

# CONSTRUCCIÓN DE MOSAICOS A PARTIR DE IMÁGENES DE SONAR DE BARRIDO LATERAL

## Trabajo Fin De Máster

*Por*

*José Manuel Bernabé Murcia*  
*josemanuel.bernabe@um.es*



MÁSTER EN NUEVAS TECNOLOGÍAS

UNIVERSIDAD DE MURCIA

Junio 2020

Tutor

Dr. Humberto Martínez Barberá

humberto@um.es



# Índice

<b>Resumen</b>	<b>2</b>
<b>1. Introducción</b>	<b>3</b>
1.1. Motivación . . . . .	7
1.2. Objetivos . . . . .	7
<b>2. Estado del Arte</b>	<b>8</b>
2.1. Correcciones radiométricas . . . . .	9
2.1.1. Requantization . . . . .	10
2.1.2. Across-track corrections . . . . .	10
2.1.3. Along-track corrections . . . . .	10
2.2. Correcciones geométricas . . . . .	11
2.2.1. Slant-range correction . . . . .	11
2.2.2. Anamorphosis . . . . .	12
2.3. Overlapping . . . . .	13
2.3.1. Métodos no simbólicos . . . . .	14
2.3.1.1. Optical Flow . . . . .	16
2.3.1.2. Phase Correlation . . . . .	22
2.3.1.3. Mutual Information . . . . .	24
2.3.1.4. Correlation Ratio . . . . .	24
2.3.2. Métodos simbólicos . . . . .	25
2.3.2.1. SIFT . . . . .	26
2.3.2.2. SURF . . . . .	29
2.3.2.3. ORB . . . . .	30
2.4. Métodos no simbólicos vs métodos simbólicos . . . . .	31
<b>3. Metodología y Herramientas</b>	<b>32</b>
3.1. Metodología . . . . .	32
3.2. Herramientas . . . . .	33
3.2.1. Hardware . . . . .	33
3.2.2. Software . . . . .	35
<b>4. Diseño y Resolución</b>	<b>37</b>
4.1. Pre-Procesado . . . . .	37
4.1.1. Misión . . . . .	37
4.1.2. Extracción, limpieza y combinación de los datos . . . . .	38
4.1.2.1. Extracción datos ROV . . . . .	39
4.1.2.2. Extracción datos sonar . . . . .	40
4.1.2.3. Limpieza de los datos . . . . .	41
4.1.2.4. Combinación de los datos . . . . .	42
4.2. Procesado . . . . .	43
4.2.1. Mapeando coordenadas . . . . .	43

4.2.1.1. Mapeando el ROV . . . . .	43
4.2.1.2. Mapeando el sonar . . . . .	44
4.2.2. Correcciones radiométricas . . . . .	46
4.2.2.1. Ecualización de Histograma . . . . .	47
4.2.2.2. Eliminación de Water Column . . . . .	47
4.2.3. Correcciones Geométricas . . . . .	49
4.2.3.1. Slant Range Correction . . . . .	49
4.2.3.2. Anamorfosis . . . . .	49
4.2.4. Generando mosaico . . . . .	50
4.3. Resultado . . . . .	54
<b>5. Conclusiones</b>	<b>60</b>



## **Declaración de Autenticidad Del Trabajo**

D. /Dña. José Manuel Bernabé Murcia, con DNI 48740534S, estudiante de la titulación de Máster Universitario en Nuevas Tecnologías en Informática de la Universidad de Murcia y autor del TF titulado “CREACIÓN DE MOSAICOS A PARTIR DE IMÁGENES DE SONAR DE BARRIDO LATERAL”.

De acuerdo con el Reglamento por el que se regulan los Trabajos Fin de Grado y de Fin de Máster en la Universidad de Murcia (aprobado C. de Gob. 30-04-2015 y modificado 22-04-2016), así como la normativa interna para la oferta, asignación, elaboración y defensa de los Trabajos Fin de Grado y Fin de Máster de las titulaciones impartidas en la Facultad de Informática de la Universidad de Murcia (aprobada en Junta de Facultad 27-11-2015)

DECLARO:

Que el Trabajo Fin de Máster presentado para su evaluación es original y de elaboración personal. Todas las fuentes utilizadas han sido debidamente citadas. así mismo, declara que no incumple ningún contrato de confidencialidad, ni viola ningún derecho de propiedad intelectual e industrial.

Murcia, a 21 de Junio de 2020

FIRMADO: JOSÉ MANUEL BERNABÉ MURCIA

## **Resumen**

La exploración submarina se ha incrementado significativamente en la última década, en diferentes campos. Esto no es una sorpresa ya que el 71 % de la tierra es agua, y solo el 5 % puede considerarse explorada. La comunidad científica y la industria, realizan tareas de investigación, exploración, instalación, reparación y mantenimiento del fondo marino. Las operaciones submarinas han requerido de la utilización de buzos para estas tareas, pero en las últimas décadas esto ha ido cambiado gracias a la aparición de los ROVs (Remote Operated Vehicle) y más recientemente de los AUV (Autonomous Underwater Vehicle).

Las operaciones que implican la visualización del fondo marino requieren la utilización de sonar acústicos, debido a la profundidad o el nivel de turbidez del agua solo un pequeño espectro de la luz solar llega al fondo marino. Los sonars transmiten una señal acústica por medio del agua gracias a un transductor. Si el fondo marino u otro objeto está en la trayectoria del impulso de sonido, el sonido rebota en el objeto y le devuelve un eco al transductor sonar, el tiempo transcurrido entre la emisión del pulso de sonido y la recepción del eco es lo que se utiliza para calcular la distancia del objeto. Algunos sistemas de sonar también miden la intensidad del eco, y esta información se puede utilizar para inferir las características de algunos de los objetos que refleja.

Este trabajo tiene como motivación principal la toma de contacto y el aprendizaje de los diferentes sistemas sonars acústicos, así como su funcionamiento, ventajas e inconvenientes. También se trata de conocer las diferentes técnicas que se usan en el campo de los vehículos submarinos para la generación de mosaicos con imágenes sonars. Además, del desarrollo de una herramienta con el fin de poder interpretar y representar el fondo marino, teniendo en cuenta la posición y pose del ROV que lo integra.

Utilizando los conocimientos adquiridos se ha desarrollado una herramienta capaz de generar mosaicos del fondo marino de forma automática, proyectando la trayectoria del ROV en las imágenes. Los principales resultados obtenidos nos dan como salida: (1) Conocer la trayectoria del ROV, (2) Analizar los distintos objetos del fondo marino y (3) Georreferenciar objetos detectados.

## 1. Introducción

La exploración submarina se ha incrementado significativamente en la última década, en diferentes campos. Esto no es una sorpresa ya que el 71% de la tierra es agua, y solo el 5% puede considerarse explorada. La comunidad científica está explorando constantemente campos como: oceanografía, geología marina, arqueología submarina, evaluación de daños, cartografía. Además de la comunidad científica, las industrias están aumentando operaciones que requieren tareas submarinas para la inspección, monitorización y reparación en: cableado, tuberías, parques eólicos marinos, perforaciones de petróleo. [18] Las operaciones submarinas han requerido de la utilización de buzos para estas tareas, pero en las últimas décadas esto ha ido cambiado gracias a la aparición de los ROVs (Remote Operated Vehicle). La utilización de ROVs tiene la ventaja de que aumenta el tiempo de operación bajo el agua y a barata el costo de la operación, como inconveniente es que requiere la supervisión y manejo de una persona para su funcionamiento, por ello en los últimos años se están desarrollado e investigando los AUV (Autonomus Underwater Vehicle), que no quieren la supervisión para el desarrollo de la tarea programada.

Las operaciones que implican la visualización del fondo marino como la exploración, búsqueda de objetos, detección de minas o mapeado del terreno, requieren el uso de sonar acústicos (Sound Navigation And Ranging). Una señal acústica o pulso de sonido, a menudo llamado ping, se transmite en el agua por una especie de altavoz subacuático conocido como transductor. El transductor puede ser montado en el casco de un barco, o puede ser remolcado en un contenedor llamado domo. Si el fondo marino u otro objeto está en la trayectoria del impulso de sonido, el sonido rebota en el objeto y le devuelve un eco al transductor sonar. El tiempo transcurrido entre la emisión del pulso de sonido y la recepción del eco es lo que se utiliza para calcular la distancia del objeto. Algunos sistemas de sonar también miden la intensidad del eco, y esta información se puede utilizar para inferir las características de algunos de los objetos que refleja. Los objetos duros, por ejemplo, producen ecos más fuertes que los objetos más suaves. Esta es una descripción de un "sonar activo". Por otro lado, tenemos sistemas de "sonar pasivos" los cuales no transmiten pulsos de sonido, sino que, escuchan los sonidos emitidos por animales marinos, barcos y otras fuentes. La principal ventaja de estos sistemas es que proporcionan imágenes de alta resolución de áreas submarinas donde apenas existe visibilidad, debido a que a partir de ciertas profundidades solo un pequeño espectro de la luz solar llega al fondo marino. [23]

Los sonar acústicos los podemos dividir en tres categorías principales: single-beam echo-sounders, multibeam echo-sounders, sidescan sonar. Podemos ver

una ilustración de cada uno de ellos extraída del libro [22] en la figura 1.

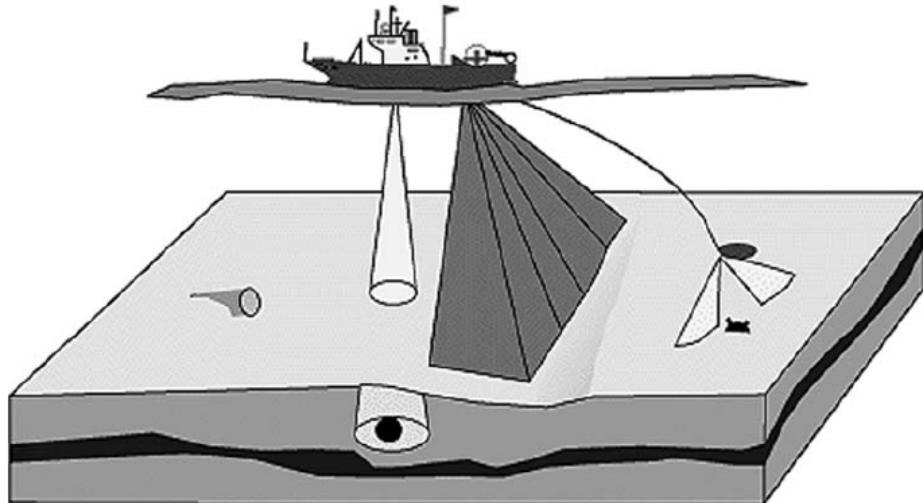


Figura 1: De izquierda a derecha single-beam, multibeam, sidescan sonar. [22]

### Single-Beam Echo-Sounders

Las ecosondas de haz simple o monohaz solo miden la profundidad directamente bajo la plataforma en la que están instaladas. Proporcionan una buena exactitud y resolución en el sentido vertical, sin embargo, proporcionan una mala información en el sentido transversal. Generalmente, usan señales de baja frecuencia transmitidas en pulsos cortos de menos 2ms, desde un único transductor. La geometría del haz es normalmente en forma de cono, la primera señal devuelta del fondo del mar corresponde a puntos más cercanos al barco, los puntos más lejanos corresponden a las diferentes capas del suelo marino.

### Multibeam Echo-Sounders

Las ecosondas multihaz nacieron a partir de las ecosondas monohaz, como una mejora de los mismos. Las ecosondas multihaz fueron más accesibles al público a finales de los 80, aunque su creación fue en la década de los 60. Actualmente es la herramienta más utilizada para el estudio del fondo marino. Un sistema multihaz utiliza múltiples transductores que apuntan en diferentes ángulos a cada lado de un barco para crear una franja de señales, cubriendo así una mayor zona. Además de precisión se gana rapidez, permitiendo en un solo desplazamiento del vehículo analizar grandes superficies

de fondo y levantar una carta batimétrica muy precisa del relieve, por tanto, un ahorro significativo en el gasto que supone cartografiar una zona.

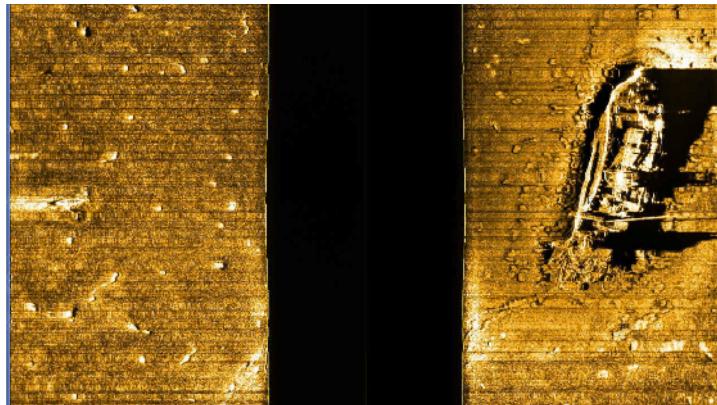


Figura 2: Imagen sonar barrido lateral. [23]

### SideScan Sonar

Sidescan sonar o sonar de barrido lateral apareció en paralelo junto con las sondas multihaz, es la herramienta predilecta elegida para la creación de mapas de alta resolución del fondo marino, trabajan a frecuencias muy altas y son capaces de cubrir como las sondas multihaz grandes áreas. A diferencia del multihaz, el sonar de barrido lateral toma medidas de la fuerza de los ecos, en lugar de datos de profundidad, por lo tanto, la cara de barrido produce imágenes en blanco y negro del fondo del mar. Por lo tanto, para el levantamiento del fondo, es decir, obtener la batimetria de una zona no son muy precisos. En la actualidad para realizar la batimetria de una zona haciendo uso de los sonar de barrido lateral es necesario una sonda monohaz o multihaz complementaria. En la figura 2 podemos ver la imagen obtenida del fondo con un sonar de barrido lateral, y en la figura 3 se aprecia la batimetria del fondo utilizando una ecosonda y un sonar de barrido lateral.

La diferencia entre la sonda y el sonar es que la ecosonda mantiene la cara radiante, del transductor, siempre en posición vertical fija, dirigida hacia el fondo del mar. Y el transductor del sonar puede operar horizontal y lateralmente a voluntad.

Aunque los sistemas multihaz y de barrido lateral tienen una aplicación final distinta la geometría de sus haces es muy similar como se aprecia en la figura 1. Las características geométricas que lo componen las podemos apreciar en la figura 4. La distancia recorrida por la señal acústica entre el transductor y el fondo marino se denomina Slant Range. El Ground Range, es la distancia horizontal entre la vertical del aparato emisor y el punto de

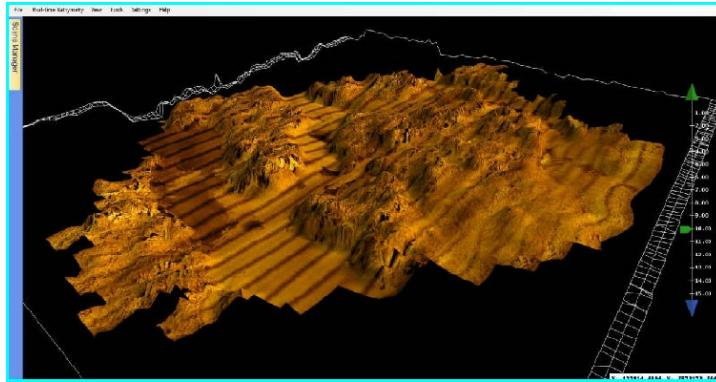


Figura 3: Batimetría + Mosaico, usando sonar barrido lateral. [23]

incidencia del rayo sonoro medida sobre el fondo marino. El ángulo formado por la dirección del haz y el fondo marino se denomina Grazing Angle, mientras que el ángulo formado por la dirección del haz y la vertical o normal del aparato emisor se denomina Elevation Angle. [23]

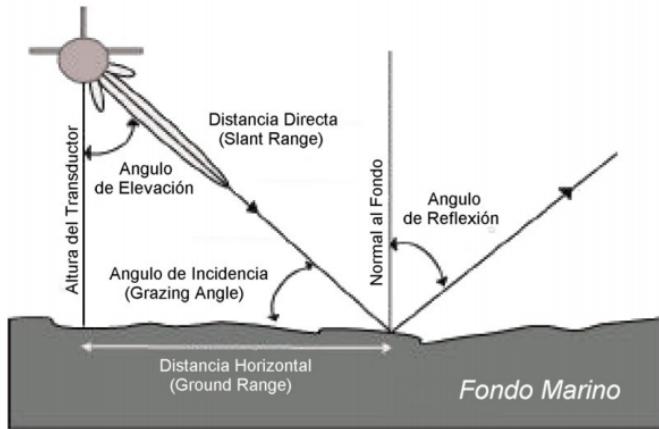


Figura 4: Geometría del haz. [23]

Hemos hablado que los sonars o ecosondas son capaces de realizar una batimetría del fondo marino, por lo que el papel que juegan los sonars acústicos en los equipos submarinos, es vital no solo ver el fondo donde no hay visibilidad apenas, sino que pueden ayudarnos a mejorar la navegación.

## SLAM

Simultaneous Localization and Mapping o SLAM, tiene como propósito la creación o actualización de un mapa de un entorno desconocido y usar ese mismo mapa para localizar, es decir, realizar un seguimiento de la ubicación

de un agente dentro de dicho mapa generado. La generación de un mapa de un entorno desconocido sirve para muy poco sino podemos ubicarnos dentro de él, por lo que es necesario tener un buen sistema de posicionamiento. En un entorno tan hostil para la propagación de señales como es el agua es un gran reto y uno de los temas de investigación actuales, determinar la posición son los cimientos para la construcción de vehículos autónomos, a día de hoy aún no se ha resuelto que un AUV interactúe con entornos complejos de manera segura. [17] [18].

