caras enfrentadas (color negro) pero también se puede observar una copa o jarrón (color blanco).

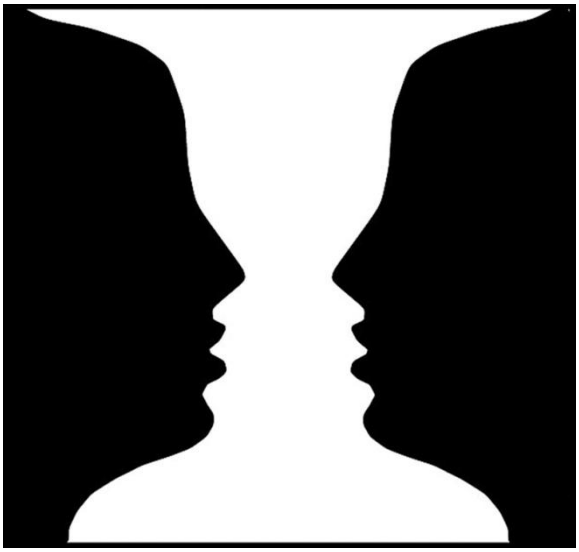


Figura 95. Ilusión óptica de imagen reversible

A.4 Implicaciones del sistema visual humano

A continuación se resumen las propiedades más relevantes del sistema visual humano (SVH) que tienen implicaciones tanto a la hora de procesar imágenes como de procesar vídeo [34].

- El SVH es más sensible a los detalles que proporciona la luminancia que a los detalles que proporciona el color. Esto hecho implica que el color (o la crominancia) puede ser reducido sin que afecte de forma significativa a la calidad de la imagen.
- El SVH es más sensible a regiones de alto contraste que a regiones de bajo contraste dentro de una imagen, lo que significa que las regiones con grandes variaciones de luminancia, como los bordes, son particularmente importantes en la apariencia de la imagen, por lo que deben ser detectados, preservados y realzados.
- El SVH es más sensible a bajas frecuencias espaciales (cambios de luminancia sobre un área grande) que a altas frecuencias espaciales (cambios rápidos de luminancia dentro de áreas pequeñas). Este es un aspecto utilizado a menudo

en las técnicas de compresión de imágenes y vídeo. Descartar contenidos redundantes de baja frecuencia espacial pero conservando los bordes supone un considerable ahorro computacional.

- El SVH es más sensible a características de la imagen que existen durante mucho tiempo, lo que significa que es importante utilizar técnicas que reduzcan al mínimo las perturbaciones o ruidos que presente la imagen.
- Las respuestas del SVH varían de un individuo a otro, lo que significa que evaluaciones subjetivas de imagen y vídeo deben llevarse a cabo con un número elevado de sujetos. Este aspecto, también refuerza la necesidad de encontrar medidas cuantitativas de la calidad de la imagen que se puedan calcular de forma automática.

Bibliografía

[1] Acharya, T.; Ray A. K. *Image processing: principles and applications*. John Wiley & Sons, 2005.

[2] Ainslie, M. A. *Principles of sonar performance modeling*. Springer, 2010.

[3] Blanchet, G. and Charbit, M. *Digital Signal and Image Processing using MATLAB*. ISTE, 2006.

[4] Blondel, P. *The handbook of sidescan sonar*. Springer, 2009.

[5] Bovik, A.C.; Huang, T.S.; Munson, D.C. The effect of median filtering on edge estimation and detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9 no. 2, pp. 181-194, 1987.

[6] Burczynski, J. Introducción al uso de sistemas sonar para la estimación de la biomasa de peces. *FAO, Documentos técnicos de pesca* nº 191, 74 págs., 1982.

[7] Canny, J. A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679-698, 1986.

[8] Chew, A. L.; Tong P. B. and Chia, C. S. Automatic detection and classification of man-made targets in side scan sonar images. *Symposium on Underwater Technology and Workshop on Scientific Use of Submarine Cables and Related Technologies*, pp. 126-132, 2007.

[9] Christ, R. D. and Wernli, R. L. *The ROV Manual: A User Guide for Observation-Class Remotely Operated Vehicles*. Elsevier, 2007.

[10] Del Grosso, V. A. New equations for the speed of sound in natural waters with comparisons to other equations. *Journal of the Acoustical Society of America* vol. 56, no. 4, pp. 1084-1091, 1974.

[11] Etter, P. C. *Underwater acoustic modeling and simulation*. Spon Press, 2003.

[12] Fraser, A.; Wilcox, K. J. Perception of illusory movement. *Nature*, vol. 281, no. 5732, pp. 565-566, 1979.

[13] González, R. C; Woods R. E. *Digital image processing (2nd edition)*. Prentice Hall, 2002.

[14] González, R. C; Woods R. E.; Eddins, S. L. *Digital image processing using Matlab*. Prentice Hall, 2003.

[15] Hackmann, W. *Seek & Strike: Sonar, anti-submarine warfare and the Royal Navy 1914-54*. HM Stationery Office, 1984.

[16] Hardie, R. C. and Boncelet, C. G. Gradient-Based Edge Detection using Nonlinear Edge Enhancing Prefilters. *IEEE Transactions on Image Processing*, vol. 4 no. 11, pp.1572-1577, 1995.

