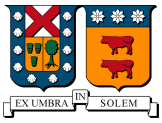
Universidad Técnica Federico Santa María
Departamento Obras Civiles



APUNTES RECTIFICACIÓN DE IMÁGENES Y ALGUNAS APLICACIONES

Autor: Esteban Opazo Verdugo

Profesor: Patricio Catalán

Enero 2022

Índice

Índice de tablas

Índice de figuras

1. Introducción

El presente documento busca dar conocer algunos de los conceptos que sustentan la rectificación de imágenes y algunas aplicaciones en el área de la oceonografía.

2. Información de Entrada (inputs)

2.1. Archivo de Video

Propiedades:

- Resolución: U x V pixeles
- Duración (DT)
- Velocidad Fotograma (FR)

Ejemplo: U = 3840, V = 2160, DT = 9 min 20 s, FR = 30 fps

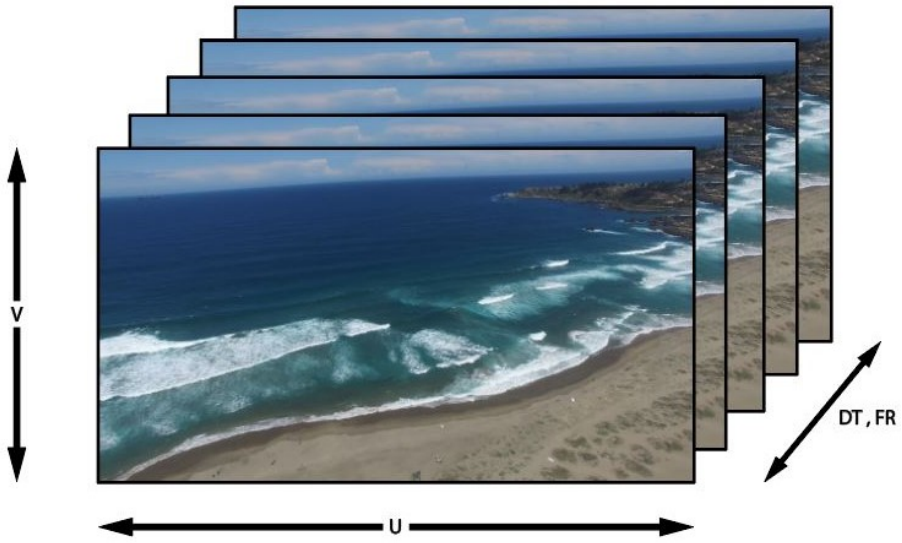


Figura 1: Representación archivo de video.

2.2. Perfil de Calibración del Lente (LCP)

El perfil de calibración del lente corresponde a los parámetros intrinsecos de este, los cuales se definen a partir de modelos matemáticos utilizados para representar el funcionamiento de una cámara. Los procedimientos de rectificación de imágenes explicados en el presende documento se basan principalmente en dos modelos de proyección que se explican a continuación.

2.2.1. Pinhole Model

El modelo de cámara estenopeica o cámara agujero de alfiler (pinhole) define que un punto en el espacio 3D se va a proyectar en el plano de imagen de acuerdo al siguiente esquema.

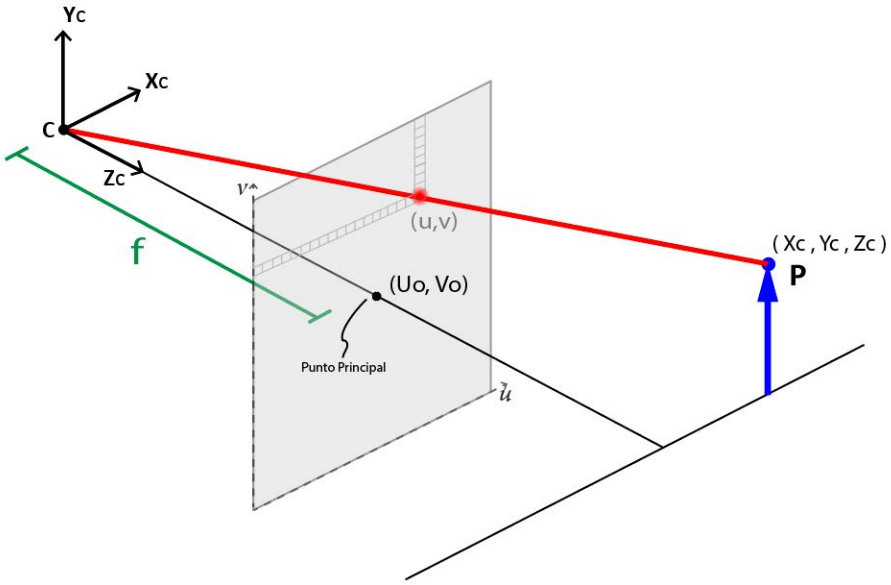


Figura 2: Representación gráfica pinhole model.

Se define un sistema de coordenadas local de la cámara en su centro óptico C, la distancia focal (f) corresponde a la distancia perpendicular entre el centro óptico y el plano de imagen, dicha intersección ocurre en un punto denominado punto principal. Además se utiliza el sistema de coordenadas de dos dimensiones u,v para representar la ubicación del punto en la imagen.

2.2.2. Modelo de distorsión radial y tangencial

Basado en el modelo de distorsión de Brown (1971), se realiza una serie de ajustes a las coordenadas de imagen para que se considere el efecto de la distorsión radial y tangencial. Las expresiones utilizadas se describen a continuación:

Coordenadas homogéneas de la cámara normalizadas:

$$x = \frac{(U - U_0)}{f_x} \quad ; \quad y = \frac{(V - V_0)}{f_y} \tag{1}$$

Se define $r^2 = x^2 + y^2$

Nuevas coordenadas normalizadas que consideran distorsión del lente:

$$\begin{Bmatrix} x_D \\ y_D \end{Bmatrix} = DistR \begin{Bmatrix} x \\ y \end{Bmatrix} + DistT$$

Distorsión Radial (DistR): $(1 + d1r^2 + d2r^4 + d3r^6)$

Distorsión Tangencial (DistT): $dX = \begin{Bmatrix} 2t_1xy + t_2(r^2 + 2x^2) \\ t_1(r^2 + 2y^2) + 2t_2xy \end{Bmatrix}$

Coordenadas de imagen que consideran distorsión:

$$U_D = x_D f_x + U_0 \qquad ; \qquad V_D = y_D f_y + V_0 \tag{2}$$

2.2.3. Parámetros Intrínsecos

De acuerdo a los modelos descritos anteriormente, los parámetros requeridos para la calibración intrínseca son los siguientes:

Parámetros	Definición
NU	Número de columnas pixeles
NV	Número de filas pixeles
U_0, V_0	Coordenadas del punto principal
f	Distancia focal
f_x	Distancia focal (en pixeles) en dirección U
f_y	Distancia focal (en pixeles) en dirección V
d_1, d_2, d_3	Coeficientes de distorsión radial
t_1, t_2	Coeficientes de distorsión tangencial

Tabla 1: Parámetros Intrínsecos.

Valores utilizados en el ejemplo de estos apuntes:

Parámetros	Valor
NU	3840
NV	2160
U_0	1957.13
V_0	1088.21
f_x	2298.59
f_y	2310.87
d_1	-0.14185
d_2	0.11168
d_3	0.0
t_1	0.00369
t_2	0.002314

Tabla 2: Valores ej. parámetros intrínsecos.

2.3. Puntos GCP (Ground Control Point)

Puntos cuyas coordenadas mundiales (Ej: UTM) son conocidas y sus respectivas coordenadas de imagen se pueden digitalizar con precisión.

Figura 3: Ubicación puntos GCP en imagen.

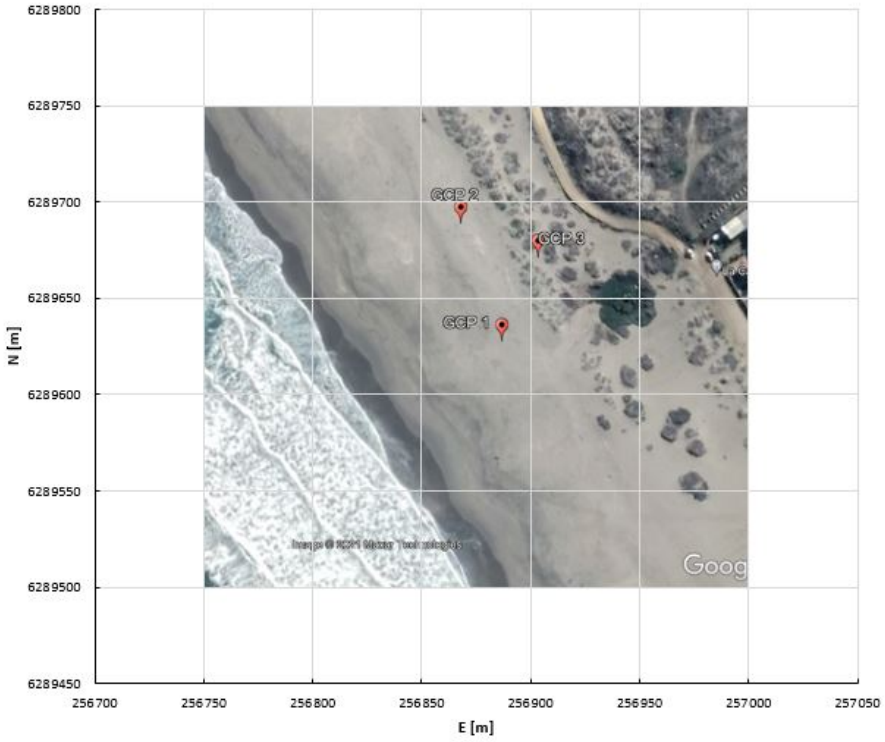


Figura 4: Ubicación puntos GCP en UTM.

2.4. Sistema de Coordenadas

2.4.1. Sistema de Coordenadas Globales:

En la figura 1 se muestra el área en estudio en un sistema de coordenadas UTM. Esta zona se ubica en el Huso 19.

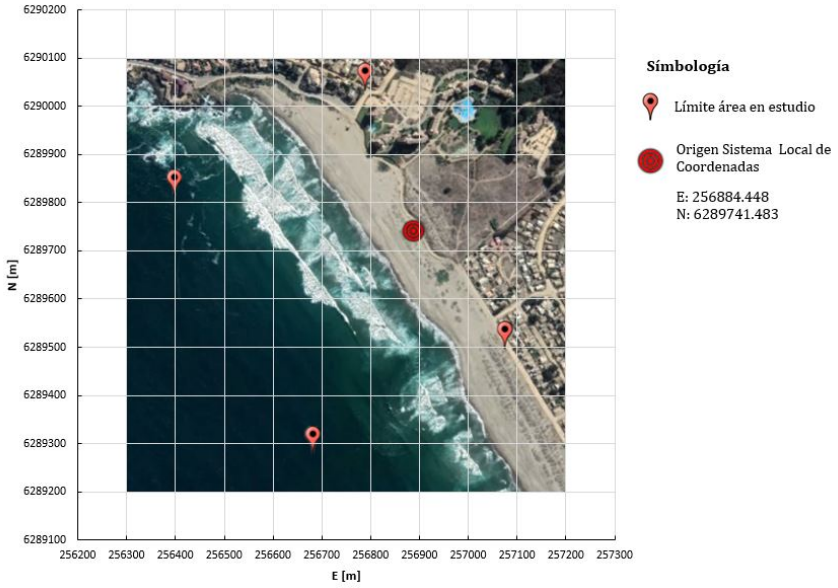


Figura 5: Sistema de Coordenadas Globales UTM

2.4.2. Sistema de Coordenadas Locales:

Para facilitar el análisis, se realiza la transformación a un sistema de coordenadas locales, cuyo origen se ubica en las coordenadas UTM: Este = 256.884,448; Norte 6.289.741,483; Huso 19. Además se realiza una rotación de 150° en sentido antihorario, de tal forma que la coordenada x apunte en la dirección perpendicular a la linea de costa.

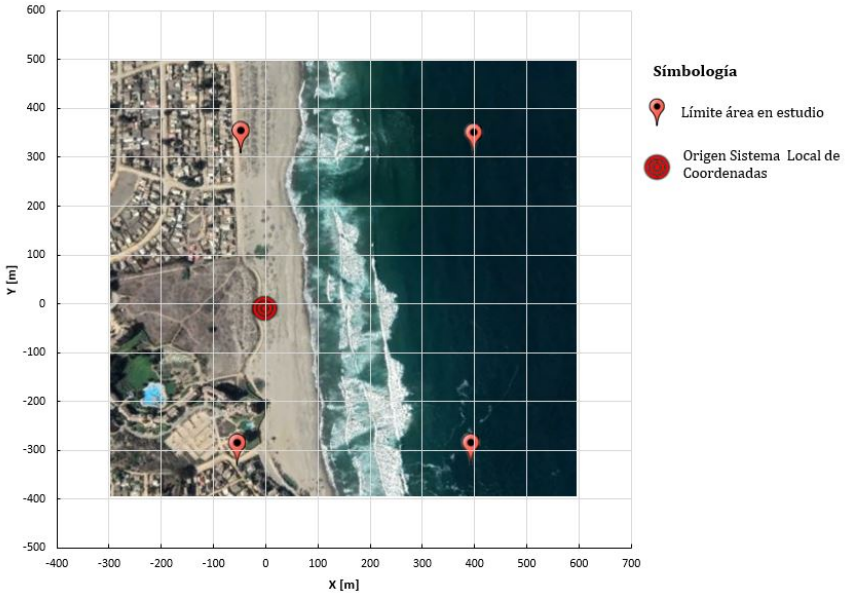
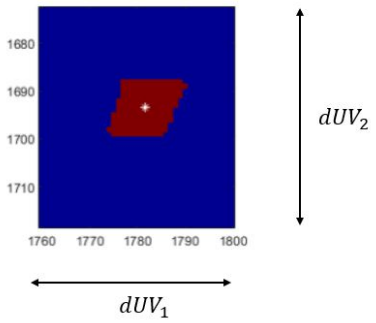


Figura 6: Sistema de Coordenadas Locales

3. Rectificación

3.1. Definición de variables iniciales

- I → Imagen, frame a rectificar (2160 x 3840 x 3 uint8)
- Ig → Imagen en escala de grises (2160 x 3840 x double)
- xyz → Puntos de Referencia GCP o GCP virtuales ; $\text{xyz} = \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{bmatrix}$
- dUV → Delta coordenadas UV que define posible ubicación puntos GCP



Obs: Existe una rutina que permite delimitar manualmente la zona donde se encuentran los puntos de referencia. Se define una zona cuadrada y un umbral de intensidad para identificar el punto.