### 1.1. Motivación

Este trabajo tiene como motivación principal la toma de contacto y el aprendizaje de los diferentes sistemas sonars acústicos, así como su funcionamiento. También se trata de conocer las diferentes técnicas que se usan en el campo de los vehículos submarinos para la generación de mosaicos con imágenes sonars. Utilizando los conocimientos adquiridos para ser capaces de aplicarlos a la creación de una herramienta capaz de generar un mosaico que sea consistente tanto espacial como temporalmente, a partir de imágenes de un sonar de barrido lateral, teniendo como datos complementarios la posición y pose de un ROV. El proceso de la creación de esta herramienta será el primer paso para la generación de mapas del fondo marino 2D y 3D, con el objetivo posterior de usar todo esto para la localización y georreferencia de un ROV o AUV en un entorno desconocido.

### 1.2. Objetivos

El objetivo principal de este trabajo fin de máster, es el procesado, interpretación y la creación de un sistema capaz de visualizar un mapa compuesto de diferentes imágenes proporcionadas por un sonar de barrido lateral incorporado en un ROV, aplicando a dichas imágenes los datos de posición y pose proporcionados por el ROV, consiguiendo la consistencia tanto espacial como temporal, es decir, existiendo una relación real entre los píxeles y los metros recorridos.

El objetivo principal, lo podemos dividir en sub-objetivos que son necesarios para el desarrollo del objetivo principal. Se enumeran a continuación.

- Estudio previo sobre los diferentes tipos de sonars y el software de control.
- Realización de la misión y recogida de datos.
- Pre-procesado y procesado de los datos.
- Generación del mosaico.

## 2. Estado del Arte

En esta sección trataremos de hacer un repaso de las diferentes técnicas que se hacen uso para la generación de mosaicos a partir de imágenes de sonar de barrido lateral y la importancia que juega las imágenes acústicas dentro del mismo.

En los orígenes de los sonar de barrido lateral, las imágenes se imprimían en largas hojas de papel, y la generación de mosaicos de la imágenes recolectadas se realizaban literalmente cortando y pegando [1], como se puede apreciar en la figura 5.



Figura 5: Formando mosaico. [1]

El proceso de crear el mosaico solo se podía hacer una vez hecha la misión, es decir, una vez recopilado y procesado los datos, con el avance y el asentamiento de la tecnología digital, estos procesos se han ido informatizando haciéndolos menos costoso en relación tiempo y coste. Inclusive pudiendo hacerse en tiempo real. [1]

Actualmente, la construcción de mosaicos se compone de líneas de píxeles adyacentes. El proceso de generación de mosaicos es simple, pero realizar las transformaciones, escalas, rotaciones y translaciones, y hacerlo de forma automática hace que aumente su complejidad. En el libro [22], se plantea unos procedimientos que llevar a cabo de antes y después de la creación del mosaico, la mayoría de artículos que componen las referencias de este trabajo hacen uso de este esquema, podemos ver en la figura 6 el diagrama de flujo de los distintos procesos que se siguen. Los procedimientos que se realizan dentro del bloque de processing, son procedimientos que ayudan a mejorar la visualización, y en algunas ocasiones necesarias para la obtención de un mosaico de calidad, por ejemplo, la solución del problema del overlapping que se encuentra en la etapa post-processing. En las siguientes subsecciones se va a hacer un repaso de las distintas correcciones que realizan trabajos parecidos a este en el sentido de la generación de mosaicos con un sonar de

barrido lateral.

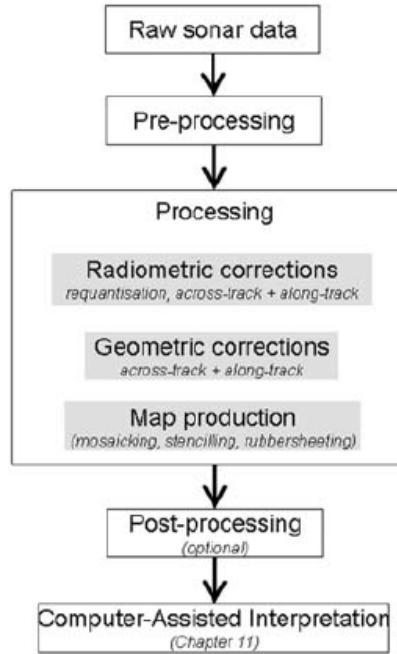


Figura 6: Procedimiento generación mosaicos. [22]

## 2.1. Correcciones radiométricas

La corrección radiométrica trata de realizar correcciones en los valores de píxel de una imagen obtenida por el sonar de barrido lateral, para obtener valores de intensidad homogéneos equilibrando histogramas o corrigiendo distorsiones presentes en los píxeles. Estas distorsiones son provocadas por muchos factores, algunos de ellos son: residuo de ganancia variable en el tiempo, patrones de haz. Muchos investigadores han realizado estudios para tratar de eliminar estos efectos. Los métodos que existen para tratar de corregir estas distorsiones no consiguen hacerlas de forma efectiva, especialmente las variaciones de sedimento del fondo marino. [19]

Dentro de las correcciones radiométricas que tratan de mejorar la calidad de la imagen, podemos encontrar métodos los siguientes métodos:

### **2.1.1. Requantization**

La mayoría de los sonar procesan paquetes con medidas de amplitud escaladas entre 0 y 255 o entre 0 y 32.767. La salida de valores del hardware no siempre sigue los mismos esquemas de cuantización por lo que puede ser necesario un recuantización. Una de las técnicas comúnmente usada es la de Bell  $\mu$ -law, la podemos ver en la expresión 1 Donde C es una constante definida por el usuario, m es el número originales de bits y n es el nuevo número de bits.

$$newvalue = C \cdot \ln \left( 1 + \frac{2^n \cdot rawvalue}{2^m} \right) \quad (1)$$

[22]

### **2.1.2. Across-track corrections**

La amplitud de la señal son órdenes de magnitud inferiores a la señal transmitida debido a la atenuación con la distancia y otros efectos. Objetos idénticos colocados en diferentes posiciones a lo largo del ancho del haz tendrán una amplitud cada vez más baja. Las ganancias variables en el tiempo (Time Varying Gains, TVG) son correcciones diseñadas para compensar este efecto. Una corrección fija generalmente se construye en el hardware del sonar, corrigiendo los efectos conocidos. Las correcciones de TVG también se pueden calcular como funciones ad hoc únicas que representan las condiciones topográficas locales, conocidas por sondas in situ. Podemos encontrar distintas formas de correcciones TVG, una de ellas se presenta en el artículo [19]

[22]

### **2.1.3. Along-track corrections**

Las distorsiones along-track producen líneas negras irregulares en las imágenes sonars, atribuibles a problemas de adquisición dentro del transductor, o pérdida de datos durante la transmisión entre el sonar y el ROV o AUV. Pero, normalmente se atribuyen a la pérdida de la transmisión debido al cambio de profundidad del sonar. Un proceso de filtrado es reemplazar los valores usando los valores promediados de las líneas adyacentes. En el artículo [19] podemos encontrar otra forma de eliminación de estas distorsiones considerando las variaciones del sedimento. [22]

## 2.2. Correcciones geométricas

Las variaciones angulares y de profundidad del sonar degradan la calidad de las imágenes de sonar de barrido lateral provocando efectos como rescalados en la imagen o pérdida de información.

### 2.2.1. Slant-range correction

Esta corrección lo que busca es tratar corregir el efecto de rescalado que provoca las variaciones antes comentadas, por lo que trata de obtener el tamaño real de lo que se está viendo en el sonar. Para ello, es necesario calcular la distancia que existe entre la distancia del AUV con el fondo y la distancia del slant con el fondo, en la figura 7 se puede ver una ilustración más clara, donde se debe calcular la distancia entre  $D$  y  $D_i$ .

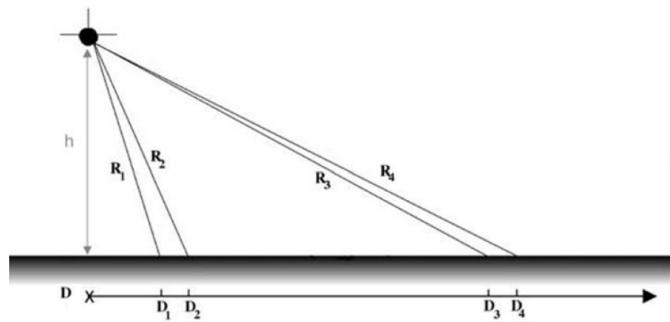


Figura 7: Distancia del slant

En la ecuación 3 calculamos la distancia del slant con el suelo, donde  $c$  es la velocidad del sonido y  $t$  el tiempo que tarda el haz del sonar en ir y volver. Una vez que obtenemos la distancia del slant con el fondo podemos calcular la distancia  $D$  con  $D_i$ , teniendo como referencia la distancia  $D_n$  que hemos configurado previamente en el comienzo de la misión, por lo que podemos desplazar los píxeles la distancia necesaria para tratar de eliminar el rescalado. En la figura extraída del artículo [21] podemos ver como actúa la corrección del slant.

$$Distance_{ground} = \sqrt{Distance_{Slant}^2 - h^2} \quad (2)$$

$$Distance_{Slant} = \frac{c \cdot t}{2} \quad (3)$$

[21]

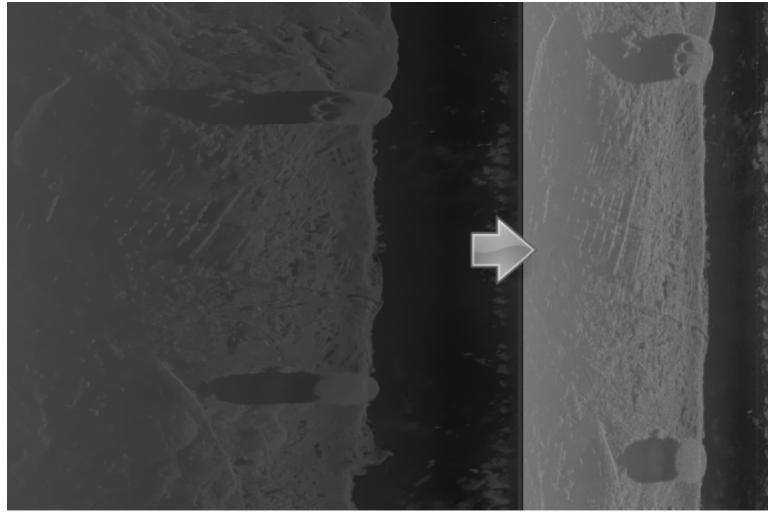


Figura 8: Ejemplo de corrección de slant range

Cabe destacar que estas ecuaciones solo funcionarían en un terreno plano, si el terreno presenta unos cambios de altura muy destacable no servirían, pero existe una variación que teniendo en cuenta una batimetría previa del terreno puede aplicarse esta corrección, podemos ver en la figura 9 una ilustración de esto, la ecuación 4 tiene en cuenta la altura del terreno.

$$Distance_{ground} = \sqrt{Distance_{Slant}^2 - (H - h)^2} \quad (4)$$

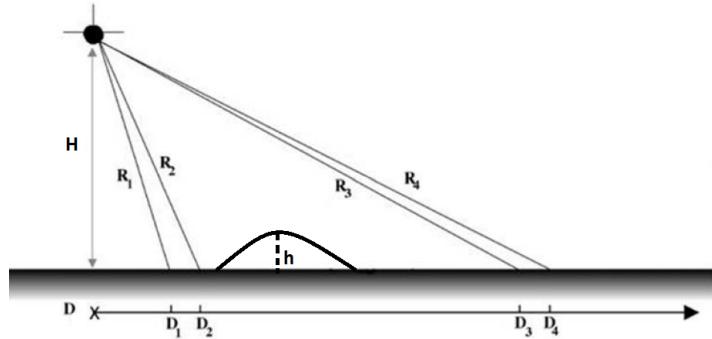


Figura 9: Distancia slant range, teniendo en cuenta la altura del terreno

### 2.2.2. Anamorphosis

Esta corrección produce una imagen en la que el espacio entre píxeles es el mismo a lo largo del recorrido, es decir, existen situaciones sobre todo

debido a una velocidad alta que provoca que el espaciado entre cada línea de barrido del sonar estén muy separadas unas de otras o muy juntas en el caso de una velocidad reducida, un ejemplo de ello lo encontramos en la figura 10 extraída del libro [22].

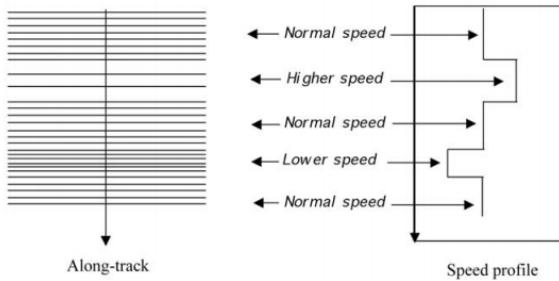


Figura 10: Anamorfosis velocidad

Es importante destacar que la resolución obtenida de las imágenes del sonar y los mosaicos, están altamente influenciados por: la corriente, giros bruscos del AUV o ROV, profundidad, la localización para crear mosaicos con precisión, es decir, que estén bien georeferenciados, más allá de una buena navegación, es muy necesario el disponer de un buen sistema de posicionamiento, lo que es un gran reto incluso a día de hoy para equipos submarinos, ya que debajo del agua los GPS's no funcionan. En este estado del arte no vamos a entrar en los distintos tipos de posicionamiento que podríamos tener, lo que se intenta recalcar es, para hacer una buena cartografía del fondo marino, es necesario un buen sistema de posicionamiento.

### 2.3. Overlapping

Cuando se construye un mosaico del fondo marino generalmente se va a producir overlapping entre las distintas imágenes que captamos con el sonar de barrido lateral. Este overlapping no tiene por qué darse en situaciones de mal manejo del ROV o de las condiciones, si no por el propio hecho de realizar un giro como se aprecia en la figura 12, o para tratar de hacer un recorrido sin dejar ninguna zona ciega. Existen diferentes soluciones a este problema, que estaría dentro de post-processing según el libro [22], las técnicas donde se solventa con mayor éxito el overlapping, generando mosaicos de calidad y que además la mayoría de investigadores sobre robótica submarina están actualmente trabajando, tienen que ver técnicas del campo de la visión artificial, podemos ver un ejemplo de mosaico de calidad en la figura 11. Estas técnicas para la realización de mosaicos realizan un registro de imágenes [2], en el campo de la visión por computación un registro de imágenes es el procedimiento de superponer dos o más imágenes de la misma escena tomadas

en diferentes momentos, desde diferentes puntos de vista y/o por diferentes sensores, alineada geométricamente [2]. En otras palabras, es el proceso de transformar una imagen en diferentes perspectivas dentro de un mismo sistema de coordenadas [3]. La mayoría de los algoritmos usados hoy en día han sido desarrollados antes de la década de los noventa, y dentro de ellos los podemos clasificar en dos métodos principales, métodos no simbólicos y métodos simbólicos. [4]

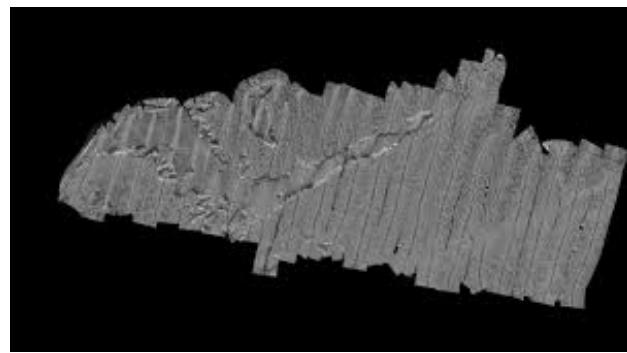


Figura 11: Mosaico de un sonar de barrido lateral

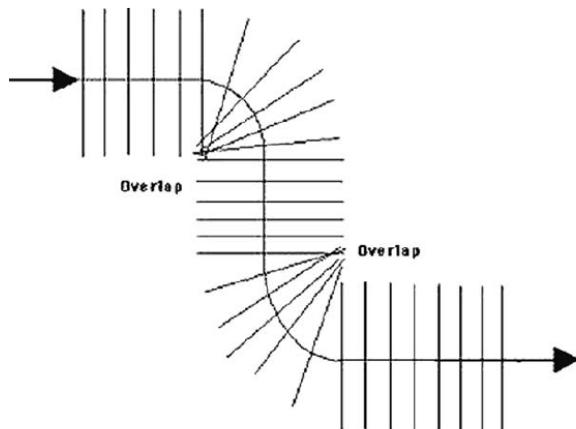


Figura 12: Overlapping provocado por los giros

### 2.3.1. Métodos no simbólicos

Los métodos no simbólicos, son métodos basados en un nivel bajo de información, como puede ser la intensidad de los píxeles. Un método no simbólico implica establecer una relación global entre los píxeles de dos imágenes utilizando solo sus niveles de gris. Los primeros algoritmos desarrollados usando métodos no simbólicos surgen a mediados de los años setenta, utilizando

un criterio basado en la suma de las diferencias cuadradas de la intensidad del nivel de gris. Consideran que las imágenes están curvadas por una transformación básica geométrica o corrompidas por ruido gaussiano blanco adicional y usaban o bien optical flow o phase correlation. [4]

Types of dependency	Similarity measures
Second order dependency [25]	Correlation
	Sum of Absolute Differences (SAD)
	Sum of Squared Differences (SSD)
	Correlation coefficient of Moravec
	Median of grey level differences
	Least median of squared differences
	Seitz measure
	Variance of Squared Differences (VOSD)
	Kurtosis
Linear dependency [25]	Zero Normalized Cross Correlation (ZNCC)
	Normalized Covariance (NCOV)
	Cross correlation coefficient
	Zero Normalized Sum of Squared Differences (ZNSSD)
	Smooth Median Absolute Deviation (SMAD)
	M-estimator (Geman MacClure)
	Pseudo-norm ( $\alpha = 0.5$ )
	Phase correlation [28]
Functional dependency [26]	Woods criterion
	<b>Correlation ratio</b>
Law dependency [27]	Dissimilarity of $\chi^2$
	Distance to independence
	<b>Mutual Information</b>
	Kolmogorov distance
	Kullback Leibler divergence
	K-divergence of Lin
	L-divergence of Lin
	Hellinger distance
	Toussaint distance
	W-divergence of Kagan
	Matusita distance
	Battacharya coefficient
	$\chi^\alpha$ divergence of Vajda
	$\alpha$ order information of Bose Einstein
	$\alpha$ order information of Rényi ( $\alpha = 3$ )
	$\alpha$ order information of Femi Dirac
	Cluster Reward Algorithm (CRA) [13]

Cuadro 1: Medidas de Similitud (Similarity Measure) [4]

Desde la década de los ochenta hasta hoy, ha habido un gran incremento de tipos de sensores, así como la mejora de los ya existentes, por lo que aparecen nueva información a tener en cuenta. En particular, en imágenes por satélite, medicina y submarina. Otro enfoque dentro de los métodos no simbólicos surgió, un enfoque estocástico basándose en una medida de similitud, que han demostrado comportarse mejor para el registro automático de imágenes sonar [4]. Existen muchas medidas de similitud, las que más destacan son:

phase correlation, mutual information y rate correlation. En la tabla 1 se puede ver la gran cantidad de medidas de similitud que existen. Este enfoque estocástico ha demostrado comportarse mejor que el enfoque determinista [4], los métodos deterministas operan de una forma sistemática, generando una coincidencia local por cada pixel (optical flow).

