FACULTAD DE MATEMÁTICA, ASTRONOMÍA, FÍSICA Y COMPUTACIÓN

UNIVERSIDAD NACIONAL DE CÓRDOBA

TRABAJO ESPECIAL DE LA LINCECIATTURA EN CIENCIAS DE LA COMPUTACIÓN

Seguimiento de partículas en videos de microscopía

REYES MARTÍN GABRIEL

DIRECTOR:
Sánchez, Jorge A.

2017

Abstract

This work presents a modular algorithm that aims to reconstruct trajectories of particles from sequences of images obtained with optical microscopy techniques, and extract quantitative statistical information about the movements of the same. In first instance, the coordinates of the particles are obtained, after applying combinations of different filters to clean the images and obtain local maxima when applying a Laplacian operator. Then the detections of the different video frames are assembled with an algorithm based on Kalman filters. The algorithm of trajectory reconstruction is evaluated and compared with known experiments. Finally, relevant information that will serve to characterize the trajectories is extracted.

Keywords: particles, trajectories, detection, linking, Kalman

Resumen

En este trabajo se presenta un algoritmo modular que tiene como objetivo reconstruir trayectorias de partículas a partir de secuencias de imágenes obtenidas con técnicas de microscopía óptica, y extraer información estadística cuantitativa acerca de los movimientos de las mismas. En primera instancia, se obtienen las coordenadas de las partículas, luego de aplicar combinaciones de distintos filtros para limpiar las imágenes y obtener máximos locales al aplicar un operador Laplaciano. Luego se ensamblan las detecciones de los diferentes cuadros de video con un algoritmo basado en filtros de Kalman. El algoritmo de reconstrucción de trayectorias se evalúa y compara con experimentos conocidos. Finalmente se extrae información relevante que servirá para caracterizar las trayectorias.

Palabras clave: partículas, trayectorias, detección, enlace, Kalman

Agradecimientos

A mi familia, por su apoyo incondicional a lo largo de toda la carrera, y en especial a mis padres Álvaro y Graciela, y a mi hermana Gabriela, por bancarme siempre y darme la posibilidad de crecer de forma personal y también profesional, dentro de la facultad y fuera de ella.

A mi novia Ivana, por acompañarme incondicionalmente y brindarme siempre sus mejores deseos.

A mis amigos aquí en Córdoba y en mi querido pueblo, que siempre estuvieron conmigo y que hicieron que todos estos años se pasen volando.

A mi director Jorge, por su infinita paciencia a lo largo de esta última etapa de carrera y por brindarme su apoyo para poder cumplir este sueño.

A Pedro y Verónica por darme la confianza para llevar a cabo este proyecto.

Finalmente, quiero agradecer a todos aquellos que pude conocer dentro de la facultad. Profesores y compañeros que me ayudaron durante todos estos años para llegar hasta aquí.

Gracias

Índice general

1. Introducción 3

2. Marco Teórico 5

2.1. Detección de partículas 5

2.1.1. Reducción de ruido en imágenes digitales 5

2.1.2. Redimensionado de las imágenes con interpolación bilineal 10

2.1.3. Detección de blobs 11

2.2. Filtros de Kalman para enlazar detecciones 15

2.2.1. Filtros de Kalman discretos 16

2.2.2. Algoritmo del filtro de Kalman 18

3. Seguimiento de partículas 21

3.1. Detección de las partículas 22

3.1.1. Problemas a la hora de realizar detecciones 23

3.1.2. Preprocesamiento de las imágenes 24

3.1.3. Detección de blobs con filtros Laplacianos 26

3.2. Enlace de las detecciones 28

3.2.1. Problemas a la hora de enlazar detecciones 28

3.2.2. Enlace con método de distancias mínimas 29

3.2.3. Enlace de partículas con filtros de Kalman 34

4. Evaluación del algoritmo propuesto 39

5. Análisis de trayectorias 45

5.1. Detección automática de cambios de dirección 46

5.2. Análisis de datos para categorización de movimientos 49

6. Conclusión y trabajos futuros 53

1. Introducción

A lo largo de la historia se han estudiado conjuntos de células para entender su comportamiento y en muchas ocasiones, utilizar esta información para alterarlo voluntariamente. Comprender la mecánica de movimientos de organismos ha sido la base para diferentes tipos de aplicaciones, siendo de principal interés en la biología [1], medicina [2], física [3], ingeniería [4], etc. [5]

A la hora de obtener muestras, se utilizan dos técnicas que facilitan la tarea de reconocimiento de las partículas (i) microscopía óptica y (ii) microscopía electrónica. En particular, la microscopía óptica de fluorescencia [6, 7]) y la microscopía electrónica de transmisión [8] (la microscopía electrónica de barrido también se utiliza pero para obtener imágenes en relieve). Dentro del presente trabajo se analizan muestras obtenidas con microscopía óptica de fluorescencia. En particular, centrado a una muestra de organismos choanoflagelados.

Para poder realizar un análisis cuantitativo de los organismos, primero es necesario conocer las trayectorias de cada una de las partículas. Anotar a mano las coordenadas en cada instante de cada una de las partículas es una tarea prácticamente imposible si se trabaja con densidades grandes y con tiempos prolongados. Existen decenas de programas para obtener las trayectorias de manera automática, como por ejemplo: ImageJ [9], Amira [10], herramientas de Icy [11] como son Motion Profiler, Spot Tracking, Track Manager entre otros [12]. La efectividad de estos programas no siempre es buena. Los tracks (trayectorias reconstruidas) más largos obtenidos muchas veces son muy cortos (entre 3 a 6 segundos) y las muestras de esta duración son reducidas en muchas ocasiones, llegando a obtener entre 100 y 200 tracks. Poder automatizar el tracking (proceso de conformación de trayectorias) de las partículas para poder manejar grandes volúmenes de datos, ayuda a tener mejor información estadística.

En la literatura se conocen diversos enfoques a la hora encarar el problema de tracking. Los métodos históricamente propuestos dividen al problema en dos etapas: (i) la detección de partículas, relacionado al aspecto espacial, en el cual se busca limpiar el ruido en cada una de las imágenes, y así identificar de manera eficiente y precisa las coordenadas de cada una de las partículas (esta etapa puede dividirse en dos, una de preprocesamiento, y una de detección), y (ii) el enlace de las partículas a lo largo de diferentes cuadros de video. Es decir, relacionado al aspecto temporal, en el cual se busca enlazar(o *linckear*, en inglés) de manera eficiente cada una detecciones obtenidas en la etapa anterior.

Existen diferentes formas de abordar cada una de las etapas. Ambas presentan dificultades propias. La etapa de enlace a su vez, tiene que ser capaz de poder manejar algunos errores y omisiones de la etapa de detección. Por lo cual una buena integración de ambas partes es esencial para obtener buenos resultados en el proceso de conformación de trayectorias.

No existe un método ideal para cada una de estas etapas. Los resultados históricos marcan que hay algoritmos que obtienen mejores resultados en algunos escenarios y no tanto en otros. Una de las formas de comparar diferentes algoritmos, es mediante la comparación de resultados sobre secuencias con datos simulados. Hace algunos años se realizó una competencia llamada *Particle Tracking Challenge*[1], donde varios equipos alrededor del mundo corrieron sus algoritmos y compararon sus resultados sobre diferentes secuencias simuladas. En este trabajo se utilizan los mismos conjuntos de secuencias para evaluar de manera cuantitativa la etapa de conformación de trayectorias. Los resultados obtenidos con el método presentado en este trabajo marcan que el algoritmo propuesto se encuentra en buen camino para convertirse en proyecto base para futuros trabajos. El esquema que se propone tiene en cuenta diversas dificultades en cada etapa, permitiendo extenderlo fácilmente para una mayor adaptación y efectividad a diferentes tipos de partículas.

Luego de analizar la efectividad del algoritmo propuesto para formar las trayectorias sobre un conjunto de secuencias, se aplicó el algoritmo sobre una población de organismos choanoflagelados. El presente trabajo surge como un proyecto interdisciplinario (entre física y computación) dentro de FaMAF. Se extiende se trabajo sobre el cual se encontraban trabajando hace un tiempo [13]. El objetivo de dicha investigación es caracterizar el movimiento de los organismos choanoflagelados [14, 15]. Este trabajo tiene como objetivo brindar herramientas que faciliten dicha tarea. Con la reconstrucción de automática de trayectorias, se pueden realizar análisis que no se encuentren sesgados por un subconjunto de tracks. El objetivo suyo es categorizar las partículas según su tipo de movimiento (browniano, lineal, curvo, y con algunos cambios de dirección). En este trabajo se obtienen algunos datos estadísticos sobre cada trayectoria, y se discuten posibles formas de clasificar (de manera parcial).

A la hora de analizar las trayectorias, el principal interés se centra en identificar los cambios de dirección a lo largo del las mismas. Se propone un algoritmo para identificar de manera automática las coordenadas de cambios de dirección y los ángulos.

[1] http://www.bioimageanalysis.org/track/

2. Marco Teórico

En esta sección se referencian las herramientas necesarias para la implementación del algoritmo propuesto de reconstrucción de trayectorias de partículas (tracking). Dado que el problema de tracking se divide en dos etapas, una de detección, y otra de enlace, se detallan las herramientas correspondientes a cada etapa en diferentes secciones. En la sección 2.1 se detallan herramientas correspondientes a la etapa de detección, y en la sección 2.2 las correspondientes a la etapa de enlace de partículas.

2.1. Detección de partículas

Debido a imperfecciones en las imágenes generadas por el proceso de adquisición, es necesario utilizar algoritmos que mejoren la calidad de las imágenes y que permitan extraer información de la locación de las partículas. En esta sección se discuten problemas y bloques de procesamiento que se utilizarán dentro del algoritmo de detección.

2.1.1. Reducción de ruido en imágenes digitales

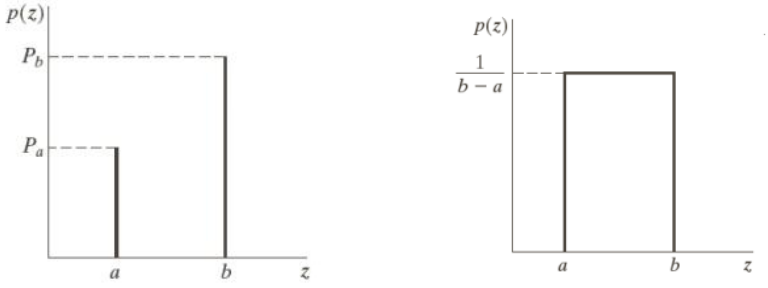
Una *imagen digital* es una función $f : \mathbb{R}^2 \mapsto \mathbb{R}$ cuyo dominio es una grilla de coordenadas discretas (x, y) (correspondientes a las coordenadas de los píxeles de la imagen) y cuya imagen son las intensidades de los píxeles sobre dichas coordenadas dentro del rango (0-1). Las imágenes en escala de grises están definidas con una única función de intensidad, mientras que las imágenes a color están formadas por triplas, donde cada elemento se obtiene con una función de intensidad que representa a los colores R (rojo), G (verde) y B (azul). Es decir, $f(x, y) = (f_R(x, y), f_G(x, y), f_B(x, y))$. Una imagen digital en escala de grises puede ser representada por una matriz numérica bidimensional donde los valores de cada elemento de la matriz se corresponden con los valores de $f(x, y)$.

El ruido en imágenes digitales es la variación aleatoria de la intensidad en cada canal de color. La intensidad g de un píxel situado en las coordenadas (x, y) de la matriz correspondiente a una imagen se modela como la adición del valor real de la imagen f y el ruido agregado r:

$$g(x, y) = f(x, y) + r(x, y) \tag{2.1}$$

Para mejorar la calidad de las imágenes hay que reducir el ruido de forma tal que el valor de intensidad de un píxel se corresponda con el real, es decir $g(x, y) = f(x, y)$. Existen diferentes técnicas dentro de la literatura para recuperar f, pero es necesario conocer que tipo de características tiene el ruido r.

Existen diferentes tipos de ruidos, los cuales están caracterizados por sus funciones de densidad de probabilidad. Entre ellos se encuentran el ruido Gaussiano, uniforme, sal y pimienta (o impulsivo), entre otros. Cada uno de estos se genera en situaciones particulares. El ruido impulsivo se produce en el proceso de digitalización. El uniforme es aquel en el cual distintos píxeles toman valores en un determinado intervalo de forma equiprobable.

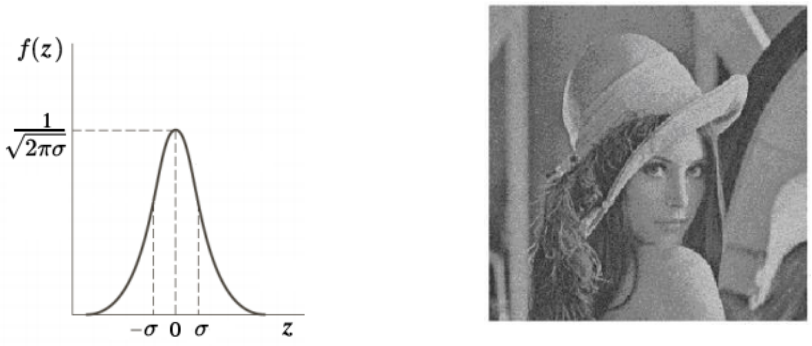


(a) ruido impulsivo

(b) ruido uniforme

Figura 2.1: Funciones de densidad del ruido

El ruido blanco Gaussiano se genera a causa de los circuitos electrónicos, falta de iluminación y variaciones de temperatura de sensores. En los videos analizados, este es el tipo de ruido que afecta en mayor medida las imágenes. Son señales aleatorias, caracterizadas porque sus valores en instantes de tiempo distintos no tienen relación alguna entre sí. Es decir, no existe correlación estadística entre sus valores. La función de densidad del ruido blanco Gaussiano se corresponde con una distribución normal con media cero.



(a) función de densidad

(b) imagen con ruido Gaussiano

Figura 2.2: ruido Gaussiano

Para poder transformar las imágenes originales en otras cuyo nivel de ruido sea menor, se utilizan filtros digitales. Existen diferentes tipos de filtros pero en este trabajo, se hace referencias a filtros paso bajo y filtros paso alto. Los filtro paso bajo tienen como objetivo reducir el ruido. Este tipo de filtros atenúa las frecuencias altas y mantiene sin variaciones las bajas. Como resultado de aplicar este tipo de filtros, se obtiene un suavizado de las imágenes originales donde el ruido se reduce pero los bordes se difuminan. Los filtros paso alto tienen como objetivo realzar los bordes. Este tipo de filtros atenúa las

frecuencias bajas y mantiene sin variaciones las altas. Se refuerzan los contrastes de las imágenes en los bordes.

2.1.1.1. Suavizado mediante promedios no locales

Un algoritmo utilizado para filtrar el ruido blanco Gaussiano es el suavizado de promedios no locales, o *non-local means denoising* [16], por su siglas en inglés. El principio de este método es bastante simple, reemplazar el color de un píxel con un promedio de los colores de los píxeles más parecidos. Si se pueden encontrar nueve píxeles parecidos para cada píxel de la imagen, se reduciría el ruido en un factor de tres (en caso de considerar los dieciséis píxeles más parecidos, el ruido se reduciría en un factor de cuatro). Sin embargo los píxeles más parecidos pueden no encontrarse cerca en absoluto. En particular, en este algoritmo, se dice que dos píxeles son parecidos comparando no solo los píxeles directamente, sino comparando una ventana alrededor de los mismos.

Supongamos que tenemos una imagen con un solo canal (escala de grises) a la cual queremos aplicar este filtro de suavizado, y nos encontramos suavizando un píxel en las coordenadas (x, y), al cual referenciaremos con p. El valor del nuevo píxel denotado por $\widehat{u}(p)$ se obtiene de la siguiente forma:

$$\widehat{u}(p) = \frac{1}{C(p)} \sum_{q \in B(p,r)} u(q) w(p,q) \tag{2.2}$$

$$C(p) = \sum_{q \in B(p,r)} w(p,q) \tag{2.3}$$

donde $u(q)$ es el valor de intensidad correspondiente al píxel q, $B(p, r)$ son los vecinos (ubicados en una ventana cuadrada) centrados en p que se encuentran dentro de una ventana de $(2r + 1) \times (2r + 1)$ píxeles. Esta restricción se pone debido a que comparar con todos los píxeles de la imagen puede tener un costo computacional muy alto. Los pesos $w(p, q)$ dependen de la distancia euclidiana al cuadrado respecto a las coordenadas simétricas en los parches de tamaño $(2f + 1) \times (2f + 1)$, centrados respectivamente en p y q. Es decir,

$$d^2(B(p,f), B(q,f)) = \frac{1}{(2f+1)^2} \sum_{j \in B(0,f)} (u(p+j) - u(q+j))^2 \tag{2.4}$$

Esto significa que el valor de cada píxel se actualiza con un promedio de los píxeles más parecidos. Para evitar el cálculo de los pesos $w(p, q)$, se utiliza un kernel exponencial:

$$w(p,q) = e^{-\frac{\max(d^2 - 2\sigma^2, 0)}{h^2}} \tag{2.5}$$

donde σ es la desviación estándar del ruido y h es un parámetro de filtrado que depende del valor de σ. La función de peso se establece con el fin de promediar los parches similares, incluso con ruido. Esto

es, los parches con distancias cuadráticas menores que $2\sigma^2$ se establecen en 1, mientras que en distancias cuadráticas grandes decrece rápidamente de acuerdo al kernel exponencial.