- Ur, Vr → Coordenadas UV de los puntos de referencia GCP encontrados a partir de la zona indicada y umbral especificado.

$$U_r = \begin{bmatrix} U_{r1} \\ U_{r2} \\ U_{r3} \end{bmatrix} ; V_r = \begin{bmatrix} V_{r1} \\ V_{r2} \\ V_{r3} \end{bmatrix}$$

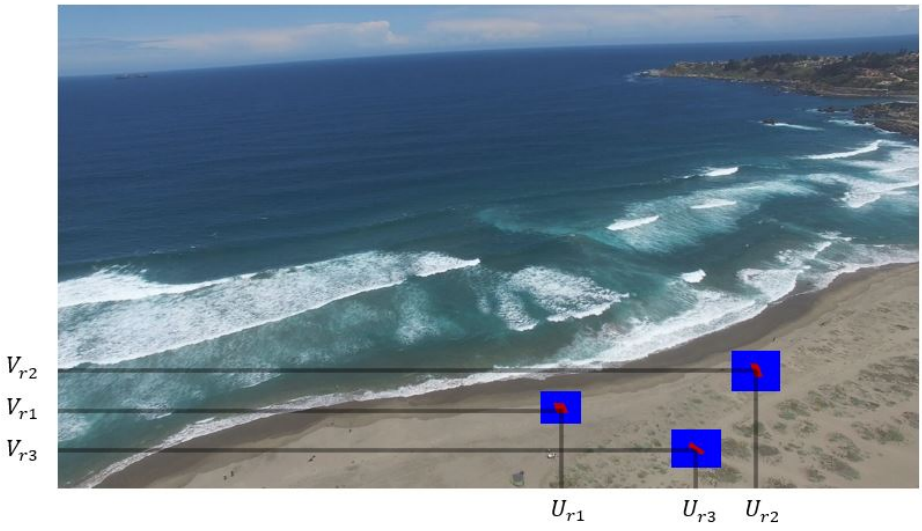


Figura 7: Ubicación UV de puntos de referencia

3.2. Parámetros extrínsecos

Corresponden a los parámetros que definen completamente la ubicación de la cámara con respecto al sistema de referencia utilizado. Estos parámetros juntos con los parámetros intrínsecos permiten construir la matriz proyectiva que relaciona las coordenadas del mundo real con las ubicaciones en la imagen (más detalle en la siguiente sección), esta transformación corresponde a la operación base para poder realizar el proceso de rectificación.

Los parámetros extrínsecos se muestran en figura 14 y estos son las coordenadas xyz de la cámara, denominados xCam, yCam y zCam; y los ángulos Azimuth, Tilt y Roll que definen todas las rotaciones angulares posibles de la cámara.

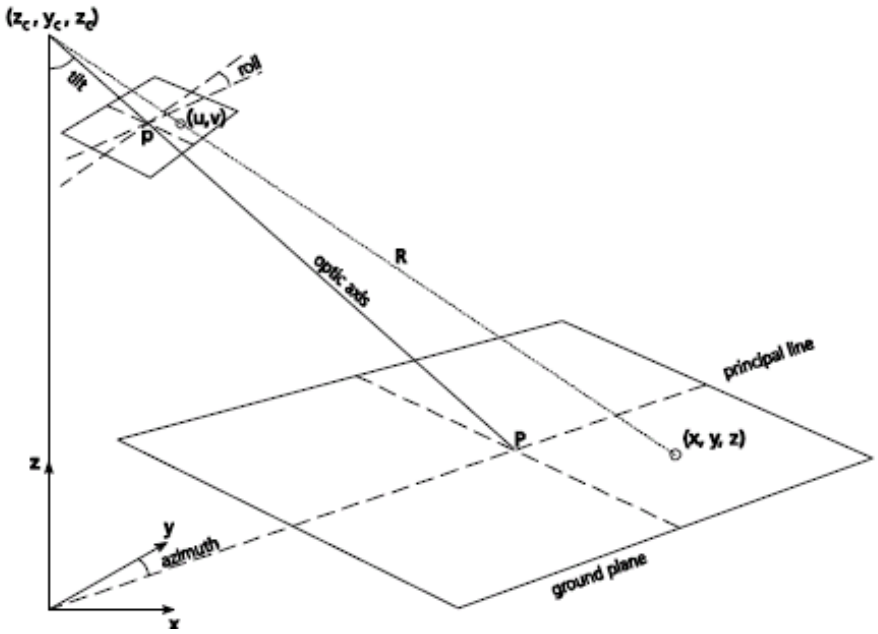


Figura 8: Esquema parámetros extrínsecos

Para encontrar los parámetros extrínsecos se realiza un proceso denominado calibración extrínseca, el cual consiste en optimizar estos parámetros al comparar las coordenadas de imágenes medidas (puntos de referencia) y pronosticadas (modelo findUV explicado más adelante).

3.3. Modelo findUV

Rutina principal del proceso de rectificación que cálcula las coordenadas UV a partir de coordenadas xyz, los parámetros intrínsecos y extrínsecos.

Para poder encontrar las ubicaciones de imagen (coordenadas 2D) a partir de las coordenadas en el mundo real (coordenadas 3D xyz), se multiplican por una matriz de transformación denominada matriz proyectiva P. A continuación se entrega el detalle algebraico.

Matrices de parámetros:

Matriz de parámetros intrínsecos: $K = \begin{bmatrix} f_x & 0 & U_0 \\ 0 & -f_y & V_0 \\ 0 & 0 & 1 \end{bmatrix}$

Matríz de rotación: R , donde a = azimuth, t = tilt, s = swing (roll)

$$R = \begin{bmatrix} cos(a)cos(s) + sin(a)cos(t)sin(s) & -cos(s)sin(a) + sin(s)cos(t)cos(a) & sin(s)sin(t) \\ -sin(s)cos(a) + cos(s)cos(t)sin(a) & sin(s)sin(a) + cos(s)cos(t)cos(a) & cos(s)sin(t) \\ sin(t)sin(a) & sin(t)cos(a) & -cos(t) \end{bmatrix}$$

Coordenadas 3D de la cámara: $C = \begin{bmatrix} xCam \\ yCam \\ zCam \end{bmatrix}$

Construcción matriz de proyección P:

$$IC = \begin{bmatrix} 1 & 0 & 0 & -xCam \\ 0 & 1 & 0 & -yCam \\ 0 & 0 & 1 & -zCam \end{bmatrix}$$

$P = K \cdot R \cdot IC$

$P = P/P(3,4) \rightarrow$ (homogenización)

Obtención coordenadas UV:

Puntos en coordenadas 3D: $xyz = \begin{bmatrix} x_1 & y_2 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \\ \vdots & \vdots & \vdots \\ x_n & y_n & z_n \end{bmatrix}_{N \times 3}$

$$UV = [P] \cdot \begin{pmatrix} [xyz]^T \\ [1 \cdots 1] \end{pmatrix}_{4 \times N} = [P] \cdot \begin{pmatrix} \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ y_1 & y_2 & \cdots & y_n \\ z_1 & z_2 & \cdots & y_n \end{bmatrix} \\ [1 \quad 1 \quad \cdots \quad \cdots \quad 1] \end{pmatrix}_{4 \times N}$$

$UV = UV./repmat(UV(3,:),3,1) \rightarrow$ (homogenización)

Distorsión angular y tangencial:

Los ajustes a las coordenadas de imagen de acuerdo a su distorsión están incluidos en la función distortCaltech(u,v,lcp) y esta opera de acuerdo a las ecuaciones descritas descritas en la sección 2.2.

[U ,V] = distort (UV(1 , :) , UV (2 , :) , lcp)

distortCaltech(u,v,lcp)

$$x = \frac{(u(:) - U_0)}{f_x} \quad ; \quad y = \frac{(v(:) - V_0)}{f_y} \quad ; \quad r^2 = x^2 + y^2 \tag{3}$$

Factor de distorsión radial: fr → Interpolación lineal utilizando el perfil de calibración del lente (lcp) y se evaluan los nuevos puntos r.

- fr = interp1(lcp.r , lcp.fr , r)

Factor de distorsión tangencial: dx, dy → Interpolación lineal 2D.

- dx = interp2(lcp.x , lcp.y , lcp.dx , x , y)
- dy = interp2(lcp.x , lcp.y , lcp.dy , x , y)

Corrección distorsión:

$x2 = x \cdot f_r + dx$

$y2 = y \cdot f_r + dy$

$u_d = x2 \cdot f_x + U_0$

$v_d = y2 \cdot f_y + V_0$

3.4. Rectificación

Calibración Extrínseca:

Se realiza una regresión no lineal utilizando los puntos de referencia (xyz y UV_r). En el algoritmo tiene la siguiente forma:

betaNew = nlinfit(xyz ,[Ur(:) ; Vr(:)], 'findUVnDOF' , $betas_i$) ;

Donde nlinfit( ) corresponde a un método de regresión no lineal de matlab, findUVnDOF es el modelo que relaciona las coordenadas de imagen con las coordenadas 3D (descrito anteriormente). Al utilizar nlinfit( ) se obtiene un vector estimado de coeficientes que corresponden a los parámetros externos.

beta = betaNew = [xCam yCam zCam Azimuth Tilt Roll]

Rectificación Grilla:

Grilla de Análisis: rectxy = [X_{min} dX X_{max} Y_{min} dY Y_{max}] → Input

z = 0

Puntos grilla: xyz = $\begin{bmatrix} x_1 & y_1 & 0 \\ x_1 & y_2 & 0 \\ \vdots & \vdots & \vdots \\ x_1 & y_{max} & 0 \\ x_2 & y_1 & 0 \\ \vdots & \vdots & \vdots \\ x_{max} & y_{max} & 0 \end{bmatrix}$

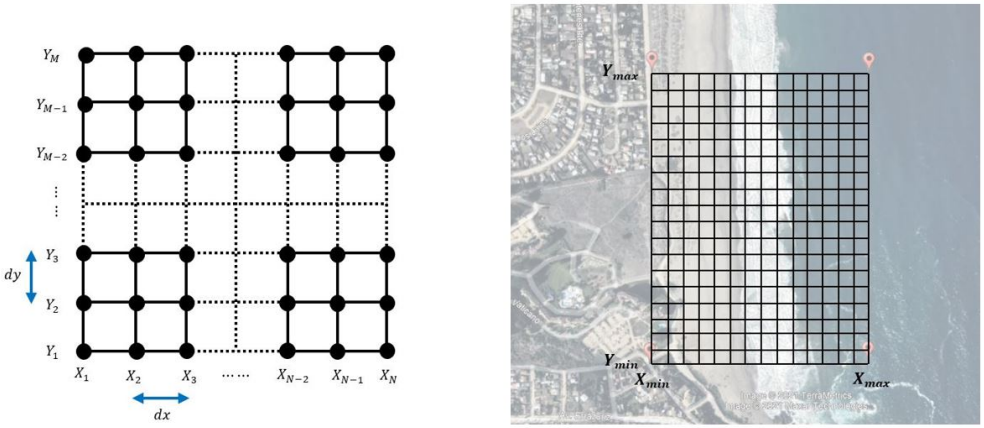


Figura 9: Grilla de Análisis

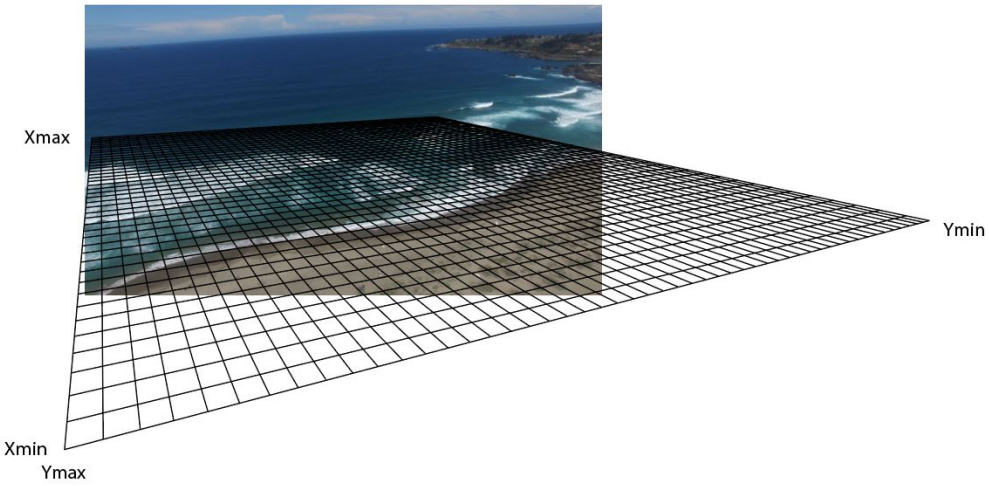


Figura 10: Representación grilla en perspectiva en la imagen

La rectificación en esta etapa consiste en aplicar la rutina principal findUV( ) sobre la grilla definida, es decir sobre las coordenadas xyz, obteniendo de esta forma sus coordenadas de imagen UV respectivas.

xyz → findUV( ) → UV (Ej. 1082101x2)

Una vez obtenidas las coordenadas de imagen UV, reordeno los pixeles correspondientes con esas coordenadas en la grilla de análisis definida, esto se logra realizando una serie de operaciones basadas en indexaciones de arreglos o matrices. Al finalizar esta etapa, el algoritmo entrega como resultado una imagen rectificada como se muestra a continuación.

Figura 11: Imagen de ejemplo rectificada

4. Corrientes Longshore

El algoritmo propuesto por Chickadel permite obtener velocidades de corrientes promediadas en dirección paralela a la línea de la costa. Los cálculos asociados a dicho algoritmo, así como las principales variables que intervienen se presentan en las siguientes secciones.

El algoritmo se encuentra disponible en:
`https://github.com/Coastal-Imaging-Research-Network/Video-Currents-Toolbox/`

4.1. Inputs

La información de entrada que requiere el algoritmo de Chickadel queda definido por una estructura de datos denominada instrumento de pixel del tipo vbar, el cual sigue las convenciones establecidas por el sistema de monitoreo de video Argus, y determina la imagen timestack que analiza el algoritmo.

Instrumento vBar

Matriz de coordenadas que define cuáles son los pixeles que se van a extraer para analizar, así como la forma en que se van a extraer estos pixeles (tipo de estructura).

Tipo de instrumento: vbar (linea paralela a linea de costa)

$$\text{xyz} = \begin{bmatrix} x_0 & y_{min} & z \\ x_0 & y_{max} & z \end{bmatrix}$$

Figura 12: Instrumento vbar

Timestack

Imagen en escala de grises que se construye a partir del instrumento vbar previamente definido.