### 2.3.1.1 Optical Flow

Optical flow es la distribución de velocidades aparentes de movimiento de patrones de brillo en una imagen, causado por el movimiento relativo entre un observador (un ojo o una cámara) y la escena. En consecuencia, optical flow puede darnos información importante sobre la disposición espacial de los objetos y la tasa de cambio de estos, influenciados por las condiciones variables del entorno, tales como iluminación, sombras, reflejos y otros efectos luminosos. Las discontinuidades en optical flow puede ayudar a segmentar las imágenes en regiones que corresponden a diferentes objetos, en algunos casos incluso se puede recuperar la forma de ciertos objetos. [5]

El problema de optical flow se ha convertido en uno de los más importantes a resolver en el campo de la visión por computación, dada su gran importancia en campos como la reconstrucción 3D, compresión de vídeo, detección de objetos y navegación robótica.

El desplazamiento de los píxeles no es más que la proyección en una imagen del movimiento tridimensional. Se trata de un problema inverso en donde los valores de algunos parámetros del modelo deben ser obtenidos de los datos observados. Las imágenes son los datos observados y queremos conocer el movimiento que se registra en la escena. Esto provoca que la estimación de optical flow sea un problema mal condicionado ya que puede haber varias soluciones para un mismo desplazamiento, lo que da lugar a cierta ambigüedad [6], por lo que surgen restricciones para evitar esta ambigüedad.

Si representamos una imagen como una aplicación  $I : (x, y, t) \rightarrow I(x, y, t)$  donde  $(x, y)$  representa la coordenada espacial de la imagen y  $t$  el tiempo, se puede ver una secuencia de imágenes como la variación de la intensidad en las coordenadas de la imagen a lo largo del tiempo. El vector desplazamiento se define como la función  $h(x, y, t) = (u(x, y, t), v(x, y, t))^T$  y representa el movimiento horizontal y vertical de los píxeles a través de la sucesión de imágenes. Para detectar la correspondencias de los píxeles entre dos imágenes se suele suponer que alguna propiedad de la imagen no varía a lo largo del tiempo. Esta suposición la vemos representada en la ecuación 5 donde  $f$  es algún tipo de propiedad en la imagen, y  $t+1$  representan dos imágenes de la escena en distintos instantes de tiempo.

$$f(x + u, y + v, t + 1) - f(x, y, t) = 0 \quad (5)$$

La intensidad de los píxeles de una imagen es un valor que indica la cantidad de radiación luminosa reflejada por la superficie de los objetos. Existen muchos modelos para representar los distintos tipos de superficies. Uno de los más simples el modelo lambertiano, que asume un mismo brillo para las diferentes perspectivas, es decir, se mantendrá el mismo valor de intensidad en todas las secuencias de imágenes. Por este motivo, bastaría con sustituir  $f$  por  $I$  en la ecuación 5 para representar esta invarianza. Si analiza la expresión nos damos cuenta de que no es lineal, esta no linealidad la podemos evitar realizando un desarrollo de Taylor de dicha expresión y desecharmos los términos de orden superior. La ecuación resultante es conocida como ecuación de restricción de flujo óptico.

$$I_x u + I_y v + I_t = 0 \quad (6)$$

donde los subíndices indican derivadas parciales. A partir de la expresión 6, no es posible determinar el vector desplazamiento, ya que tenemos dos incógnitas en una sola ecuación.

[6]

### **Clasificación de los Métodos**

En esta sección vamos a realizar un breve recorrido de los algoritmos más importantes que se utilizan en optical flow, y agruparlos basándonos en [7] y [8].

Existen muchos métodos para calcular optical flow, y por ello varios investigadores han aportado sus diferentes clasificaciones, en este trabajo los agruparemos en los tres grupos principales y que más han aportado al campo de optical flow: métodos diferenciales, métodos basados en la frecuencia y métodos basados en la correlación [7].

### **Métodos diferenciales**

Los métodos diferenciales calculan el desplazamiento de los píxeles a partir de las derivadas espaciales (o espaciotemporales) de las intensidades de la imagen. Una limitación que tiene este tipo de métodos es que obliga a que las derivadas sean computables en el dominio de la imagen.

Dentro de los métodos basados en derivadas, podemos encontrar algoritmos basándose en la primera derivada ([5]), y otros en la segunda derivada.

Beauchemin-Barron ([7]) estableció la siguiente subcategoría para los métodos diferenciales: locales, globales, de superficie, de contornos y multirestricciones. Los métodos locales y globales se basan en la ecuación de restricción del flujo óptico (6), como no es posible estimar el flujo óptico únicamente con esa ecuación, es necesario una restricción adicional.

Dentro de los métodos globales, el más representativo es el de Horn y Schunck toman que toman como segunda restricción una restricción de suavidad.

### **Restricción de constancia de brillo, Horn y Schunck**

El cambio de brillo entre dos cuadros consecutivos del video es cero o aproximadamente cero, tomando como referencia la ecuación 6. Para el cálculo de las derivadas parciales espaciales, así como para la derivada temporal se utiliza el método de diferencias finitas. Existen muchas fórmulas para realizar este cálculo, Horn y Schunck propusieron las siguientes ecuaciones 7, 8, 9. Donde E representa el brillo en un punto de la imagen

$$E_x \approx \frac{1}{4} \{ E_{i,j+1,k} - E_{i,j,k} + E_{i+1,j+1,k} - E_{i+1,j,k} \\ + E_{i,j+1,k+1} - E_{i,j,k+1} + E_{i+1,j+1,k+1} - E_{i+1,j,k+1} \}, \quad (7)$$

$$E_y \approx \frac{1}{4} \{ E_{i+1,j,k} - E_{i,j,k} + E_{i+1,j+1,k} - E_{i,j+1,k} \\ + E_{i+1,j,k+1} - E_{i,j,k+1} + E_{i+1,j+1,k+1} - E_{i,j+1,k+1} \}, \quad (8)$$

$$E_t \approx \frac{1}{4} \{ E_{i,j,k+1} - E_{i,j,k} + E_{i+1,j,k+1} - E_{i+1,j,k} \\ + E_{i,j+1,k+1} - E_{i,j+1,k} + E_{i+1,j+1,k+1} - E_{i,j+1,k} \} \quad (9)$$

El objetivo es referenciar el mismo punto en la imagen al mismo tiempo, por lo que es importante que las estimaciones de  $E_x$ ,  $E_y$  y  $E_t$ , sean consistentes. Utilizan un conjunto que les da una estimación de  $E_x$ ,  $E_y$ ,  $E_t$  en un punto en el centro de un cubo formado por ocho mediciones. La relación en el espacio y el tiempo entre estas mediciones se muestra en la figura 13 sacada del artículo [5]. Cada una de las estimaciones es el promedio de cuatro primeras diferencias tomadas sobre las mediciones adyacentes en el cubo. Este método consiste en aproximar la derivada considerando el peso de un conjunto discreto de mediciones de brillo disponibles alrededor del pixel  $E(x,y,t)$ .

Una vez obtenidas las aproximaciones de las derivadas, y al poner una de las velocidad en función de otra en la ecuación 6, se observa que lo que se

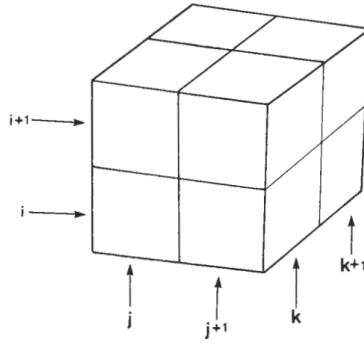


Figura 13: Aquí el índice de columna  $j$  corresponde a la dirección  $x$  en la imagen, el índice de fila  $i$  a la dirección  $y$ , mientras  $k$  se encuentra en la dirección del tiempo.

obtiene es la ecuación de una recta dentro de la cual se encuentra la solución para la determinación de optical flow. Se muestra en la Figura 14 la recta correspondiente a la ecuación  $v = \frac{E_x}{E_y}u - \frac{E_t}{E_y}$  [9]

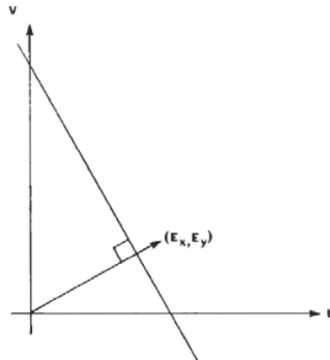


Figura 14: Línea de restricción del flujo óptico comprendida en el plano formado por las velocidades  $u$  y  $v$ . El vector perpendicular a la línea de restricción muestra el gradiente de brillo  $E_x, E_y$ . [5]

Como se observa en la Figura 14, a pesar de disponer de la ecuación que describe la restricción de constancia de brillo, el flujo óptico puede encontrarse a lo largo de la recta descrita en el plano  $u$ - $v$ .

### Restricción de suavidad

La restricción de suavidad utilizada por Horn y Schunck asume, que los pixeles que conforman los objetos de tamaño finito en la imagen (patrones de brillo) tienden a someterse a movimientos rígidos como un todo, por lo

que casi nunca se encuentran píxeles con movimientos independientes de sus vecinos cercanos. Esto genera un campo de velocidades de los patrones de brillo que varía suavemente en casi toda la imagen, debido a determinadas excepciones como en el caso de presencia de texturas. Esta restricción implica la minimización de las derivadas parciales espaciales de las componentes de la velocidad. Para poder realizar esta minimización se puede plantear la suma de los Laplacianos en los ejes  $x$  e  $y$ , como plantea Horn en su artículo de 1981, con las ecuaciones 10 y 11. En estas ecuaciones se muestra una equivalencia donde se consideran promedios locales de las velocidades ( $\bar{u}$  y  $\bar{v}$ ) así como un factor proporcional  $k$ . Para lograr la minimización de los Laplacianos de las componentes de la velocidad, estos deben ser igualados a 0. [9]

$$\nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} (\bar{u}_{i,j,k} - u_{i,j,k}) \quad (10)$$

$$\nabla^2 v = \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} (\bar{v}_{i,j,k} - v_{i,j,k}) \quad (11)$$

Al igual que se utilizaron diferencias finitas para la aproximación de las derivadas parciales, el cálculo del Laplaciano utiliza el mismo método de aproximación, tomando las ponderaciones de un cuadrado de valores de los vecinos cercanos y sustrayéndolo del valor central, como se muestra en la Figura 15.

$\frac{1}{12}$	$\frac{1}{6}$	$\frac{1}{12}$
$\frac{1}{6}$	-1	$\frac{1}{6}$
$\frac{1}{12}$	$\frac{1}{6}$	$\frac{1}{12}$

Figura 15: Cuadrado de ponderaciones con los pesos correspondientes a cada vecino para el cálculo del Laplaciano en las componentes de  $u$  y  $v$  del flujo óptico. [5]

El cálculo de los promedios locales de las componentes de la velocidad se realiza utilizando las ecuaciones 12 y 13, donde se realiza la ponderación planteada en la Figura 15. La restricción de suavidad es una suposición bastante buena para el cálculo de optical flow excepto cuando existen objetos

en la imagen que ocultan a otros, provocando de esta manera una discontinuidad en el flujo [9].

$$\begin{aligned}\bar{u}_{i,j,k} = & \frac{1}{6} \{ u_{i-1,j,k} + u_{i,j+1,k} + u_{i+1,j,k} + u_{i,j-1,k} \} \\ & + \frac{1}{12} \{ u_{i-1,j-1,k} + u_{i-1,j+1,k} + u_{i+1,j+1,k} + u_{i+1,j-1,k} \}\end{aligned}\quad (12)$$

$$\begin{aligned}\bar{v}_{i,j,k} = & \frac{1}{6} \{ v_{i-1,j,k} + v_{i,j+1,k} + v_{i+1,j,k} + v_{i,j-1,k} \} \\ & + \frac{1}{12} \{ v_{i-1,j-1,k} + v_{i-1,j+1,k} + v_{i+1,j+1,k} + v_{i+1,j-1,k} \}\end{aligned}\quad (13)$$

Por otro lado, los métodos locales utilizan la información en una vecindad alrededor de un píxel para estimar su movimiento. El método más representativo de esta familia es el de Lucas-Kanade ([10]), cuyo desarrollo es el mismo que el de Horn, pero difieren en la segunda restricción. Lucas y Kanade, plantean un método que calcula el desplazamiento a partir de la minimización de la ecuación del flujo óptico alrededor de una ventana centrado en un píxel (ecuación 14) donde  $W(x, y)$  es la ventana centrada en el píxel  $(x, y)$  y  $N$  es la vecindad.

$$\sum_{(x,y) \in N} W^2(x, y) (E_x u + E_y v + E_t)^2 \quad (14)$$

El mayor inconveniente de este tipo de métodos (locales) es que sólo es posible detectar el movimiento en aquellas zonas donde exista variaciones en la imagen. En zonas homogéneas, donde puede haber movimiento éste no es detectable. Por ello, los campos de desplazamiento no son densos.

### Métodos basados en la Frecuencia

Los métodos basados en la frecuencia o también llamados, métodos basados en energía espacio-temporal, son métodos que utilizan la transformada de Fourier para calcular el flujo ópticos a través del dominio de la frecuencia, adaptando la ecuación que describe cualquier invarianza (6) en el dominio de la imagen, al dominio de la frecuencia obtenemos la ecuación 15

$$\hat{I}(k, f) - \hat{I}_0(k) \delta(\omega^T k + f) = 0 \quad (15)$$

donde  $\hat{I}_0(k)$  es la transformada de Fourier de  $I(x, y, 0)$ ,  $\delta$  es la delta de Dirac y  $k, f$  es la frecuencia espacio-temporal. Según esta ecuación, cualquier patrón

que se mueva de una imagen a otra por una simple translación, se manifiesta en el dominio de Fourier como un cambio de fase. Este tipo de métodos suele resultar muy útil para la detección del movimiento de objetos que son difíciles de capturar, como es el caso de puntos aleatorios. [6]

### Métodos basados en la correlación

Los métodos basados en la correlación realizan la búsqueda de correspondencias utilizando ventanas o patrones alrededor de cada píxel. La idea que subyace a estos métodos es que es mucho más fácil encontrar las correspondencias entre los píxeles a través de la comparación de regiones entre las imágenes por maximización de alguna medida de similaridad. Una ventaja que tienen estos métodos es que al usar mayor información la búsqueda de las correspondencias es más efectiva.

$$C(\omega) = \int_{\Omega} f(x + \omega) g(x) dx \quad (16)$$

donde  $\omega$  es el desplazamiento del píxel  $x$  y  $\Omega$  es el dominio de la imagen. En el caso discreto la medida de la correlación en un punto que se suele tomar es el siguiente:

$$C(\omega) = \frac{\sum_{\delta\omega=-(a,b)}^{(a,b)} (f(x + \delta\omega) - \bar{f}(x)) (g(x + \omega + \delta\omega) - \bar{g}(x + \omega))}{(2a+1)(2n+1)\sigma_f(x)(x+\omega)} \quad (17)$$

donde  $(a, b)$  representa las dimensiones de la ventana de correlación,  $\bar{f}(x)$  la media de la imagen  $f$  en esa ventana y  $\sigma_f(x)$  la desviación estándar.