Los pesos $w(p,p)$ se establecen como el máximo de los pesos en las vecindades $B(p,r)$. De otra manera, $w(p,p)$ tomarían el valor 1, y debería ser necesario un valor alto de h para asegurar la reducción de ruido.

El tamaño de los parches (ventanas alrededor de los píxeles que se comparan a p para determinar similitud) y de las ventanas de observación (ventana alrededor del píxel p, sobre los que se realiza la comparación de parches) dependen del valor de σ. Cuando σ aumenta, se necesita un tamaño de parche mayor para realizar una comparación más robusta. A su vez se necesita incrementar el tamaño de la ventana de observación para aumentar la capacidad de remover ruido encontrado en píxeles más similares.

El parámetro de regulación de fuerza del filtro es $h = k\sigma$. El valor de k decrece, mientras que el tamaño del parche aumenta. Valores altos de h remueven mejor el ruido, pero también remueve detalles de la imagen. Valores bajos de h mantienen detalles de la imagen pero también dejan ruido. En el artículo [17] se encuentran más detalles acerca de los tamaños que deben tener los parches y las ventanas de observación dependiendo del σ seleccionado.

En la siguiente imagen se muestra un ejemplo de la aplicación de este algoritmo:

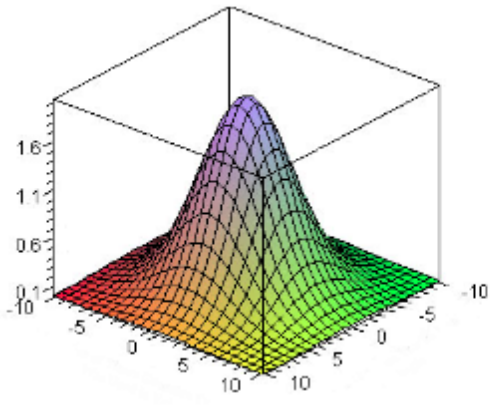
(a) imagen original (b) imagen con ruido (c) aplicación non-local means

Figura 2.3: Aplicación del algoritmo Non-Local Means

2.1.1.2. Suavizado Gaussiano

El filtro de mayor importancia que se utilizó para reducir el ruido dentro de este trabajo, es el filtro Gaussiano. Éste, es un filtro lineal. Se realiza una operación de convolución entre la imagen a ser filtrada y una máscara (o *kernel*). Tal cual como su nombre lo indica, este filtro reduce el ruido de tipo Gaussiano.

El kernel utilizado es una matriz de $k \times k$ donde los valores se corresponden con una distribución Gaussiana:



1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

Figura 2.4: Kernel Gaussiano

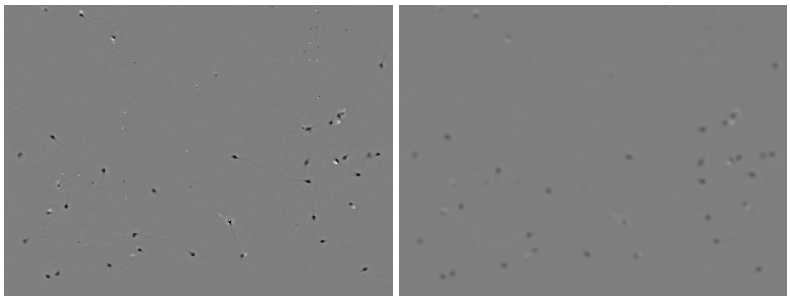
Como estos kernel siguen una distribución normal, los valores correspondientes a las celdas están directamente relacionados con la desviación estándar. Ésta, es la que rige entonces que tan fuerte es el suavizado. Mientras que el tamaño de la ventana del kernel, es la que establece cuantos píxeles alrededor del punto influyen en la obtención del nuevo valor del píxel.

Los valores de cada uno de los píxeles en la imagen filtrada se obtienen al promediar con distintos pesos (determinados por el kernel) los valores de los píxeles vecinos en una ventana cuadrada alrededor del mismo.

El tamaño del núcleo(kernel) se establece empíricamente como $k \times k$, donde:

$$k = 2 \lfloor 3\sigma \rfloor + 1 \tag{2.6}$$

Esto dice que el radio del kernel es de 3σ. Se utiliza este radio debido a que este valor comprende el $99{,}7\,\%$ del total de la integral de la Gaussiana. En otras palabras, los píxeles dentro de este área van a afectar de una manera importante el valor del cálculo, mientras que si se tienen en cuenta más píxeles circundantes, no aportarían demasiada información, y el costo computacional aumentaría. En la siguiente imagen se muestra un ejemplo de la aplicación de este filtro.



(a) imagen original (b) suavizado Gaussiano

Figura 2.5: Aplicación de suavizado Gaussiano

2.1.2. Redimensionado de las imágenes con interpolación bilineal

Dado a que las dimensiones de las imágenes pueden ser muy grandes, el costo computacional a la hora de aplicar los distintos filtros puede ser grande. Es por ello que de acuerdo al video con el que se esté trabajando, puede resultar conveniente reducir el tamaño de las imágenes.

Además de reducir los tiempos, el redimensionado puede ayudar a disminuir el ruido. Por ejemplo, si tenemos una imagen con partículas de diámetro mucho más grande que el del ruido, se puede reducir el tamaño de la imagen utilizando alguno de los métodos de interpolación, y así eliminar el ruido, o hacerlo de forma parcial.

Existen diferentes métodos para redimensionar imágenes. En el presente trabajo se utilizó un método de redimensionado con interpolación bilineal. Para obtener los valores de los píxeles de la nueva imagen, al hacer una reducción de tamaño, se posiciona la matriz de coordenadas de la nueva imagen sobre la posición real que ocuparían en la imagen original. El valor de cada píxel en la imagen reducida se obtiene utilizando los valores de los cuatro píxeles más cercanos dentro de la imagen original, ubicados en las diagonales del píxel a calcular. De estos cuatro píxeles, se obtiene el promedio ponderado y se calcula el valor interpolado.

La interpolación bilineal extiende a la interpolación lineal para poder interpolar funciones de dos variables. Básicamente se realiza una interpolación lineal sobre un eje, y luego sobre el otro. La elección del orden en que se realizan las interpolaciones no afecta los resultados. Si bien estos pasos son lineales, la interpolación bilineal no lo es.

Supongamos que tenemos la situación de la Figura 2.6, donde se desea obtener la intensidad del píxel P en la imagen escalada. La correspondencia de el centro dicho punto en la imagen original es un punto (x, y). Dentro de la imagen original se van a considerar los cuatro píxeles cuyos centros se encuentren rodeando al punto (x, y). En la figura, las posiciones de estos centros se encuentran en color rojo.

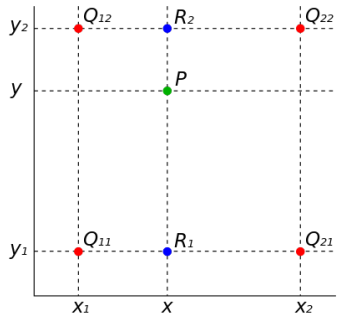


Figura 2.6

Se elige uno de los ejes y se realiza la interpolación lineal. Supongamos que elegimos el eje horizontal, entonces obtenemos los valores R_1 y R_2 de acuerdo a las siguientes fórmulas:

$$R_1 = \frac{x_2 - x}{x_2 - x_1} * Q_{11} + \frac{x - x_1}{x_2 - x_1} * Q_{21} \tag{2.7}$$

$$R_2 = \frac{x_2 - x}{x_2 - x_1} * Q_{12} + \frac{x - x_1}{x_2 - x_1} * Q_{22} \tag{2.8}$$

Luego se aplica interpolación lineal sobre el eje vertical para obtener el valor de P:

$$P = \frac{y_2 - y}{y_2 - y_1} * R_1 + \frac{y - y_1}{y_2 - y_1} * R_2 \tag{2.9}$$

El resultado que se obtiene con este tipo de interpolación es un suavizado. Por lo tanto, la intensidad de los píxeles correspondientes al ruido se atenúa con los del fondo.

2.1.3. Detección de blobs

Para realizar la detección de las partículas, se utilizará una técnica de detección de *blobs* (Binary Large OBjects). Éstos, son regiones uniformes que se encuentran en las imágenes. Para lograr esta detección se aplicará el operador Laplaciano sobre las imágenes para destacar las regiones uniformes correspondientes a las partículas, y luego se extraerán las coordenadas de máximos locales correspondientes a las posiciones de los centros de las mismas.

2.1.3.1. Diferenciación de partículas aplicando el operador Laplaciano

La técnica que se utilizará para destacar las partículas es la de *matching* de las estructuras de las partículas con la de un *kernel*. En particular, luego de aplicar los diferentes filtros en la etapa de preprocesamiento, las estructuras de las partículas en las imágenes resultantes se corresponden con un punto central con mayor intensidad y difusivo hacia los costados. Al aplicar una convolución con una máscara con estas características, se obtendrían respuestas máximas cuando la estructura se corresponda con la de la máscara. El operador que tiene la forma detallada es el Laplaciano. El mismo, hace uso de las derivadas de segundo orden. Las derivadas de una función digital se definen en términos de los píxeles adyacentes. Existen varias formas de definirlas, pero dichas definiciones deben cumplir ciertos criterios. En especial, para definir las derivadas de segundo orden, éstas deben cumplir con ser:

- 0 en regiones de gris constante
- distinto de 0 en los extremos de una rampa y escalones
- 0 a lo largo de rampas

Veamos una posible definición para las derivadas de segundo orden. Calculando la serie de Taylor de orden dos sobre uno de los ejes, tenemos:

$$f(x \pm \Delta x) = f(x) \pm \Delta x \;\; f'(x) + \frac{\Delta x^2}{2} f''(x) + O(\Delta x^3) \tag{2.10}$$

Dejando de lado los términos independientes, podemos sumar ambos resultados para cancelar las derivadas de primer orden:

$$f(x + \Delta) + f(x - \Delta) \approx 2f(x) + \Delta x^2 f''(x) \tag{2.11}$$

Dentro de este trabajo, Δx es 1 debido a que estamos trabajando con imágenes y por lo tanto asumimos valores discretos, donde la menor distancia que se puede considerar es de un píxel. Entonces la derivada resulta:

$$f''(x) \approx f(x-1) - 2f(x) + f(x+1) \tag{2.12}$$

Aplicar el operador derivada segunda sobre una imagen digital, se corresponde con realizar una convolución con la máscara:

1	-2	1

En la Figura 2.7 se muestran las respuestas obtenidas por este filtro sobre una imagen unidimensional. Estas derivadas también se pueden utilizar para detectar bordes, correspondiéndose a los cruces por cero.

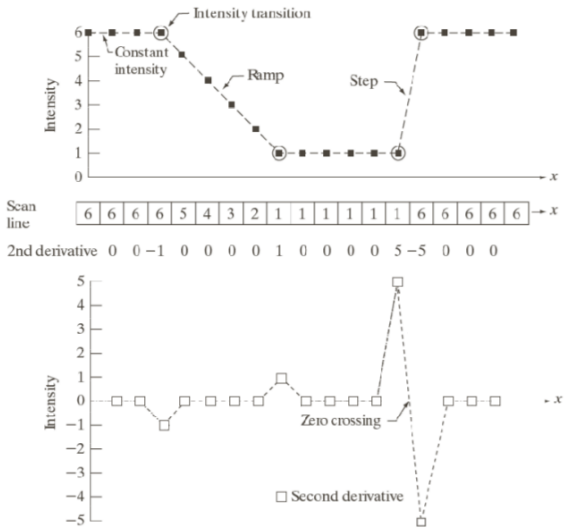


Figura 2.7: respuesta de convolución con máscara correspondiente a la segunda derivada sobre imagen unidimensional

De manera similar se puede obtener la derivada respecto al eje y y su correspondiente máscara de convolución:

$$f''(y) \approx f(y-1) - 2f(y) + f(y+1) \tag{2.13}$$

1
-2
1

Para obtener la segmentación de las partículas, se utilizará el operador Laplaciano. Éste, es un operador isotrópico, lo que significa que es invariante ante rotación, es decir, que aplicar el operador Laplaciano

sobre una imagen y sobre una rotación de la misma tendría el mismo resultado. El Laplaciano se define como la suma de las segundas derivadas en las direcciones de los ejes x e y:

$$\nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2} \tag{2.14}$$

Y cuyas definiciones de las derivadas de segundo orden son las de las ecuaciones 2.12 y 2.13. Por lo tanto:

$$\nabla^2 f(x,y)) = [f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1)] - 4f(x,y) \tag{2.15}$$

Esta operación puede expresarse en forma de convolución con el siguiente kernel:

0	1	0
1	-4	1
0	1	0

Si bien existen diferentes definiciones para las derivadas de segundo orden, en el presente trabajo se utilizan las mencionadas.

Los filtros que utilizan derivada de segundo orden son muy sensibles al ruido en las imágenes, es por ello que es de gran importancia utilizar previamente un filtro Gaussiano para eliminar ruido. Sin embargo, ambos filtros se pueden combinar, dando lugar a un solo operador que realice el suavizado y la detección de bordes, conocido como Laplaciano de Gaussianas (LoG) y cuyo núcleo puede calcularse componiendo ambos. Es decir convolucionar la imagen con el Laplaciano de una función Gaussiana bidimensional:

$$LoG(x,y) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right] e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.16}$$

El Laplaciano de Gaussianas tiene la forma de un sombrero mexicano. Un ejemplo del LoG se muestra a continuación:

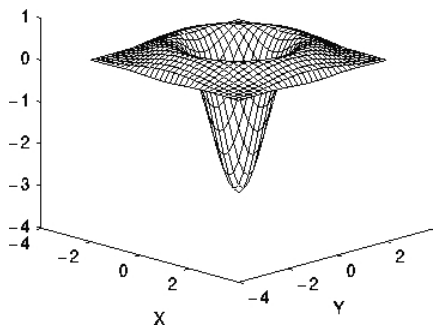


Figura 2.8: Laplaciano de Gaussianas

En el siguiente ejemplo, se muestra la respuesta que tiene este filtro sobre una región uniforme en una imagen unidimensional:

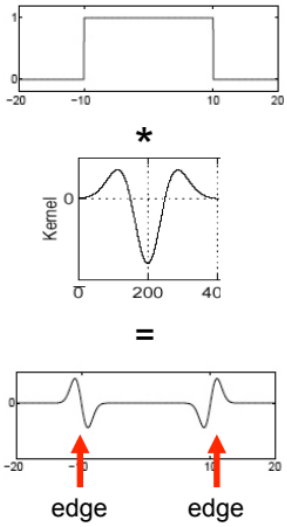


Figura 2.9: Respuesta LoG sobre un borde

Sin embargo, si la región uniforme se corresponde con el ancho del Laplaciano, la respuesta que obtendríamos sería la siguiente (en una imagen unidimensional):

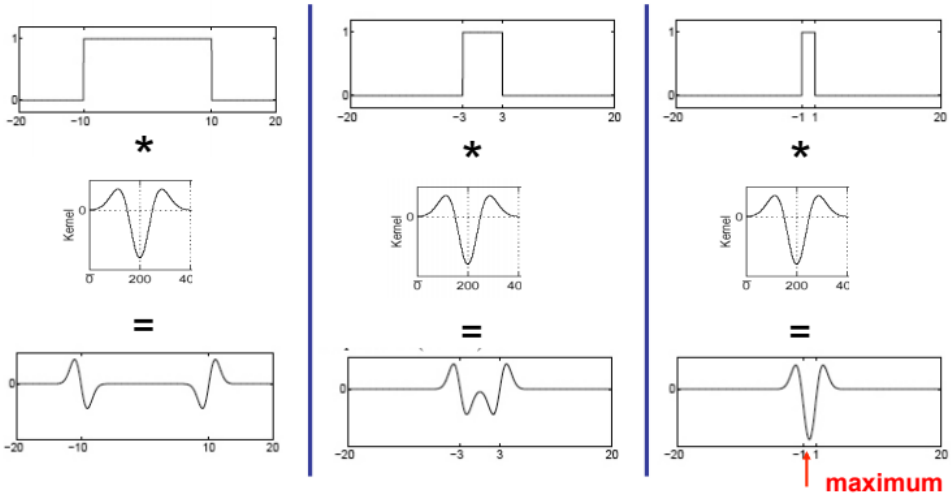


Figura 2.10: LoG sobre distintos anchos de borde

Como se puede observar, si se da la situación donde el ancho de la campana del Laplaciano se corresponde con el ancho del borde, se fusionan los picos de ambos bordes y arroja una respuesta mayor. Entonces, considerando la respuesta del Laplaciano ahora sobre imágenes bidimensionales, y como las partículas se encuentran en las imágenes como regiones uniformes, si se consigue que el filtro arroje un pico máximo en los centros de las partículas, obtendríamos las coordenadas de cada una de ellas en los picos donde la respuesta es máxima. En particular el LoG arroja valores máximos en los centros de regiones con un radio medio de $\sqrt{2}\sigma$ [18]. Si bien en la imagen el pico más grande se observa como un mínimo, se pueden invertir los colores de las imágenes para obtener picos máximos.