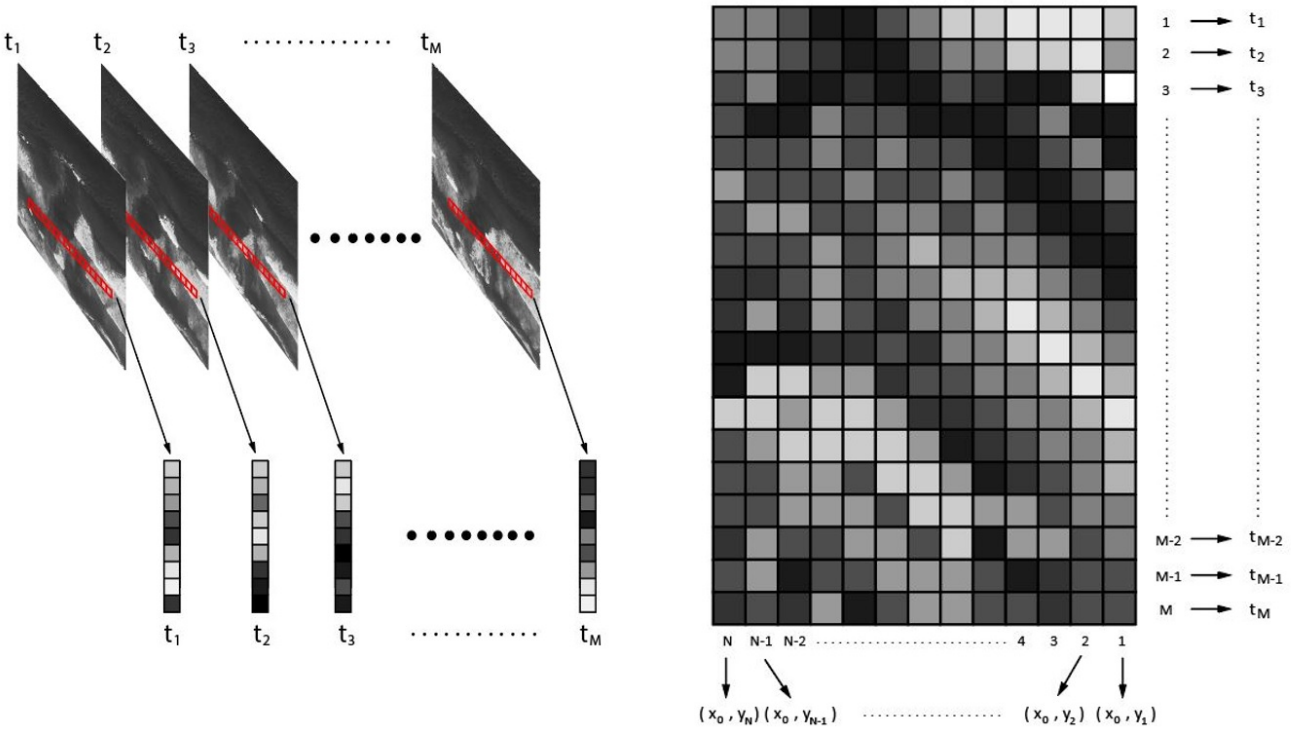


Figura 13: (a)Construcción timestack. (b) Esquema timestack.

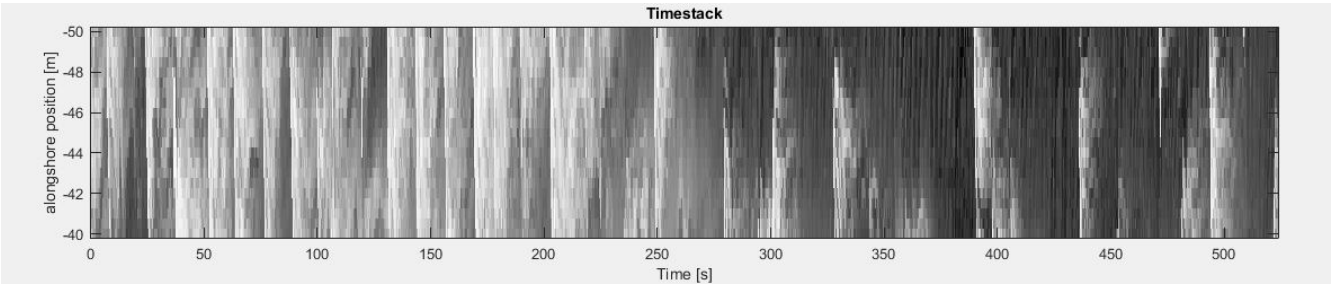


Figura 14: Ejemplo timestack

La imagen timestack se trabaja como matriz (stack) en un formato de 8 bits, es decir la intensidad en la imagen varía entre 0 y 255.

$$\text{stack} = \begin{bmatrix} I_{11} & I_{12} & \cdots & I_{1N} \\ I_{21} & I_{22} & \cdots & I_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ I_{M1} & I_{M2} & \cdots & I_{MN} \end{bmatrix} \text{ ; donde } 0 \leq I_{ij} \leq 255$$

Coordenadas timestack y vector temporal

$$\text{time} = \begin{bmatrix} t_1 & t_2 & t_3 & \cdots & t_M \end{bmatrix}$$

$$\text{xyzAll} = \begin{bmatrix} x_0 & y_1 & z \\ x_0 & y_2 & z \\ x_0 & y_3 & z \\ \vdots & \vdots & \vdots \\ x_N & y_N & z \end{bmatrix}$$

Ventanas de tiempo

En el algoritmo se definen las ventanas de tiempo T_{win} y T_{step} necesarias para el análisis a realizar.

T_{win} : Tiempo de la ventana FFT. Parámetro que define la longitud de la ventana de tiempo utilizado en el análisis de fourier y su respectivo ajuste al modelo.

T_{step} : Tiempo que define cada cuánto tiempo en el timestack se realiza el análisis.

La siguiente figura esquematiza la ubicación de T_{win} y T_{step} en el timestack.

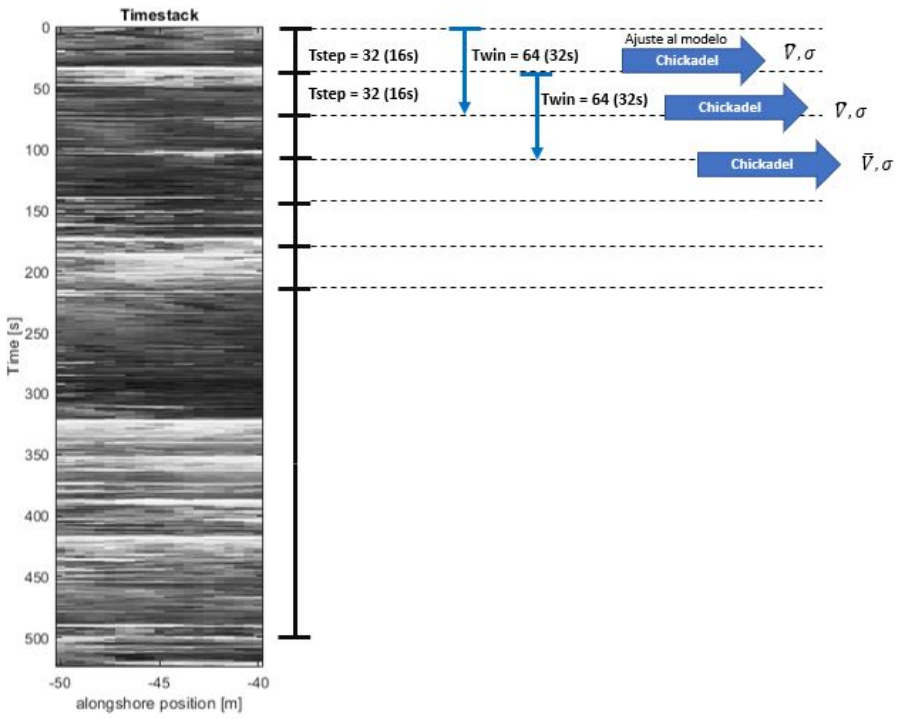


Figura 15: Esquema timestack con T_{step} y T_{win}

4.2. Algoritmo OCM

El algoritmo del tipo OCM (Optical Current Meter) propuesto por Chickadel, consiste en cuatro pasos:

- Cálculo del espectro número de onda - frecuencia $S(f, k_y)$ en dos dimensiones.
- Transformación del espectro $S(f, k_y)$ al espectro número de onda - velocidad.
- Integración sobre el número de onda para producir un "espectro" de velocidad.
- Estimación la velocidad más representativa para ese segmento.

4.2.1. Cálculo $S(f, k_y)$

El primer paso es transformar la matriz de intensidades pixel, I(t,y), desde el dominio del espacio (y) y el tiempo (t) al dominio de la frecuencia (f [Hz]) y el número de onda (k_y [1/m]) usando una transformada de Fourier en dos dimensiones, obteniendose $\hat{I}(f, k_y)$ de acuerdo a la siguiente ecuación:

$$\hat{I}(f, k_y) = \iint B(t, y) I(t, y) e^{-i2\pi f t} e^{-i2\pi k_y y} dt dy \tag{4}$$

Donde B(t,y) corresponde al filtro de Bartlett, que se multiplica para reducir la fuga espectral. Luego el espectro en dos dimensiones, $S(f, k_y)$, es calculado como:

$$S(f, k_y) = \hat{I}(f, k_y)\hat{I}(f, k_y)* \tag{5}$$

Donde el asterisco (*) denota el complejo conjugado

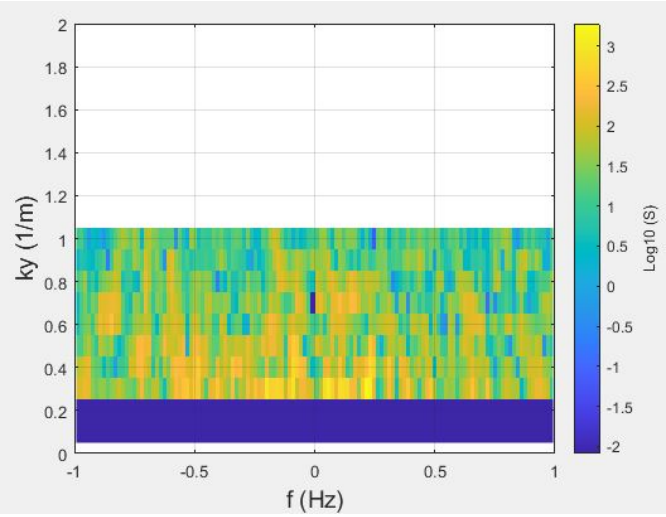


Figura 16: Espectro $S(f, k_y)$

Observación: La matriz I(t,y) se trabaja centrada en cero, es decir se le resta el promedio a cada elemento. Además el espectro $S(f, k_y)$ también se desplaza de tal forma que la frecuencia de valor cero queda en el centro, para más información consulte la función fftshift(s) en documentación de matlab.

4.2.2. Transformación del espectro $S(f, k_y)$

La velocidad de la corriente en dirección y estará asociada a una gran variedad de números de onda y frecuencia, y esta relación estará dada por la siguiente relación:

$$v = \frac{f}{k_y} \tag{6}$$

Esta relación es usada para transformar el espectro al espacio velocidad-número de onda. Para conservar la varianza en la transformación,

$$\text{var}\{S(f, k_y)\} = \iint S(f, k_y) df dk_y = \iint S(v, k_y)|k_y| dv dk_y \tag{7}$$

donde $|k_y|$ es el determinante Jacobiano y $S(v, k_y)$ es el espectro velocidad-número de onda, por lo tanto

$$S(v, k_y) = \frac{1}{|k_y|} S(f, k_y) \tag{8}$$

La velocidad se limita entre 3 y -3 m/s con la finalidad de eliminar una posible contaminación con las olas incidentes oblicuas que pueden mover a mayores velocidades la espuma de mar, estas velocidades usualmente exceden ese rango por lejos, mientras que las velocidades de corrientes están tipicamente dentro de ese rango.

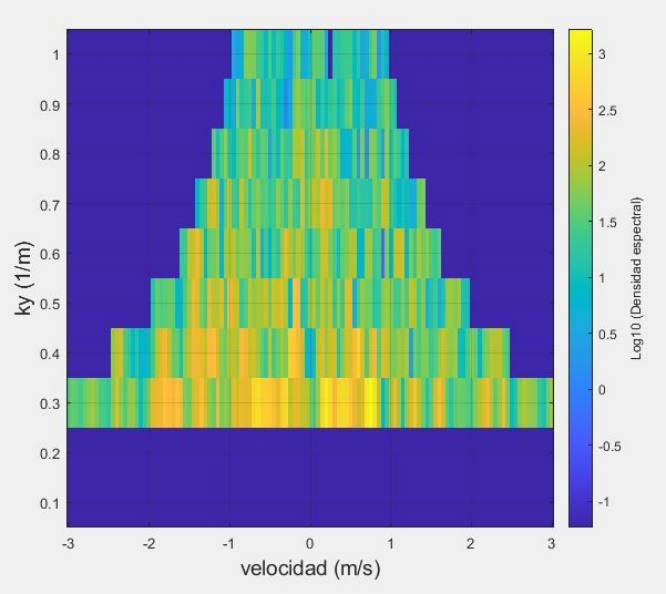


Figura 17: Espectro $S(v, k_y)$

4.2.3. Transformación al espectro $S(v)$

El espectro $S(v, k_y)$ puede ser integrado con respecto al número de onda, obteniendose un "espectro" de velocidad,

$$S(v) = \int_{k_{min}}^{k_{nyq}} S(v, k_y) dk_y \tag{9}$$

La contaminación debido a las olas oblicuas es minimizada excluyendo la energía asociada a los números de ondas menores a un k_{min} (elegido como 0.125 1/m). El límite superior de la integral es

$$k_{nyq} = \frac{1}{2\Delta y} \tag{10}$$

donde k_{nyq} es el número de onda Nyquist y Δy es el espaciado de muestreo.

El espectro de velocidad resultante (ver figura 18) tiene varias características típicas:

- Un nivel de energía de fondo debido al ruido del video.
- Patrones de intensidad de baja frecuencia.
- Un gran peak de energía que representa las trazas de espuma.