[6]

Hemos hecho un breve repaso de optical flow, de sus principales métodos y los dos más usados. Actualmente, optical flow es muy usado en odometría visual, este método es relativamente eficiente con imágenes nítidas, pero en imágenes con poca nitidez, como pasa con las imágenes submarinas donde hay mucho ruido, este método no da buenos resultados, por lo tanto, en la práctica no se hace uso de él, ya que es muy sensible a las variaciones luz y a la oclusión. [4]

#### 2.3.1.2 Phase Correlation

Phase Correlation (PC) se calcula con la transformada rápida de Fourier en dos imágenes, y usar la diferencia entre la fase de los componentes espec-

trales, para evaluar el desplazamiento espacial entre ambas imágenes. Un cambio de fase en el dominio del espacio, correspondería a un desplazamiento. Las diferencias de fase están sujetas a una transformación inversa que determina una superficie de correlación, compuesta de picos cuyas posiciones corresponden a la norma de movimientos entre imágenes. Esto se puede describir de la siguiente manera: dejando  $f_1(x, y)$  y  $f_2(x, y)$  ser dos imágenes que se diferencian únicamente por un desplazamiento  $(x_0, y_0)$

$$f_2(x, y) = f_1(x - x_0, y - y_0) \quad (18)$$

$$F_2(\varepsilon, \eta) = e^{-j2\pi(\varepsilon x_0 + \eta y_0)} * F_1(\varepsilon, \eta) \quad (19)$$

La ecuación 18 está relacionada con su correspondiente transformada de Fourier 19. El espectro de potencia cruzada de dos imágenes  $f$  y  $f'$  con transformada de Fourier  $F$  y  $F'$  se define en la ecuación 20

$$\frac{F(\varepsilon, \eta)F' * (\varepsilon, \eta)}{|F(\varepsilon, \eta)F'(\varepsilon, \eta)|} = e^{-j2\pi(\varepsilon x_0 + \eta y_0)} \quad (20)$$

donde  $F^*$  es el conjugado de  $F$ . El teorema de desplazamiento garantiza que la fase del espectro de potencia cruzada es equivalente a la diferencia de fase entre las dos imágenes. Al tomar la transformada inversa de Fourier de la representación en el dominio de la frecuencia, obtendremos una función que es un impulso, o llamada correlación superficial; es decir, es aproximadamente cero en todas partes, excepto donde el desplazamiento es adecuado para el registro óptimo. Luego aparecen picos en estos lugares, debemos de elegir cual de estos picos es el que más apropiado para representar el movimiento real. Luego se aplica una coincidencia de bloque orientada (block matching) en las imágenes de nivel de gris correspondientes (normalizadas), que compara cada conjunto candidato, centrado en uno de los picos de superficie de correlación en la primera imagen, con su correspondiente en la segunda imagen de nivel de gris. De hecho, un vector candidato se define desde el centro de la imagen referenciada a cada uno de los picos de la superficie de correlación. El máximo de correlación en los niveles de gris normalizados revela el vector de movimiento válido. En consecuencia, el uso de una estrategia de desplazamiento y correlación nos permite discriminar los vectores válidos de los falsos, entre los vectores candidatos. En resumen, el proceso de registro se realiza en dos etapas que son un análisis espectral, seguido de un método de coincidencia de bloque (block matching de correlación). La principal ventaja de phase correlation es su excelente robustez contra ruido aleatorio. [11]

### 2.3.1.3 Mutual Information

Mutual Information se ha hecho popular en diferentes campos, como en el registro de imágenes médicas o radares, particularmente debido a su gran indiferencia a los cambios de iluminación. Mutual Information es una medida de dependencia estadística entre dos variables aleatorias A y B, y está dada por:

$$I(A, B) = H(A) + H(B) - H(A, B) \quad (21)$$

Donde  $H(A)$  es la entropía de una variable aleatoria, la cual esta definida como  $H(A) = \int_{-\infty}^{+\infty} p(A) \ln(p(A)) dA$  y la entropía conjunta es  $H(A, B) = -\int_{-\infty}^{+\infty} p(A, B) \ln(p(A, B)) dAdB$ . La entropía de una variable aleatoria X mide la cantidad de 'información' proporcionada por una observación de X. A partir de esta definición, se observa que cuanto menos probable es un evento, más información recibimos cuando ocurre.

Uno de los inconvenientes de mutual information es que no tiene en cuenta la relación espacial entre píxeles; de hecho, los histogramas no proporcionan información espacial sobre los píxeles de la imagen. Consecuentemente, muchos autores trabajaron en el tema para mejorar la información mutua, Russakoff sugirió tener en cuenta una vecindad de píxeles en el cálculo de MI, Marti propuso usar matrices de coeficiente de gris en lugar de histogramas. [11]

### 2.3.1.4 Correlation Ratio

El Correlation Ratio mide la dependencia funcional entre dos variables X e Y. Toma valores entre 0 (sin dependencia funcional) y 1 (dependencia puramente determinista). Se define por:

$$\eta(X|Y) = 1 - \frac{Var[(Y - (Y|X))]^2}{Var(Y)} \quad (22)$$

$\eta(X|Y)$  es invariante a los cambios multiplicativos en Y, es decir,  $\forall K, \eta(kY|X) = \eta(Y|X)$ . A diferencia de la información mutua, la relación de correlación es asimétrica y depende de qué variable se use para predecir la otra, de hecho, las variables no juegan el mismo papel en la relación funcional; eso significa  $\eta(Y|X) \neq \eta(X|Y)$ . Algunos autores proponen usar un ratio de correlation simétrico para encontrar una forma de evitar este inconveniente al evaluar la varianza condicional normalizada como la suma de dos razones de correlación.

$$\eta_{symmetric} = \eta(X|Y) + \eta(Y|X) \quad (23)$$

En el artículo [4], podemos encontrar la realización de mosaicos utilizando una medida de similaridad SM, en concreto 35 medidas, de las cuales intenta buscar identificar que medida es la más adecuada para la generación de mosaicos, se generan mediante un algoritmo de block matching, para realización correcta del mosaico, lleva a cabo una transformación rígida global, y una elastica local. Los resultados experimentales en el trabajo de [4] mostraron que un mosaico con buen efecto solo se podía obtener cuando había una ligera variación en la rotación y en la escala entre las dos imágenes coincidentes. Sin embargo, según las intensidades de píxeles o sus distribuciones estadísticas, el rendimiento de los métodos no simbólicos se verá influenciado por el ruido complejo del entorno oceánico y las ganancias variacionales. Además, el estudio concluye que de todas las medidas las que mejor resultado dieron fue CR (Correlation Ratio) y MI (Mutual Information), capaces de complementarse una a la otra. [3]

### 2.3.2. Métodos simbólicos

Los métodos simbólicos se basan principalmente en la extracción y coincidencia de características. Las características incluyen: formas, bordes, detección de esquinas, etc. Los algoritmos que tienen propiedades de invariancias en rotación, escala, afinidad y perspectiva, han demostrado tener resultados muy positivos [3]. Los algoritmos más conocidos que nos encontramos para estas aplicaciones son: SIFT, SURF y ORB.

Estos algoritmos también han sido usados para navegación, por ejemplo, en el artículo [8], hace uso de SIFT para el registro de imágenes, con el fin de mejorar los errores de navegación de los sistemas iniciales, ya que el registro de imagen proporciona un feedback capaz de usarse para compensar dichos errores. En este caso, no genera un mosaico, pero la conclusión que podemos sacar es que se abren vías para el desarrollo de un sistema complementario capaz de corregir los errores provocados por los sistemas iniciales a partir de imágenes de ultra sonidos.

En algunos casos como en el artículo [8] o [12] hacen uso del algoritmo RANSAC, debido a que se puede hacer uso de diferentes herusitcas para la coincidencia entre imágenes, y usando una heurística como nearest-neighbour tiende a dar muchos falsos positivos, por lo que para poder lidiar con esos outliers y estimar correctamente la transformación del modelo aplican el algoritmo RANSAC.

### 2.3.2.1 SIFT

SIFT (Scale-Invariant Feature Transform), es ampliamente usado en visión por computación y fue desarrollado por David Lowe [13]. SIFT es capaz de extraer puntos de interés (keypoints) de una imagen aunque esta esté rotada, escalada, sometida a cambios de iluminación, ruido y pequeños cambios de pose o perspectiva. Esto se consigue a partir de características locales, que se almacenan en los descriptores, los descriptores tratan de describir localmente zonas importantes de la imagen con determinadas variables, entre ellas el gradiente.

El algoritmo está estructurado en cuatro fases:

- Scale-space extrema detection
- Keypoint localization
- Orientation assignment
- Keypoint descriptor

#### Scale-space extrema detection

En esta fase, se busca un primer conjunto de puntos de interés, en las etapas posteriores, estos puntos de interés o keypoints, se irán descartando por no cumplir ciertos requisitos. En esta etapa, la búsqueda se realiza sobre todas las localizaciones y todas las escalas de la imagen. Para detectar localizaciones invariantes a cambios de escala, se utiliza la función conocida como scale-space:  $L(x, y, \sigma)$ , utilizaremos una función gausiana para obtener dicha función a partir de nuestra imagen original ( $I(x, y)$ ), en la ecuación 24 podemos ver como quedaría la expresión, donde el operador \* indica la convolución entre la imagen y la gausiana  $G$ .

$$L = (x, y, \sigma) * I(x, y) \quad (24)$$

Para calcular todo el espacio  $L(x, y, \sigma)$  hay que construir una pirámide gausiana, convolucionando con diferentes filtros  $G(x, y, \sigma)$  variando el parámetro  $\sigma$ . Definiremos dos términos que se hacen uso en esta fase:

- Octava: Conjunto de imágenes del espacio L con el mismo tamaño que difieren en el filtrado  $\sigma$  con el que han sido obtenidas.
- Escala: Conjunto de imágenes del espacio L filtradas con el mismo parámetro  $\sigma$  pero con diferentes tamaños.

Existe una condición que debe cumplirse en este proceso,  $\sigma_i = k^{i-1} = 2^{1/n^0 \text{escalas}-2}$ .

Una vez completada toda la pirámide, para detectar puntos de interés estables, se utiliza la función Difference of Gaussian (DoG) ya que es una función que hace una aproximación cercana a escala-normalizada laplaciana gaussiana. DoG se calcula restando imágenes vecinas de una misma octava, ecuación 25.

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) \end{aligned} \quad (25)$$

Lindeberg demostró que la normalización de los laplacianos con el factor  $\sigma^2$  es necesaria para la invariancia de escala real. Mikolajczyk (2002) descubrió que los máximos y mínimos de  $\sigma^2 \nabla^2$  produce unas características de imagen más estables en comparación con otras funciones i.e. gradient, Hessian, or Harris corner function.

A partir de los cálculos anteriores, se calculan los máximos y mínimos locales del espacio  $D(x, y, \sigma)$  (local extrema detection).

[13]

### Keypoint localization

La segunda fase del método SIFT se centra en almacenar toda la información disponible de cada keypoint. Es decir, para cada punto de interés encontrado se guardará a qué escala y octava de la pirámide pertenece, y su posición [fila, columna] dentro de la imagen correspondiente. Esta información permite rechazar puntos que tienen poco contraste (y por lo tanto son sensibles al ruido) o que están mal localizados a lo largo de un borde

Para descartar los puntos con un contraste bajo se utiliza la expresión de Taylor de la función  $D(x, y, \sigma)$ , hasta el término cuadrático. Si definimos un punto  $p$  de los seleccionados como keypoint en el apartado anterior, tal que  $p = (x, y, \sigma)^T$ , se obtiene la expresión 26

$$D(p) = D + \frac{\delta D^T}{\delta p} p + \frac{1}{2} p^T \frac{\delta^2 D}{\delta p^2} p \quad (26)$$

El extremo  $\hat{p}$  es determinado por la derivada de su función respecto a  $p$  y estableciéndola a cero, obtenemos:

$$\hat{p} = - \left( \frac{\delta^2 D^{-1}}{\delta p^2} \cdot \frac{\delta D}{\delta p} \right) \quad (27)$$

La función  $D(\hat{p})$  es muy útil para descartar puntos de bajo contraste, esta puede ser obtenida sustituyendo la ecuación 26 dentro de la ecuación 25, dando como resultado la ecuación 28. Para ello, se establece un umbral mínimo al que deben llegar los keypoints para no ser rechazados.

$$D(\hat{p}) = D + \frac{1}{2} \frac{\delta D^T}{p} \hat{p} \quad (28)$$

La función  $D(x, y, \sigma)$  tiene una gran respuesta ante puntos situados sobre los bordes. Muchos de esos puntos no serán suficientemente estables. Un keypoint que esté situado sobre un borde tendrá una respuesta muy pobre en la dirección del borde, pero muy elevada en la dirección perpendicular. La realización de este computo es el mismo que el que propusieron en 1988 Harris and Stephens. En esta fasa, mucho de los keypoints deducidos en la fase anterior son eliminados, quedando los que realmente son invariantes al cambio.

[13]

### Orientation assignment

Esta etapa del algoritmo, se centra en calcular las orientaciones de cada keypoint. Una vez terminada esta etapa, se podrá dar paso a la creación de los descriptores (keypoint descriptor). Al asignar una orientación coherente a cada keypoints en función de las propiedades de la imagen local, el descriptor del keypoint se puede representar en relación con esta orientación y, por lo tanto, lograr la invariabilidad en la rotación de la imagen.

La escala del keypoint es usada para seleccionar la imagen suavizada de Gauss,  $L$ , con la escala más cercada, de modo que todos los cálculos se realicen de manera invariable. Para cada muestra de imagen,  $L(x, y)$ , a esta escala, la magnitud del gradiente,  $m(x, y)$ , y la orientación,  $\theta(x, y)$ , pre-calculada usando la diferencia de pixeles:

$$\begin{aligned} m(x, y) &= \sqrt{(L(x+1, y) - L(x-1, y))^2 + (L(x, y+1) - L(x, y-1))^2} \\ \theta(x, y) &= \tan^{-1}((L(x, y+1) - L(x, y-1)) / (L(x+1, y) - L(x-1, y))) \end{aligned} \quad (29)$$

Una vez realizado el cálculo, se genera un histograma a partir de las orientaciones de gradiente, un para cada keypoint. Los picos más altos de cada histograma son las direcciones dominantes de los gradientes locales, y por tanto, la orientación final, en ocasiones no nos quedaremos con el pico más alto.

[13]

### **Keypoint Descriptor**

En las operaciones anteriores, hemos asignado a cada keypoint una localización, escala y orientación, estos parámetros forman un sistema de coordenadas 2D local, en el que se describe localmente cada región de la imagen, por lo que proporciona la invariancia a esos mismos parámetros.

El siguiente paso es proporcionar la invariabilidad, sobre variaciones como podrían ser cambios de luz o punto de vista 3D. El vector se normaliza a la unidad, un cambio en el contraste de la imagen en el que cada valor de píxel se multiplica por una constante multiplicará los gradientes por la misma constante, por lo que este cambio de contraste se cancelará mediante la normalización del vector. Un cambio de brillo en el que se agrega una constante a cada píxel de la imagen no afectará los valores de gradiente, ya que se calculan a partir de las diferencias de píxeles. Por lo tanto, el descriptor es invariante para los cambios afines en la iluminación. Sin embargo, los cambios de iluminación no lineales también pueden ocurrir, debido a la saturación de la cámara, o debido a cambios de iluminación que afectan a las superficies 3D, con diferentes orientaciones en diferentes cantidades. Estos efectos pueden causar un gran cambio en las magnitudes relativas de algunos gradientes, pero es menos probable que afecten las orientaciones de los gradientes. Por lo tanto, reducimos la influencia de grandes magnitudes de gradiente al limitar los valores en el vector de características de la unidad para que cada uno no sea mayor que 0.2, y luego renormalizar a la longitud de la unidad. Esto significa que igualar las magnitudes para grandes gradientes ya no es tan importante, y que la distribución de orientaciones tiene mayor énfasis. El valor de 0.2 se determinó experimentalmente usando imágenes que contenían diferentes iluminaciones para los mismos objetos 3D.

[13]

Una vez realizadas estas modificaciones, el proceso de construcción de los descriptores se da por finalizado. El método SIFT posteriormente comparará cada uno de los descriptores.

#### **2.3.2.2 SURF**

Speeded up robust features o SURF, es un algoritmo derivado de SIFT, nació para solucionar los problemas que tenía SIFT en tiempo de ejecución, que es donde se hace la extracción de características y la comparación de descriptores, por lo que según su autor en estos aspectos, SURF es más eficiente y robusto.

El algoritmo SURF consta de tres etapas:

- interest Point Detection.
- interest Point Description.
- Search (matching relationship)

No vamos a entrar en detalle dentro de cada punto, ya que el algoritmo es muy similar a SIFT, lo que si se va a describir es en que se diferencian.

En SURF se aproxima el DoG con box filters. En vez de utilizar gaussianas para promediar la imagen, se utilizan cuadrados (aproximaciones). Hacer la convolución de la imagen con un cuadrado es mucho más rápido si se utiliza la imagen integral, esto se puede hacer en paralelo para diferentes escalas. SURF utiliza un detector BLOB que se basa en la matriz de Hesse para encontrar los puntos de interés, para la asignación de orientación, utiliza respuestas wavelet en direcciones horizontales y verticales mediante la aplicación de pesos gaussianos adecuados. La descripción de la característica, también se utiliza las respuestas wavelet. Se selecciona un vecindario alrededor del punto clave y se divide en subregiones y luego, para cada subregión, las respuestas wavelet se toman y se representan para obtener el descriptor de característica SURF. El signo de Laplaciano que ya se calcula en la detección se utiliza para los puntos de interés subyacentes, que distingue las manchas brillantes sobre fondos oscuros. En caso de coincidencia, las características se comparan solo si tienen el mismo tipo de contraste (basado en el signo) que permite una coincidencia más rápida. [14]

### 2.3.2.3 ORB

Oriented FAST and rotated BRIEF u ORB, creado por Ethan Rublee en 2011 [15], según los autores este algoritmo es más eficiente y robusto que SURF, y por tanto, que SIFT. Resultado de la fusión del detector de puntos de interés FAST y del descriptor BRIEF. El algoritmo no se ve afectado significativamente por el ruido de la imagen. ORB es capaz de utilizarse en tiempo real.

### FAST

Features from Accelerated Segment Test o FAST, es una técnica que localiza bordes y esquinas. Este método recibe como parámetro el umbral de la diferencia de intensidad entre el píxel central y aquellos situados en un círculo alrededor del centro. Este detector no produce una medida que cuantifique que tanto un punto puede ser considerado como una esquina, y de esta manera también produce altas respuestas a lo largo de bordes y esquinas. Por

esta razón, se emplea una medida empleada en la técnica de detección de esquinas de Harris para ordenar los puntos de acuerdo a su importancia, resaltando aquellos correspondientes a esquinas y obtener un número determinado de puntos de interés. así, en el algoritmo debe establecerse un umbral suficientemente bajo para obtener más de los N puntos de interés deseados, para, una vez hallados, ordenarlos según la medida de Harris y seleccionar únicamente los N puntos más significativos. [16]

## BRIEF

Binary Robust Independent Elementary Features o BRIEF, BRIEF es un descriptor binario, basado en simples tests de diferencia de intensidad. Existen varias técnicas propuestas en la literatura para acelerar el proceso de encontrar correspondencias y reducir consumo de memoria, como reducción dimensional (Principal Component Analysis), cuantización usando algunos bits de los descriptores existentes e incluso binarización. Pero todas estas aproximaciones requieren primero la computación del descriptor entero mientras que BRIEF computa directamente los vectores binarios para trozos de imágenes. El ser un descriptor binario es lo que le hace más rápido que otros enfoques distintos como el que usa SIFT o SURF. [16]

En la práctica en nuestro tema sobre la generación de mosaicos con imágenes submarinas a través de medios acústicos, el artículo [12] compara los tres algoritmos, SIFT, SURF y ORB, sobre las distintas transformaciones que han de hacerse, rotación, translación, escala. ORB se comporta bien en ciertas situaciones, como por ejemplo, donde hay pocas sombras, pero en general SIFT, está por encima, incluso de SURF.