2.1.3.2. Identificación de partículas con máximos locales

Una vez resaltadas las regiones uniformes se deben determinar cuales de estas corresponden a partículas y cuales no. Dado que el tamaño de las partículas se relaciona con el valor de σ utilizado al aplicar el Laplaciano de Gaussianas, los centros de las partículas se van a corresponder con intensidades superiores al resto. Para extraer las coordenadas de cada una de estas, se realiza una búsqueda de máximos locales dentro de vecindades. Es decir, comparando cada píxel con los píxeles adyacentes.

El ruido de las imágenes también puede generar máximos locales. Sin embargo las respuestas al aplicar el operador Laplaciano van a ser menores a las arrojadas sobre las regiones correspondientes a partículas. Es por ello que se deben filtrar las respuestas que superen un umbral, que refleja la similitud del kernel Laplaciano con la porción de imagen observada.

Debido a que existen diferentes combinaciones de colores dentro de los videos (fondos oscuros con partículas más claras o fondos claros con partículas más oscuras), como el método utilizado obtiene las coordenadas de las partículas buscando máximos locales, en ciertas ocasiones hay que realizar un invertido de los colores (en escala de grises).

2.2. Filtros de Kalman para enlazar detecciones

Dentro de esta sección se detallarán los *filtros de Kalman*. Los mismos se utilizarán dentro de la etapa de enlace de partículas para tomar decisiones acerca del enlace entre detecciones, utilizando información del tiempo previo al momento observado.

Los filtros de Kalman son filtros recursivos Bayesianos para funciones lineales sujetas a ruido Gaussiano. Esencialmente, son un conjunto de ecuaciones matemáticas que proveen una solución recursiva y eficiente del método de cuadrados mínimos. Esta solución, permite calcular un estimador lineal, insesgado y óptimo del estado de un proceso a tiempo t, con base en la información ruidosa que se encuentra disponible en tiempo $t-1$, y actualizar con información adicional disponible en tiempo t, dichas estimaciones. Los filtros de Kalman constituyen el principal algoritmo para estimar sistemas dinámicos representados en la forma de espacio-estado [19], los cuales son esencialmente una notación conveniente para la estimación de modelos estocásticos, donde se asumen errores en la medición del sistema.

En particular, para este trabajo, esto significa que a medida que se procesen los cuadros de video en forma lineal, se obtendrá para cada una de las partículas, un modelo que se relacione directamente con el movimiento que ha realizado hasta el momento observado, es decir con cada track. Este modelo, se actualizará con las detecciones reales captadas por el módulo de detección hasta el tiempo observado, y será capaz de estimar una nueva posición para el siguiente cuadro. Esta estimación se utiliza para decidir si un track activo debe enlazarse con una nueva detección o no.

Los filtros de Kalman fueron presentados en el documento “Kalman (1960)” [20], y allí, se describe una solución de manera recursiva del problema de filtrado lineal de datos discretos. La derivación de estos filtros se realizó en un contexto de modelos de espacio-estado, cuya base está sustentada principalmente en la solución de cuadrados mínimos con pesos recursivos. Desde entonces, gracias a su bajo costo computacional, propiedades recursivas, y a su posición como el estimador óptimo para sistemas lineales

de variables continuas con distribución Gaussiana, los filtros de Kalman han sido objeto de numerosas investigaciones y aplicaciones. Entre ellas, se encuentran sistemas de recepción de posicionamiento global, suavizado de salidas de los trackpads en notebooks, navegación autónoma y asistida, rastreo de misiles, teléfonos, videojuegos, entre otros. Uno de los usos más famosos de aplicación de estos filtros es dentro de una computadora de navegación utilizada en la misión Apolo 11.

Con el método de cuadrados mínimos con peso recursivo [21], es posible estimar un estado a partir de una cantidad fija de mediciones de manera recursiva. Sin embargo, cuando la cantidad de mediciones no es fija, como en el caso de la estimación de posición de un móvil en vivo, es necesario extender el método de cuadrados mínimos con el filtro de Kalman.

En resumen, el filtro de Kalman es un procedimiento matemático que opera mediante predicción y corrección. El algoritmo pronostica un estado nuevo a partir de estimaciones previas, agregando un término de corrección que es proporcional al error de la predicción, de manera que éste es minimizado estadísticamente.

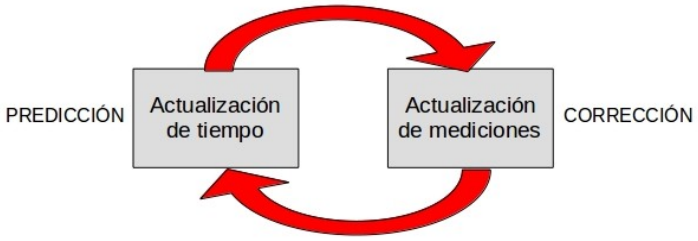


Figura 2.11: Ciclo de filtro de Kalman

2.2.1. Filtros de Kalman discretos

Un estado de un sistema es un vector que contiene toda la información relativa al sistema en cierto tiempo. Como pueden ser por ejemplo: posición, velocidad, dirección, etc. El filtro de Kalman funciona sobre sistemas que pueden ser descriptos con modelos lineales estocásticos, donde, tanto el error asociado al sistema, como la información adicional que se incorpora, tienen distribución normal con media cero y varianza determinada.

De manera formal, el filtro de Kalman sirve para estimar el estado $x \in \mathbb{R}^n$ de un proceso controlado en tiempo discreto, que se rige por la siguiente ecuación lineal:

$$x_k = F_k x_{k-1} + B u_{k-1} + w_{k-1} \tag{2.17}$$

donde:

- x_k: es el vector de estado que contiene los términos de interés del sistema en tiempo k.
- F_k: es la matriz $n \times n$ de transición de estados, la cual aplica los efectos del estado en tiempo $k-1$.

- B_k: es la matriz $n \times l$ de control de entrada. Es la que rige el control que tienen los parámetros de control de entrada u_k (o mediciones).
- u_k: es un vector l-dimensional que contiene todas las entradas externas (mediciones reales del sistema).
- w_k: es un vector que contiene los términos de ruido del proceso para cada parámetro en el vector de estado.

Las mediciones del sistema $z \in \mathbb{R}^m$ se realizan de acuerdo a la siguiente fórmula:

$$z_k = Hx_k + v_k \tag{2.18}$$

donde:

- z_k: es el vector m-dimensional de mediciones.
- H_k: es la matriz $m \times n$ de transformación, que mapea el vector de estado al dominio de las mediciones.
- v_k: es un vector que contiene los términos de ruido de las mediciones para cada observación en el vector de mediciones.

Las variables aleatorias w_k y v_k se asumen independientes una de otra, blancas, y con distribución de probabilidad normal multivariada:

$$p(w) \sim N(0, Q) \tag{2.19}$$

$$p(v) \sim N(0, R) \tag{2.20}$$

Si bien en la práctica, las matrices de covarianza del ruido del proceso Q y del ruido de las mediciones R, podrían cambiar en el tiempo, por simplicidad, en general se asumen constantes. De igual manera, las matrices F y H pueden variar en el tiempo, pero por simplicidad, en general se asumen constantes.

Al igual que en el caso de cuadrados mínimos recursivos, se pueden plantear las ecuaciones anteriores (adecuando subíndices) como:

$$x_{k+1} - Fx_k = 0 \qquad \text{con error } Bu_k + w_k \tag{2.21}$$

$$z_k = Hx_k \qquad \text{con error } v_k \tag{2.22}$$

y combinar estas ecuaciones en un solo sistema.

Definimos $\widehat{x}_k^- \in \mathbb{R}^n$ como el estado estimado a *priori* del estado x_k, dado el conocimiento que tenemos del proceso previo al paso k. Y definimos $\widehat{x}_k \in \mathbb{R}^n$ como el estado estimado a *posteriori* del estado x_k, dada la medición z_k y el proceso previo. Donde:

$$\widehat{x}_k = F_k \widehat{x}_{k-1} + B_k u_k \tag{2.23}$$

Podemos definir entonces los errores estimados a priori y a posteriori en comparación de estos estados estimados con el estado real:

$$\begin{aligned} e_k^- &= x_k - \widehat{x}_k^- \\ e_k &= x_k - \widehat{x}_k \end{aligned} \tag{2.24}$$

Entonces, la covarianza de los errores estimados a priori y a posteriori respectivamente son:

$$P_k^- = \mathrm{E}[e_k^- e_k^{-T}] \tag{2.25}$$

$$P_k = \mathrm{E}[e_k e_k^T] \tag{2.26}$$

La derivación del filtro de Kalman tiene como objetivo encontrar un conjunto de ecuaciones que permitan computar un estado estimado a posteriori $\widehat{x}_k$ como una combinación lineal de un estado estimado a priori $\widehat{x}_k^-$, y una diferencia con peso entre el conjunto de mediciones z_k y la predicción de las mediciones $H\widehat{x}_k^-$.Esto es:

$$\widehat{x}_k = \widehat{x}_k^- + K_k(z_k - H\widehat{x}_k^-) \tag{2.27}$$

Donde la diferencia $(z_k - H\widehat{x}_k^-)$ se llama *innovación*, y la matriz K_k de tamaño $n \times m$ se llama *ganancia de Kalman*, y es la que minimiza la ecuación de la covarianza del error a posteriori 2.26. Luego de realizar la derivación sobre las fórmulas anteriores, se obtiene la ganancia de Kalman que minimiza la ecuación 2.26:

$$K_k = \frac{\widehat{P}_k^- H^T}{H\widehat{P}_k^- H^T + R} \tag{2.28}$$

Más detalles acerca de la derivación del filtro de Kalman, puede encontrarse en diversos artículos [22, 23].

2.2.2. Algoritmo del filtro de Kalman

El filtro de Kalman se divide en dos etapas. La primera, de *detección*, y posteriormente, la de *actualización*. Las ecuaciones correspondientes a la predicción son las encargadas de estimar estados en

tiempo t, tomando como referencia al estado en tiempo $t-1$, y de realizar la actualización intermedia de la matriz de covarianza de estado.

$$\widehat{x}_k^- = F\widehat{x}_{k-1} + Bu_{k-1} \qquad (P1)$$
$$P_k^- = FP_{k-1}F^T + Q \qquad (P2)$$

Mientras que las ecuaciones de actualización son las encargadas de corregir la estimación anterior mediante la incorporación de nueva información. Para ello, primero se calcula la ganancia de Kalman. Luego se incorporan las mediciones para obtener un estado estimado, y finalmente se actualiza la matriz de covarianza del error.

$$K_k = \frac{P_k^- H^T}{HP_k^- H^T + R} \qquad (A1)$$
$$\widehat{x}_k = \widehat{x}_k^- + K_k(z_k - H\widehat{x}_k^-) \qquad (A2)$$
$$P_k = (I - K_k H)P_k^- \qquad (A3)$$

3. Seguimiento de partículas

Para poder llevar a cabo la tarea de tracking de partículas, es necesario poder primero discernir que cosas son partículas y que cosas son elementos indeseados. Poder identificar cuadro por cuadro, las coordenadas precisas de cada partícula no es una tarea sencilla. Para facilitar esta tarea, es de vital importancia tener imágenes limpias, sin ruido. Es decir, imágenes donde solamente manchas uniformes correspondientes a partículas resalten sobre un fondo de color uniforme.

La limpieza de las imágenes se debe realizar acorde al tipo de imágenes a analizar. Existen diferentes tipos de ruido por el cuál se pueden ver afectadas las mismas. Ruido producido por la manipulación, y ruido generado por el ambiente de la muestra. Debido a los métodos utilizados en la obtención de las imágenes, el ruido principal que se genera dentro de la manipulación, es el ruido de tipo Gaussiano.

Una vez detectadas todas las partículas y sus correspondientes coordenadas (aspecto espacial), se realiza la tarea de enlace (*linking*, en inglés). En esta etapa se tiene en cuenta el aspecto temporal. Tiene como objetivo poder enlazar las detecciones de la etapa anterior, vinculando las que corresponden a una misma partícula física. Para realizar esta tarea se aplica un conjunto de criterios que se ven afectados por el tipo de movimiento que existe entre las partículas.

La mayoría de los algoritmos que existen en la actualidad [24] para llevar a cabo la tarea de tracking, aplican la etapa de detección y enlace una única vez para obtener los tracks. Sin embargo, algunos algoritmos se basan en aplicaciones iterativas sobre estas etapas. Dentro del presente, se utiliza una sola vez cada etapa. El esquema del algoritmo entonces es el siguiente:

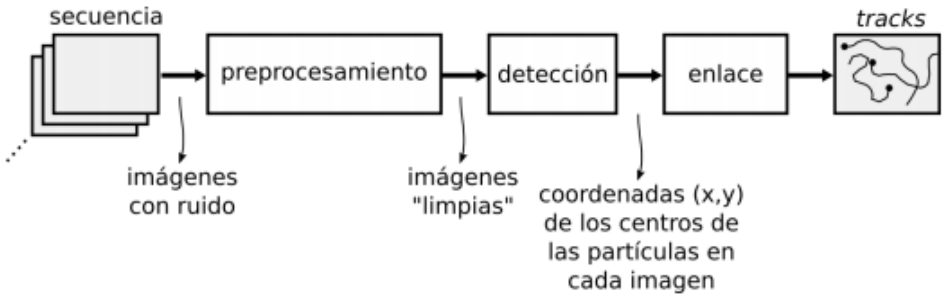


Figura 3.1: Esquema del algoritmo

Primero se realiza un preprocesamiento de las imágenes con el fin de obtener imágenes limpias que faciliten el proceso de detección de las partículas. Luego se realiza el proceso de detección de las coordenadas de las partículas utilizando información de máximos locales obtenidos luego de aplicar

el operador Laplaciano sobre las imágenes. Estas dos primeras etapas pueden coalecer si se utiliza el Laplaciano de Gaussianas. Finalmente se enlazan las detecciones a lo largo las secuencias de imágenes de forma que las trayectorias obtenidas se correspondan con las reales. Para esta etapa se plantean dos algoritmos. Uno que tiene en cuenta solo las distancias físicas y temporales de las partículas, y otro que incorpora comportamiento previo al momento analizado, utilizando filtros de Kalman. Una vez obtenidos los tracks, se puede realizar un análisis sobre los mismos para poder obtener información de interés.

El algoritmo de tracking funciona sobre videos con diferentes características mediante el ajuste de algunas variables que se detallarán a lo largo del capítulo.

3.1. Detección de las partículas

La etapa de detección tiene como objetivo identificar las coordenadas espaciales de cada partícula en cada uno de los cuadros de video. Debido a que las partículas se manifiestan como regiones uniformes o *blobs*, es de vital importancia poder discernir entre las partículas y los elementos que no lo son. Entre estos últimos se puede encontrar ruido producido por los métodos de obtención del video, pero también ruido relacionado al ambiente sobre el cuál se tomaron las imágenes. Por ejemplo, puede haber manchones producidos por suciedad, o marcas producidas por el fluido sobre el cual se tomó la muestra.

Existen diferentes técnicas para poder "limpiar" cada una de las imágenes y también existen diferentes técnicas para poder detectar blobs. A lo largo de los últimos años se realizaron varios estudios para poder encontrar las mejores técnicas para realizar estas tareas. *Cheezum et al.* [25] comparó cuatro métodos para poder detectar partículas fluorescentes de manera individual. Otros estudios más finos fueron presentados por *Carter et al.* [26] y *Smal et al.* [27]. Los autores de estos artículos concluyen que todos los métodos comparados se comportan de buena manera para la relaciones de señal-ruido (SNR por sus siglas en inglés) mayores a 5. Esta relación se define como la proporción existente entre la potencia de la señal que se transmite y la potencia del ruido que la corrompe. Se calcula de la siguiente manera:

$$SNR = \frac{I_0 - I_b}{\sqrt{I_0}} \tag{3.1}$$

donde I_0 es la intensidad máxima de las partículas, e I_b es la intensidad media del fondo.