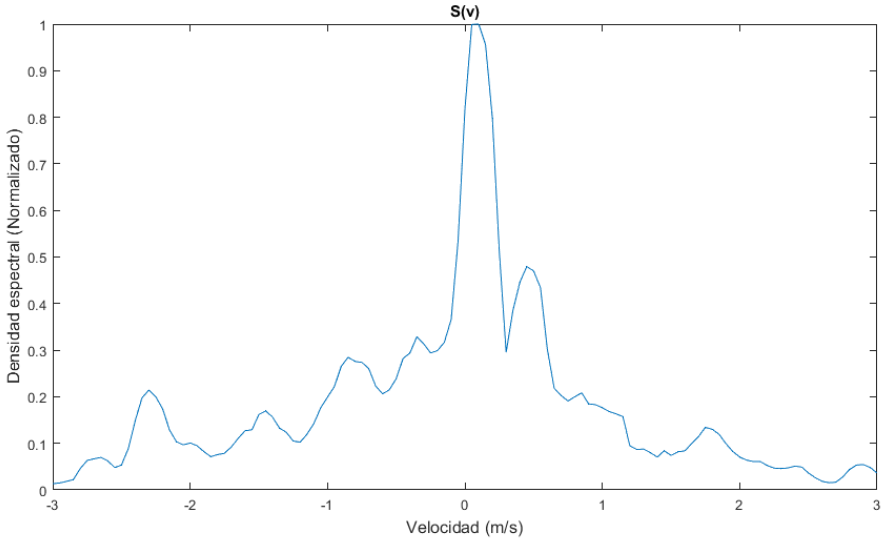


Figura 18: Espectro $S(v)$

4.2.4. Modelo S(v)

Chickadel propone un modelo de espectro de velocidad S_m(v) el cual ajusta al espectro observado S(v), para esto utiliza una rutina de mínimos cuadrados no lineal. El modelo incluye dos componentes, el ruido de fondo y las trazas de espuma oblicuas.

$$S_m(v) = S_{foam}(v) + S_{noise}(v) \tag{11}$$

El espectro de la traza de espuma, S_{foam} es modelado como una curva Gaussiana

$$S_{foam}(v) = A_{foam}\exp\left[-\frac{(v-\bar{v})^2}{\sigma^2{}_{foam}}\right] \tag{12}$$

Donde
A_{foam}: valor peak de S(v)
$\bar{v}$: promedio de la velocidad de espuma
σ_{foam}: desviación estandar de la velocidad de espuma

Para modelar el ruido se asume una serie de tiempo de ruido blanco (señal aleatoria que contiene todas las frecuencias y todas ellas muestran la misma energía asociada) distribuido uniformemente sobre $S(f, k_y)$. Transformando este ruido blanco al espectro de velocidad se tiene

$$S_{noise}(v) = \begin{cases} A_{noise}\frac{f_{\mathrm{nyq}}^2}{2v^2} & si \quad |v| \leq \frac{f_{\mathrm{nyq}}}{k_{\mathrm{nyq}}} \\ A_{noise}\frac{k_{\mathrm{nyq}}^2}{2} & si \quad |v| > \frac{f_{\mathrm{nyq}}}{k_{\mathrm{nyq}}} \end{cases} \tag{13}$$

Donde
A_{noise}: es el factor de amplitud del ruido
f_{nyq}: Frecuencia Nyquist
k_{nyq}: Número de onda Nyquist.

El modelo depende entonces de cuatro parámteros A_{foam}, $\bar{v}$, σ_{foam}, A_{noise}. La rutina utilizada aplica un método iterativo para encontrar el mejor ajuste de estos parámetros. Finalmente el valor $\bar{v}$ será la velocidad más representativa del intervalo en análisis. Un ejemplo del espectro obtenido con el modelo se muestra a continuación.

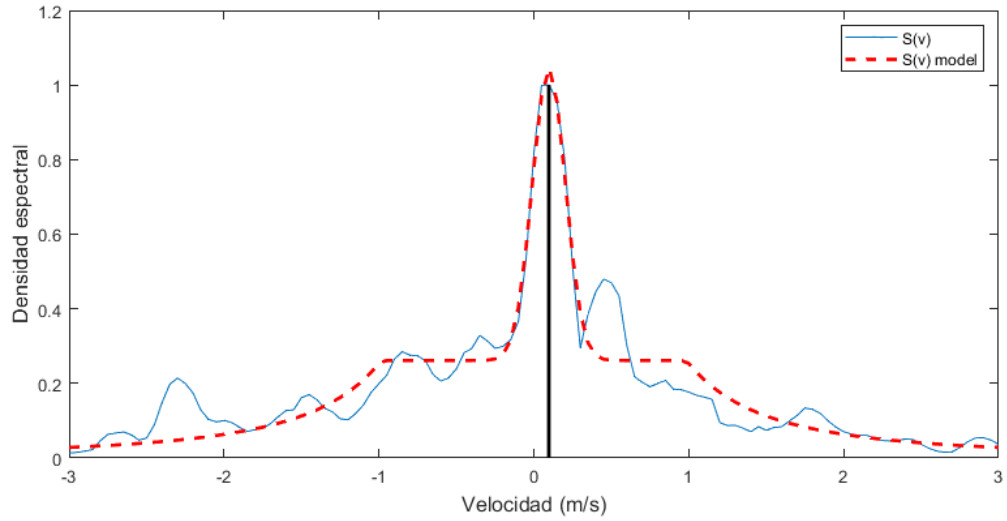


Figura 19: Ejemplo 1: Espectro $S(v)$ y $S_m(v)$

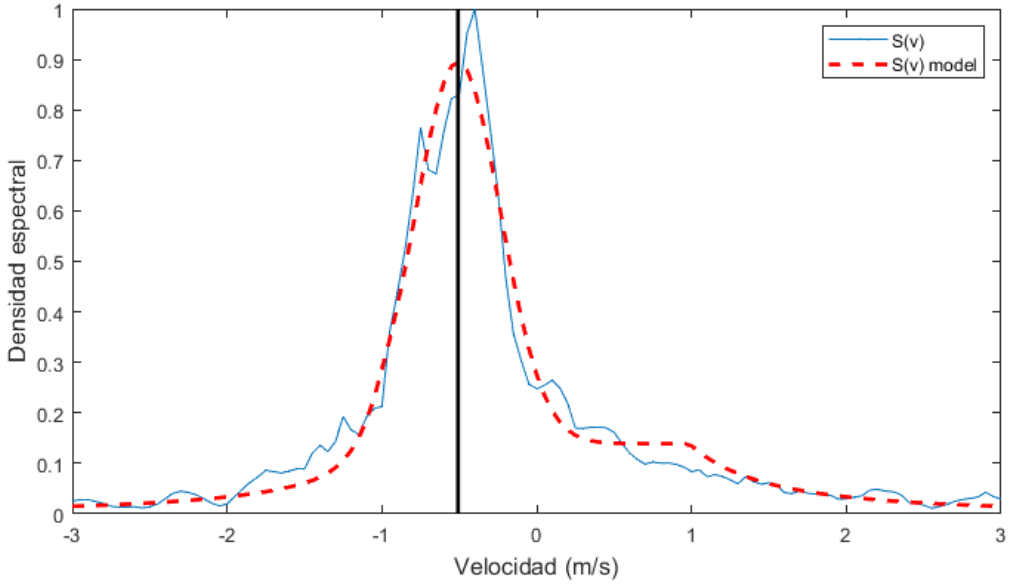


Figura 20: Ejemplo 2: Espectro $S(v)$ y $S_m(v)$

4.3. Cálculo serie de tiempo velocidad de corriente

Es posible estimar una serie de tiempo de velocidad de corriente longshore (ver figura 21), aplicando el algoritmo OCM repetidamente y de forma gradual a lo largo del registro de video de modo que la ventana de análisis, T_{win}, se superponga a la ventana anterior por cierta cantidad de tiempo. La duración de tiempo de la ventana de anásis, T_{win}, y la duración del tiempo de paso, T_{step}, son constantes sin restricciones. El aumento de T_{win} aumenta la estabilidad de la estimación resultante, pero disminuye la resolución temporal de la serie.

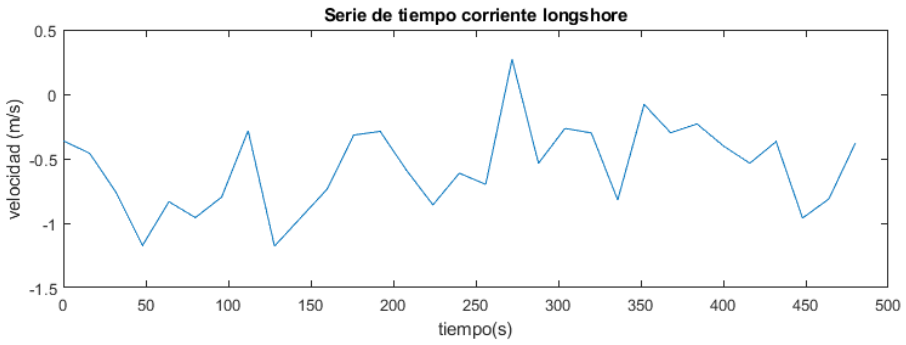


Figura 21: Serie de tiempo velocidad de corriente longshore

4.4. Calidad de la estimación

La calidad de la velocidad estimada se evaluó calculando algunas medidas de bondad y los mejores parámetros de ajuste a los datos. Además, debido a que este método se basa en la presencia de espuma debido a la ruptura de la onda, Chickadel ideó un método para proporcionar un filtro objetivo para identificar y rechazar estimaciones calculadas a partir de imágenes que no contenían suficiente contraste de espuma. Estas medidas de calidad proporcionan una base para ignorar los casos en los que no se esperan resultados útiles.

Los métodos utilizados se muestran a continuación:

Test χ^2

La calidad del ajuste del modelo a los datos se describe usando el test estadístico χ^2.

$$\chi^2 = \sum_{i=1}^{N} \left[\frac{S(v(i)) - S_m(v(i); \hat{\beta}}{\sigma(i)} \right]^2 \tag{14}$$

donde $\hat{\beta}$ es el set de parámetros con mejor ajuste y σ es la desviación estándar del error de medición en cada punto i.

Indicador rotura y espuma residual

Se calcula un indicador para el grado de rotura y espuma residual en el video, I_{range}, en función del histograma de intensidad de la ventana del timestack

$$I_{range} = I_{95} - I_{50} \tag{15}$$

donde I_{95} y I_{50} son los valores de intensidad correspondientes a los percentiles 95 y 50 respectivamente.

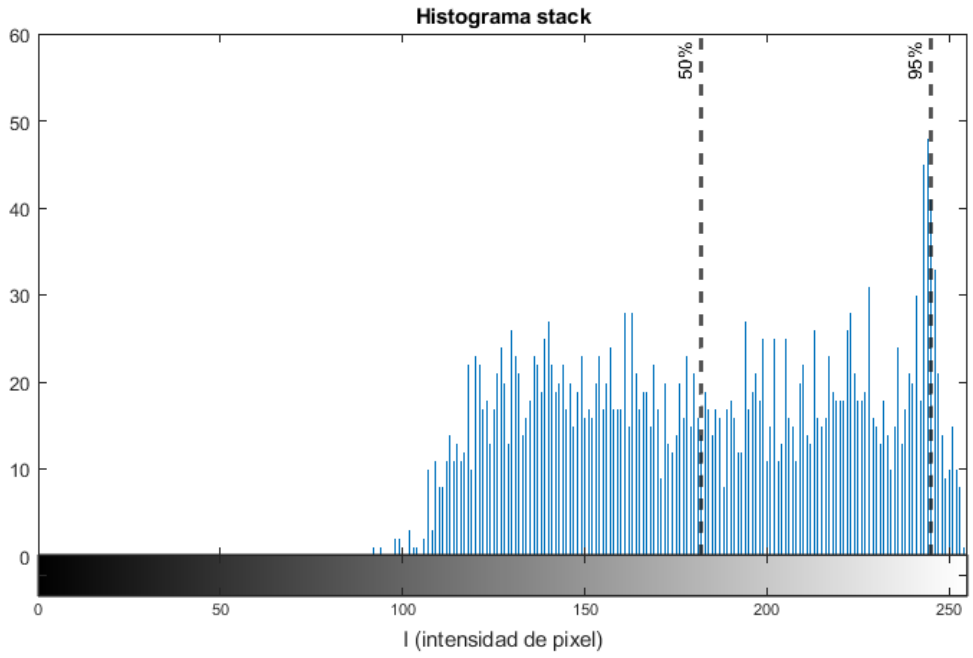


Figura 22: Histograma stack (32s), las lineas discontinuas representan los persentiles 50 y 95.

A medida que aumenta el valor de I_{range}, también lo hace el contraste, el grado de rotura de ola y la espuma residual observada en el timestack.

4.4.1. Criterio de aceptación serie temporal

De acuerdo con Chickadel, las estimaciones individuales del OCM de la velocidad de superficie, v_s, se consideran aceptables si satisfacen tres criterios:

- Nivel de confianza del ajuste al modelo (Test χ^2) mayor al 90 %
- Intervalos de confianza de 95 % de la estimación de los parámetros (del modelo ajustado) menor que 0.2 m/s
- I_{range} >40

Chickadel consideró validas las corrientes medias si al menos 10 de las 63 estimaciones, para cada serie de tiempo, pasaban los criterios anteriores, lo que equivale a un 16 % aproximadamente.