### 2.4. Métodos no simbólicos vs métodos simbólicos

Los métodos simbólicos son más intuitivos que los métodos no simbólicos, ya que como se han descrito, se enfocan principalmente en dos pasos: un proceso de segmentación asociado con un paso de clasificación para detectar y etiquetar características, y por último un proceso de correspondencia entre dichas características extraídas. Los principales inconvenientes de estos métodos son el tiempo el tiempo requerido para la segmentación y extracción de características.

Los métodos no simbólicos se basan en información de bajo nivel, mientras que los simbólicos en información de alto nivel. Los métodos no simbólicos, destacan principalmente en su rapidez y su mayor tolerancia hasta cierto punto a distorsiones, variaciones de iluminación y ruido en las imágenes del sonar. [4]

### 3. Metodología y Herramientas

#### 3.1. Metodología

La metodología desarrollada han conllevado una búsqueda y estudio para determinar que procedimientos llevar a cabo para la realización de mosaicos con un sonar de barrido lateral, y se ha llegado a la conclusión de que no existe una metodología estándar, pero tanto en este trabajo como en trabajos anteriores realizados por investigadores, se realiza una metodología o procedimientos muy parecidos e intuitivos, que se puede ver en la figura 16 . La metodología propuesta está basada en la descrita en el estado del arte en el libro [22]

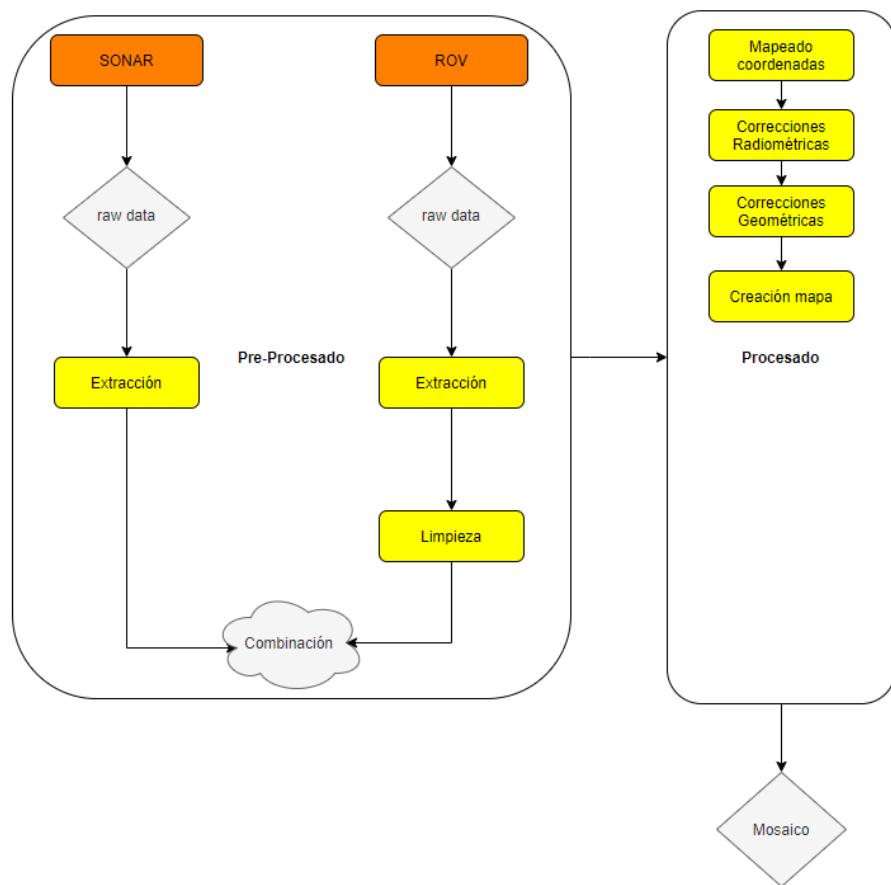


Figura 16: Metodología.

La metodología o procedimientos, la podemos dividir en dos bloques: pre-procesado y procesado. En el bloque del pre-procesado se encuentran, la

adquisición, combinación de los datos y limpieza de los datos, por otro lado, en el bloque procesado, podemos encontrar el mapeado de coordenadas UTM a coordenadas en píxeles, correcciones radiométricas, geométricas y la creación del mosaico.

Como se puede observar en la figura 16, el procedimiento empieza con la generación de datos de los dos flujos: Sonar y ROV. En este punto nos encontramos realizando la misión, y recopilando todos los datos. Podemos intuir que son dos flujos independientes esto quiere decir que, aunque el sonar este incorporado en el ROV, la extracción de datos se hace de forma independiente, es decir, su funcionamiento se hace por distinto software. Más adelante, se detallará la razón de esto y los problemas que conlleva. Por otro lado, cabe destacar que algunos autores de trabajos relacionados con imágenes sonar deciden no realizar ningún tipo de corrección en la parte de procesado, mostrando las imágenes sonar exactamente como se reciben.

## 3.2. Herramientas

La realización de este trabajo, no solo ha requerido de herramientas software, sino también hardware. En esta sección vamos a detallar brevemente las herramientas que han sido necesarias a lo largo de todo el proyecto, es decir, todas aquellas herramientas necesarias desde la adquisición de los datos hasta la generación del mosaico.

### 3.2.1. Hardware

#### ROV

Por la parte hardware, se ha hecho uso de un vehículo ROV, llamado Sibiu Pro de la empresa Nido Robotics, con la colaboración del Grupo de Investigación de Ingeniería Aplicada de la Universidad de Murcia están realizando proyectos de investigación. El más destacado entre todas las investigaciones que se están actualmente desarrollando, es el de un posicionamiento submarino, con un equipamiento de bajo coste. Este trabajo aprovecha dicho posicionamiento para realizar los mosaicos con imágenes sonar.

El equipamiento del que dispone este ROV es el siguiente:

- Ecosonda monohaz. Ping1D BlueRobotics.
- DVL. DVL1000 Nortek.
- Sonar 360 grados. Ping360 BlueRobotics
- GPS. Vectornav vn-200.

Frecuencia	680 kHz.
Horizontal beamwidth	0.7°.
Vertical beamwidth	60°.
Rango	10 a 100 metros.
Máxima profundidad	100 metros.

Cuadro 2: Especificaciones sonar barrido lateral BR-ROV

- Iniciales. Vectornav vn-200.
- Sensor presión. Bar30 Bluerobotics.
- Cámara. Low-Light HD USB Camera BlueRobotics.
- Sonar de barrido lateral. Deep Vision BR-ROV.



Figura 17: SibiuNano y SibiuPro.

### Sonar barrido lateral

Por otro lado, junto con el ROV uno de los pilares de este trabajo es el sonar de barrido lateral. El sonar de barrido lateral es de la empresa Deep Vision con sede en Suecia, encargados de crear sistemas sonars de bajo coste y bajo consumo, con un alto rendimiento. El modelo de sonar de barrido lateral con el que se ha trabajado es el BR-ROV, las características más importantes de este sonar de barrido lateral se pueden ver en la tabla 2

Para que el dispositivo funcione correctamente, es necesario que la altura sea un 10 % del rango sobre el que se está operando, es decir, si hemos establecido un rango de 20 metros para capturar el fondo marino, la distancia mínima que debemos mantener con el suelo es de 2 metros. Cabe destacar, que toda la gama de productos de Deep Vision es completamente privativo, se intentó

contactar con la empresa para que nos indicara el protocolo de comunicación que se usaba, con el fin de poder integrarlo dentro del software del ROV, y así poder tener todo totalmente sincronizado, velocidad, posiciones, pose. Por desgracia, la empresa no quiso compartir el protocolo, conllevando a una serie de problemas que se pueden intuir, pero que en secciones posteriores se verá con más detalle.



Figura 18: Deep Vision BR-ROV.

### 3.2.2. Software

#### DasBoot

DasBoot es el software que el Grupo de Investigación de Ingeniería Aplicada ha desarrollado para el control y recopilación de datos del AUV, así como, diferentes utilidades como reconocimiento de marcadores, mapeado del fondo marino o batimetría y posicionamiento submarino.

En este trabajo se hace uso de este software, para la recopilación de datos como el posicionamiento, velocidad y pose. Este software permite grabar las misiones, y reproducirlas como si de una simulación se tratase, esto tiene la gran ventaja de poder adquirir los datos después de realizar la misión, así como de añadir nuevas funcionalidades sin tener que volver a realizar otra misión.



Figura 19: Interfaz DasBoot.

## **DeepView LT**

DeepView LT, es el software de control, grabación y reproducción del sonar de barrido lateral. Permite medir distancias, alturas y hacer zoom sobre la imagen generada, existen otras versiones del mismo software que son capaces de georeferenciar e incluso realizar mosaicos, tal y como se busca en este trabajo, la versión utilizada ha sido DeepView LT, debido a que es la que venía con el dispositivo de forma gratuita, Deep Vision también pone a disposición una versión completamente gratuita que solo es capaz de reproducir grabaciones. La grabación de archivos se guarda en un formato de archivo propio binario, siendo incapaz de poder leer fichero para poder manipularlo, otros dispositivos de barrido lateral con su propio software graban en formatos de archivos estándar, siendo abierto y que permiten su lectura.

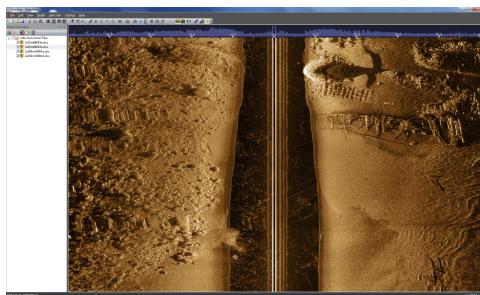


Figura 20: Interfaz Deep View LT.

## **Lenguaje y librerías utilizadas**

El desarrollo de este trabajo se ha realizado bajo el lenguaje de programación python, un lenguaje ampliamente conocido y asentado entre la comunidad de programadores. La decisión por optar por este lenguaje, es debido a su sencillez a la hora de programar y comodidad para trabajar con matrices, ya que posee librerías muy potentes como numpy. Además, de la librería numpy, también se ha hecho uso de OpenCV, una librería muy famosa en el campo de la visión artificial. No se va a entrar en detalle a hablar sobre las distintas tecnologías mencionadas, pues se sobreentiende que el lector ya está familiarizado.

## 4. Diseño y Resolución

En este capítulo vamos a ver como se han desarrollado los objetivos y subobjetivos mencionados en la introducción, así como el flujo de desarrollo planteado en la sección de metodología. Una vez visto en detalle el proceso para la generación de mosaicos, veremos los resultados obtenidos, teniendo como resultado un mapa del fondo marino de la trayectoria realizada por el ROV.

### 4.1. Pre-Procesado

Este es el primer bloque de la metodología planteada, en este bloque se detalla los procesos de: La misión a realizar, la captación de los datos generados por los distintos dispositivos y la combinación de estos.

#### 4.1.1. Misión

El primer paso para generar el mosaico del fondo marino es establecer el recorrido que se va a realizar con el ROV durante misión. La ubicación en la que se desarrolla la misión es en la central térmica de Endesa en Carboneras, Almería. El escenario que se plantea es la generación de un mosaico del fondo para la búsqueda de ánforas que se depositan en el fondo para posteriormente venderlas, en la figura 21 vemos exactamente la zona donde el ROV realizará el recorrido.



Figura 21: Ubicación de la misión.

Se ha subrayado en varias ocasiones, en capítulos anteriores que fijar un recorrido es muy importante, pues la calidad del mapa que se genere con el

mosaico de imágenes depende prácticamente de un buen recorrido, por ello unos requisitos casi imprescindibles a la hora realizar el recorrido son:

- No tener giros bruscos.
- Mantener una velocidad constante.
- No tener cambios significativos de profundidad.

Sino se tienen en cuenta estos requisitos obtendríamos imágenes totalmente distorsionadas, normalmente se dan situaciones donde es inevitable tener perturbaciones en el recorrido provocadas, por ejemplo, por la corriente del mar, pero perturbaciones que no provocan una distorsión total de la imagen son posibles de corregir. En el estado del arte hemos hecho un repaso de cómo algunos autores de trabajos relacionados con la robótica submarina resuelven algunas de estas distorsiones.

Durante la misión es donde se van a generar los diferentes datos que posteriormente, en la reproducción recopilaremos, en la figura 22 se puede apreciar el recorrido que ha realizado el ROV, al final de este capítulo veremos el mismo recorrido con las imágenes del fondo marino proporcionadas por el sonar.



Figura 22: Recorrido del ROV visualizado en DasBoot.

#### 4.1.2. Extracción, limpieza y combinación de los datos

En esta sección vamos a hacer la extracción, limpieza y combinación de datos, como bien se ha podido ver en el flujo del procedimiento en el capítulo 3 en la sección metodología, la extracción de datos se hace de forma independiente. Por tanto, por un lado debemos extraer los datos del ROV y por otro, los del sonar de barrido lateral, realizar la limpieza oportuna y combinarlos. Además, se comentará los distintos problemas que han surgido

con el sonar de barrido lateral y como se han tomado medidas para que este trabajo pudiese seguir adelante.

#### 4.1.2.1 Extracción datos ROV

Como se ha dicho anteriormente, una vez realizada la misión y gracias al sistema de grabación y reproducción de misiones implementado en DasBoot, somos capaces de realizar la extracción de datos del ROV sin necesidad de hacerlo durante la misión. La información que nos proporcionan el ROV se vuelca a tres ficheros, donde se distinguen por:

##### Fichero Posición

En el fichero posición almacenamos las coordenadas UTM del recorrido que ha seguido el ROV. Las coordenadas UTM te devuelven el norte y el este, que corresponde a  $x$  e  $y$ , ya que están basadas en una proyección cartográfica. Estas coordenadas vienen dadas inicialmente por el GPS cuando el ROV emerge a la superficie, de esta forma captamos la posición inicial para posteriormente, utilizar navegación por estima cuando se encuentra sumergido.

##### Fichero pose

En el fichero pose se almacenan las rotaciones pitch, roll y yaw del ROV. Estas rotaciones son muy utilizadas en la navegación, sirven para conocer la orientación del ROV en las tres dimensiones. En nuestro caso: el pitch corresponde a un giro alrededor del eje  $y$ , el roll alrededor del eje  $x$  y el yaw un giro alrededor del eje  $z$ .

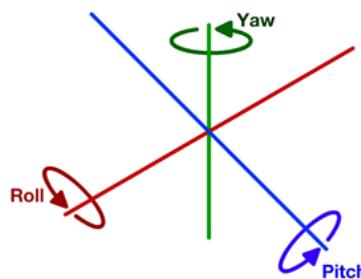


Figura 23: Ilustración del Pitch, Roll y Yaw.

## Fichero velocidad

En este fichero almacenamos la velocidad lineal y lateral, en metros por segundo. Para la realización de esta misión el ROV estaba equipado con un DVL, un Doppler Velocity Logger es un dispositivo que utilizando el efecto doppler es capaz de medir la velocidad del ROV respecto al fondo. La velocidad obtenida viene de este dispositivo, en caso de que no tuviéramos este dispositivo se estaría utilizando una velocidad virtual, lo que provocaría un error mucho más grande en la navegación por estima. Cabe destacar que una navegación por estima tiene un error que se va acumulando a lo largo del tiempo, por lo que, dependiendo de los distintos sensores que dispongamos podremos minimizar más o menos ese error, un buen DVL es capaz de minimizar este error lo suficiente siendo despreciable el margen de error con la posición real.

La lectura de los distintos ficheros se realiza desde la herramienta creada en python, el fichero se procesa línea por línea, teniendo como separador los espacios. En la figura 24 podemos ver un ejemplo del fichero pose, y un extracto del código de la lectura del fichero en el listado de código 1.

```
-0.0029670597283903604 -0.04886921905584124 -2.8231030497407206  
-0.0029670597283903604 -0.04886921905584124 -2.8231030497407206  
-0.0029670597283903604 -0.04886921905584124 -2.8231030497407206  
-0.0054105206811824215 -0.04555309347705201 -2.8249191531353643  
-0.0054105206811824215 -0.04555309347705201 -2.8249191531353643  
-0.0054105206811824215 -0.04555309347705201 -2.8249191531353643  
-0.010122909661567111 -0.0427605667386108 -2.828933410414956
```

Figura 24: Estructura fichero pose (pitch, roll, yaw).

```
1 @staticmethod  
2     def ReadFromFile(filename):  
3         file = open(filename, "r+")  
4         temp = file.read().splitlines()  
5         resultPose = []  
6         for line in temp:  
7             resultPose.append(Pose(line.split()[0],  
8                                     line.split()[1], line.split()[2]))  
9     return resultPose
```

Listing 1: Lectura de fichero

### 4.1.2.2 Extracción datos sonar

Los dispositivos sonar normalmente devuelven un array de amplitudes por cada eco recibido del barrido que van realizando, en este caso la extracción de datos del sonar planteó un serio problema a este trabajo, pues como se ha venido comentando en capítulos anteriores, el sonar es un dispositivo totalmente privativo, la empresa no quiso dar información del protocolo de comunicación, ni del formato del fichero, lo que supuso que la extracción

de datos resultase un reto. Esto supone un gran inconveniente pues existen correcciones que solo se pueden hacer si se tienen la salida de las amplitudes con sus tiempos (ida y vuelta del tiempo del haz), además, una desventaja muy importante es la falta de sincronización con los datos del ROV. Descartado el intento de Ingeniería inversa, tanto en el protocolo como en la lectura del fichero por el tiempo que supondría, la única alternativa que quedaba para poder extraer datos ha sido la grabación de la interfaz del software que venía con el sonar, Deep View. En la figura 20 se puede ver la interfaz del Deep View, lo que se ha grabado es la imagen de salida del sonar de barrido lateral, quedando como se ve en la figura 25, la grabación se ha realizado con el programa OBS, y mientras se hacia la grabación nos dimos cuenta de que la reproducción con Deep View se hacia al contrario, es decir, empezaba por el final, por lo que se ha tenido que dar la vuelta con un programa de edición de vídeo como es OpenShot. La librería OpenCV es la que se usa para el procesamiento del vídeo que se ha grabado, frame por frame, pero teniendo en cuenta que de un frame a otro apenas hay nuevas líneas de píxeles, pues la reproducción del vídeo se reproduce en forma de arrastre, por lo que para obtener nuevos frames totalmente diferentes y poder capturar cada línea, se ha contabilizado cuanto tarda una línea de pixel en desaparecer del plano secuencia.