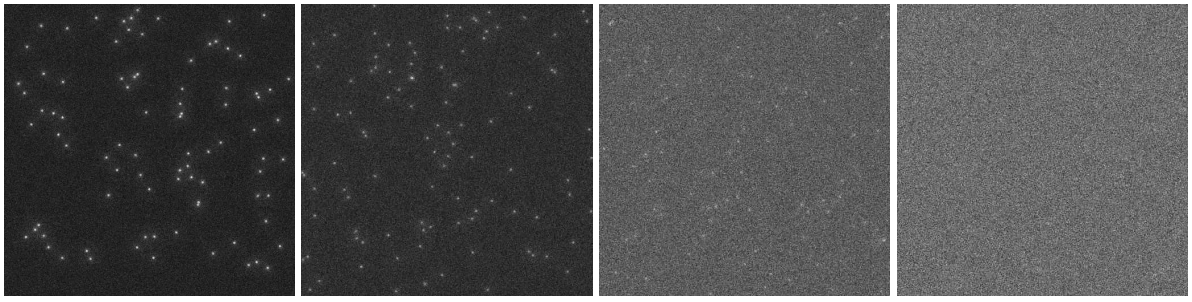
Dado que dentro de este trabajo no se consideran de interés experimentos de laboratorio simulados, los escenarios con $SNR < 4$ no se utilizan para obtener métricas en las pruebas de eficiencia. Cabe destacar que videos con $SNR = 2$, las partículas son muy difíciles de detectar para un observador humano, siendo prácticamente imposible para $SNR = 1$.

Dentro de este trabajo se aplicaron distintas combinaciones de filtros para poder limpiar las imágenes y posteriormente se detectan las partículas utilizando la técnica de detección de máximos locales luego de aplicar un operador Laplaciano sobre las imágenes. Este método históricamente ha arrojado buenos resultados dentro de la detección de partículas. A lo largo de esta sección se discutirán detalles que se deben tener en cuenta para mejorar la detección de partículas, y se describe el método propuesto para la detección misma.

3.1.1. Problemas a la hora de realizar detecciones

Para poder determinar las trayectorias de partículas en videos de microscopía óptica, la etapa más importante es la etapa de detección. Ésta, se realiza de manera individual cuadro por cuadro dentro del video. Es necesario poder determinar de manera efectiva y precisa las coordenadas de cada partícula que se mueve dentro de un fluido, en los ejes verticales y horizontales de las imágenes. En algunas ocasiones también es de interés obtener las coordenadas de profundidad dentro del fluido. El hecho de que las partículas se muevan dentro de un fluido y puedan acercarse o alejarse al lente del plano focal de la cámara, hace que su tamaño varíe. Siendo de gran importancia el proceso de la toma de muestras.

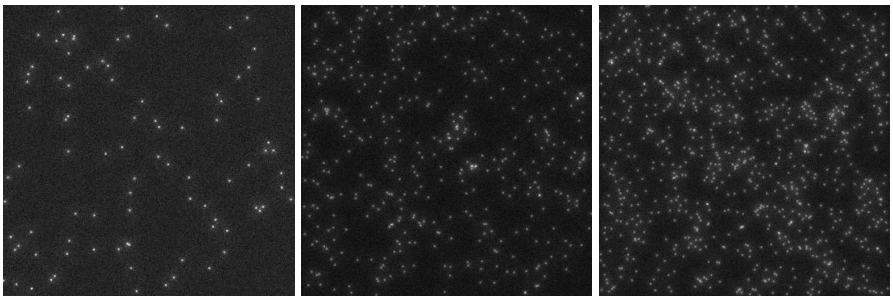
La microscopía óptica consiste en pasar luz visible, refractada sobre el objeto de estudio, a través de lentes ópticos simples o múltiples. La obtención de imágenes no es un proceso perfecto y los resultados se ven afectados por los circuitos electrónicos, la iluminación, temperaturas, entre otros. Esto hace que las imágenes posean ruido y se dificulte la detección de las partículas. Dependiendo de que tan ruidosa sea una imagen, se clasifica en distintos niveles de ruido. Mientras más bajo es el nivel de SNR, la imagen es más ruidosa. En la siguiente imagen se muestra una misma imagen con diferentes niveles de SNR y se aprecia como afectan estos a la intensidad de los píxeles de la imagen:



(a) SNR = 7 (b) SNR = 4 (c) SNR = 2 (d) SNR = 1

Figura 3.2: Niveles de SNR. Se aumentaron los contrastes de las imágenes para una mejor visualización debido a la poca variación de intensidades entre los píxeles

Otro de los factores que interviene directamente en la tarea de detección es la densidad de población. Obtener detecciones de partículas cuando la densidad es pequeña es significativamente más sencillo. Cuando la densidad es mayor, existen mayor cantidad de situaciones de cruces de partículas y distancias relativas más pequeñas. La tarea tanto de detección, como de enlace en estas situaciones es mucho más compleja. Dentro de la detección, dos partículas que se encuentran cruzando una por encima de la otra dentro del fluido se fusionan visualmente de manera parcial o total. En la imagen 3.3 se muestran imágenes con distintos niveles de densidad:



(a) bajo (b) medio (c) alto

Figura 3.3: Niveles de densidad de partículas. Se aumentaron los contrastes de las imágenes para una mejor visualización debido a la poca variación de intensidades entre los píxeles

Por último también hay que tener en cuenta el tipo de partícula que se está analizando, y el ambiente en el que se encuentran. Debido a que las partículas se mueven dentro de fluido, el movimiento a través del mismo, puede generar halos alrededor de la partícula. Este tipo de información puede ser realmente útil para poder considerar la profundidad a la cual se encuentra inmersa la partícula dentro del fluido. Tener en cuesta esta información, mejora la obtención de estadísticas como por ejemplo la velocidad que posee la partícula en un momento dado.

Debido a la complejidad de este tipo de análisis, en el presente trabajo no se tienen en cuenta imágenes con halos muy acentuados. Una de las formas de evitar esta particularidad a la hora de obtener las imágenes, es reducir la profundidad del fluido. Esto se logra presionando los vidrios sobre el cual se coloca la muestra.

3.1.2. Preprocesamiento de las imágenes

La imagen de cada cuadro de video, está formada por la combinación de tres canales (RGB). Debido a que el método de detección utilizado tiene en cuenta regiones uniformes o blobs, se realiza una transformación de las imágenes a escala de grises. Esto también hace que el cálculo posterior sea menor. Dentro del preprocesamiento de imágenes se aplicaron de manera secuencial distintos algoritmos para eliminar el ruido producido por la obtención de las muestras, y el ruido que existe dentro de las muestras. El algoritmo propuesto para limpiar las imágenes es el siguiente:

1. **Sustracción de fondo**: se eliminan elementos no deseados que se encuentran de manera estática dentro del ambiente de la muestra. En la sección 3.1.2.1 se detallará esta subetapa.

2. **Redimensionado de las imágenes**: tiene como objetivo principal reducir tiempos de cómputo. A su vez al cambiar la escala de las imágenes, se cambia el tamaño efectivo de los filtros de procesamiento. Si bien no es la tarea principal, el método aplicado para redimensionar las imágenes (interpolación bilineal) ayuda a eliminar parte del ruido. Más detalles del método de redimensionamiento aplicado se describen en la sección 2.1.2.

3. **Suavizado de imágenes**: se busca eliminar el ruido blanco Gaussiano que se encuentra dentro de las imágenes. Se consideran dos filtros de suavizado. El filtro de suavizado con promedios no

locales, y el filtro de suavizado Gaussiano. El primero se puede elegir si se aplica o no, mientras que el suavizado Gaussiano se realiza siempre. Esto es debido a que el filtro de suavizado Gaussiano en general elimina en buena medida la totalidad del ruido blanco Gaussiano, y aplicar otro método de suavizado refiere a mayor cómputo. Más detalles acerca de estos filtros se describen en las secciones 2.1.1.1 y 2.1.1.2.

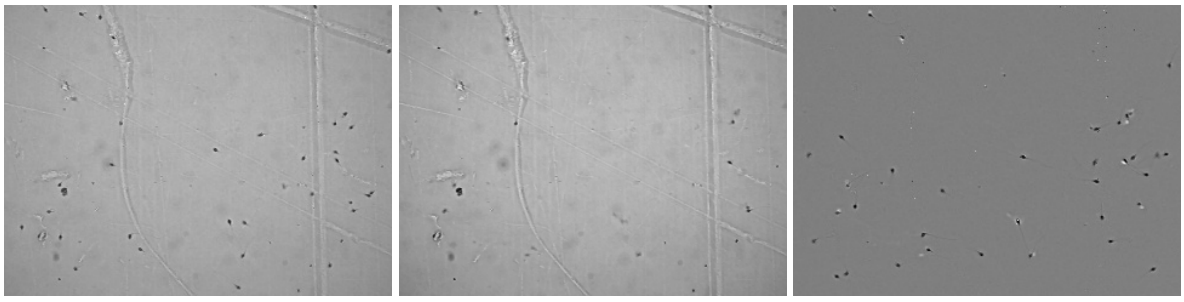
Las etapas de sustracción de fondo y de redimensionado pueden elegir aplicarse o no dependiendo del video analizado.

3.1.2.1. Sustracción de fondo

Dentro de las imágenes obtenidas, pueden aparecer elementos extraños o manchas que no corresponden con las partículas. Por ejemplo, puede haber defectos generados por el microscopio, por la cámara, o generados por la refracción de luz. También puede haber marcas generadas por el fluido o suciedad dentro de las muestras. Estos detalles se manifiestan generalmente de manera estática en partes del video o en su totalidad, y pueden causar detecciones falsas o inferir en la detección de partículas reales. Es por ello que dentro del módulo de detección se realiza un filtrado previo, donde se resta la imagen promedio del video. Hay que tener en cuenta que al realizar esta resta, también pueden desaparecer partículas que se encuentran pegadas al vidrio que contiene la muestra, o que se mueven dentro de un área muy pequeña.

Si bien la sustracción de fondo ayuda a limpiar las imágenes de parte del ruido no deseado, puede haber inconvenientes en imágenes que poseen un alto nivel de ruido, o cuya densidad de población es grande. Esto se debe a que la imagen promedio en algunas regiones puede tomar valores cercanos a los que correspondería con el de una partícula. Para estos casos, y también por costo computacional, una mejor opción es elegir aleatoriamente un conjunto de N imágenes sobre las cuales se calculará el promedio. La elección de este parámetro suele definirse en torno al $10\,\%$ del total de las imágenes del video.

Como el objetivo de este trabajo es obtener datos estadísticos relacionados al movimiento de las partículas, no son de interés los ejemplares que se encuentran muertos o pegados al vidrio. Aplicando este método, se logran eliminar estas partículas o difuminarlas lo suficiente para luego no ser tenidas en cuenta por el detector. En la Figura 3.4 se muestra un cuadro original de video, la imagen promedio, y la imagen luego de realizar la sustracción de fondo:



(a) imagen original (b) imagen promedio (c) sustracción de fondo

Figura 3.4: Proceso de sustracción de fondo

3.1.3. Detección de blobs con filtros Laplacianos

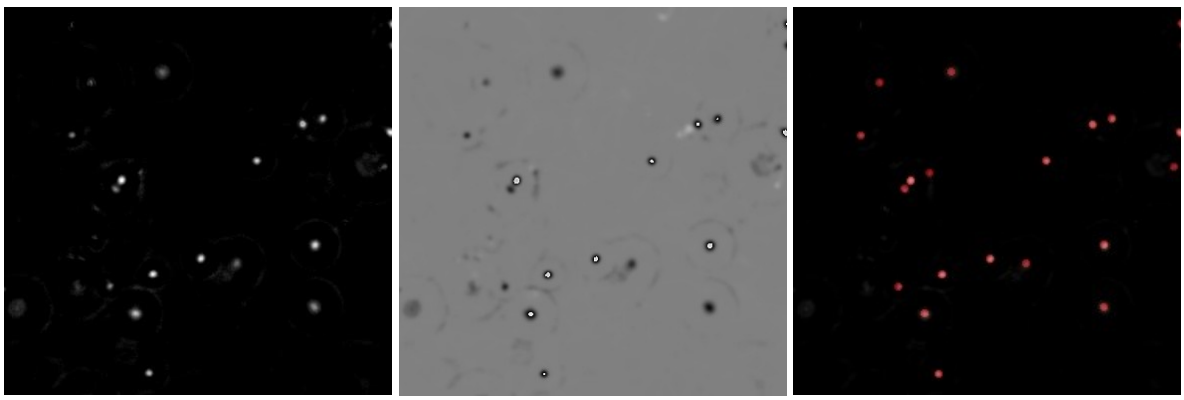
Como Laplaciano de Gaussianas arroja valores máximos en los centros de regiones con un radio medio de $\sqrt{2}\sigma$, si se conoce el tamaño de las partículas de antemano se pueden obtener respuestas máximas. A la hora de tomar las muestras, el microscopio permite realizar mediciones, y con las mismas se puede obtener las dimensiones reales de las partículas. De la misma forma, obtienen las relaciones que existen entre los píxeles de las imágenes y las dimensiones reales.

Como no es cierto que el tamaño de las partículas es fijo (en las imágenes obtenidas), debido a que éstas se encuentran en movimiento sobre el fluido, y que dentro de la población pueden existir diferentes tipos de partículas, se debería encontrar un valor de σ que arroje respuestas máximas que superen el umbral propuesto solo para las partículas. Idealmente se debería realizar un análisis multidimensional, con diferentes escalas sobre σ para obtener respuestas máximas acordes a las dimensiones estimadas de cada partícula. Sin embargo, esto aumentaría el costo computacional y la complejidad del sistema.

El método de máximos locales es uno de los métodos que históricamente ha arrojado buenos resultados si el preprocesamiento se hace de manera eficiente. Dentro de la competencia de Particle Challenge (2012) [28] los algoritmos mejor rankeados fueron los que utilizaban la detección con máximos locales con diferentes filtrados previos, siendo el que utilizaba Laplaciano de Gaussianas el que mejor resultados obtuvo.

La detección de máximos locales se realizó en una vecindad con los píxeles adyacentes. Es decir con aquellos que rodean al píxel observado.

En las Figuras 3.5- 3.8 se muestran algunos de los resultados obtenidos luego de aplicar este método en videos con distintos tipos de partículas.

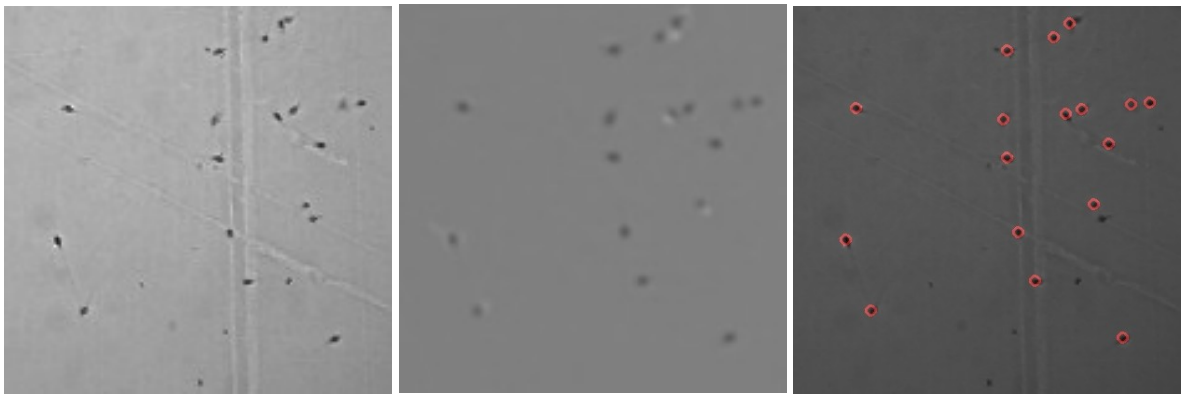


(a) cuadro original

(b) cuadro filtrado (con colores invertidos) previo a aplicación de máximos locales

(c) detecciones obtenidas

Figura 3.5: Video: choanoflagelados (fragmento de una imagen para apreciar los resultados)

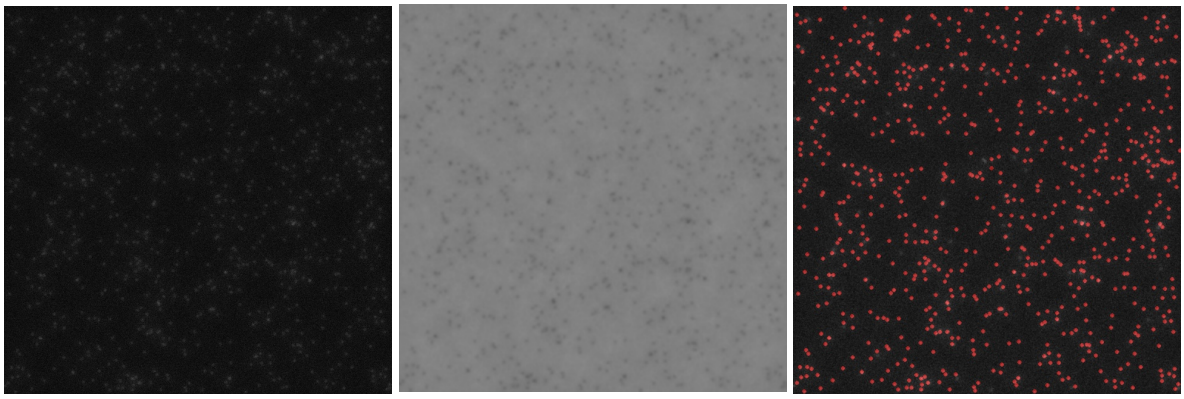


(a) cuadro original

(b) cuadro filtrado previo a aplicación de máximos locales

(c) detecciones obtenidas

Figura 3.6: Video: espermatozoides (fragmento de una imagen para apreciar los resultados)