4.5. Resumen algoritmo OCM

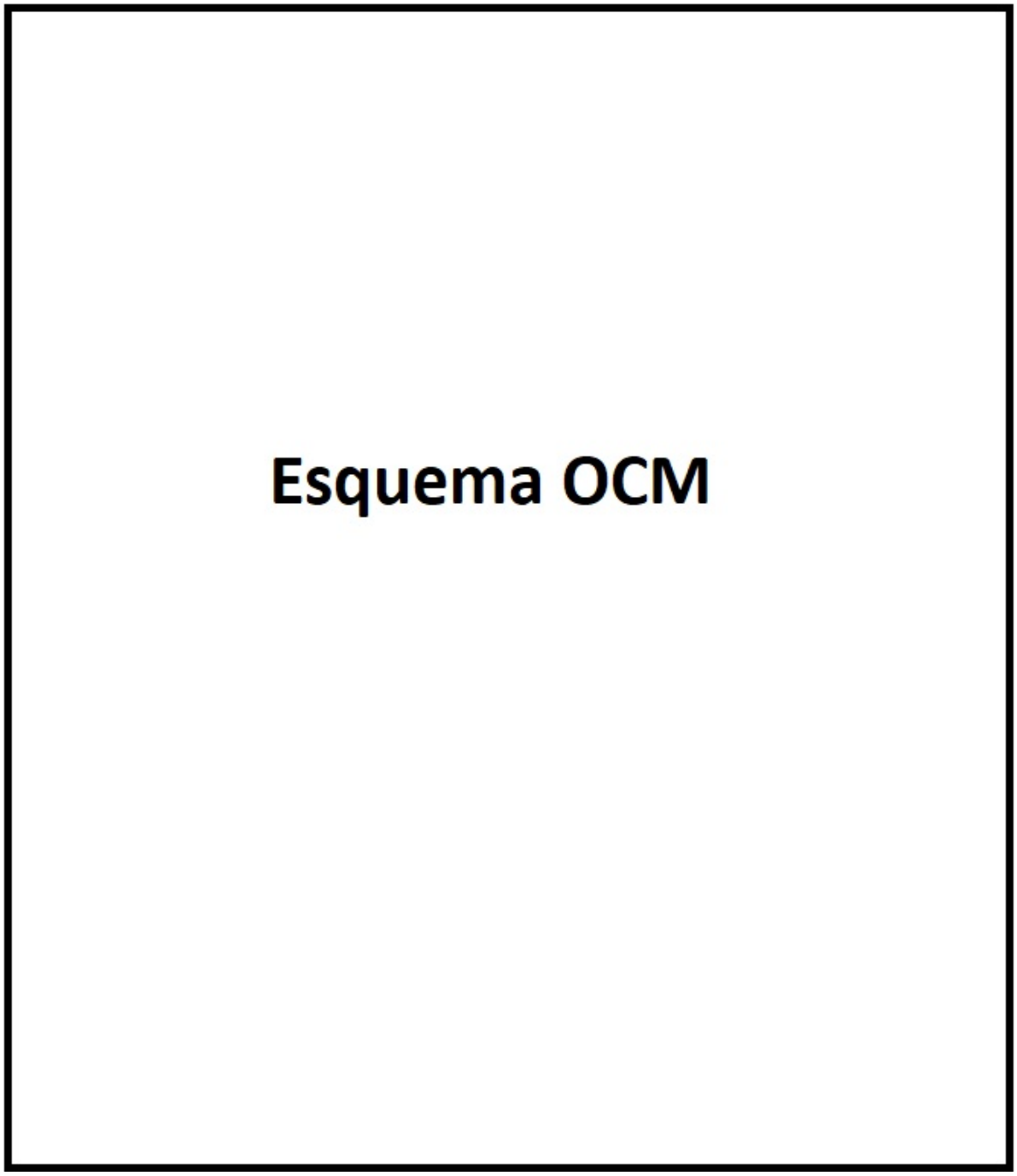


Figura 23: Esquema resumen algoritmo OCM

5. Cálculo campo de velocidades

5.1. Malla vBar

Se comienza definiendo una malla de instrumentos vBar dispuestos uno al lado del otro. Cada uno de estos posee una longitud en dirección paralela a linea de costa, Ly, y una separación en direción x, dx.

Dimensiones Instrumentos vBar:

- $Ly \rightarrow$ Longitud instrumentos vBar
- $dx \rightarrow$ Separación instrumentos en dirección x
- $Xlim = [Xl_1, Xl_2] \rightarrow$ Límites grilla en dirección x
- $Ylim = [Yl_1, Yl_2] \rightarrow$ Límites grilla en dirección y

Un esquema de la malla de instrumentos definida y un ejemplo se muestra en la siguiente figura:

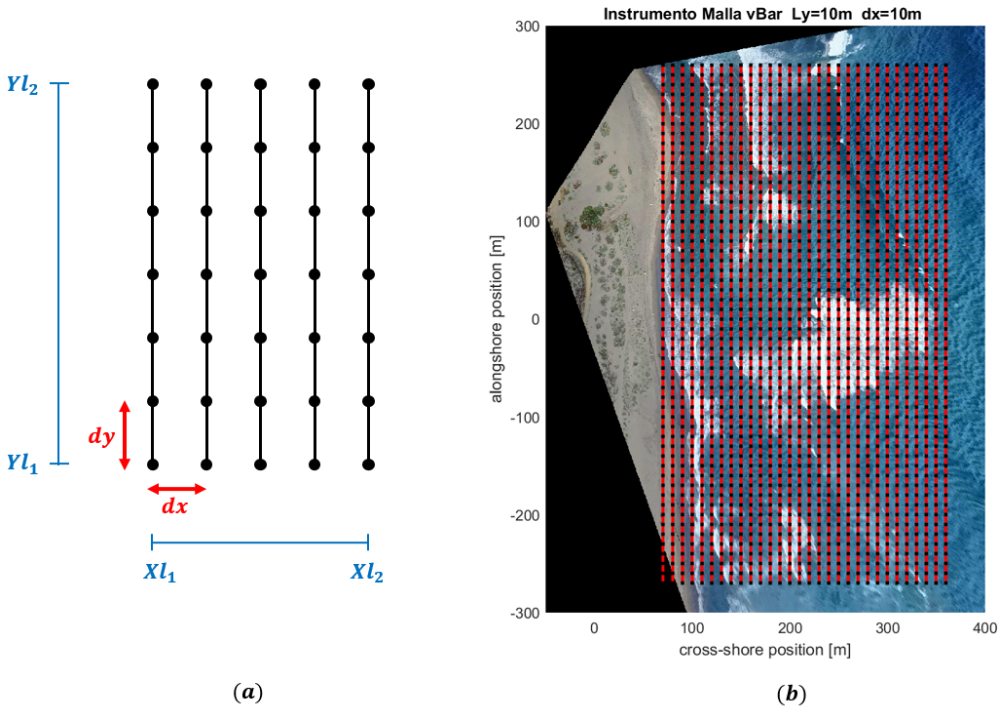
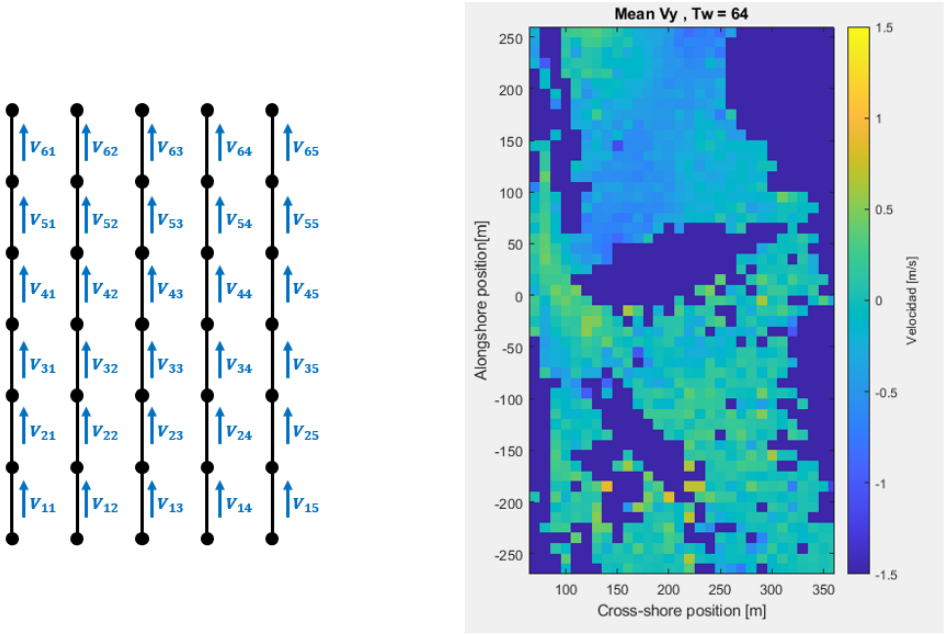


Figura 24: (a) Esquema grilla vBar. (b) Ejemplo grilla vBar

5.2. Matriz de velocidades

El proceso siguiente consiste en aplicar el algoritmo de Chickadel sobre todos los instrumentos vbar definidos en la grilla, para los cuales obtengo su velocidad de corriente longshore correspondiente, como se muestra en la figura25.b



(*a*) (*b*)

Figura 25: (a) Esquema velocidades (ejemplo matriz 6x5). (b) Velocidades obtenidas promediando los resultados obtenidos para 31 ventanas de tiempo (ejemplo matriz 53x30).

5.3. Binarización y filtrado

Figura 26: Imágenes binarias utilizadas como molde para la interpolación de las velocidades faltantes

Con el objetivo de eliminar algunos elementos NaN (Not a Number) en la matriz de velocidades y reemplazarlos por valores interpolados linealmente, se aplican una serie de filtros y operaciones morfológicas a una imagen binarizada, esto permitirá, en pasos posteriores (siguiente sección), aplicar las ecuaciones de continuidad sin tanta interrupción en los datos.

En primer lugar, la matriz de velocidades se binariza asignando unos a los elementos que contienen un valor de velocidad y ceros a los que no. Posteriormente se aplica una serie de operaciones morfológicas y filtrados gaussianos pertenecientes al conjunto de funciones de procesamiento de imágenes de Matlab.

En la figura 26 se muestra la primera imagen binarizada, una correspondiente a una fase intermedia y la imagen final utilizada para realizar la interpolación de velocidades.

Una vez aplicado el filtro con ayuda de las imágenes binarias, se realiza la interpolación de las velocidades. Un ejemplo de esto se muestra en la figura 27.

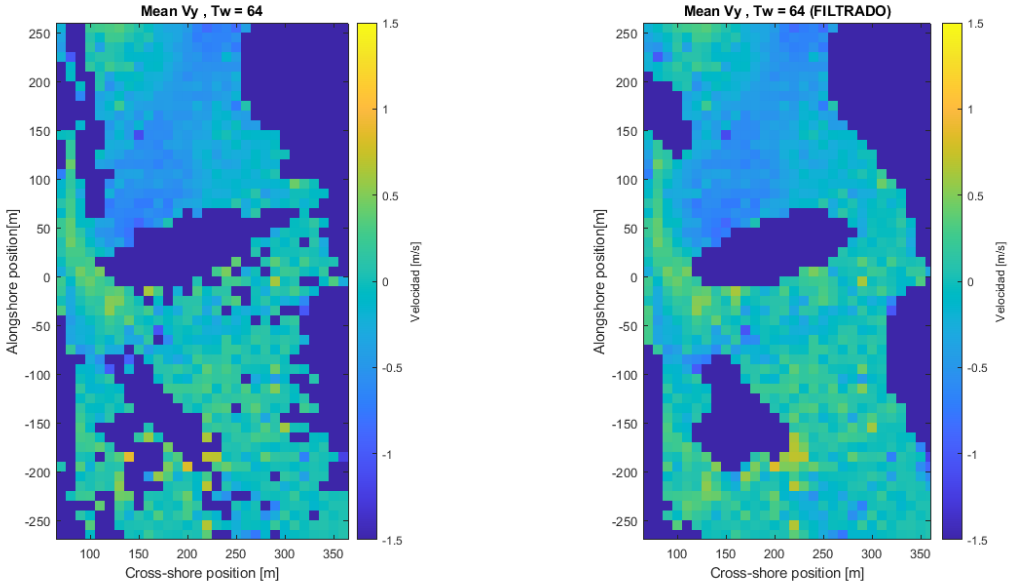


Figura 27: (izq) Matriz de velocidades original. (der) Matriz con velocidades interpoladas

5.4. Cálculo velocidad de corriente cross-shore (U)

Para obtener la velocidad de corriente en dirección cross-shore (U) se utiliza un modelo bidimensional donde se asume que el flujo en el plano horizontal predomina por sobre lo que ocurre en dirección vertical, de esta forma, utilizando la discretización definida en la malla vBar, se establece una malla cuadricular compuesta por elementos cuadrados donde cada lado tiene asociado un caudal o velocidad (U en dirección x, V en dirección y) y profundidad, ver figura 28. Para este caso, la variable que se quiere encontrar es el campo de velocidades U, es decir, el vector de incógnitas está compuesto por todos los elementos U_{ij}.

El campo de velocidades en todo el dominio es posible obtenerlo a partir de un balance de flujos, es decir, utilizado la ecuación de continuidad,

$$\frac{\partial \bar{\eta}}{\partial t} + \frac{\partial \bar{Q}_x}{\partial x} + \frac{\partial \bar{Q}_y}{\partial y} = 0 \tag{16}$$

Esta ecuación nos permite afirmar que los flujos que ocurren en la zona de la rompiente estarán determinados por un balanceo entre los gradientes de flujo másico en las dos direcciones horizontales ($\bar{Q}_x$ y $\bar{Q}_y$) y el nivel medio ($\bar{\eta}$).

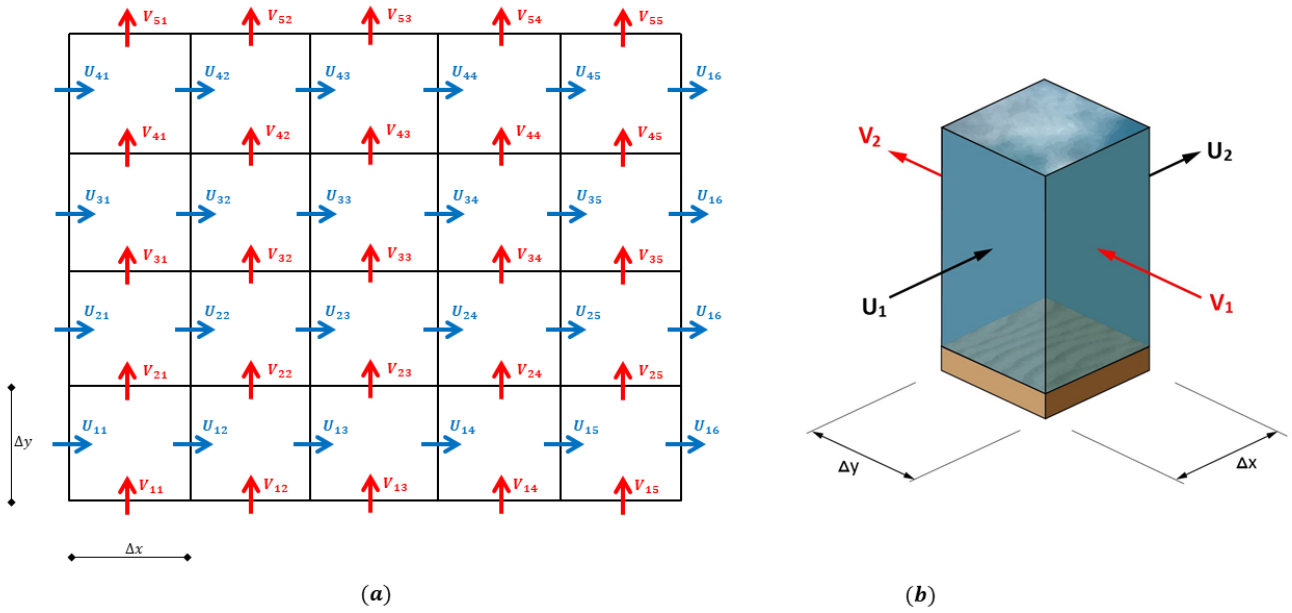


Figura 28: (a) Esquema representativo malla UV. (b) Volumen de control elemento grilla UV

Para efectos de este estudio se asume un flujo permanente, es decir, existe una condición de equilibrio donde los gradientes de flujo en dirección $\hat{x}$ están balanceados con los gradientes de flujo en dirección $\hat{y}$, de esta forma es posible asumir que el nivel medio del mar permanece constante.

$$\cancelto{0}{\frac{\partial \bar{\eta}}{\partial t}} + \frac{\partial \bar{Q_x}}{\partial x} + \frac{\partial \bar{Q_y}}{\partial y} = 0 \tag{17}$$

Utilizando como volumen de control los elementos definidos en la grilla UV, se aplica la ecuación de continuidad de forma discreta sobre cada columna de agua, una representación de esto se muestra en la figura 28 (b) y en la figura 29, donde se considera un flujo entrante con velocidad U_1 y V_1 en direción $\hat{x}$ e $\hat{y}$ respectivamente y un flujo de salida V_2 y U_2, además, cada lado del volumen de control tiene asociada una profundidad h_{x1}, h_{x2}, h_{y1} o h_{y2}.