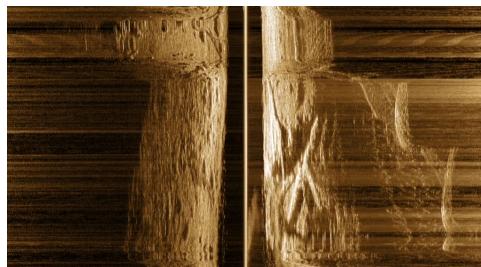


Figura 25: Imagen salida del sonar de barrido lateral.

En resumen, en vez de tener un array de amplitudes por cada línea barrida por el sonar con la que poco a poco van formando una imagen, donde tenemos control total del dispositivo, tenemos la grabación de un vídeo y la extracción se hace línea por línea de ese vídeo.

#### 4.1.2.3 Limpieza de los datos

El proceso de limpieza de datos, es una parte fundamental en cualquier campo, ya sea en procesos ETL (Extracción, Transformación y Carga) de la rama del Business Intelligence o Big Data, así como del machine learning. Pues tener unos datos de calidad es clave y se va a haber reflejado en este caso, en un mosaico de calidad. Los datos obtenidos por el ROV, en general

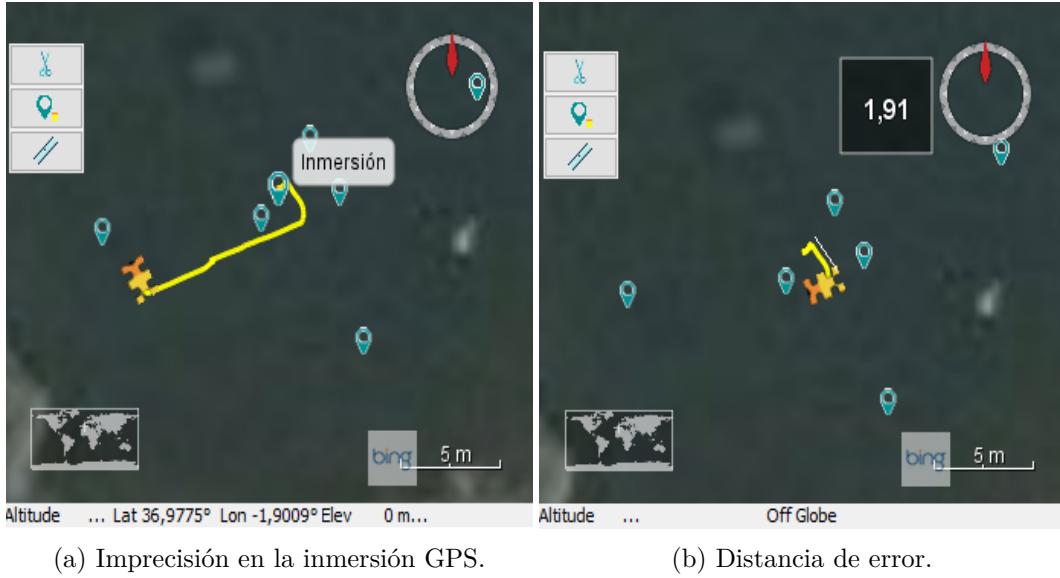


Figura 26: Imágenes del error provocado por el GPS

ya están tratados por el propio software creado para el manejo de este (DasBoot), pero existe unas imprecisiones en los datos de las posiciones, por lo que se realiza un proceso de eliminación de estas, provocadas por el GPS. Estas imprecisiones vienen dadas por la pérdida del posición del GPS, esto ocurre cuando el ROV realiza una inmersión, por lo tanto, la señal se pierde y en este caso el GPS da una posición errónea diferente al punto donde se realiza la inmersión, véase un ejemplo en la figura 26a , podemos ver que el punto donde se realiza la inmersión, si el GPS no hubiese dado tal imprecisión la línea saldría totalmente recta y no tendríamos una discrepancia de 2metros como se ha calculado en la figura 26b. Por lo que se ha decidido la eliminación de dichas posiciones ya que no aportan valor ni información, todo lo contrario.

#### 4.1.2.4 Combinación de los datos

Como se ha comentado anteriormente la única desincronización que existe es entre el ROV y el sonar, ya que los datos del ROV se sincronizan internamente con el software DasBoot. Por desgracia, no existe una forma de poder sincronizar los datos del ROV con el sonar, debido a lo anteriormente comentando, la empresa proveedora del dispositivo no ha querido dar ninguna información sobre el protocolo utilizado para la comunicación con el sonar, por tanto, no tenemos el control del dispositivo y no tenemos datos sobre angulos y tiempos de del haz en ir y volver, y un impedimento más es que a la hora de reproducir la grabación del barrido que se ha realizado

con Deep View, este reproduce un barrido que durante la misión ha tenido una duración de 4 minutos en 24 segundos. Por lo tanto, se ha tomado la decisión que cada línea de pixel del sonar de barrido lateral se corresponda a cada dato tomado del ROV (posición, pose, velocidad). En el supuesto caso de que existiese el control del dispositivo la sincronización se realizaría por instantes de tiempo, es decir, en un instante de tiempo determinado se recogen de los distintos hilos posición, pose, velocidad, sonar. Por lo que para hacer la sincronización simplemente habría que hacer coincidir los instantes de tiempo de cada dispositivo.

Por lo tanto, se recupera un frame que se almacena memoria y con ese frame extraemos una fila y toda su columna, con lo que se obtiene la línea de pixel y tratamos sus valores de píxeles como amplitudes, tal y como se haría con la salida de un sonar. Por otro lado, extraemos teniendo en cuenta un índice para poder relacionar la línea de sonar con las características del ROV (posición, velocidad y pose) y desde aquí se transmite al bloque de procesado, donde pasa por todos los procesos convenientes para obtener la salida de la línea del píxel dentro de una imagen con la correcciones que se verán más adelante. Además, al hacer la transformación de posición UTM a pixel y ubicar la línea de pixel dentro de esas posiciones somos capaces de georreferenciarla.

## 4.2. Procesado

Una vez completado el bloque de pre-procesado, nos ubicamos en el punto donde tenemos todo preparado para la proyectar nuestro mosaico en una imagen, esta proyección requiere pasar por las distintas etapas que tenemos dentro del bloque de procesado. Por tanto, en este bloque se detalla los procesos de: mapeado de coordenadas, rotaciones de las coordenadas, correcciones de las imágenes del sonar y la generación del mosaico.

### 4.2.1. Mapeando coordenadas

En esta sección se va a ver el proceso en detalle junto con los algoritmos necesarios, para el mapeado de las coordenadas de tanto del ROV como del sonar, así como las rotaciones geométricas necesarias para obtener un mapeado lo más fiel posible.

#### 4.2.1.1 Mapeando el ROV

El primer paso que se realiza en este bloque es el mapeo, el relacionar nuestras coordenadas UTM a coordenadas en píxeles, para ello tomamos como

referenciar el tamaño de la línea de píxeles en el eje x del vídeo del sonar (línea horizontal), en este caso la línea corresponde a un tamaño de 908 píxeles, y en la configuración del dispositivo para realizar el barrido se asignó un tamaño total de 30 metros de rango, 15 metros para cada lado. Por lo tanto, tenemos la referencia de 908 píxeles equivalen a 30 metros.

Las coordenadas UTM representan valores muy grandes por lo que se toma de referencia la primera coordenada, para restarla al resto de coordenadas. En la ecuación 30 podemos ver como pasamos una coordenada UTM a píxeles, donde  $k$  representa la relación  $\frac{\text{Pixel}}{\text{Metro}}$ .

$$\text{Pixel} = k \cdot \text{metros} \quad (30)$$

En el listado de código 2 vemos la ecuación programada en python, podemos ver que al resultado de la ecuación 30 se le resta un offset, esto se realiza para que el mosaico se genere en el centro de la imagen.

```

1 def mapping(xMeters, yMeters, offsetX, offsetY):
2     k = SIZE_BEAM_PIXELS/SIZE_BEAM_METERS
3     return int(np.abs(k*xMeters-offsetX)),
4         int(np.abs(k*yMeters-offsetY))

```

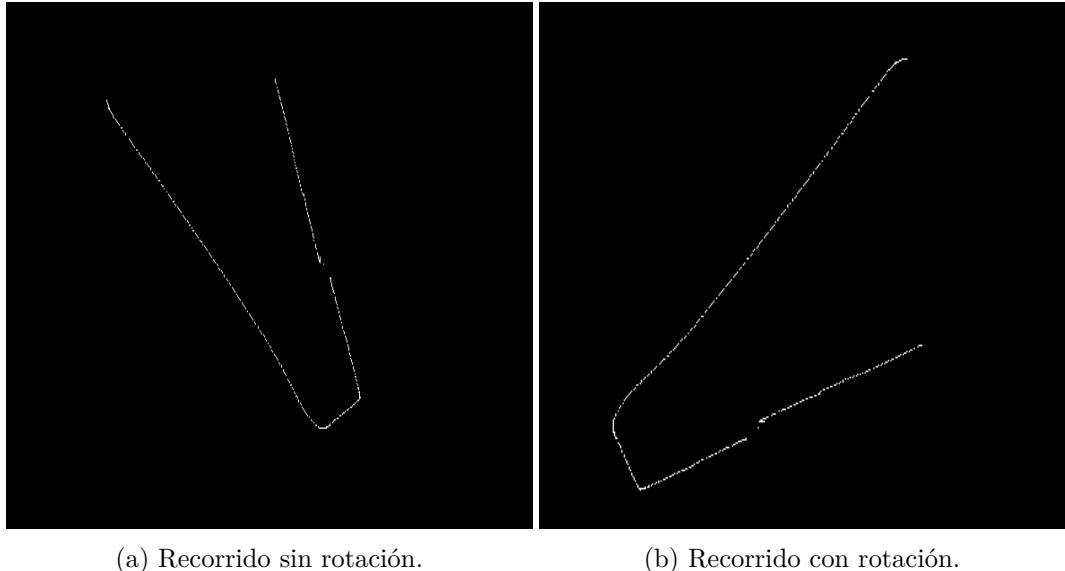
Listing 2: Función de mapeado

## Rotación geométrica

A la hora de realizar el mapeo este mostraba una rotación de 270 grados en sentido antihorario, por lo que se aplica una rotación geométrica alrededor del origen. Para la realización de esta rotación, se sigue la regla que nos indica que pasamos de tener nuestra coordenada  $R_{270^\circ}(x, y) = (y, -x)$ . En la figura 27a podemos ver el recorrido sin aplicar la rotación y en la figura 27b podemos contrastar el recorrido con el de la figura 22 donde se mostraba el recorrido del ROV dentro de Dasboot.

### 4.2.1.2 Mapeando el sonar

Una vez tenemos la posición del ROV como referencia, lo siguiente es determinar las coordenadas en píxeles del espacio que ocupa el haz del sonar de barrido lateral. Existen dos formas de obtener las coordenadas del sonar: sumandole y restandole los píxeles que hemos obtenido de la posición del ROV, ya que estamos cogiendo como referencia el propio sonar, o sumarle y restarle a las posiciones UTM los metros correspondientes y realizar posteriormente el mapeado de metros a píxeles. En este trabajo se realiza el cálculo con las coordenadas UTM, por lo que a cada posición del ROV a



(a) Recorrido sin rotación.

(b) Recorrido con rotación.

Figura 27: Recorrido del ROV

lo largo del recorrido debemos de sumarle y restarle 15 metros (izquierda y derecha), que es el rango que corresponde al tamaño del haz. El cálculo se hace sobre el eje  $y$  de las coordenadas UTM. El motivo de hacerlo al eje  $y$  es debido a que el sonar de barrido lateral realiza el barrido sobre el eje  $y$  del ROV. En la figura 28 se puede ver una ilustración de esto.

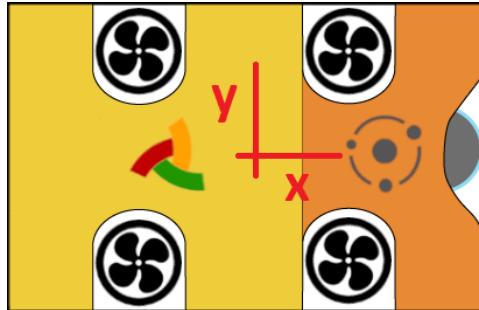


Figura 28: Posición de los ejes del AUV.

De esta forma obtenemos dos coordenadas  $(x, y)$  en la imagen donde se realiza el mosaico, que corresponden a la distancia máxima del haz por el lado izquierdo del ROV (babor), y a la distancia máxima del haz por el lado derecho del ROV (estribor), estas coordenadas junto con las coordenadas del ROV serán esenciales para más adelante proyectar la imagen del sonar.

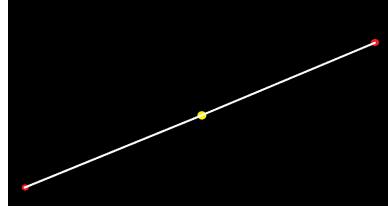


Figura 29: Puntos del ROV y sonar.

### Utilizando el yaw

El yaw como se ha comentado anteriormente, es un giro alrededor del eje z del ROV, es muy importante a la hora de generar el mosaico tener en cuenta esta rotación, pues vamos a poder proyectar en la re-construcción del recorrido del ROV los giros que este ha realizado. Las coordenadas del sonar de barrido lateral que hemos mapeado, son las mismas coordenadas que vamos a utilizar para aplicar el yaw. El haz del sonar es el que va a realizar la rotación alrededor del ROV, es decir, el punto donde se sitúa el ROV no rota, sino que es el sonar el que rota sobre el. Las ecuaciones 31 y 32 representan las nuevas coordenadas que se obtendrían en función de la rotación.

$$newX_{pixel} = \cos \cdot (x - ROV_x) - \sin \cdot (y - ROV_y) + ROV_x \quad (31)$$

$$newY_{pixel} = \sin \cdot (x - ROV_x) + \cos \cdot (y - ROV_y) + ROV_y \quad (32)$$

En la figura 29, se puede observar tres puntos, el centro corresponde al ROV y en los extremos a la distancia del sonar. Por tanto, ya tenemos mapeadas todas las coordenadas necesarias con sus rotaciones aplicadas.

#### 4.2.2. Correcciones radiométricas

En esta etapa, nos encontramos ya con las correcciones de las imágenes del sonar de barrido lateral, las correcciones tienen como fin, poder visualizar mejor lo que capta el sonar. Primero se aplica la corrección de esta sección, la radiométrica, que tiene como fin la mejora visual a partir de la transformación de niveles basada en el histogramas. Cabe destacar que Deep View nos presenta la salida de la imagen del sonar en color con tono sepia, para este trabajo se ha decidido realizarlo en escala de grises por simplicidad a la hora de realizar las diferentes correcciones.

#### 4.2.2.1 Ecualización de Histograma

La ecualización de histogramas busca un equilibrio entre los distintos valores de los píxeles que componen la imagen, tratando así de mejorar el contraste, para ello utiliza la distribución de probabilidad. El objetivo principal de esta corrección es obtener una mejoría visual para poder identificar mejor objetos, estructuras o incluso el propio terreno del fondo marino. Se ha aplicado una variante de la ecualización de histograma adaptativo, llamado Contrast Limited AHE (CLAHE).