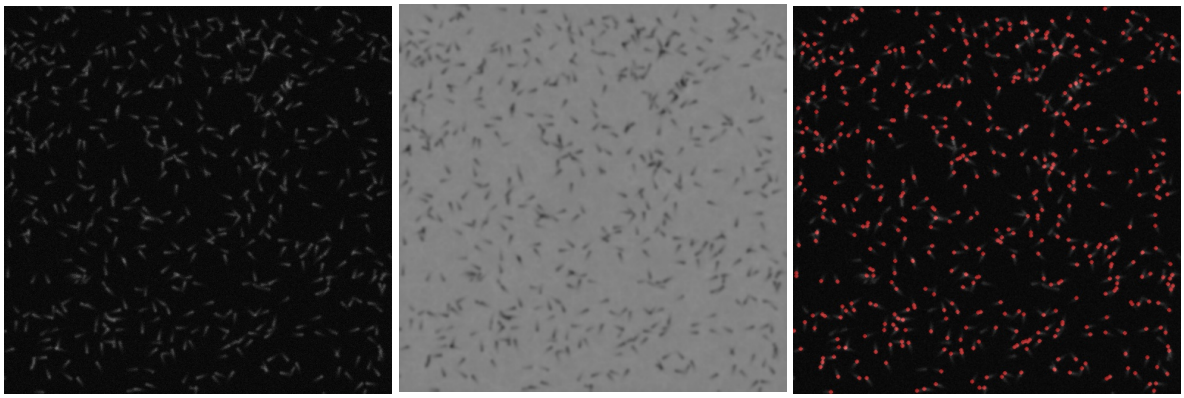


(a) cuadro original

(b) cuadro filtrado previo a aplicación de máximos locales

(c) detecciones obtenidas

Figura 3.7: Video: VESICLES, densidad: alta, SNR: 4



(a) cuadro original

(b) cuadro filtrado previo a aplicación de máximos locales

(c) detecciones obtenidas

Figura 3.8: Video: MICROTUBULE, densidad: alta, SNR: 7

3.2. Enlace de las detecciones

La etapa de enlace es la encargada de trazar las trayectorias que sigue cada una de las partículas. Dado el conjunto de detecciones en cada uno de los cuadros, se encarga de buscar correspondencias de partículas en los diferentes cuadros. Por lo cual, la etapa de detección es de vital importancia ya que influye directamente en la complejidad del problema de enlace.

Existen distintos métodos históricamente conocidos que intentan llevar a cabo esta tarea de forma eficiente. Dentro de la competencia del "Particle Challenge", el equipo que utilizó filtros de Kalman es uno de los que terminó mejor rankeado.

Para poder obtener enlaces buenos (enlaces que se corresponden con la situación real), es necesario tener en cuenta distintas situaciones que pueden ocurrir en este tipo de videos, y también es necesario tener en cuenta caracterísitcas del objeto de estudio. Entre ellos, se encuentran cruces, velocidades, direcciones, tipo de movimiento, entre otros. En el presente trabajo se plantea una primera solución que tiene en cuenta las distancias espaciales que existen entre las distintas detecciones, y luego se incorporan filtros de Kalman sobre esta idea.

Si bien el primer método aplicado arrojó muy buenos resultados, la aplicación posterior con filtros de Kalman ayudó a mejorar la performance.

3.2.1. Problemas a la hora de enlazar detecciones

Una vez obtenidas las coordenadas de las partículas en los diferentes cuadros de video se deben enlazar o unir las mismas con las diferentes correspondencias en los diferentes cuadros. Esta tarea no es sencilla y está condicionada a diversos factores, como son: tipo de movimiento, densidad de partículas y relación señal-ruido de las imágenes. Dentro de la etapa de detección, el tipo de movimiento no es un factor influyente, como si lo son la densidad de partículas y la cantidad de ruido que poseen las imágenes.

La etapa de enlace, sin embargo, se ve afectada por las tres categorías. La relación señal-ruido afecta en relación a cuanto afecta en la etapa de detección. Si esto hace que las partículas no se detecten, es un problema para luego poder enlazarlas. Por ejemplo, si una partícula conocida es detectada en los cuadros C_1 y C_3 pero no en el cuadro C_2, el algoritmo debe ser capaz de enlazar esa partícula entre los cuadros C_1 y C_3.

Debido a que las partículas se mueven dentro de un fluido a lo largo de tres ejes de simetría y con la cámara solo se registran dos, la partícula simplemente puede desaparecer de la vista, o desaparecer temporalmente y luego aparecer con la misma trayectoria o en un lugar inesperado con otra trayectoria. El presente trabajo se enfoca a estudios sobre muestras que solamente contienen imágenes de video obtenidas desde un solo ángulo. Sin embargo, existen otros métodos para obtener muestras con información adicional y que ayudan a mejorar la robustez en la tarea de detección y enlace.

La densidad de partículas, además de afectar en la detección también afecta a la etapa de enlace. Al igual que ocurre con la relación señal-ruido, algunas detecciones pueden perderse (debido al cruce de partículas). Para poder enlazar partículas en momentos posteriores a cruces hay que tener conocimiento de como era el movimiento previo. De lo contrario una partícula que estaba en reposo y sobre la cual

cruza otra que viene con movimiento y trayectoria constante podrían confundirse. Pero, si la densidad en el número de partículas es muy grande, poder recolectar información previa es difícil.

Por último, el tipo de movimiento hace que recolectar información buena sobre las trayectorias tenga distinta complejidad. Si las partículas se mueven de forma aleatoria, en momentos de cruces es prácticamente imposible poder determinar en el cuadro posterior, cuales deberían ser las correspondencias.

A causa de los diferentes filtros previos que se aplican en la etapa de detección, las coordenadas provistas pueden no ser las más precisas. Por lo cual una partícula que en realidad tiene una trayectoria recta, no sigue la misma trayectoria si consideramos las coordenadas del detector. Pueden existir pequeñas fluctuaciones.

Poder incorporar información adicional además de las coordenadas obtenidas por el detector, como lo son velocidades y/o aceleraciones, registros de actividades en cada punto del track, tiempos entre cuadros de video, ayudan a mejorar en enlace.

3.2.2. Enlace con método de distancias mínimas

El primer algoritmo propuesto está compuesto por dos etapas. La primera donde se busca obtener enlaces en cuadros de video contiguos. En esta etapa se busca obtener tracks que no pasaron por situaciones de pérdida de información en las detecciones. Es decir que las detecciones correspondientes a cada track se encuentran siempre en cuadros contiguos de video. En la segunda etapa se busca unir los tracks ya formados, pudiendo saltar cuadros intermedios.

A lo largo de esta sección y la siguiente se hará referencia al concepto de *tracks activos* a tiempo t, definidos como los tracks cuyos últimos puntos corresponden a detecciones de tiempo $t - \Delta t$. Donde Δt es la cantidad de cuadros de video que puede "saltar" una partícula sin estar ligada a una detección. También se hará mención a los siguientes conceptos de distancia:

- **Distancia entre partículas de un mismo cuadro de video**: distancia espacial entre ambos puntos (Figura 3.9).

- **Distancia entre un track y un punto**: distancia espacial entre el último punto del track y el punto correspondiente al superponer ambas imágenes (Figura 3.10).

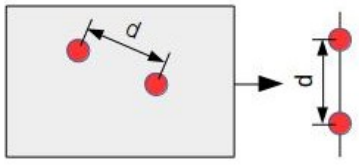


Figura 3.9: distancia de un track a un punto, y representación

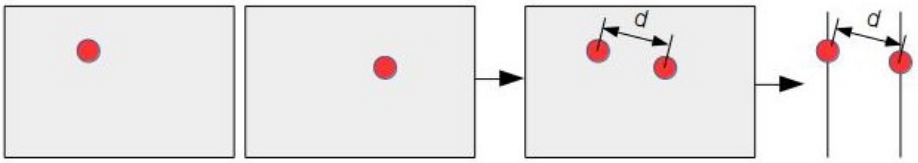


Figura 3.10: distancia entre partículas en un mismo cuadro, y representación

En la primera etapa del algoritmo propuesto, se realiza un forward pass, recorriendo los cuadros de video en orden. En cada uno de los cuadros se tienen cuenta las partículas detectadas dentro del mismo cuadro, y los tracks que se encuentran activos en el cuadro anterior. Si al estar parado sobre un cuadro, éste no posee tracks activos al momento, entonces cada una de las detecciones se considera como el inicio de un nuevo track. Caso contrario, se debe elegir para cada una de las detecciones, si corresponden con la continuación de alguno de los tracks activos o no. Para ello se analiza el escenario respecto a dos criterios:

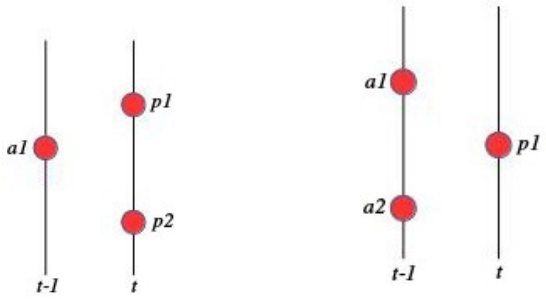
1. **Relación de distancia con otras detecciones**: tiene en cuenta tanto las distancias a tracks activos, como también las detecciones dentro del mismo cuadro.

2. **Relación de distancia de salto**: tiene en cuenta las distancias promedio entre los distintos puntos de cada track.

En la mayoría de los casos si un supuesto enlace pasa ambos criterios debería considerarse como bueno. Hay un caso particular donde tener el visto bueno por ambos criterios no es suficiente. A continuación se detallan los dos criterios de selección y posteriormente se describe el problema específico y la solución propuesta para esta situación en particular.

Criterio de distancias con otras detecciones

El objetivo de este criterio es asegurarse que ante un posible enlace, no existan conflictos con otras detecciones. Por un lado, asegura que una partícula que se presupone puede ser la continuación de un track, no entre en conflicto con otras partículas del mismo cuadro. Si en un cuadro C_t, correspondiente al tiempo t, hay dos partículas p_1 y p_2 que se encuentran muy cercanas, y en el cuadro C_{t-1} hay un track activo que se encuentra a una distancia pequeña y similar tanto a p_1, como a p_2 (Figura 3.11 (a)), enlazar el track al que posea la distancia mínima puede ser una mala elección debido a que no se conoce las trayectorias que traían las partículas, ni sus velocidades. Por otro lado, este criterio asegura, que no existan dos tracks que intenten enlazarse con la misma partícula. En el caso que existan dos tracks cuya distancia a la presunta partícula a enlazarse sea pequeña y similar (Figura 3.11(b)), ambos tracks van a optar por "continuar" sus trayectorias sobre este punto. Debido a que el algoritmo utiliza solamente relaciones de distancias y no información de comportamiento de los tracks, si ambos tracks se unieran a un mismo punto, estos se pegarían y a partir de entonces siempre elegirían los mismos puntos de enlace como continuación.



(a) (b)

Figura 3.11: Situaciones problemáticas

Dado que en este criterio solo se utiliza información relacionada a distancias, en las situaciones de cruce de partículas, es difícil tomar decisiones correctas a la hora de realizar los enlaces. A continuación se muestran posibles situaciones de cruces, donde las partículas pueden tener distancias más cortas que otras a los tracks activos a pesar que la correspondencia pueda ser errónea. Esto se debe a que las detecciones no se realizan en tiempo lineal, sino en tiempos discretos. Entonces, los saltos que pueden pegar las partículas entre un cuadro y otro pueden hacer que los verdaderos puntos de enlace no se correspondan con la distancia mínima.

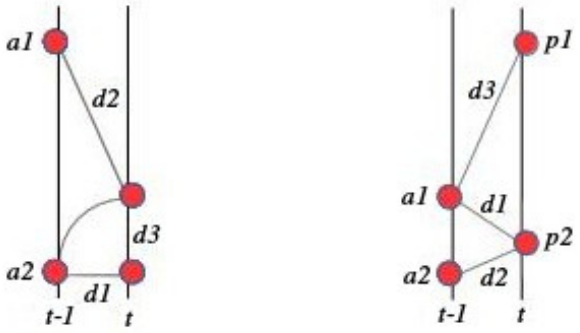


Figura 3.12: Situaciones problemáticas

Cuando no exista posibilidad de confusión en ningún sentido, entonces los tracks activos deben enlazarse con las partículas correspondientes. Por ejemplo, en la Figura 3.13 el track activo que termina con el punto a_1 se debe enlazar con el punto p_1, y el track que termina con el punto a_2 con p_2.

Figura 3.13

Supongamos que estamos parados en el cuadro C_t. Se analiza cada una de las partículas detectadas dentro de este cuadro. Y supongamos que estamos observando una partícula p dentro del cuadro C_t. Se obtienen las distancias euclidianas a los dos tracks activos más cercanos, que en este caso serían los tracks activos en tiempo $t-1$. Supongamos que las dos distancias mínimas encontradas son d_{a1} y d_{a2}, correspondientes a la menor y a la segunda menor distancia. A su vez se obtiene cual es la distancia mínima a otra partícula dentro del cuadro C_t. Supongamos que esta es d_c. Luego, la partícula p cumplirá el criterio de distancias si se cumple lo siguiente:

$$\frac{d_{a1}}{\text{mín}(d_{a2}, d_c)} < \psi \tag{3.2}$$

donde ψ es un límite de relación de distancias mínimas que condiciona el enlace entre partículas.

Notar que solamente se tiene en cuenta la distancia mínima entre d_{a2} y d_c. Esto se debe a que ésta distancia, es la que posiblemente puede generar algún tipo de confusión a la hora de enlazar.

Criterio de distancias de salto

El concepto de distancias de salto se corresponde a las distancias espaciales que tienen los puntos contiguos de un track entre un cuadro de video y el siguiente.

En ocasiones, una detección puede haber pasado el criterio de distancias con otras detecciones pero el enlace puede no ser correcto. El criterio de distancias de salto, tiene en cuenta las distancias de saltos promedio que tiene cada uno de los tracks entre sus puntos a la hora de decidir los enlaces. Esto quiere decir que si una partícula tiene una distancia d al track activo más cercano, se enlazará al mismo dependiendo de que tan grande sea esa distancia.

Al igual que en el criterio anterior se analiza cada detección dentro de un cuadro de video por separado. Supongamos que estamos observando una partícula p en un cuadro C_t y que el track activo (en tiempo $t-1$) más cercano es a, el cual está conformado por n detecciones. Para que el enlace sea válido respecto a este criterio se utiliza la distancia promedio de los $n-1$ saltos que tuvo la partícula a lo largo del track a.

Una partícula puede variar su velocidad, por lo cual las distancias pueden variar entre los diferentes puntos que conforman al track. Debido a que las velocidades pueden ir aumentando, si se utilizan las distancias promedio de los track activos como distancia radial alrededor del último punto del track sobre los cuales pueden caer las partículas a enlazarse, algunos enlaces se perderían(cuando la partícula acelera). Es por ello que el algoritmo tiene en cuenta estas situaciones.

Supongamos que la distancia de p al track activo a es d_a y que la distancia de saltos promedios del track a es d_m. Entonces el enlace del track activo a con la partícula p pasará el criterio de distancias de salto si:

$$d_a < \text{máx}(\varphi, \gamma * d_m) \tag{3.3}$$

donde φ es una distancia espacial mínima hasta el cual dos partículas se pueden enlazar, sin tener en cuenta el comportamiento previo, y γ es un factor de crecimiento de distancias de saltos entre cuadros, respecto a las distancias promedios hasta el momento.

El valor de γ limita la aceleración que puede tener una partícula. Haciendo que entre cuadro y cuadro, las distancias entre los puntos de un track puedan aumentar o disminuir, teniendo en cuenta el comportamiento previo del track, pero dándole un margen, ya sea superior o inferior, para poder realizar saltos más grandes o más pequeños. El valor de φ es de gran importancia para tracks que recién comienzan, debido a que no hay información previa y enlaces muy cercanos podrían llegar a que el track no acepte nuevas uniones por esperar detecciones demasiado cercanas. Pueden existir partículas que comiencen con una velocidad determinada y disminuyan la misma hasta cierto punto, pero luego de un determinado número de saltos, volver a aumentar la velocidad, pegando saltos más grandes entre los distintos cuadros de videos.