Obs 1 : Se utilizan los subindices 1 y 2 en las variables para simplificar la notación.

Obs 2 : Las profundidades h_{x1}, h_{x2}, h_{y1} y h_{y2} son datos que se deben conocer para poder determinar el campo de velocidades U, por lo tanto, la batimetría de la playa es un dato necesario. Esta puede ser determinada utilizando cBathy (ver sección 6) o utilizando algún otro método que se estime conveniente.

$$\frac{\partial Q_x}{\partial x} = -\frac{\partial Q_y}{\partial y} \tag{18}$$

$$Q_x = -\frac{\partial Q_y}{\partial y} \cdot \partial x + C_1 \quad \text{(integrando)} \tag{19}$$

$$Q_2 = -\frac{\Delta Q_y}{\Delta y} \cdot \Delta x + Q_1 \quad \text{(discretizando)} \tag{20}$$

$$U_2 \cdot h_{x2} = -\frac{V_2 \cdot h_{y2} - V_1 \cdot h_{y1}}{\Delta y} \cdot \Delta x + U_1 \cdot h_{x1} \quad \text{(aplicando definición de flujo másico)} \tag{21}$$

$$U_2 = \frac{1}{h_{x2}} \left(\frac{\Delta x}{\Delta y} \left(V_2 \cdot h_{y2} - V_1 \cdot h_{y1} \right) + U_1 \cdot h_{x1} \right) \tag{22}$$

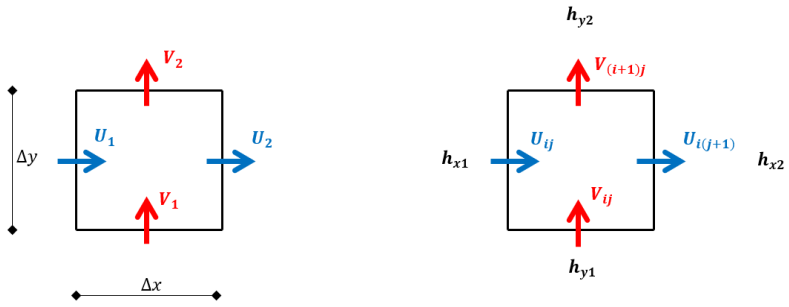


Figura 29: Diagrama volumen de control

La ecuación 22 permite determinar el campo de velocidades U en todo el dominio, salvo en ocasiones donde alguna velocidad no se conoce, por lo que se requiere imponer ciertas condiciones. Las condiciones impuestas en la frontera así como los casos donde falta información se describen a continuación.

5.4.1. Condiciones en caso de no existir data

Si $U_1 \neq$ NaN , $V_1 =$ NaN y $V_2 \neq$ NaN

$\Rightarrow V_1 = 0$

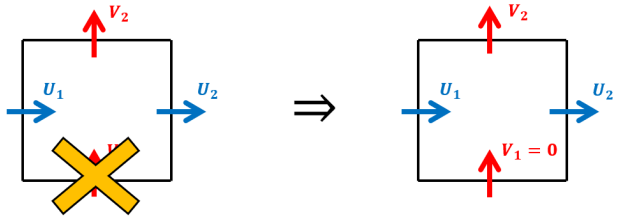


Figura 30: Caso 1

Si $U_1 \neq$ NaN , $V_1 \neq$ NaN y $V_2 =$ NaN

$\Rightarrow V_2 = 0$

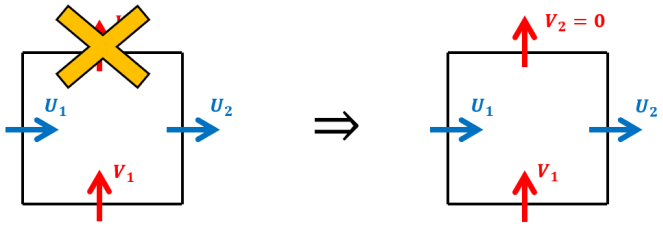


Figura 31: Caso 2

Si $U_1 \neq \text{NaN}$, $V_1 = \text{NaN}$ y $V_2 = \text{NaN}$

$\Rightarrow U_2 = \text{NaN}$

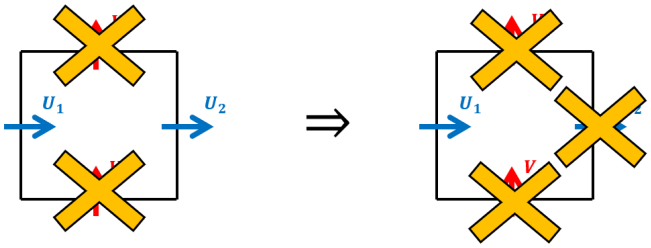


Figura 32: Caso 3

Si $U_1 = \text{NaN}$, $V_1 \neq \text{NaN}$ y $V_2 \neq \text{NaN}$

$\Rightarrow U_1 = 0$

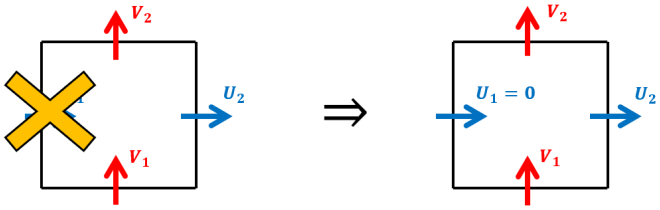


Figura 33: Caso 4

Si $U_1 = \text{NaN}$, $V_1 = \text{NaN}$ y $V_2 \neq \text{NaN}$

$\Rightarrow U_1 = 0$ y $V_1 = 0$

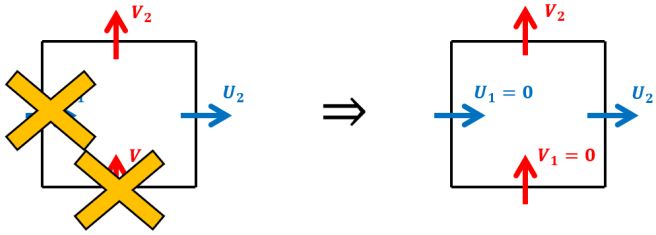


Figura 34: Caso 5

Si $U_1 = \text{NaN}$, $V_1 \neq \text{NaN}$ y $V_2 = \text{NaN}$

$\Rightarrow U_1 = 0$ y $V_2 = 0$

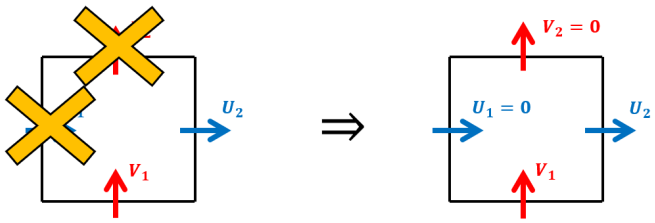


Figura 35: Caso 6

Si $U_1 = \text{NaN}$, $V_1 = \text{NaN}$ y $V_2 = \text{NaN}$

$\Rightarrow U_2 = \text{NaN}$

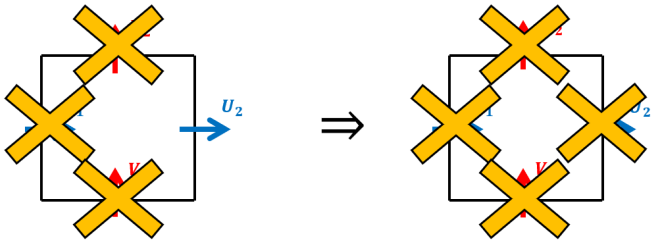


Figura 36: Caso 7

5.4.2. Cálculo campo de velocidades

A continuación se muestra un ejemplo de los resultados obtenidos, estos se obtuvieron al trabajar con valores promediados (e interpolados en ciertos lugares), tal como se muestra en la figura 27 b. La primera imagen 37 muestra un campo de velocidades U obtenido a partir de la ecuación de continuidad, mientras que la figura 38 muestra el campo de velocidades completo, incluyendo la magnitud de la corriente y las trayectorias que definen.

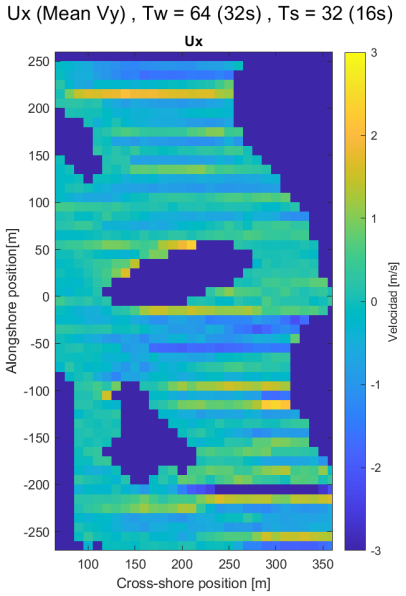


Figura 37: Campo de velocidades U

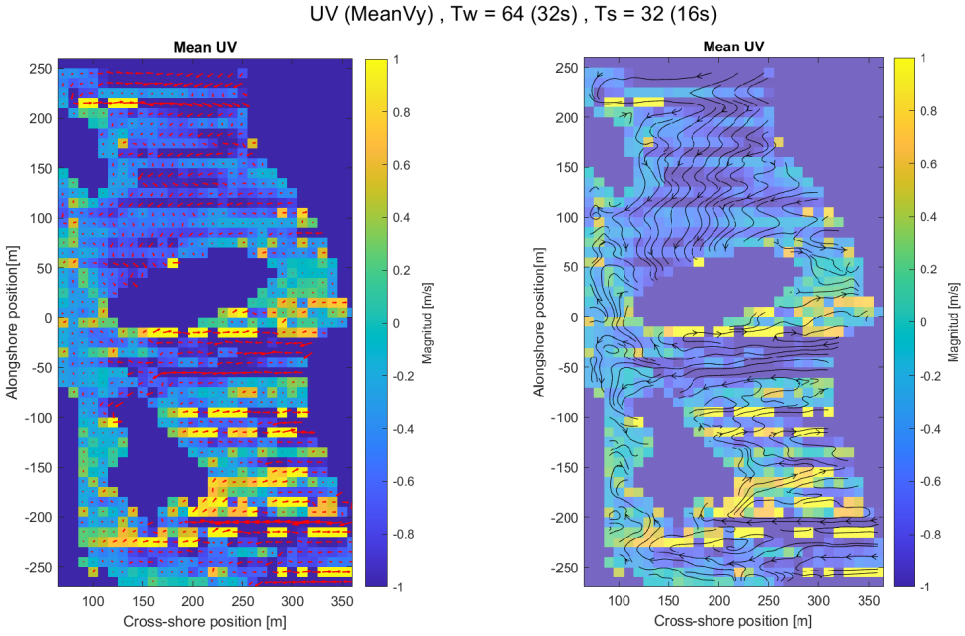


Figura 38: Campo de velocidades UV

5.5. Cálculo velocidad U: Método Matricial

El sistema de ecuaciones planteado en la sección anterior, también se puede resolver de manera matricial. Para esto, se realiza la formulación matricial a partir de las ecuaciones de continuidad de cada volumen de control analizado. A continuación se presenta el desarrollo para un sistema de 12 grados de libertad, lo que equivale a 4 celdas dispuestas según se esquematiza en la figura 39. Posteriormente se generaliza para un sistema más grande.

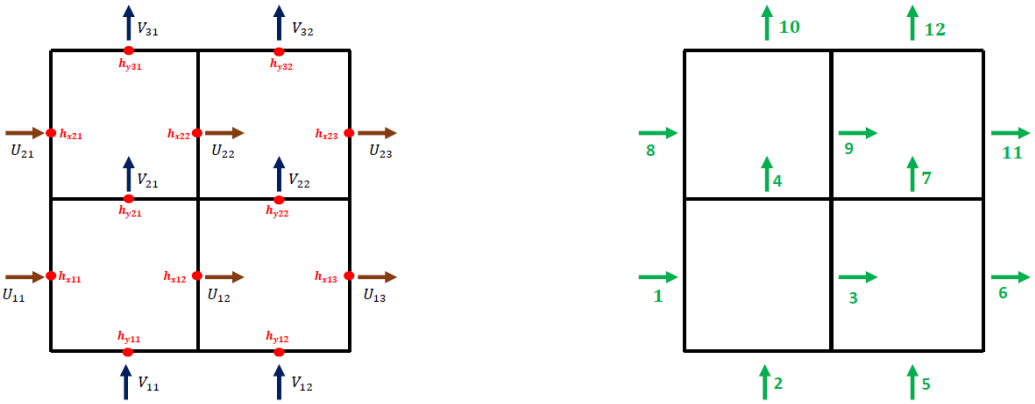


Figura 39: (a) Esquema de velocidades, sistema de 4 celdas. (b) Grados de libertad para un sistema de 4 celdas

Es conveniente analizar un bloque (volumen de control) en un sistema local y luego llevarlo a uno global. Se considera que un bloque o celda tiene un total de 4 grados de libertad (uno por cada lado), y cada grado de libertad tiene asociado una velocidad y profundidad, tal como se ilustra en la figura 40.

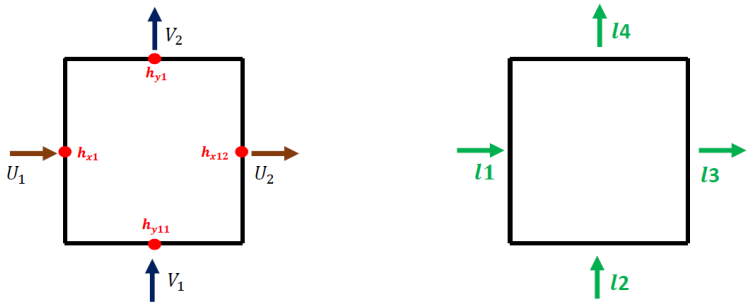


Figura 40: (a) Esquema volumen de control de un bloque. (b) Esquema grados de libertad de un bloque

El primer paso es plantear la ecuación de continuidad del bloque. Se asume que Δx es igual a Δy por lo que no intervienen en la ecuación.

$$\begin{bmatrix} h_{x1} & h_{y1} & -h_{x2} & -h_{y2} \end{bmatrix} \cdot \begin{bmatrix} U_1 \\ V_1 \\ U_2 \\ V_2 \end{bmatrix} = 0 \tag{23}$$

Definiendo la matriz de profundidades local H_L y vector de velocidades UV_L :

$$[H_L] \cdot \{UV_L\} = 0 \tag{24}$$

El objetivo siguiente es compatibilizar los grados de libertad locales con los globales, por lo que se definen las matrices de transformación C. Estas matrices permiten llevar las variables locales a una matriz global representativa de todo el sistema.