CLAHE limita la amplificación del contraste para así evitar en la medida de lo posible generar ruido en la imagen. La implementación de esta función se realiza bajo la librería de OpenCV, en el listado de código 3 se puede observar la función que realiza este proceso, una vez aplicada esta función la imagen se convierte a escala de grises. En la figura podemos ver la diferencia de contraste.

```
1 def adjust_light(image):
2     clahe = cv.createCLAHE(clipLimit=6.0, tileSize
3     =(8,8))
4     lab_image = cv.cvtColor(image, cv.COLOR_RGB2Lab)
5     lab_planes = cv.split(lab_image)
6     lab_planes[0] = clahe.apply(lab_planes[0])
7     lab_image = cv.merge(lab_planes,lab_planes[0])
8     result = cv.cvtColor(lab_image, cv.COLOR_Lab2RGB)
9     return result
```

Listing 3: CLAHE

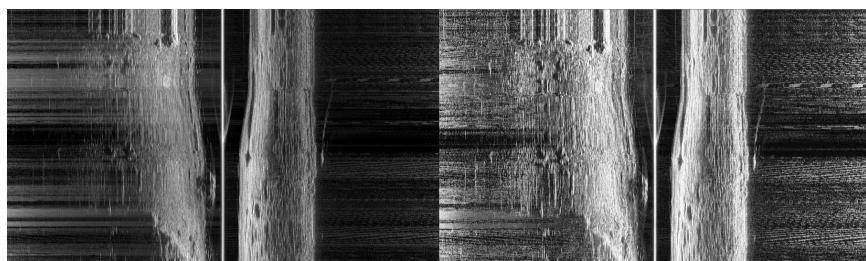


Figura 30: Izquierda sin filtro CLAHE vs Derecha con filtro CLAHE

#### 4.2.2.2 Eliminación de Water Column

Al comienzo de cada ping el transductor no recibe ecos, ya que la señal se pierde a través del agua sin ser recogida de nuevo por el transductor, esto genera un área oscura en el centro de la imagen llamada columna de agua o water column. El water column es una zona completamente ciega, llena de ruido que no contiene ninguna información útil por lo que normalmente

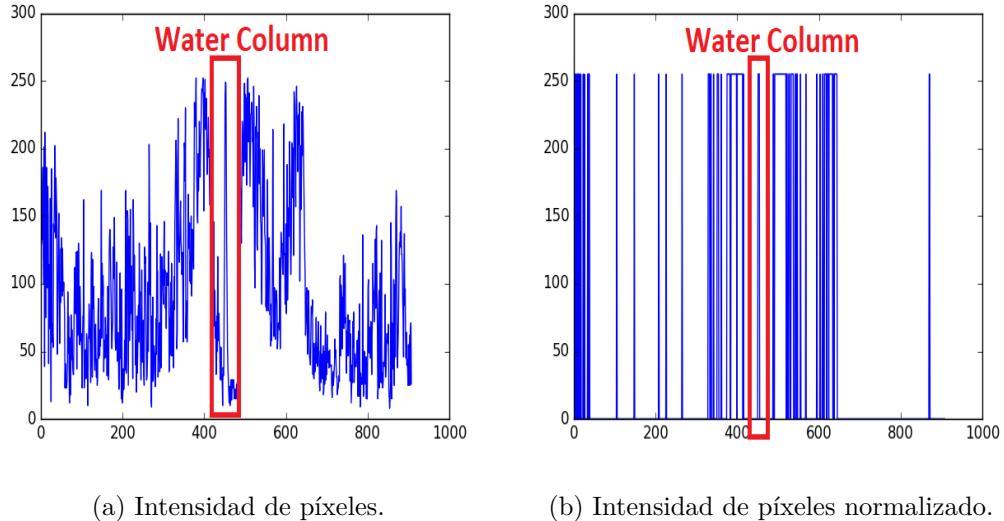


Figura 31: Análisis para la identificación de water column.

se opta por eliminarla. Existe una ecuación que dado el ángulo del haz y el ángulo de apertura, el resultado que da se toma como un rango entre el resultado negativo y el positivo y si los ecos están entre esos rangos no se tienen en cuenta. En nuestro caso no tenemos el valor del ángulo del haz, por lo tanto, para la eliminación del water column se ha analizado la intensidad de los píxeles en el centro con ayuda de la librería pyplot, para el dibujado de gráficos y se ha encontrado un patrón. Este patrón tal y como podemos apreciar en la figura 31a el water column presenta una alta intensidad en el centro de la imagen, que va disminuyendo drásticamente hasta prácticamente tener una intensidad de cero, esto sucede durante toda la grabación, por lo que para facilitar su eliminación, se normaliza aplicando una segmentación binaria con la ayuda de OpenCV, el resultado obtenido lo podemos ver figura 31a con esto conseguimos que aquellos píxeles que pueden dar la sensación de contener información establecerlos a cero por no superar cierto umbral.

Para llegar a identificar un buen threshold, donde se distinguiera perfectamente lo que compone el water column, se decidió por crear una pequeña herramienta haciendo uso de las funciones de OpenCV, para cambiar y visualizar en tiempo real el umbral que se usa, se puede apreciar en la figura 32.

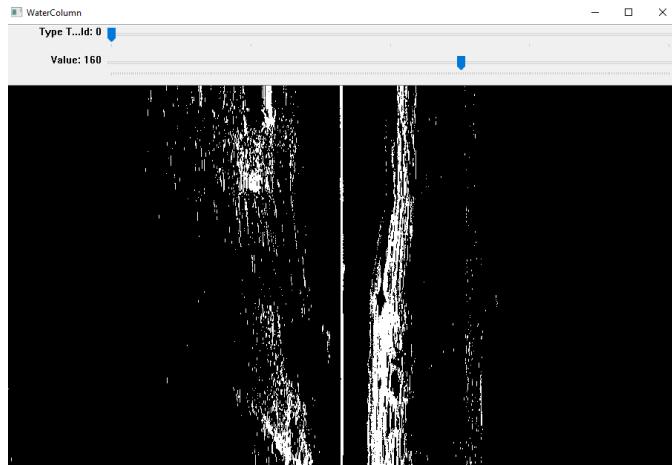


Figura 32: Herramienta para seleccionar el threshold

#### 4.2.3. Correcciones Geométricas

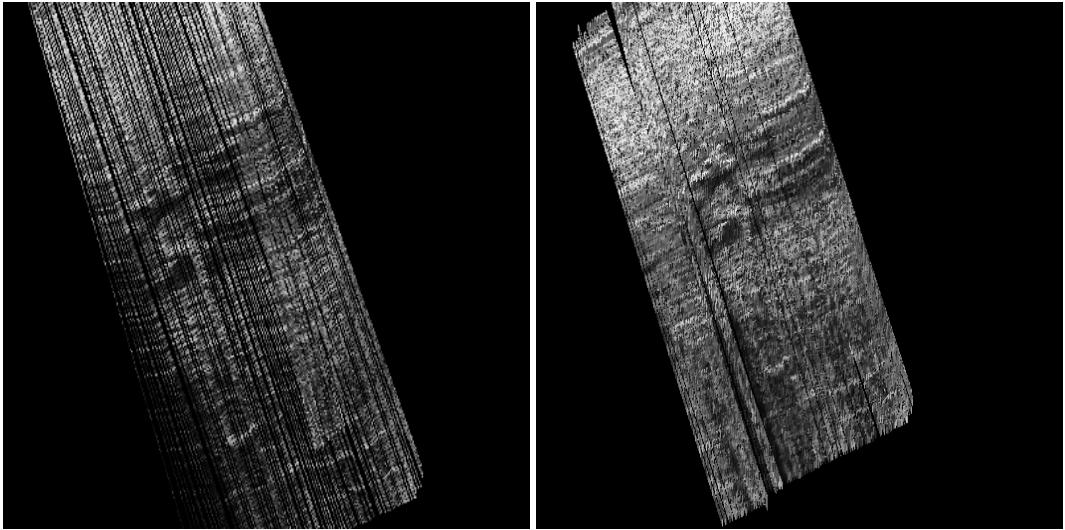
Con estas correcciones tal y como se dijo en el estado del arte, vamos a tratar de corregir aquellas distorsiones que provocan rescalados en la imagen o pérdida de información.

##### 4.2.3.1 Slant Range Correction

Esta corrección se explica en detalle a nivel teórico en el estado del arte, pues en principio iba a estar incluida en este trabajo, pero como no se dispone de las variables que son necesarias para su implementación (tiempo del haz en ir y volver), debido a que no se tiene acceso al dispositivo ni a su protocolo como se ha comentado anteriormente no se ha podido implementar.

##### 4.2.3.2 Anamorfosis

En este trabajo para solventar este problema se tiene en cuenta la distancia en metros de cada posición, utilizando las coordenadas UTM para hallar la distancia euclídea, dependiendo de la distancia reescalamos más o menos la imagen con OpenCV los píxeles necesarios para evitar separaciones muy grandes y así tratar de no tener espacios en negro. Hay que tener en cuenta, que para hacer uso del rescalado la distancia que se toma es la actual y la siguiente, por tanto, existe un retardo para que se pueda cumplir esta corrección. El escalado por defecto es de tamaño cinco, este tamaño se ha establecido porque se ha demostrado que era el tamaño adecuado para una buena visualización sin una pérdida de información provocada por el ruido



(a) No teniendo en cuenta la corrección de anamorfosis.  
(b) Teniendo en cuenta la corrección de anamorfosis.

Figura 33: Comparación corrección geométrica.

que lleva estirar una imagen, este tamaño puede variar si el cálculo de la ecuación 33, es mayor que el rescalado por defecto, donde  $k$  representa la relación  $\frac{\text{Pixel}}{\text{Metro}}$ .

$$height = distancia \cdot k \quad (33)$$

Por otro lado, en cuanto a los giros evitamos hacer este rescalado debido a dos razones: la falta de sincronización entre posición e imagen del sonar y a que no se cumplió uno de los requisitos anteriormente mencionados durante la misión, que es el de no realizar giros bruscos. Tratar de rescalar la imagen cuando el giro es demasiado rápido, estiraría demasiado la imagen, creando un efecto contrario al que buscamos, que es el de mejorar la visualización. Un ejemplo de esta corrección la tenemos en la figura 33a y 33b.

#### 4.2.4. Generando mosaico

Ya nos encontramos al final de este bloque de procesado, en esta última etapa, vamos a crear el mapa del fondo marino que se ha barrido durante la misión, ya tenemos preparadas: coordenadas, correcciones radiométricas y las correcciones geométricas, ya solo falta generar el mosaico. Para ello, en esta sección se presenta en detalle: la proyección de las distintas líneas del sonar de barrido lateral en un lienzo vacío y tratamiento al problema del overlapping.

## Proyectando la línea

Al principio de este bloque hemos mapeado las coordenadas al lienzo o imagen que vamos a generar para realizar el mosaico, por tanto, ahora debemos de proyectar la línea de pixel que tenemos sobre los puntos del sonar hasta el punto del ROV, tanto para la izquierda como para la derecha. Una ilustración de lo que se debe hacer podemos verlo en la figura 34. Para volcar la información de los píxeles y poder trazar la línea se hace uso del algoritmo de Bresenham.

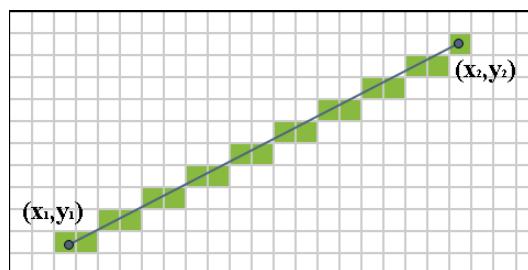


Figura 34: Trazado de línea

El algoritmo de Bresenham es un método para el trazado rápido y eficiente de líneas teniendo como entrada dos puntos, como anteriormente hemos comentado, tenemos las coordenadas en píxeles del sonar de barrido lateral y las del ROV, aplicando el algoritmo de Bresenham nos devuelve todas las posiciones hasta llegar al punto final que le indiquemos, en este caso, el punto inicio es el ROV y el punto final el rango máximo del sonar. El inconveniente que tiene el algoritmo de Bressham es el aliasing, en este caso, tenemos que la distancia en píxeles del ROV al rango del sonar es 454 píxeles, tal como se puede observar en la propia figura 34 esto no va a ocurrir, se va a dar el caso en que tengamos más posiciones dadas por el algortimo de Bresenham que imagen tenemos, este es el problema del aliasing, existen otros algoritmos que tratan de solucionar esto, como es el algortimo de Xiaolin Wu, pero se ha optado por no desarrollarlo debido a que la mejora producida no iba a ser tan significativa para que merezca la pena, además, el algoritmo de Bresenham es más ligero, y esto si que beneficia a la herramienta desarrollada, para que genere el mosaico de forma más rápida. En la figura 35 podemos ver la proyección de una línea de pixel obtenida de la imagen del sonar de barrido lateral aplicando el algoritmo de Bressenham, también podemos ver la implementación del algoritmo en el lista de código 4

```
1 def bresenham_algorithm(v1, v2):
2     vResult, incYi, incXi, incYr, incXr = [], 0, 0, 0
3     dY = (v2[1] - v1[1])
4     dX = (v2[0] - v1[0])
5
```

```

6     if(dY >= 0):
7         incYi = 1
8     else:
9         dY = -dY
10        incYi = -1
11
12    if(dX >= 0):
13        incXi = 1
14    else:
15        dX = -dX
16        incXi = -1
17
18    if(dX >= dY):
19        incYr = 0
20        incXr = incXi
21    else:
22        incXr = 0
23        incYr = incYi
24        dX, dY = dY, dX
25
26 # Inicializar valores (y de error).
27 x, y = v1[0], v1[1]
28 avR = (2 * dY)
29 av = (avR - dX)
30 avI = (av - dX)
31 vResult.append([x,y])
32 while x != v2[0]:
33     if(av >= 0):
34         x = (x + incXi)
35         y = (y + incYi)
36         av = (av + avI)
37     else:
38         x = (x + incXr)
39         y = (y + incYr)
40         av = (av + avR)
41     vResult.append([x,y])
42
43 return vResult

```

Listing 4: Algoritmo de Bresenham

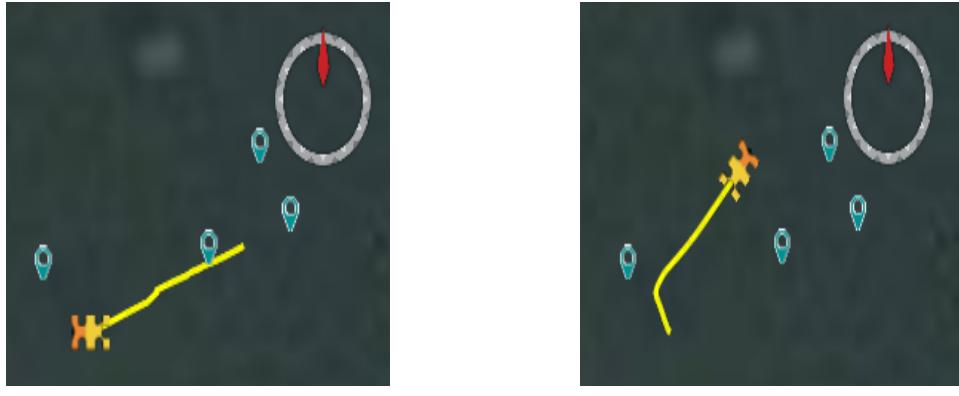
## Overlapping

El problema del overlapping es crítico, desde el punto de vista visual, ya que una buena resolución de este problema conllevará a una gran calidad en nuestro mosaico. En el estado del arte se comentó extensamente los diferentes métodos de resolverlo que actualmente se están llevando a cabo, en este trabajo se opta por resolver el overlapping de una forma sencilla como es la media entre los píxeles. La razón de esto, es que este trabajo no está enfocado a la resolución óptima visual del mapeado, sino todo el trabajo



Figura 35: Algoritmo de Bresseham

que hay que lleva hasta el punto donde nos encontramos que es el final para tener como salida nuestro mosaico, por lo tanto, todo lo que ha desarrollado es el pilar fundamental para que posteriormente en una ampliación de este trabajo se pueda aplicar SIFT, SURF u ORB.



(a) Recorrido A.

(b) Recorrido B.

Figura 36: Recorrido dividido.

### 4.3. Resultado

Los resultados obtenidos de la misión teniendo en cuenta el recorrido que se realizó y escaso número de pruebas que se pudo realizar en dos días en Carboneras, han sido muy satisfactorios. Los resultados se presentan con un recorrido dividido en dos partes (A y B), pues al realizar un recorrido tan cercano el uno con el otro, esto ha provocado un overlapping durante todo el recorrido, además, uno presentaba más distorsión que el otro, provocando que la forma de resolver el problema del overlapping (haciendo la media entre píxeles) distorsionará completamente la imagen. En la figura 36 vemos el recorrido dividido.

En la figura 37 se presenta la zona escaneada del recorrido A, se han identificado tres tipos diferentes, los cuales se han clasificado en colores:

- Azul: ánforas.
- Verde: tuberías.
- Marrón: sedimento del fondo marino.

Por otro lado, con la misma agrupación en la figura 38 tenemos el barrido del recorrido B, podemos apreciar que en la parte inferior derecha de la imagen es idéntica, incluso con un poco más de barrido, que la imagen de la figura 37 que se corresponde al recorrido A, la perspectiva es diferente, por eso hacer la media distorsionaba gravemente la imagen.

En las figuras 39, 40, 41 comparamos tres imágenes extraídas del vídeo del sonar con el mosaico generado, se aprecia que con la corrección radiométricas somos capaces de identificar mejor objetos como las ánforas, por otro lado, como se observa en la figura 40 parece que la ánfora es de menor ta-

maño, que la imagen registrada por el sonar, esto es debido a dos factores: (1) Corrección geométrica, en concreto anamorfosis, al tener las posiciones y velocidades, se hace la corrección de reescalado o (2) fluctuaciones en el yaw provocando la intersección entre píxeles, teniendo como consecuencia la eliminación del anterior, se ha explicado la razón de esta decisión anteriormente, y era debido a que hacer la media entre píxeles no da buenos resultados debido al recorrido que ha seguido en este caso el ROV.

Somos capaces de identificar que lo que hay en azul es una anfora, o lo que hay en verde es una tubería, porque en esta ocasión para esta misión, el agua estaba lo suficientemente clara, además de no haber una gran profundidad, por lo que los rayos del sol penetraban hasta llegar al fondo. En las figuras 42 y 43 podemos ver unas imágenes que nos sirven para contrastar lo que estamos viendo con las imágenes sonar.