Si bien una partícula puede tener el visto bueno respecto a los dos criterios mencionados, todavía puede existir una situación donde el enlace no es correcto. Analicemos la Figura 3.14:

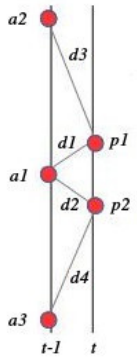


Figura 3.14

donde se cumple que:

- $d_1 \approx d_2$
- $d_3 \gg d_1$
- $d_4 \gg d_2$

En este caso si el valor de γ elegido es mayor a $0{,}5$(aproximadamente), las partículas p_1 y p_2 van a haber pasado el criterio de distancias respecto a otras detecciones. Supongamos que también se cumple el criterio de distancias de saltos. Entonces tanto la detección p_1 como p_2 van a intentar enlazarse con el mismo track activo (a_1). Lo cual es una incoherencia, ya que significa que el track se divide en dos.

Debido a esta situación, cuando dos detecciones elijan a un mismo track para enlazarse y pasen ambos los criterios mencionados anteriormente, dejarán de lado dicha unión y se convertirán ambos en nuevos tracks.

Una vez realizado el forward pass con esta implementación, se obtienen como resultados un conjunto de tracks surgidos del análisis de puntos en cuadros contiguos. Para considerar detecciones omisas, se

aplica una segunda etapa de enlace donde se busca unir tracks ya existentes y distanciados por un número de cuadros de video menores a un máximo establecido como K. Esto se hace de una manera incremental en el número de saltos de cuadros hasta alcanzar el máximo definido por este parámetro.

Consideremos un track a_1, cuyo último punto se encuentra en tiempo t. Sea otro track a_2 cuyo primer punto se encuentra en tiempo $t + k$, y sea d la distancia espacial entre estos. Es decir, la distancia entre el último punto de a_1 y el primer punto de a_2. Estos tracks se enlazarán uno con el otro si:

- $1 < k \leqslant K$
- $d < \sqrt{T(a_1, a_2)} * d_{m_{a_1}}$, donde T es la función que devuelve el número de cuadros intermedios entre estos dos tracks, y $d_{m_{a_1}}$ es la distancia promedio de saltos del track a_1.
- no existen dos tracks que se quieran unir al mismo.

Este primer algoritmo resulta efectivo sobre ciertas condiciones:

- no existen cruces de partículas, lo cuál directamente involucra el concepto de densidad de partículas.
- el fluido sobre el cual se mueven las partículas no es demasiado profundo. Es decir, las partículas no desaparecen o aparecen con demasiada frecuencia. En particular no lo hacen con demasiada frecuencia a distancias cortas de los tracks activos.

Si bien este primer algoritmo de enlace resulta básico y no tiene en consideración el manejo de algunas situaciones, en la experimentación ha arrojado muy buenos resultados. Además por el tipo de cálculos que se realiza, el costo computacional es bajo.

3.2.3. Enlace de partículas con filtros de Kalman

Los filtros de Kalman ayudan a eliminar algunos de los problemas del algoritmo de enlace de partículas basadas en distancias mínimas. Esto es debido a la capacidad de obtener estimaciones de las posiciones donde las partículas se deberían encontrar de acuerdo al comportamiento que han obtenido hasta el momento, y a las detecciones brindadas por el algoritmo de detección.

Uno de los beneficios más importantes que ofrece la aplicación de filtros de Kalman es la posibilidad de calcular trayectorias de partículas en cruces, gracias a la información sobre las posiciones y velocidades que traían las partículas hasta el momento.

También ayuda de manera significativa a evitar falsos enlaces. Es decir, cuando por ejemplo aparece una detección nueva que se encuentra más cercana al último punto de un track en un momento determinado, en el algoritmo anterior, el track continuaba con esta nueva detección suponiendo que esta había desacelerado de manera drástica o habría realizado giros drásticos entre ambos cuadros de video.

A su vez, como el algoritmo va aprendiendo sobre el registro histórico, no se limitan de manera previa las distancias máximas de los saltos que pueden realizar las partículas, además agrega una restricción a las distancias mínimas que pueden realizar.

Para poder realizar estimaciones fiables con filtros de Kalman sobre el comportamiento que posee cada una de las partículas dentro de un video, primero se debe observar durante un lapso de tiempo pequeño como se comportan dichas partículas para poder obtener información notable y no realizar estimaciones erróneas.

Es por ello, que antes de realizar estimaciones con los filtros de Kalman, primero se recoge información sobre algunas mediciones. Esto se realiza con la utilización del algoritmo de enlace basado en distancias mínimas. Y una vez recogido un mínimo de mediciones en un track, se comienza a utilizar el filtro de Kalman para estimar sus próximas posiciones y así tomar decisiones acerca de los posibles enlaces. La cantidad de mediciones mínimas a considerar está influenciada en gran medida por la densidad de partículas.

Dentro del espacio-estado a la hora de realizar las estimaciones se va a considerar información relativa a las coordenadas de las detecciones y a la velocidad de las partículas para considerar más detalles acerca del comportamiento. Es decir,

$$\widehat{x} = \begin{bmatrix} X_x \\ X_y \\ V_x \\ V_y \end{bmatrix}$$

donde X se corresponde con las coordenadas, y V las velocidades.

Suponiendo que no hay factores que modifiquen las velocidades, siendo las mediciones cada Δt segundos, y sin errores, el nuevo estado sería:

$$\begin{bmatrix} X'_x \\ X'_y \\ V'_x \\ V'_y \end{bmatrix} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} X_x \\ X_y \\ V_x \\ V_y \end{bmatrix}$$

Es decir, el nuevo estado se obtendría de la siguiente forma:

$$\begin{aligned} X'_x &= X_x + \Delta t V_x \\ X'_y &= X_y + \Delta t V_y \\ V'_x &= V_x \\ V'_y &= V_y \end{aligned}$$

Por lo cual la matriz de transición de estados A es:

$$F = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Y la matriz de función de medición es:

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

Se considera que no existen fuerzas externas. Si bien las partículas pueden modificar su movimiento independientemente de factores externos, se presupone que siguen el mismo comportamiento y no se consideran como fuerzas externas. Además para poder considerarlas se necesitaría obtener dicha información, y es justamente una de las cosas que tratamos de averiguar. Por lo cual, suponemos $Bu = 0$ y $Q = 0$.

Como estado inicial, se eligen las coordenadas de las partículas correspondientes a cada inicio de los tracks, con una velocidad inicial de 1 en cada eje. Y como matriz de covarianza inicial, se elige una matriz de 4×4 (debido a posición y velocidad sobre ejes x e y) donde los elementos de la diagonal son 1000 (lo cual refleja incertidumbre respecto a los datos iniciales) y el resto 0 (refleja la falta de correlación entre las cuatro variables). Es decir,

$$\widehat{x}_0 = \begin{bmatrix} x_0 \\ y_0 \\ 1 \\ 1 \end{bmatrix} \qquad P_0 = \begin{bmatrix} 1000 & 0 & 0 & 0 \\ 0 & 1000 & 0 & 0 \\ 0 & 0 & 1000 & 0 \\ 0 & 0 & 0 & 1000 \end{bmatrix}$$

Para poder obtener matrices de covarianza buenas, es decir que no sean aleatorias, y que reflejen una relación más real con los tracks formados hasta el momento, primero se realiza una corrida previa (dry run) de dicho algoritmo con una cantidad de cuadros de video, la cual se establece dependiendo el video observado. Luego se promedian todas las matrices de covarianza de cada uno de los filtros de Kalman correspondiente a cada partícula para obtener una nueva matriz de covarianza que será en adelante utilizada como matriz de covarianza inicial.

Suponemos también que no existen errores dentro de las mediciones, es decir que los datos de mediciones se corresponden con las posiciones reales del centro de las partículas.

Una vez iniciada la etapa de aplicación de filtros de Kalman para un track, éste, posee una cierta longitud de puntos. Por lo tanto la decisión acerca de cuantos puntos debe "recolectar" un track con el algoritmo de enlace basado en distancias mínimas refleja la importancia que tiene la información previa para poder realizar estimaciones de mayor correlación con el comportamiento real de las partículas. Principalmente al realizar las primeras estimaciones.

Los filtros de Kalman, permiten realizar estimaciones aunque en algunos momentos no se tengan mediciones. Es por ello que se permite, al igual que en el algoritmo anterior, una determinada cantidad de detecciones omisas.

El algoritmo propuesto, ya en la etapa de enlace con filtros de Kalman (es decir con una cantidad mínima de puntos para los tracks), obtiene estimaciones de las coordenadas (para cada track) donde las partículas se deberían encontrar utilizando los datos previos de los tracks, y compara las posiciones de las estimaciones con las posiciones de las detecciones reales más cercanas a cada punto. Supongamos

que estamos analizando un track a, conformado hasta el momento hasta tiempo $t - k$ (última medición real en tiempo $t - k$) y nos encontramos observando el tiempo t. Si la predicción arrojada por el filtro de Kalman acerca del lugar donde se debería encontrar la partícula en tiempo t es p_r, entonces el track a se enlazará a una nueva detección p en tiempo t si:

- $1 < k \leqslant K$
- $d_p \leqslant r$
- p es la partícula más cercana, temporalmente y espacialmente, al track activo a que cumple con las dos primeras condiciones.

donde K la cantidad máxima de saltos permitidos entre cuadros de video, d_p es la distancia euclídea entre la predicción del filtro de Kalman (p_r) y la partícula p, y r es la distancia máxima radial permitida que existe entre las predicciones y las detecciones para considerar enlaces.

En caso de haber enlace bueno, el filtro de Kalman correspondiente a dicho track se actualiza con el nuevo punto como una nueva medición. Es decir la matriz de covarianza y la estimación con el nuevo estado.

Al terminar de obtener los tracks, se realizan predicciones con los filtro de Kalman parar estimar las posiciones intermedias donde no se encontraron mediciones reales. De esta forma se obtienen predicciones confiables sobre las detecciones omisas.

4. Evaluación del algoritmo propuesto

A fin de obtener un análisis estadístico bueno sobre los tracks obtenidos, es importante que los tracks sean verídicos. Para poder evaluar de forma empírica y cuantitativa el algoritmo de obtención de tracks se utilizó el conjunto de imágenes y protocolos de evaluación utilizados en la competencia de *Particle Tracking Challenge* [28]. Los resultados obtenidos se compararon con los obtenidos por los catorce equipos participantes.

Dentro de la competencia se utilizaron 48 secuencias de imágenes generadas mediante simulación y categorizadas según tres aspectos:

- **Dinámica (tipo de movimiento)**: dividida en cuatro subconjuntos.
  - ◇ `Browniano` (VESICLE): movimientos aleatorios. Similar al que tienen las vesículas en el citoplasma.
  - ◇ `Directos` (MICROTUBULE): movimiento dirigido a velocidad casi constante. Similar a los microtúbulos.
  - ◇ `Conmutación aleatoria` (RECEPTORS): conmutación entre movimientos brownianos y directos con orientación aleatoria. Similar a las membranas receptoras.
  - ◇ `Conmutación con orientación para la componente directa` (VIRUS): conmutación entre movimientos brownianos y directos con orientación restringida en la componente de dirección. Similar a distintos virus.
- **Densidad (número de partículas)**: dividida en tres subconjuntos.
  - ◇ `Low`: ~ 100 partículas
  - ◇ `Mid`: ~ 500 partículas
  - ◇ `High`: ~ 1000 partículas
- **Señal (relativo al ruido)**: nivel de ruido dentro de la imagen.
  - ◇ `SNR = 1`
  - ◇ `SNR = 2`
  - ◇ `SNR = 4`
  - ◇ `SNR = 7`

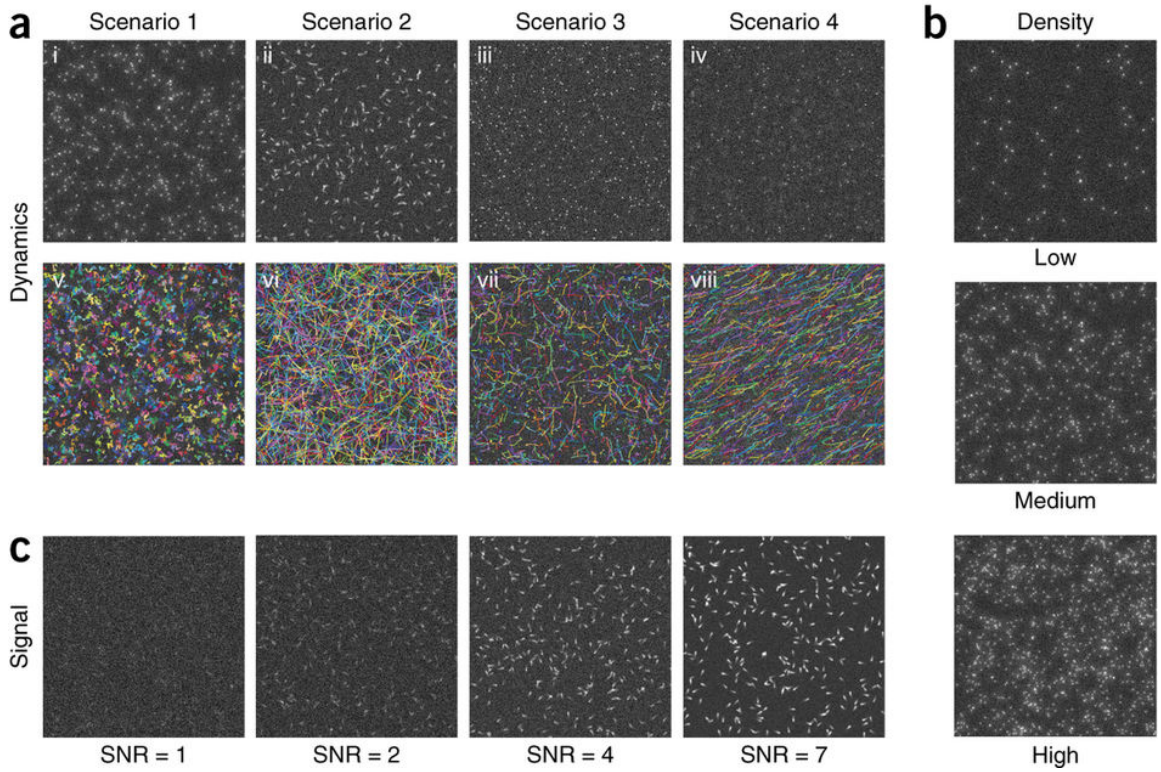


Figura 4.1: Caracterización de escenarios utilizados en *Particle Challenge*. Figura extraída del análisis de resultados de la competencia [28]

Para medir la performance de los distintos algoritmos sobre estas secuencias, se tuvieron en cuenta catorce aspectos diferentes, relacionados a los tracks, que se categorizaron en en cinco medidas de performance:

1. α: indica el grado de *matching* (emparejamiento) de tracks verdaderos con los tracks obtenidos con los algoritmos, sin tener en cuenta tracks espurios (no emparejados). Su valor está entre 0 (peor) y 1 (mejor).

2. β: es α pero con una penalización para los tracks no emparejados. Su valor está entre 0 (peor) y el valor obtenido con la medición α (mejor).

3. JSC: similitud de coeficientes de Jaccard para puntos del track. Estos coeficientes mide el grado de similitud entre dos conjuntos, sea cual sea el tipo de elementos. En este caso caracteriza la performance de la detección de partículas. Su valor está entre 0 (peor) y 1 (mejor).

4. JSC_θ: similitud de coeficientes de Jaccard para el conjunto de todos los tracks. Su valor está entre 0 (peor) y 1 (mejor).

5. $RMSE$: es una medida de precisión en las coordenadas de las detecciones sobre los tracks correctamente emparejados.

Las formulación matemática de las distintas métricas puede encontrarse en Chenouard et al. (2014) [28].