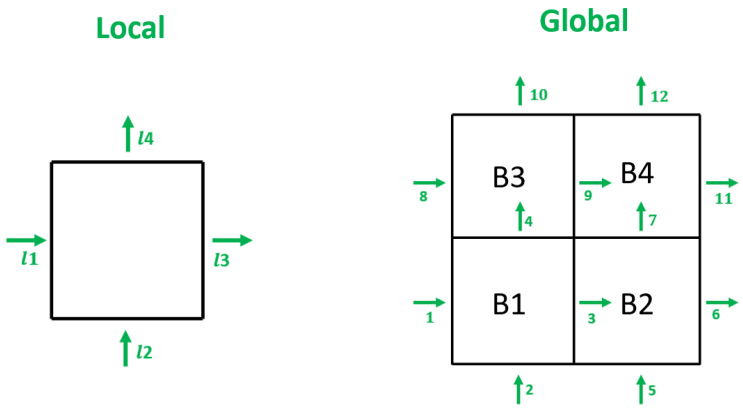


Figura 41: (a) Grados de libertad sistema local. (b)Grados de libertad sistema global

$$\begin{bmatrix} g_1 \\ g_2 \\ g_3 \\ g_4 \\ g_5 \\ g_6 \\ g_7 \\ g_8 \\ g_9 \\ g_{10} \\ g_{11} \\ g_{12} \end{bmatrix} \Rightarrow \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}}_{C_1} \cdot \begin{bmatrix} l_1 \\ l_2 \\ l_3 \\ l_4 \end{bmatrix} ; \underbrace{\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}}_{C_2} \cdot \begin{bmatrix} l_1 \\ l_2 \\ l_3 \\ l_4 \end{bmatrix} ; \underbrace{\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}}_{C_3} \cdot \begin{bmatrix} l_1 \\ l_2 \\ l_3 \\ l_4 \end{bmatrix} ; \underbrace{\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{C_4} \cdot \begin{bmatrix} l_1 \\ l_2 \\ l_3 \\ l_4 \end{bmatrix}$$

Aplicando las transformaciones a las matrices de profundidad:

$$[H_{L_1}] \cdot {C_1}^T = \begin{bmatrix} h_{x11} & h_{y11} & -h_{x12} & -h_{y21} \end{bmatrix} \cdot {C_1}^T = \begin{bmatrix} h_{x11} & h_{y11} & -h_{x12} & -h_{y21} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \text{ fila 1}$$

$$[H_{L_2}] \cdot {C_2}^T = \begin{bmatrix} h_{x12} & h_{y12} & -h_{x13} & -h_{y22} \end{bmatrix} \cdot {C_2}^T = \begin{bmatrix} 0 & 0 & h_{x12} & 0 & h_{y12} & -h_{y13} & -h_{y22} & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \text{ fila 2}$$

$$[H_{L_3}] \cdot {C_3}^T = \begin{bmatrix} h_{x21} & h_{y21} & -h_{x22} & -h_{y31} \end{bmatrix} \cdot {C_3}^T = \begin{bmatrix} 0 & 0 & 0 & h_{y21} & 0 & 0 & 0 & h_{x21} & -h_{x22} & -h_{y31} & 0 & 0 \end{bmatrix} \text{ fila 3}$$

$$[H_{L_4}] \cdot {C_4}^T = \begin{bmatrix} h_{x22} & h_{y22} & -h_{x23} & -h_{y32} \end{bmatrix} \cdot {C_4}^T = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & h_{y22} & 0 & h_{x22} & 0 & -h_{x23} & -h_{y32} \end{bmatrix} \text{ fila 4}$$

Por lo tanto la matriz de profundidades H queda:

$$\mathrm{H} = \begin{bmatrix} h_{x11} & h_{y11} & -h_{x12} & -h_{y21} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & h_{x12} & 0 & h_{y12} & -h_{y13} & -h_{y22} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & h_{y21} & 0 & 0 & 0 & h_{x21} & -h_{x22} & -h_{y31} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & h_{y22} & 0 & h_{x22} & 0 & -h_{x23} & -h_{y32} \end{bmatrix}$$

Ecuación de continuidad para todo el sistema:

$$\mathrm{H} \cdot \mathrm{UV} = 0$$

$$\begin{bmatrix} h_{x11} & h_{y11} & -h_{x12} & -h_{y21} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & h_{x12} & 0 & h_{y12} & -h_{y13} & -h_{y22} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & h_{y21} & 0 & 0 & 0 & h_{x21} & -h_{x22} & -h_{y31} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & h_{y22} & 0 & h_{x22} & 0 & -h_{x23} & -h_{y32} \end{bmatrix} \cdot \underbrace{\begin{bmatrix} U_{11} \\ V_{11} \\ U_{12} \\ V_{21} \\ V_{12} \\ U_{13} \\ V_{22} \\ U_{21} \\ U_{22} \\ V_{31} \\ U_{23} \\ V_{32} \end{bmatrix}}_{UV} = 0$$

Destacando columnas y filas asociados a grados de libertad conocidos:

$$\begin{bmatrix} h_{x11} & h_{y11} & -h_{x12} & -h_{y21} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & h_{x12} & 0 & h_{y12} & -h_{y13} & -h_{y22} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & h_{y21} & 0 & 0 & 0 & h_{x21} & -h_{x22} & -h_{y31} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & h_{y22} & 0 & h_{x22} & 0 & -h_{x23} & -h_{y32} \end{bmatrix} \cdot \begin{bmatrix} U_{11} \\ V_{11} \\ U_{12} \\ V_{21} \\ V_{12} \\ U_{13} \\ V_{22} \\ U_{21} \\ U_{22} \\ V_{31} \\ U_{23} \\ V_{32} \end{bmatrix} = 0$$

Dado que todas las velocidades V_{ij} se conocen y algunos valores de U_{ij} también (por condiciones de borde), es posible reescribir la ecuación matricial eliminando las columnas de la matriz H correspondientes con los grados de libertad conocidos (marcados en color rojo) y generando un vector, B, que contiene la multiplicación de las velocidades y profundidades asociadas a los grados de libertad conocidos:

$$\underbrace{\begin{bmatrix} -h_{x12} & 0 & 0 & 0 \\ h_{x12} & -h_{x13} & 0 & 0 \\ 0 & 0 & -h_{x22} & 0 \\ 0 & 0 & h_{x22} & -h_{x23} \end{bmatrix}}_{Hr} \cdot \underbrace{\begin{bmatrix} U_{12} \\ U_{13} \\ U_{22} \\ U_{23} \end{bmatrix}}_{Ur} = \underbrace{-\begin{bmatrix} h_{x11} \cdot U_{11} + h_{y11} \cdot V_{11} - h_{y21} \cdot V_{21} \\ h_{y12} \cdot V_{12} - h_{y22} \cdot V_{22} \\ h_{y21} \cdot V_{21} + h_{x21} \cdot U_{21} - h_{y31} \cdot V_{31} \\ h_{y22} \cdot V_{22} - h_{y32} \cdot V_{32} \end{bmatrix}}_{B}$$

En el sistema reducido se definen las siguientes matrices:

Hr : Matriz de profundidades reducida
Ur : Vector de velocidades U reducido
B : Matriz B

Finalmente las incógnitas Ur, se obtienen invirtiendo la matriz Hr y premultiplicando por B:

$$U_r = Hr^{-1} \cdot B \tag{25}$$

Generalización

Generalizando para un sistema de (M-1) x N celdas (como por ejemplo el mostrado en la figura 28.a de 4 x 5), donde Δx es igual a Δy, con nx grados de libertad en dirección $\hat{x}$ y ny grados de libertad en dirección $\hat{y}$, se plantea la ecuación matricial de continuidad para el sistema:

$$\mathrm{H} \cdot \mathrm{UV} = 0$$

donde:

H : Matriz general de profundidades (Nc x Ng)

fila i de H = $\mathrm{H}_{Li} \cdot C_i^T$ i = 1, 2, 3, ..., Nc

H_{Li} : Matriz de profundidad local de la celda i (1x4)

dada por la continuidad a nivel local en cada celda, $[H_{Li}] \cdot \{UV_{Li}\} = 0$

UV : Vector general de velocidades (Ng x 1)

C_i : Matriz de transformación que compatibiliza grados de libertad locales con globales

UV_{Li} : Vector de velocidades local de la celda i (4x1)

V : Matriz de velocidades longshore (M x N)

Nc : Número de celdas, Nc = (M-1) x N

Ng : Número total de grados de libertad, Ng = nx + ny

nx = Número de grados de libertad en dirección $\hat{x}$, nx = (M-1) x (N+1)

ny = Número de grados de libertad en dirección $\hat{y}$, ny = M x N

Dado que existen grados de libertad conocidos, gk, y grados libertad no conocidos, gu, asociados a las incógnitas buscadas), el sistema se puede reducir a:

$$Hr \cdot U_r = B \tag{26}$$

Hr : Matriz de profundidades reducida (Nc x Nc)

columnas gu de la matriz H

Ur : Vector de velocidades U reducido (Nc x 1)

B : Vector B, cada elemento se calcula como $\mathrm{B}_i = -\sum_{jk} \mathrm{h}_{i,jk}\mathrm{UV}_{jk}$

donde $\rightarrow$ subíndice i: asociado a la celda i

subíndice jk : grados de libertad conocidos

Obs: Dado que la formulación se realiza suponiendo que no existen valores NaN dentro del sistema, el número de grados de libertad no conocidos coincide con el número de celdas, Nc.

De esta forma, las velocidades U_r se encuentran invirtiendo la matriz H_r y premultiplicando por B:

$$U_r = Hr^{-1} \cdot B \tag{27}$$

En caso de no existir data, se aplican las mismas condiciones expuestas en la sección anterior, tratándolo como grado de libertad conocido en caso de que la velocidad sea cero, y eliminando el grado de libertad en el caso que sea NaN.

6. Cbathy: un algoritmo robusto para estimar batimetrías

cBathy es un algoritmo para la estimación de batimetría a partir de imágenes de video que se basa en la conocida dependencia de la celeridad de onda, c, con la profundidad. A partir de pruebas exhaustivas, el algoritmo actual es bastante preciso y resistente al ruido.

cBathy se basa en la relación de dispersión lineal:

$$\sigma^2 = gk * tanh(kh) \tag{28}$$

Donde σ es la frecuencia angular, k es el número de onda, g la aceleración de gravedad y h la profundidad.

Para más información puede visitar el repositorio online donde se encuentra almacenado el algoritmo `https://github.com/Coastal-Imaging-Research-Network/cBathy-Toolbox/`, o directamente el artículo dónde se expone con más detalle las bases de cBathy `https://agupubs.onlinelibrary.wiley.com/doi/full/10.1002/jgrc.20199`.

6.1. Inputs

Datos de entrada para el análisis principal de cBathy

- Matriz de muestreo espacial: $xyz = \begin{bmatrix} x_1 & y_2 & 0 \\ x_2 & y_2 & 0 \\ x_3 & y_3 & 0 \\ \vdots & \vdots & \vdots \\ x_M & y_M & 0 \end{bmatrix}_{M\times 3}$
- Matriz de muestreo temporal: $t = \begin{bmatrix} t_1 & t_2 & t_3 & \cdots & t_N \end{bmatrix}_{1\times N}$
- Matriz de series de tiempo: data $= \begin{bmatrix} I_{11} & I_{12} & \cdots & I_{1M} \\ I_{21} & I_{22} & \cdots & I_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ I_{N1} & I_{N2} & \cdots & I_{NM} \end{bmatrix}_{N\times M}$
  Por cada pixel definido según la matriz xyz, hay una serie temporal que representa la variación de intensidad en el tiempo.

6.2. Parámetros de configuración

La caja de herramientas (Toolbox) de cBathy contiene un archivo de configuración donde se deben definir todos los parámetros de análisis. A continuación se describen los parámetros principales:

- dxm y dym : Espaciamiento deseado para los puntos de análisis, espaciamiento de salida.
- xyMinMax : Mínimo y máximo de x e y.
- MINDEPTH y MAXDEPTH: Límites para la profundidad.
- Lx y Ly : Parámetros que definen la vecindad que se analiza por cada punto.
- fB : Rango de frecuencias de análisis.
- nKeep : N° de frecuencias que se consideran en el análisis (las que poseen mayor coherencia dentro de fB).

6.3. Algoritmo cBathy

La figura 42 muestra la matriz de muestreo de píxeles (diezmada por 2 para evitar saturación). Los puntos azules corresponden a las ubicaciones $[x_p, y_p]$, cada uno de los cuales le corresponde una serie temporal de intensidad de píxel. El análisis se lleva a cabo secuencialmente en una serie de puntos de análisis seleccionados por el usuario,$[x_m, y_m]$, uno de los cuales se indica con un asterisco rojo, y se basa en datos de los píxeles inmediatamente circundantes (puntos verdes) dentro de un rango especificado por el usuario, L_x, L_y. Dentro de cada uno de esos mosaicos, el objetivo es estimar el número de onda, k , para cada uno de un conjunto de frecuencias candidatas, f_b, que abarcan la banda de onda incidente. Para cada par (f_b, k), se puede estimar una profundidad $\bar{h}(x_m, y_m)$, usando la ecuación de dispersión lineal (ecuación 28), o una sólo profundidad , $\hat{h}(x_m, y_m)$, que mejor se ajuste a todas las frecuencias.