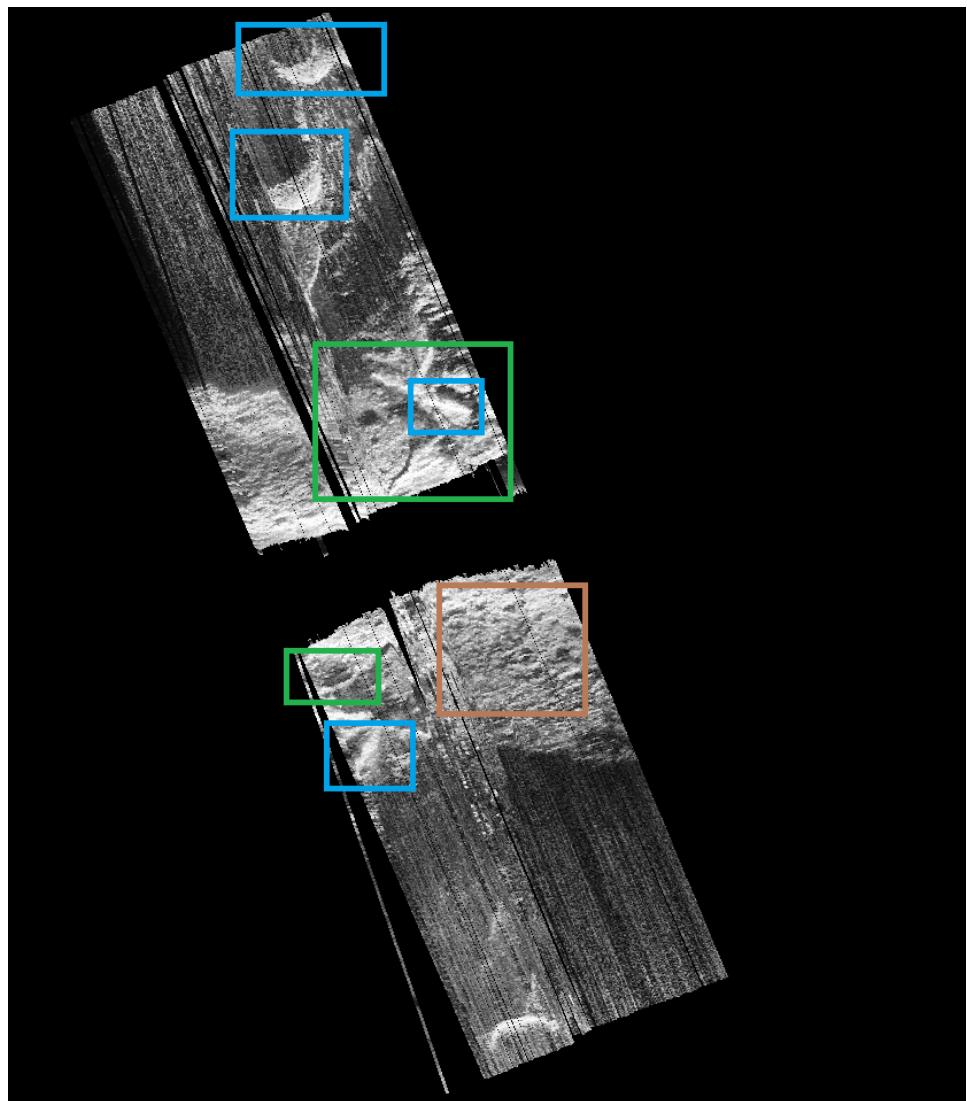


Figura 37: Barrido recorrido A.

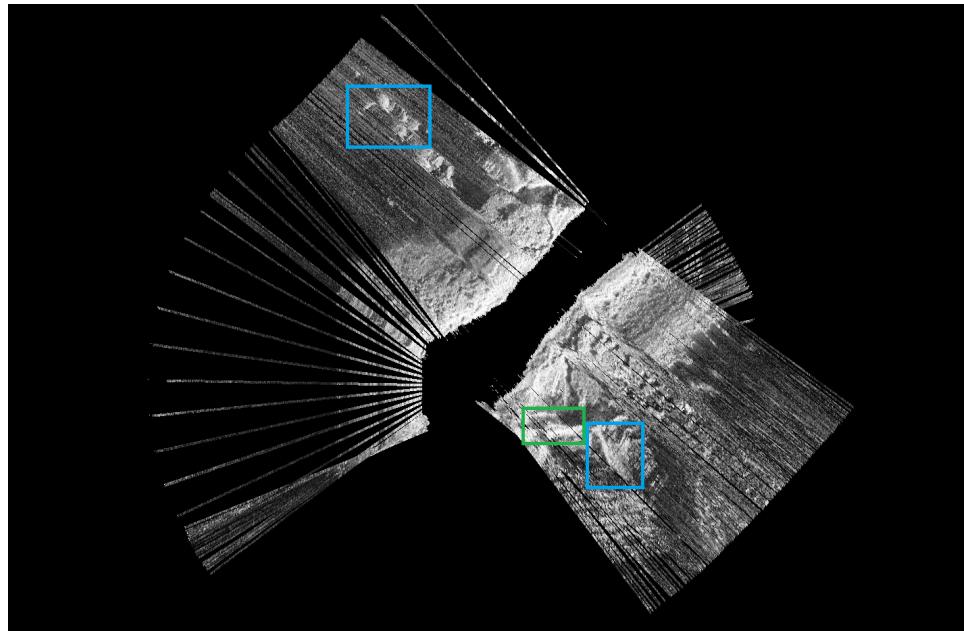


Figura 38: Barrido recorrido B.

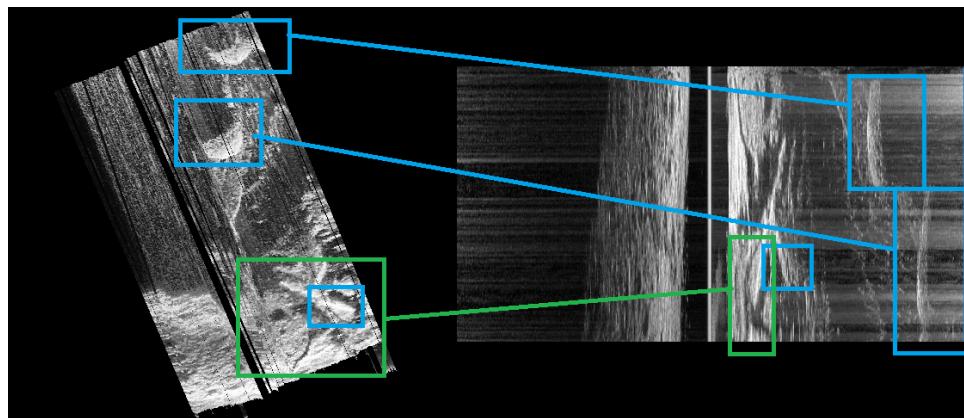


Figura 39: Comparando barrido recorrido A con imagen sonar sin tratar.

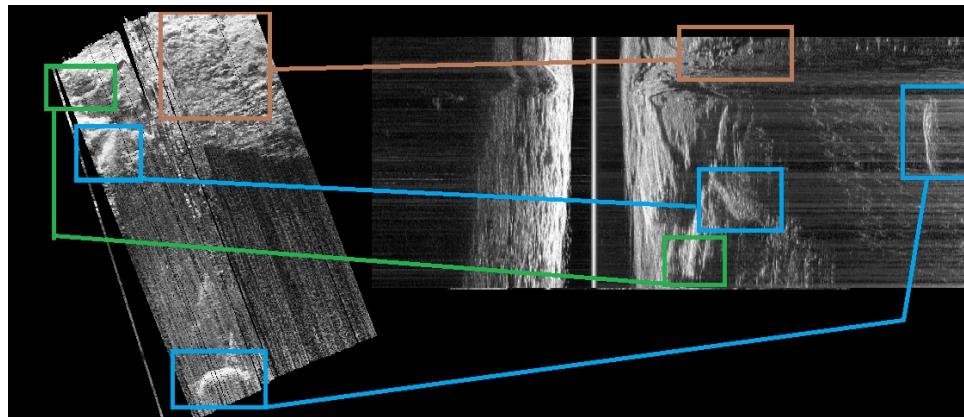


Figura 40: Comparando barrido recorrido A con imagen sonar sin tratar.

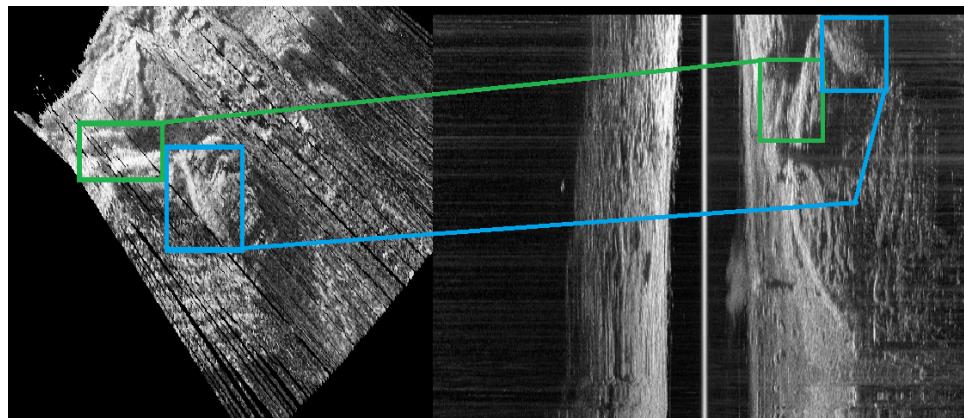


Figura 41: Comparando barrido recorrido B con imagen sonar sin tratar.

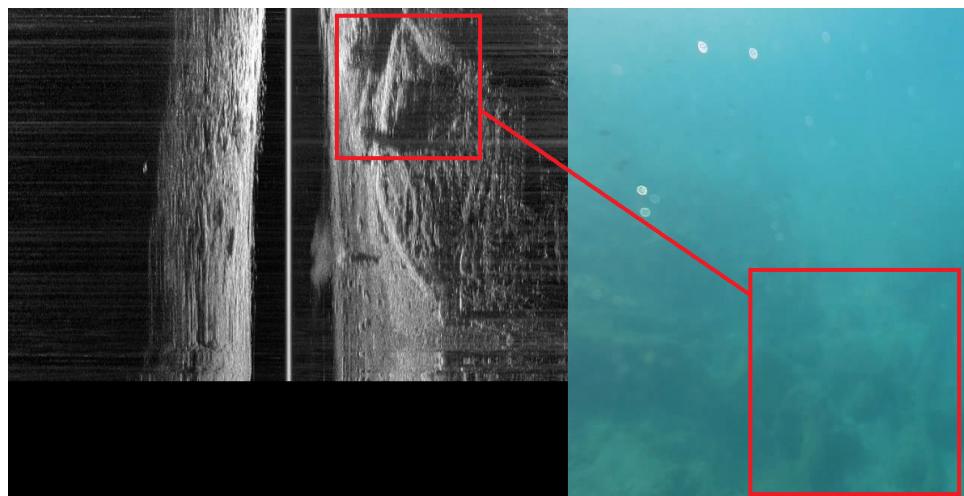


Figura 42: Comparación del recorrido B con cámara del ROV.

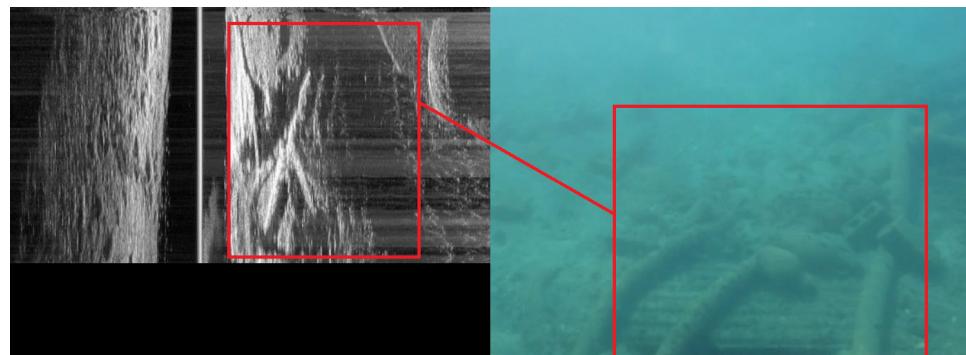


Figura 43: Comparación del recorrido A con cámara del ROV.



Figura 44: Imagen de la cámara del ROV.

## 5. Conclusiones

El desarrollo de este trabajo ha sido una herramienta capaz generar mosaicos del fondo marino de forma automática, por medio de un sonar de barrido lateral, además, se tiene en cuenta la trayectoria del ROV, por lo que se traza la trayectoria a partir de las imágenes, de este modo conseguimos consistencia a nivel espacial y temporal.

El desarrollo de este trabajo ha llevado consigo un estudio previo de las diferentes tecnologías de sonars acústicos que existen y su funcionamiento, también ha sido necesario recordar conceptos claves de la álgebra lineal como son las rotaciones. Como se ha podido apreciar en la lectura de este trabajo, la realización del mosaico no es solo la de concatenar imagen tras imagen, sino que quiere de un pre-procesado y un procesado de los datos, en este trabajo se recoge todo lo necesario para la generación de mosaicos del fondo marino, siendo un pilar fundamental para la construcción de la batimetría. En definitiva, ha sido una experiencia enriquecedora a nivel académico y a nivel presencial, debido a que las pruebas de funcionamiento se desarrollan fuera del entorno habitual al que esta habituado normalmente el informático, como es el ordenador y la oficina.

Los resultados obtenidos no son buenos desde el punto de vista de lo que quería mostrar en este trabajo, esta misión que se ha mostrado como resultado no tenía como fin entrar en este proyecto, ya que la misión que se ha mostrado se estaba realizando para otras pruebas del ROV, pero debido a la situación que ha atravesado el país no se han podido tomar datos posteriormente, los datos mostrados en este trabajo son datos que se recogieron dos semanas antes del confinamiento. Aun así, las conclusiones que podemos sacar son, por un lado, lo importante que es marcar un buen recorrido, teniendo en cuenta como buen recorrido aquel que intente solapar lo menos posible el rango del haz e intentar mantener siempre el mismo ángulo de recorrido, es decir, que las líneas trazadas sean lo más paralelas posibles, por otro lado, destacar de la importancia de un buen algoritmo que resuelva el overlapping, se ha demostrado que realizar la media entre los píxeles no daba buenos resultados, por lo que se optó por deshabilitar la opción y dividir el recorrido en dos. Destacar, este trabajo deja abierta como ampliación la detección de objetos, resolver con métodos simbólicos el overlapping y levantamiento del fondo marino.

Por último, destacar el gran inconveniente que ha sido el disponer de un dispositivo totalmente privativo, y que además la empresa no cediera la información del protocolo por miedo a que se filtrase, ya que se habían planteado correcciones, funcionalidades y lo que es más importante la sincronización total con el ROV, que solo hubiera sido posible teniendo el control total del dispositivo.

## Referencias

- [1] Bennell James D. . Mosaicing of sidescan sonar images to map seabed features. En: Jon Davies. Marine Monitoring Handbook. 2001. PG1-5.
- [2] Zitová Barbara , Flusser Jan. Image Registration Methods. En: Todorovic S. Image and Vision Computing. 21, 2003, PG 977-1000.
- [3] Zhao Jianhu, Wang Aixue, Zhang Hongmei, Wang Xiao. Mosaic method of side-scan sonar strip images using corresponding features. En: IET. 6, 2013, PG616-623
- [4] Chailloux Cyril, Le Caillec Jean-Marc, Gueriot Didier, Zerr Benoit. Intensity-Based Block Matching Algorithm for Mosaicing Sonar Images. En: IEEE. 4, 2011, PG627-644.
- [5] Horn K.P., Schunck G. Brian. Determining Optical Flow. En: Science-Direct. Artificial Intelligence. 1, 1981, PG185-203
- [6] Salgado de la Nuez Agustín Javier. Métodos Variacionales para la Estimación del Flujo Óptico y Mapas de Disparidad [master's thesis]. Lugar: Universidad de las Palmas de Gran Canaria. 2010. 217P.
- [7] Barron J. L. Beauchemin S. S. . The Computation of Optical Flow. En: ACM Computing Surveys. 1995, 35P.
- [8] Vandish P., Vardy A., Walker D., Dobre A. O. .Side-scan Sonar Image Registration for AUV Navigation. 6. P1-7.
- [9] Pazmiño Reyes Ricardo Esteban. Implementación y comparación de los algoritmos de determinación de flujo óptico de Horn-Schunck y Lucas-Kanade para rastreo de objetos. En: Universidad San Fransico De Quito. 46P.
- [10] Lucas Bruce D., Kanade Takeo. An Iterative Image Registration Technique with an Application to Stereo Vision. Proceedings DARPA Image Understanding Workshop. 1981. P674-679.
- [11] Chailloux Cyril, Zerr Benoit. Non Symbolic Methods to register sonar images. En: Oceans. 1. 2005. PG276-281.
- [12] Dhana Lakshmi M., Vimal Raj M., Sakthivel Murugan S. . Feature Matching and Assessment of Similarity rate on geometrically Distorted Side Scan Sonar Images. En: TEQIP III Sponsored International Conference on Microwave Integrated Circuits, Photonics and Wireless Networks (IMICPW). 2019, PG208-212.
- [13] Lowe G. David. Distinctive Image Features from Scale-Invariant Keypoints. En: Int. J. Comput. Vision. 60. 2004, PG91-110.

- [14] Akalanka Perera Shehan . A Comparison of SIFT , SURF and ORB. 2018. Disponible en: <https://medium.com/@shehan.a.perera/a-comparison-of-sift-surf-and-orb-333d64bcaa>
- [15] E. Rublee, V. Rabaud, K. Konolige and G. Bradski, .°RB: An efficient alternative to SIFT or SURF,"2011 International Conference on Computer Vision, Barcelona, 2011, pp. 2564-2571.
- [16] Godoy Olivera Yesmar Andrés, Ducuara Oyuela Álvaro Isledier. Análisis Comparativo de las técnicas SURF y ORB para la detección de puntos de interés en fotografías aéreas. [master's thesis]. Lugar: Universidad de Ibagué. 2019. 71P.
- [17] Reed S., Ruiz I. T., Capus C., Petillot Y. . The fusion of large scale classified side-scan sonar image mosaics. En: IEEE Transactions on Image Processing, vol. 15, 7. 2006. PG2049-2060.
- [18] Palomar, A., Ridao, P., Ribas, D. . Multibeam 3D Underwater SLAM with Probabilistic Registration. Sensors 2016, 16, 560.
- [19] Zhao Jianhu, Yan Jun, Zhang Hongmei and Meng Junxia. A New Radiometric Correction Method for Side-Scan Sonar Images in Consideration of Seabed Sediment Variation. 2017.
- [20] Burguera A, Oliver G High-Resolution Underwater Mapping Using Side-Scan Sonar. PLoS ONE 11(1). 2016.
- [21] Gautier Sylvain and Morissette Guillaume. Unsupervised extraction of underwater regions of interest in sidescan sonar imagery. 2019.
- [22] Blondel Philippe . The Handbook of Sidescan Sonar. (2009).
- [23] Sgt. Al. León Martín, José David. Del Escandallo a los Sondadores Multihaz. Escuela de Hidrografía. (2004).