Debido a que las imágenes con $SNR = 1$ y $SNR = 2$ la tarea de detección es muy difícil de llevar a cabo, incluso para un observador humano, dichos escenarios no son de interés para contrastar los resultados de este trabajo. Se compararon los resultados obtenidos con los de los catorce equipos de la competencia sobre las secuencias de VESICLES y RECEPTORS. A continuación, se muestran los resultados sobre los coeficientes α, β, JSC (amarillo el método propuesto):

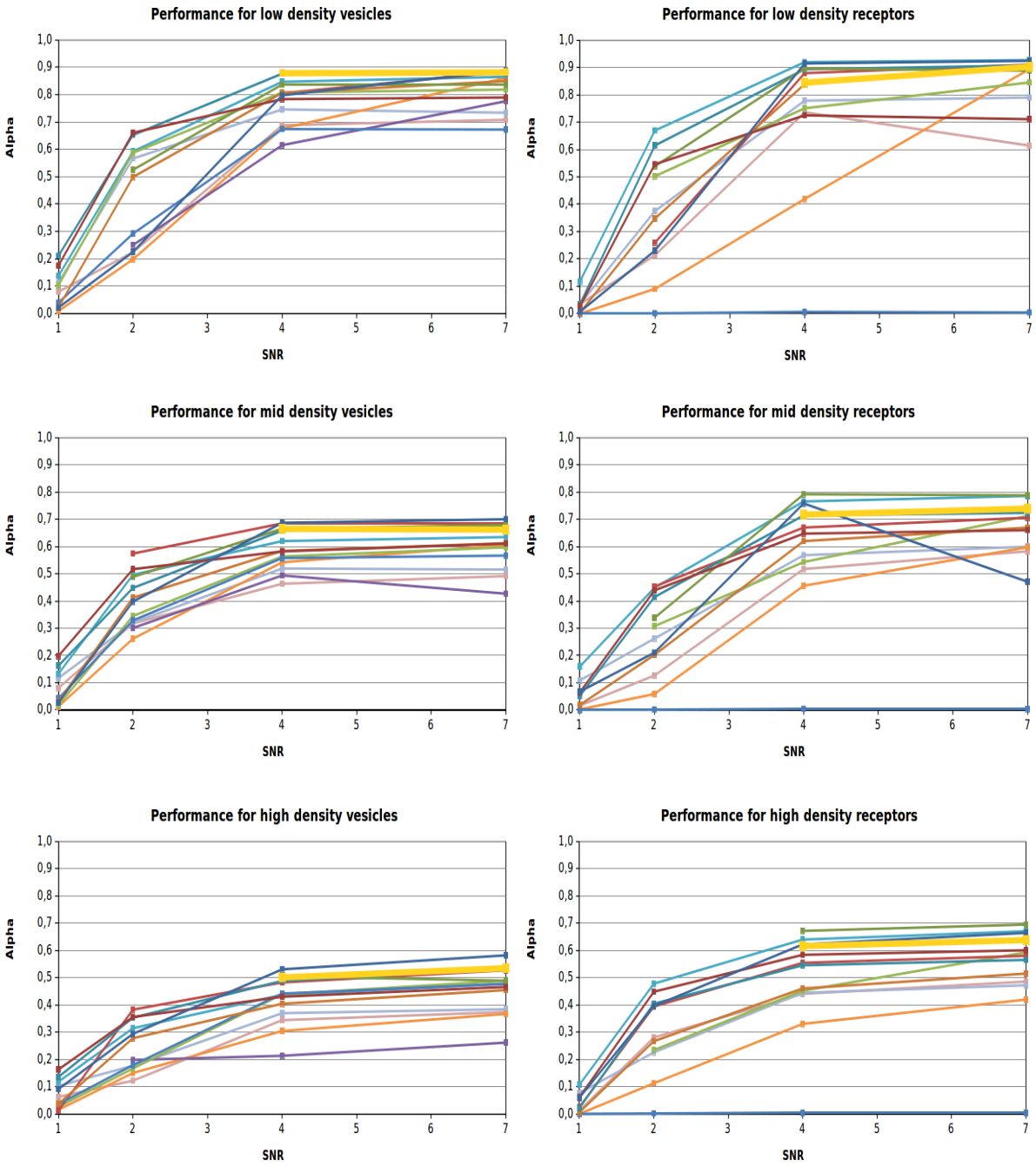


Figura 4.2: Coeficiente α sobre VESICLES (izquierda) y RECEPTORS (derecha) para los tres niveles de densidad (baja, media y alta) en función de la relación SNR. Se comparan los resultados obtenidos el método propuesto (amarillo) con el de los catorce equipos de la competencia.

Como se puede observar en las gráficas correspondientes al coeficiente α, las performance de los diferentes algoritmos es baja con niveles de SNR bajos y aumenta mientras el ruido desaparece. También disminuye cuando la densidad de parículas aumenta. Estas situaciones se dan también sobre las demás medidas de performance. El número de tracks emparejados sin tener en cuenta los tracks espurios es alto en comparación con los demás algoritmos de la competencia. Más información acerca de cada uno de los métodos de los presentados en la competencia puede encontrarse en [28].

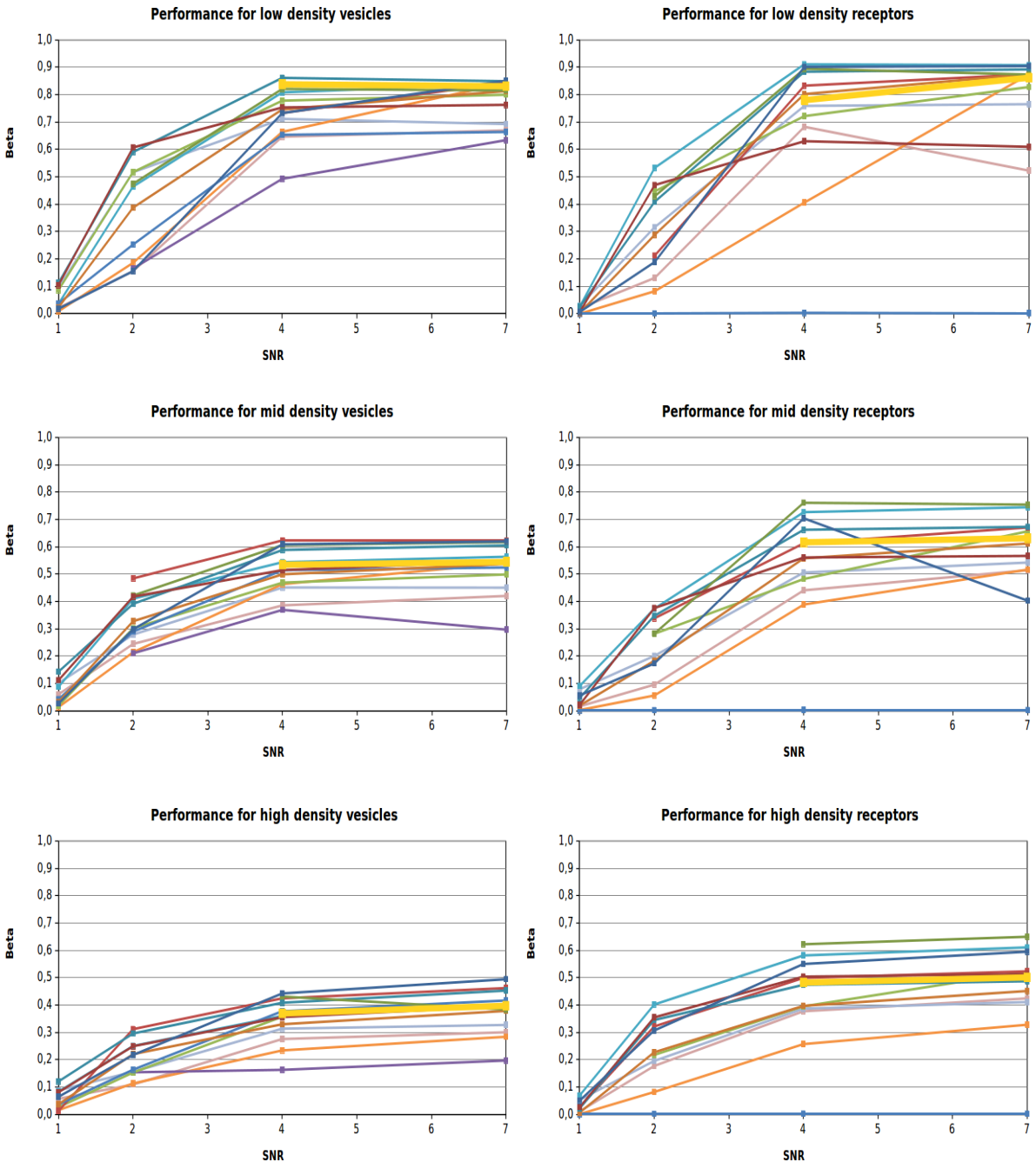


Figura 4.3: Coeficiente β sobre VESICLES (izquierda) y RECEPTORS (derecha) para los tres niveles de densidad (baja, media y alta) en función de la relación SNR. Se comparan los resultados obtenidos el método propuesto (amarillo) con el de los catorce equipos de la competencia.

Respecto al coeficiente β, que tiene en cuenta los tracks emparejados y los tracks espurios la eficiencia disminuyó un poco en comparación a los catorce equipos, posicionándose un poco por encima de la media. Lo que quiere decir que si bien los tracks emparejados de forma correcta son muchos, también hay varios tracks obtenidos con el método propuesto que no se corresponden con tracks reales.

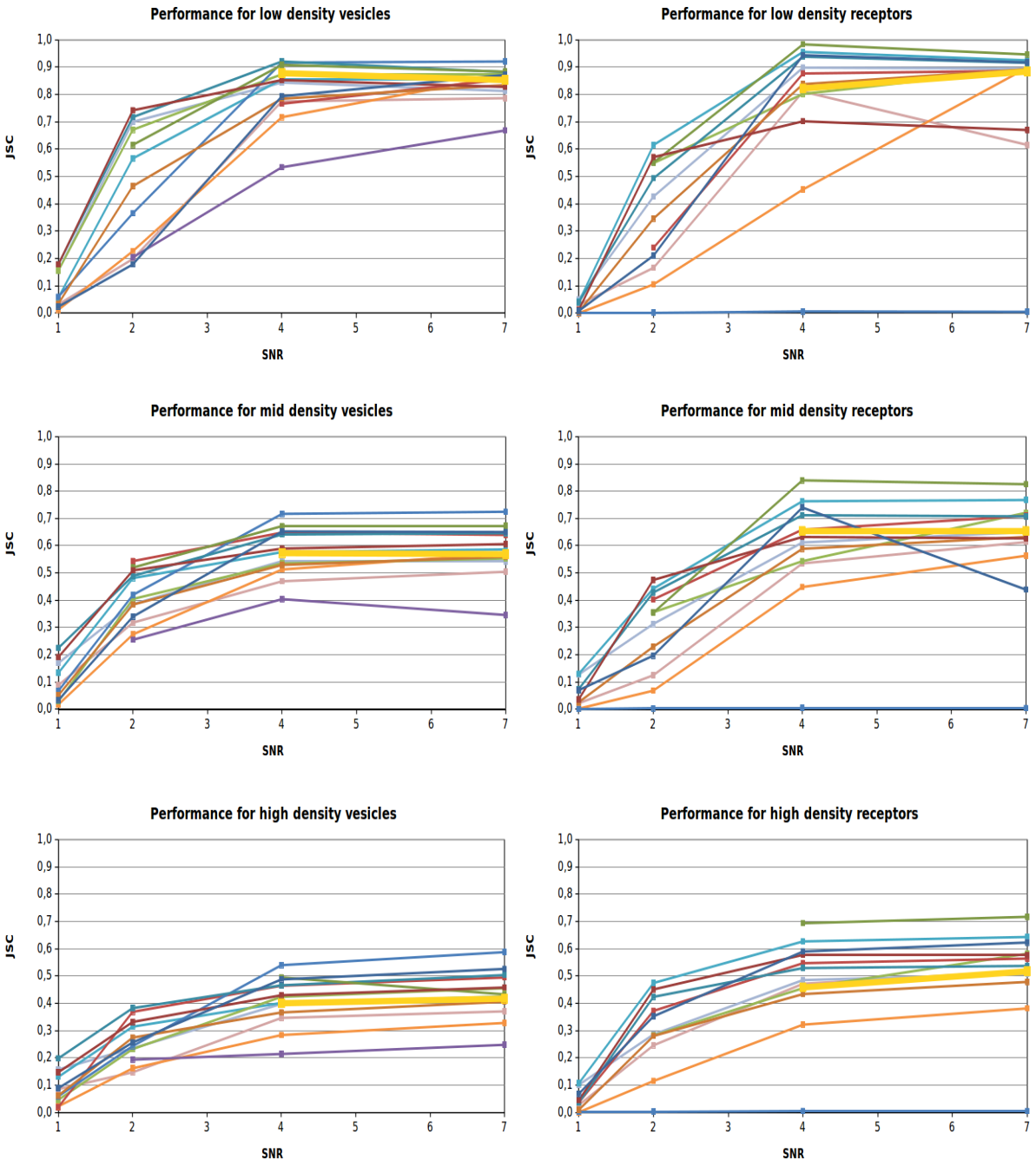


Figura 4.4: Coeficiente JSC sobre VESICLES (izquierda) y RECEPTORS (derecha) para los tres niveles de densidad (baja, media y alta) en función de la relación SNR. Se comparan los resultados obtenidos el método propuesto (amarillo) con el de los catorce equipos de la competencia.

El coeficiente JSC sirve para medir la performance en la etapa de detección, comparando las detecciones obtenidas con las reales. El algoritmo propuesto obtiene buenas detecciones cuando las densidades de partículas son bajas, pero a medida que estas aumentan, la performance respecto a este coeficiente baja más rápido en comparación a los coeficientes anteriormente observados. Los resultados se posicionan alrededor de la media (VESICLES) y un poco por debajo de la media (RECEPTORS). Esto nos dice que para mejorar el algoritmo propuesto, hay que poner gran importancia en la etapa de detección.

De forma general, los resultados obtenidos posicionan al algoritmo presentado en este trabajo con muy buenos resultados. En especial con densidades bajas y medias.

5. Análisis de trayectorias

Las trayectorias obtenidas luego de aplicar las etapas de detección y enlace son de particular importancia para poder obtener información cuantitativa del comportamiento de las partículas.

A la hora de realizar el análisis es importante tener en cuenta las dimensiones reales. Hasta el momento las coordenadas de las detecciones representan la posición espacial dentro de las imágenes, a nivel de píxeles. Sin embargo, según el tipo de estudio que se realice, puede ser necesario tener que trabajar sobre las dimensiones verdaderas. En particular, para poder obtener las velocidades de cada partícula, se necesita este tipo de información. Se utilizó esta relación de píxeles de las imágenes con las distancias originales (medidas en μm) . Lo mismo ocurre con el tiempo. Para obtener las velocidades se necesita saber en que tiempo se recorrió un trayecto. Esa información se obtiene a partir de los datos mismos del video, utilizando el dato de la cantiad de cuadros por segundo. Dado un pedido especial de un conjunto de investigadores de FaMAF y CONICET, se realizó un estudio preliminar sobre un video que brindaron ellos sobre una población de organismos llamados Choanoflagelado (Choanoflagellatea). Se obtuvieron algunos datos correspondientes a cada track. En la actualidad se siguen realizando estudios más extensos sobre los datos obtenidos por el método propuesto para la reconstrucción de las trayectorias. Los datos extraídos de forma preliminar fueron los siguientes:

- **Número de puntos**: cantidad de detecciones que se obtuvieron.
- **Velocidad media (**μm**)**: considerando las velocidades entre cada par de detecciones.
- **Distancia recorrida (**μm**)**: uniendo las detecciones de manera lineal.
- **Error cuadrático medio (**μm**)**: respecto a la recta que une el primer punto de un track con el último.
- **Tiempo de vida (**s**)**: tiempo total de visualización del track.

Como principal objetivo dentro de sus investigaciones, buscan caracterizar las trayectorias de nado de los choanoflagelados, por lo cual necesitan datos estadísticos acerca del tipo de movimiento de las partículas. Dentro de estas poblaciones, se encuentran dos categorías de partículas, las lentas y las rápidas. Las rápidas tienen un diámetro más pequeño. Dentro del preprocesamiento de las imágenes se usa un σ intermedio entre los valores óptimos que debería adoptar para cada tamaño. De esta forma la construcción de tracks es la misma para toda la población, y con la velocidad media se establece un umbral para considerarlas lentas o rápidas. En el futuro se planea extender la información con la velocidad instantánea promedio.

A la vista, se observan diferentes tipos de movimientos bien definidos dentro de las trayectorias:

- **Lineal**: trayectorias rectas o con variaciones muy pequeñas.
- **Browniano**: trayectorias con movimientos aleatorios.
- **Curvo**: trayectorias con curvas.
- **Con tumbos**: trayectorias con cambios de dirección bruscos.

5.1. Detección automática de cambios de dirección

Una partícula puede cambiar el tipo de movimiento entre un instante y otro. En el presente trabajo se desarrolló un método para detectar tumbos en los tracks, que será de gran ayuda en la tarea de caracterización de las trayectorias.