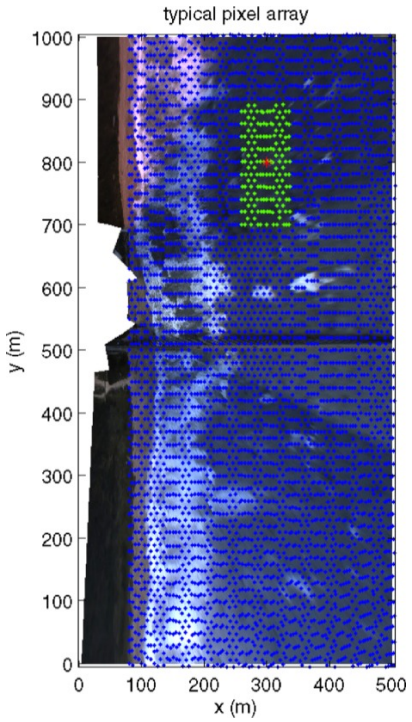


Figura 42: Matriz de píxeles de ejemplo utilizada para el análisis de cBathy

Las estimaciones pueden ser deficientes o imposibles a veces debido al clima, el resplandor del sol o el mar en calma, por lo que las estimaciones de las recopilaciones de datos por hora se promedian objetivamente para producir una profundidad promedio móvil estable $\bar{h}(x_m, y_m)$. Así, el análisis final de cBathy en cada punto consta de tres etapas:

- Fase 1 : Análisis dependientes de la frecuencia de f_b, k, α y $\bar{h}$ donde α el ángulo de onda, es un producto colateral.
- Fase 2 : Obtención de una profundidad única $\hat{h}$ que mejor se ajuste a todas las frecuencias.
- Fase 3 : Estimación de una profundidad promedio móvil, $\bar{h}$ (utilizando filtro de kalman).

6.4. Estructura bathy

La variable bathy que devuelven los algoritmos de cBathy es una estructura compacta que contiene todos los resultados de análisis relevantes en un solo lugar. La estructura general contiene seis campos iniciales que documentan los detalles del análisis, seguidos de tres subestructuras, una para cada fase del análisis. Un ejemplo del formato de la estructura cbathy es:

```
bathy =
epoch: '1285174800'
sName: [1x64 char]
params: [1x1 struct]
tide: [1x1 struct]
xm: [1x43 double]
ym: [1x41 double]
fDependent: [1x1 struct]
fCombined: [1x1 struct]
runningAverage: [1x1 struct]
```

Epoch

Hora de inicio considerada para el análisis de los datos en formato estándar Unix (segundos desde la medianoche del 1 de enero de 1970).

sName

Nombre del archivo que se analizó.

Params

Subestructura que contiene todas las configuraciones de entrada del archivo de configuración (argus02a).

Tide

Estructura que contiene la estimación del nivel de la marea que se utiliza para corregir las estimaciones locales de la profundidad de cBathy (medidas en relación con la superficie del mar) a un marco de referencia vertical local, por ejemplo, NGVD88 o AHD. Si bathy.tide.zt == nan, su valor predeterminado, entonces no se ha realizado ninguna corrección del nivel de la marea, por lo que este campo es importante para documentar si se ha realizado o no esta corrección.

xm y ym

xm e ym son los vectores de los puntos de análisis para los que se estima bathy y están determinados por el archivo de configuración (argus02a).

fDependent

La estructura contiene la salida de la fase 1 de cBathy en la que todos los resultados se expresan aún como una función de la frecuencia. Cada campo incluirá mapas de resultados que tengan el mismo número de filas que hay en ym, el mismo número de columnas que hay en xm, y el número de planos (mapas) es igual a nKeep, el número de frecuencias que tiene el usuario definió guardar en el archivo de configuración.

fCombined

La subestructura fCombined contiene los resultados de la fase 2 del algoritmo cBathy. Estos son mapas de las estimaciones de profundidad en cada ubicación de análisis que mejor se ajustan a una combinación ponderada de

toda la información de frecuencia y número de onda disponible. Esto produce un mapa de batimetría único, junto con información de error de ajuste.

runningAverage

La subestructura runningAverage contiene resultados de la etapa de filtrado de Kalman, fase 3 del algoritmo cBathy. Contiene los resultados finales de cBathy, con errores, así como información utilizada en la operación de filtrado de Kalman. Mientras que las fases 1 y 2 de cBathy dependen solo de la recopilación de datos actual, el filtrado de Kalman depende tanto del resultado de bathy actual como del anterior. Dado que las recopilaciones de datos a menudo se recuperan de una estación Argus fuera de servicio, las fases 1 y 2 generalmente se ejecutan mediante un código separado (analyseBathyCollect) para un conjunto de recopilaciones de datos, luego el filtrado de Kalman de la fase 3 se ejecuta más tarde (usando runningFilterBathy), después de que todas las recopilaciones de datos se hayan recuperado y analizado hasta la fase 2, para completar la estructura runningAverage. El filtrado es un cálculo rápido y también se puede volver a realizar en cualquier momento si el usuario desea cambiar el filtro de alguna manera, por ejemplo, cambiando el error del proceso.

6.5. Ejemplo cBathy

1 - Archivo de entrada

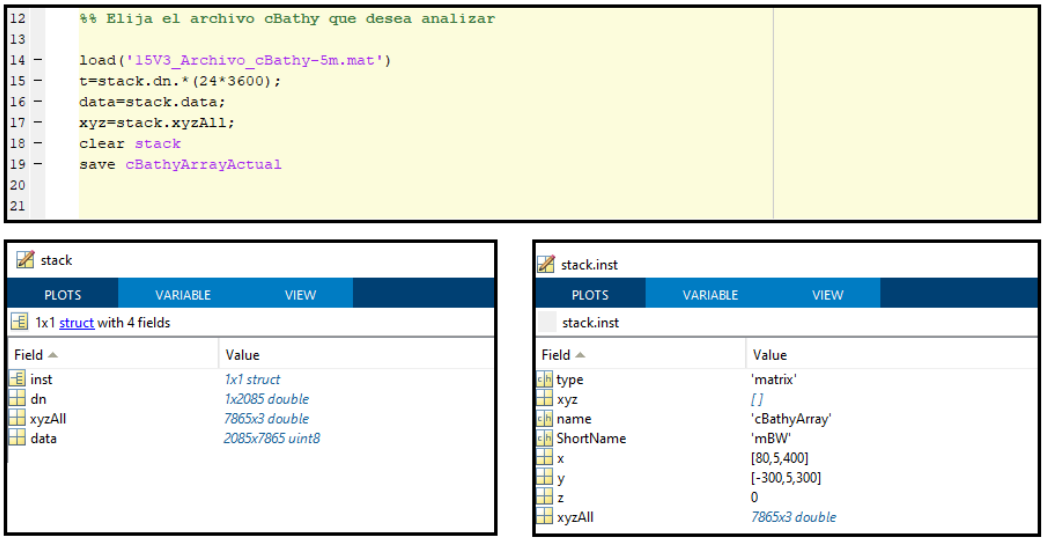


Figura 43: Ejemplo cBathy. Carga de archivo con el instrumento de pixel tipo matriz, previamente definido

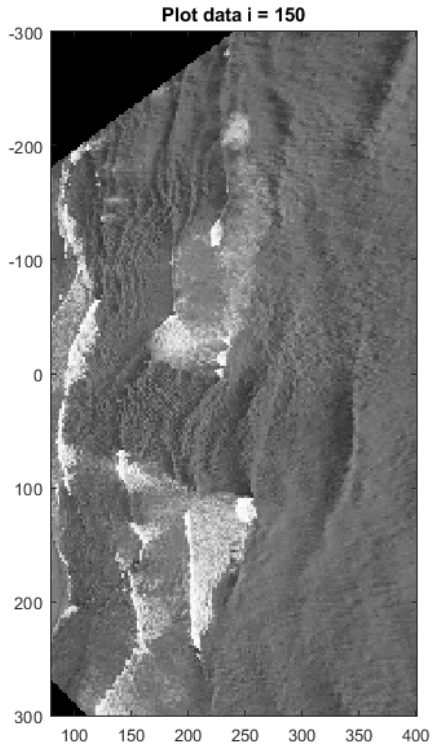


Figura 44: Plot frame de ejemplo, contenido en la estructura data

2 - Configuración de parámetros

```
%%% Site-specific Inputs
params.stationStr = 'argus02a';

params.dxm = 10;  %4 %5                % analysis domain spacing in x
params.dym = 10;  %8 %10               % analysis domain spacing in y
params.xyMinMax = [80 400 -300 300];   % min, max of x, then y
                                       % default to [] for cBathy to choose
params.tideFunction = 'cBathyTide';    % tide level function for evel

%%%%%%%   Power user settings from here down   %%%%%%%
params.MINDEPTH = 0.25;              % for initialization and final QC
params.minValsForBathyEst = 4;       % min num f-k pairs for bathy est.

params.QTOL = 0.5;                   % reject skill below this in csm
params.minLam = 10;                  % min normalized eigenvalue to proceed
params.Lx = 3*params.dxm;            % tomographic domain smoothing
params.Ly = 3*params.dym;            %
params.kappa0 = 2;                   % increase in smoothing at outer xm
params.DECIMATE = 1;                 % decimate pixels to reduce work load.
params.maxNPix = 80;                 % max num pixels per tile (decimate excess)

% f-domain etc.
params.fB = [1/18: 1/50: 1/4];       % frequencies for analysis (~40 dof)
params.nKeep = 4;                    % number of frequencies to keep

% debugging options
params.debug.production = 1;
params.debug.DOPLOTSTACKANDPHASEMAPS = 1;  % top level debug of phase
params.debug.DOSHOWPROGRESS = 1;           % show progress of tiles
params.debug.DOPLOTPHASETILE = 1;          % observed and EOF results per pt
params.debug.TRANSECTX = 200;          % for plotStacksAndPhaseMaps
params.debug.TRANSECTY = 900;          % for plotStacksAndPhaseMaps

% default offshore wave angle.  For search seeds.
params.offshoreRadCCWFromx = 0;

```

Figura 45: Ejemplo cBathy. Configuración de parámetros

3 - Ejecución algoritmo cBathy (Fase 1 y 2)

```
%% RUN the cBathyDemo


% Do a single cBathy analysis, returning the bathy structure (which you
% would normally save somewhere) and displaying the results.

stationStr = 'argus02a';
stackName = 'cBathyArrayActual';
bathy = analyzeSingleBathyRunNotCIL(stackName, stationStr);
bathy.params.debug.production=1;

plotBathyCollect(bathy)

```

Figura 46: Ejemplo cBathy. Ejecución cBathy

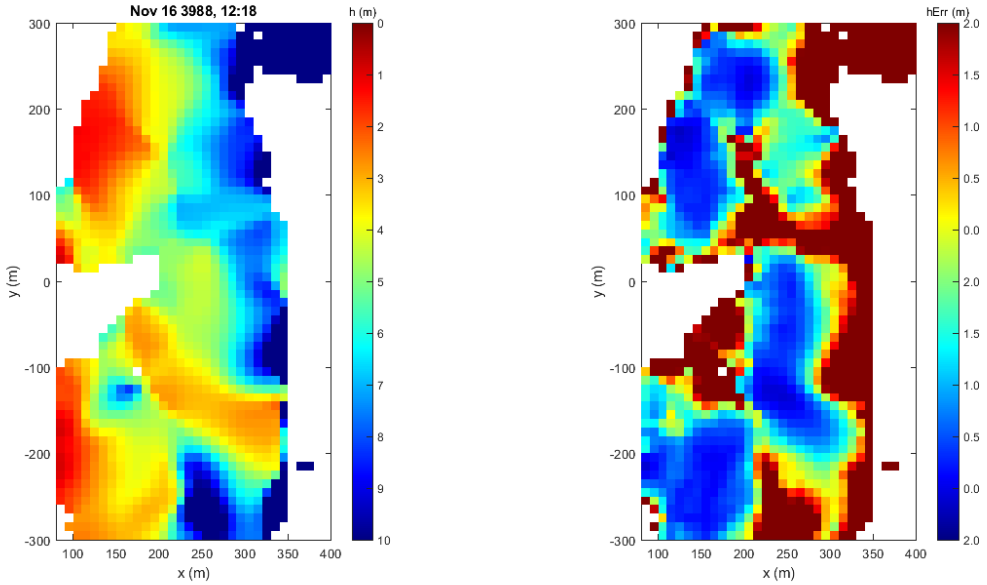


Figura 47: Plot resultados fase 2. Profundidad y error asociado

4 - Aplicación filtro de Kalman

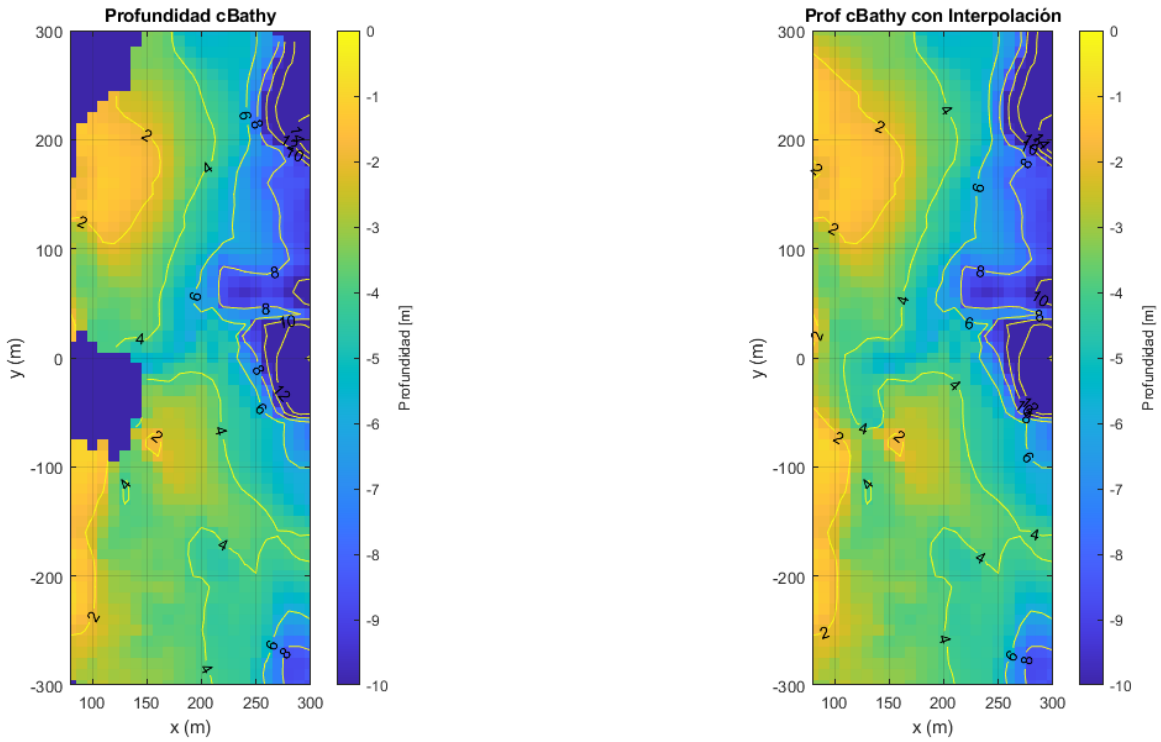


Figura 48: Ejemplo cBathy. Profundidad obtenida luego de haber aplicado el filtro de kalman utilizando 7 videos