[17] Hardie, R. C. and Boncelet, C. G. LUM Filters: A Class of Rank-Order-Based Filters for Smoothing and Sharpening. *IEEE Transactions on Signal Processing* vol. 41 no. 3, pp. 1061-1076, 1993.

[18] Jensen, J. R. *Introductory digital image processing: A remote sensing perspective*. Prentice Hall, 2004.

[19] Konsberg. Underwater surveillance and protection information, C'Inspector. www.kongsberg.com/en/kds/products/navalsystems/underwatersurveillanceprotection/

[20] Lee, Y.; Fam, A. An edge gradient enhancing adaptive order statistic filter. *IEEE Transactions on Acoustics, Speech and Signal Processing* vol. 35, no. 5, pp. 680-695, 1987.

[21] Longbotham, H. G. and Eberly, D. The WMMR filters: A class of robust edge enhancers. *IEEE Transactions on Signal Processing*, vol. 41, no. 4, 1993.

[22] Longbotham, H.G., Bovik, A.C. and Restrepo, A. Generalized order statistic filters. *In Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, ICASSP-89, pp. 1610-1613, Glasgow (Scotland), 1989.

[23] Mackenzie, K. V. Nine-term equation for sound speed in the oceans. *Journal of the Acoustical Society of America* vol. 70, no. 3, pp. 807-812, 1981.

[24] Maravall, D. *Reconocimiento de formas y visión artificial*. Ediciones Ra-ma, 1993.

[25] Marques, O. *Practical image and video processing using MATLAB*. John Wiley & Sons, 2011.

[26] Martínez Rojas, J. A.; Alpuente Hermosilla, J.; Martínez Rojas, I. and Vignote Peña, S. Fractal-based image enhancement techniques inspired by differential interference contrast microscopy. *Journal of Optics A: Pure and Applied Optics*, vol. 11, no. 6, 2009.

[27] Medwin, H. Speed of sound in water: A simple equation for realistic parameters. *Journal of the Acoustical Society of America* vol. 58, no. 6, pp. 1318-1319, 1975.

[28] Medwin, H.; Clay C. S. *Fundamentals of acoustical oceanography*. Academic Press, 1997.

[29] *Oviedo, L. M. M.; Kanashiro, A. M. y Colimbini, M. A. Fractales un universo poco frecuentado*. Ediciones Universidad Nacional del Litoral, 2004.

[30] Pérez Vega, C.; Zamanillo Sainz de la Maza, J. M. *Fundamentos de la televisión analógica y digital.* Servicio de publicaciones de la Universidad de Cantabria, 2003.

[31] Qidwai, U. and Chen, C. H. *Digital Image Processing: an algorithmic approach with Matlab*. Chapmann & Hall/CRC Press, 2010.

[32] Raichel, D. R. *The science and applications of acoustics (2nd edition)*. Springer, 2006.

[33] Recuero, M. *Ingeniería acústica*. Editorial Paraninfo, 1994.

[34] Richardson, I. E. G. *Video Codec Design: Developing Video and Image Compression Systems*. John Wiley & Sons, 2002.

[35] Russ, J. C. *The image processing handbook (4th edition)*. CRC Press, 2002.

[36] Stergiopoulos, S. *Advanced Signal Processing Handbook: Theory and Implementation for Radar, Sonar, and Medical Imaging Real-Time Systems*. CRC Press, 2001.

[37] *The Klein Image Gallery.* www.l-3klein.com/image_gallery/image_gallery.html

[38] The Mathworks. *Matlab: Creating Graphical User Interfaces*. The MathWorks Inc., 2012.

[39] The Mathworks. *Matlab: Image Processing Toolbox User's Guide*. The MathWorks Inc, 2012.

[40] Thurnhofer, S. and Mitra, S. K. A general framework for quadratic Volterra filters for edge enhancement. *IEEE Transactions on Image Processing*, vol. 5 no. 6, pp. 950-963, 1996.

[41] Tritech. SeaKing (side scan sonar or imaging sonar) on AUV/ROV. www.tritech.co.uk/product/side-scan-sonar-seaking-auv-rov

[42] Vélez Serrano, J. F.; Moreno Díaz, A. B.; Sánchez Calle, A.; Esteban Sánchez-Marín, J. L. *Visión por computador (2ª edición)*. Editorial Dykinson, 2003.

[43] Waite, A. D. *Sonar for Practising Engineers (3rd edition)*. John Wiley & Sons, 2002.

[44] Wilson, W. A. Speed of sound in sea water as a function of temperature, pressure and salinity. *Journal of the Acoustical Society of America* vol. 32, no. 6, pp. 641-644, 1960.

[45] Wood, A, B. From the board invention and research to the Royal Naval scientific Service. *Journal of the Royal Naval Scientific Service* vol. 20, no. 4, pp. 1-100, 1965.

[46] Ye, X.; Li, P. and Deng, Y. A side scan sonar image denoising algorithm based on compound of fuzzy weighted average and Kalman filter. *In Proceedings of IEEE International Conference on Mechatronics and Automation*, pp. 720-724, Chengdu (China), 2012.