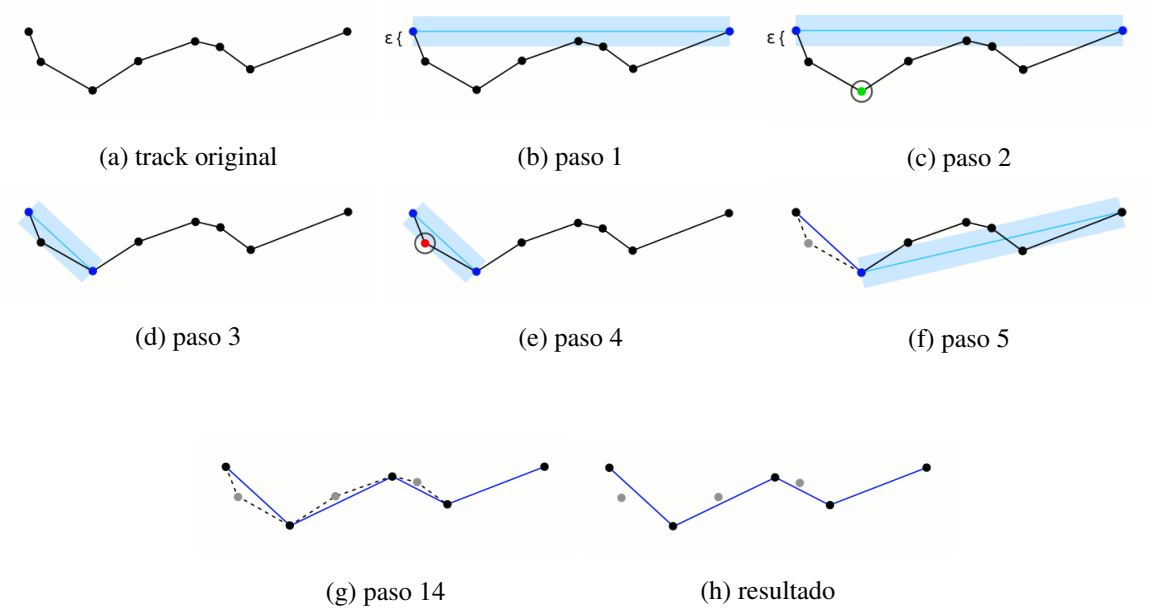
En una primera instancia se utilizó un algoritmo basado en la interpolación de dos rectas centradas a lo largo de parches de tamaño n. A continuación se muestra un pseudocódigo del algoritmo aplicado sobre cada track:

$tumble\_points \leftarrow []$
$start\_path \leftarrow first\_point\_idx + path\_size$
$end\_path \leftarrow last\_point\_idx - path\_size$
for $i = start\_path$ **to** end_path **do**
  $line1, line2 \leftarrow interpolate\_2lines(i - path\_size,\ i,\ i + path\_size)$
  if $degree(line1,\ line2) >\ min\_degree$ **then**
    $tumble\_points.append(i, degree(line1,\ line2))$
  end if
end for

Este algoritmo requiere un ángulo mínimo establecido por parámetro, ya que de lo contrario, cada parche va a arrojar una detección de tumbo, a excepción de cuando la mitad izquierda y la mitad derecha del parche interpolan las mismas líneas.

En la experimentación el primer algoritmo arrojó buenos resultados solo en algunas ocasiones. Está directamente relacionado al tamaño del parche. En algunos tracks, para obtener buenos resultados se debería aplicar distintos tamaños de parches a lo largo del track. Es por ello que se descartó la aplicación del método y se desarrolló un nuevo método utilizando el algoritmo de *Ramer–Douglas–Peucker(RDP)* [29, 30]. Este algoritmo aplica un método recursivo sobre distintos segmentos. Básicamente traza una línea entre los puntos de los extremos y calcula la distancia al punto más lejano de la recta. Si esa distancia es mayor que un ϵ, el algoritmo marca dicho punto como tumbo y aplica recursividad sobre los dos nuevos segmentos (divididos por el punto de tumbo). A continuación se muestra un pseudocódigo con el algoritmo y unas imágenes del algoritmo en funcionamiento:

function $rdp(points[\,],\ \epsilon)$:
  $dist\_max \leftarrow 0$
  for $i = first\_point\_idx + 1$ **to** $last\_point\_idx - 1$ **do**
    $d \leftarrow distance2line(points[first\_point\_idx],\ points[last\_point\_idx]),\ points[i])$
    if $distance >\ dist\_max$ **then**
      $dist\_max \leftarrow d$
      $tumble\_idx \leftarrow i$
    end if
  end for
  if $dist\_max >\ \epsilon$ **then**
    $recursive\_1 \leftarrow rdp(points[first\_point\_idx \cdots tumble\_idx],\ \epsilon)$
    $recursive\_2 \leftarrow rdp(points[tumble\_idx \cdots last\_point\_idx],\ \epsilon)$
    return $[recursive\_1,\ recursive\_2]$
  else
    return $[first\_point\_idx,\ last\_point\_idx]$
  end if



(a) track original (b) paso 1 (c) paso 2

(d) paso 3 (e) paso 4 (f) paso 5

(g) paso 14 (h) resultado

Figura 5.1: Algoritmo RDP

A la hora de considerar tumbos válidos, una de las restricciones que aportaron los investigadores, es que la partícula se debe mover al menos n cuerpos. Por lo cual el tamaño de la partícula es importante. Y debido a que las medidas brindadas se encuentran en μm, es necesario cambiar las medidas para no trabajar a nivel de píxeles. Si bien la elección del ϵ condiciona la cantidad de cuerpos mínimos que se debe desplazar una partícula, éste lo hace de manera ortogonal a la recta. Lo que quiere decir, que

para respetar mínimamente la distancia de n cuerpos en puntos ortogonales a la recta, cercanos a los extremos, por ejemplo, el ϵ debería ser igual a $2nd$ (con d el diámetro de la partícula). Consideremos la Figura 5.2. Supongamos que para ser considerado tumbo, una partícula se debe desplazar dos cuerpos. Entonces, la detección 2 no debe ser considerada como tumbo, debido a que hay un desplazamiento de aproximadamente un cuerpo desde el punto anterior. El ϵ elegido debe ser entonces mayor que la distancia ortogonal de la recta al punto 2. Esto hace que el punto 3, que es un tumbo real, no sea detectado. Si se utiliza un ϵ menor, la detección 2 se consideraría tumbo.

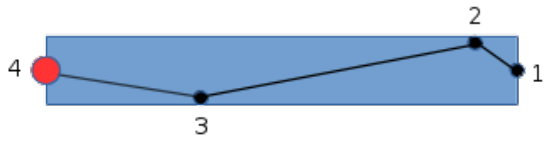


Figura 5.2: Elección del ϵ

Para abordar estas situaciones, la detección de tumbos se realiza con las siguientes cuatro etapas:

1. **Algoritmo rdp**: se realiza una corrida del algoritmo de RDP con ϵ bajo (en el caso anterior, para que 3 se considere tumbo)

2. **Forward pass**: una vez obtenido un track *simplificado* con la aplicación del algoritmo RDP, se recorre hacia adelante el track, comparando la distancia de cada punto con el siguiente. Si esa distancia es menor a los n cuerpos, entonces el punto siguiente se descarta, y se compara con el siguiente. En caso de que la distancia sea mayor a n, ambos puntos se conservan y se pasa a comparar a partir del siguiente no descartado. Esto se realiza hasta comparar las distancias con el penúltimo punto, ya que de lo contrario se podría eliminar la ultima detección, y podría haber una distancia menor a n cuerpos seguidos del cambio de dirección.

3. **Backward pass**: símil al forward pass pero recorriendo el track en sentido inverso. Esto se debe a que pueden existir casos donde el último punto del resultado del forward pass esté a menos de n cuerpos del último punto del track. Recorriendo hacia atrás se considera esta situación y se actualizan nuevamente los tumbos con la nueva información.

4. **Algoritmo rdp**: luego de haber realizado el forward y el backward pass, la simplificación de los puntos puede resultar en que nuevos puntos no deberían ser considerados. Por ejemplo en la Figura 5.3, si en el backward pass borra el punto 3 debido a la distancia de cuerpos, el punto 2 puede dejar de considerarse tumbo si entra dentro del margen de $\epsilon/2$ que une los puntos 1 y 4. Anteriormente era considerado tumbo por el margen alrededor de la línea entre 1 y 3.

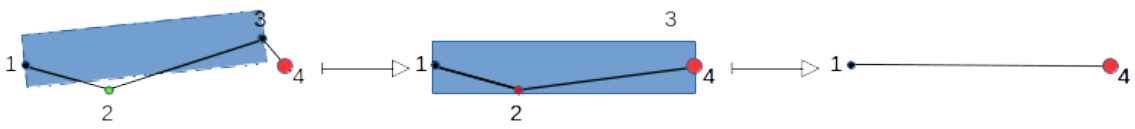


Figura 5.3: Aplicación del algoritmo RDP (última etapa)

Con estas cuatro etapas de procesamiento, el track de la Figura 5.2 conserva el punto 3 si consideramos un ϵ más pequeño, y tomamos como referencia que se debe mover al menos 2 cuerpos. El punto 2 se elimina en el forward pass.

Obtener estos puntos simplificados elimina ruido de las detecciones. Ya que las coordenadas de las mismas se obtienen a partir del análisis de las intensidades. Entonces pequeñas fluctuaciones se relacionan con variaciones de intensidad debido a los sensores y no con el movimiento real de las partículas. Con este método, además se conservan solo los puntos donde hay tumbos. Por lo cual el análisis posterior se realiza sobre menor cantidad de puntos. Debido a que las partículas que tienen movimiento browniano se mueven de manera aleatoria dentro de áreas pequeñas, al obtener un track simplificado en la primer etapa de aplicación del algoritmo de RDP, el número de puntos simplificados es mucho menor. En las zonas que no corresponden movimiento browniano, el número de simplifaciones también es pequeño por la naturaleza del algoritmo. Debido a esto, aplicar las siguientes tres etapas del método de detección de tumbos no genera gran cómputo computacional.

Una vez obtenidos los tracks simplificados, cada punto intermedio es un punto de cambio de dirección en la trayectoria. Obteniendo las diferencias de ángulos entre los dos vectores generados cada tres puntos consecutivos, se obtienen los ángulos de los cambios de dirección. A continuación se muestra como se obtiene el ángulo de tumbo correspondiente al punto 2.

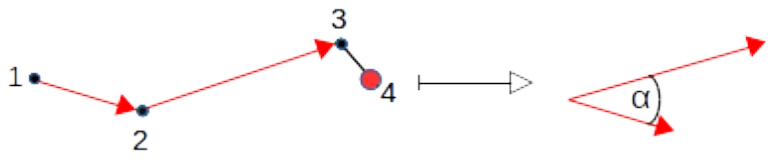


Figura 5.4: Obtención de ángulos

5.2. Análisis de datos para categorización de movimientos

Con la información obtenida hasta el momento, se pueden estudiar características de los tracks a dos niveles, microscópico y macroscópico. A nivel microscópico considerando cada una de las detecciones, y a nivel macroscópico utilizando las simplificaciones generadas por el algoritmo de detección de cambios de direcciones.

Algunas de las partículas observadas difieren su tipo de movimiento entre un instante y otro. Por ejemplo, hay partículas que por momentos tienen trayectorias de tipo lineal, y en otros momentos se mueven de manera aleatoria. Debido a esto, realizar una categorización sobre el movimiento de las partículas no es una tarea sencilla.

Las partículas que se mueven con trayectorias lineales se pueden identificar teniendo en cuenta las velocidades instantáneas. O sea, las velocidades entre cada par de detecciones consecutivas. Si un track

tiene trayectoria lineal, entonces el promedio de velocidades instantáneas debe ser muy similar a la velocidad media, considerando solo el primer y último punto del track. Es decir:

$$index\_1 = \frac{L/T}{\frac{1}{n-1}\sum_{i=1}^{n-1}\frac{\Delta l_i}{\Delta t_i}} \tag{5.1}$$

Mediante la observación de éste índice sobre un conjunto de tracks, se concluye que tracks con valores superiores a $0{,}96$ se corresponden con movimiento lineales (sobre el video de organismos choanoflagelados).

A nivel microscópico, una tarea común en este tipo de investigaciones es el estudio de cambios de dirección entre cada una de las detecciones. Es decir, punto a punto. Al comparar estos datos con los tracks simplificados, se puede obtener un estimativo acerca del nivel de ruido dentro de las detecciones.

Para obtener estadísticas que se correspondan de mejor forma con los datos reales, se debe tratar de eliminar el ruido. En particular, dentro de las muestras, en muchos casos, había trayectorias que seguían una dirección y en un solo punto intermedio había un retroceso. Las coordenadas de esas detecciones, a su vez, estaban a distancias ínfimas. Por lo cual esas coordenadas son erróneas por detalles en los cambios de iluminación, y no se corresponden con el movimiento real de las partículas. Si no se maneja este ruido, se arrastra a las estadísticas. Se realizaron pruebas ajustando un paraboloide centrado en cada partícula para obtener las coordenadas más precisas pero no se notaron cambios significativos.

Debido a que para cada partícula tenemos mediciones y estimaciones de los estados, gracias a los filtros de Kalman, se pueden reemplazar cada una de las coordenadas por estimaciones. Esto ayuda a limpiar los tracks de estas pequeñas fluctuaciones, sin cambiar el comportamiento a nivel macroscópico. También se realizaron pruebas suavizando los datos con promedio móvil, pero el suavizado aunque sea con tamaños de ventana pequeños es mayor, lo cual hace que las estadísticas sean menos fiables.

Se calcularon histogramas de los cosenos de los ángulos entre vectores consecutivos (formados por tres puntos consecutivos) dentro de cada uno de los tracks originales, divididos cada uno en 10 segmentos. En la mayoría de los tracks resaltan los valores alrededor de 1. Esto se debe a que la mayoría de los vectores generados entre tres puntos consecutivos generan ángulos pequeños.

En muchas investigaciones se utilizan estos histogramas como base para categorizar los tipos de movimientos. Dentro de la muestra de organismos choanoflagelados, hay partículas que se mueven rápido, que generalmente se mueven de forma más lineal, y las lentas, que generalmente se mueven de forma browniana. De acuerdo a esto, las detecciones de las partículas lentas se encuentran más cercanas entre sí, y como el proceso de detección no es perfecto, y tiene muchas fluctuaciones en cuanto a las coordenadas de las partículas (por el método mismo de microscopía lumínica), los ángulos entre vectores de tres puntos consecutivos son grandes. Por lo cual se considera importante realizar análisis separados de las partículas lentas y las rápidas.

Una de las observaciones que se realizó, es que en general, coincidían las partículas rápidas, que tenían un porcentaje menor al $\%80$ de ángulos casi nulos (la barra del histograma correspondiente a

cosenos de ángulos iguales a 1), con las categoría de movimientos brownianas. Dentro de futuras investigaciones se estudiarán con mayor detalle estos datos para realizar una categorización más completa.

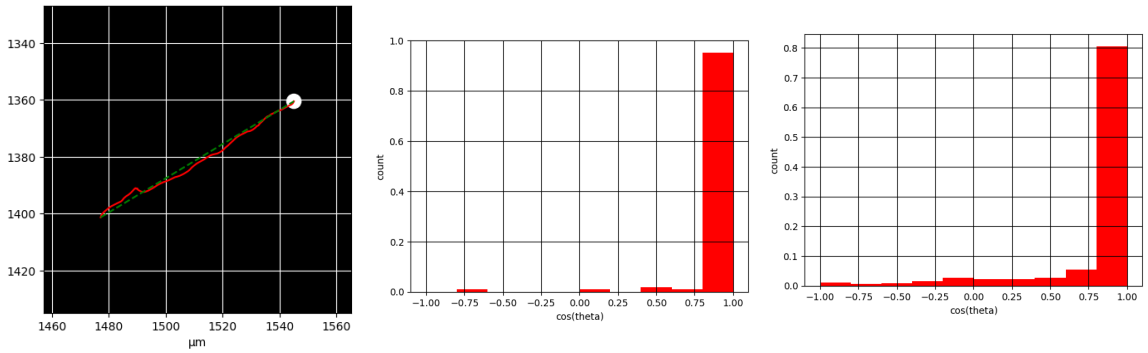
Además de los histogramas formados por tres puntos consecutivos, se obtuvieron histogramas relacionados a los cosenos de los ángulos de todos los vectores formados por puntos consecutivos. Es decir, si hay n puntos que conforman el track, hay $(n-1) \times (n-1) - n$ cosenos de ángulos (sin considerar comparaciones de los vectores consigo mismo). Estos histogramas pueden ser de gran ayuda para poder categorizar los movimientos. Tracks lineales deben tener solamente (o en su mayoría) cosenos de ángulos iguales a 1. Tracks con movimiento browniano se corresponden con histogramas chatos, con valores similares en cada segmento, a excepción de los cosenos de ángulos iguales a 1, que debería predominar, salvo que el track se caracterice por realizar cambios de dirección muy seguidos. En tracks con pocos cambios de dirección, el histograma va a tener picos sobre valores de 1 y los correspondientes a los ángulos de los cambios de dirección.

Este tipo de histogramas se cree que puede brindar información importante para detectar algunos tracks con con trayectorias curvas, pero amerita un mayor análisis. Si bien los tracks con movimiento browniano se corresponden con histogramas chatos, una partícula que se mueva en círculos por ejemplo, también generará un histograma similar. La categorización de los tracks en muchos casos queda sujeta a la interpretación. Tracks con varios cambios de dirección pueden o no considerarse con movimiento browniano por ejemplo.

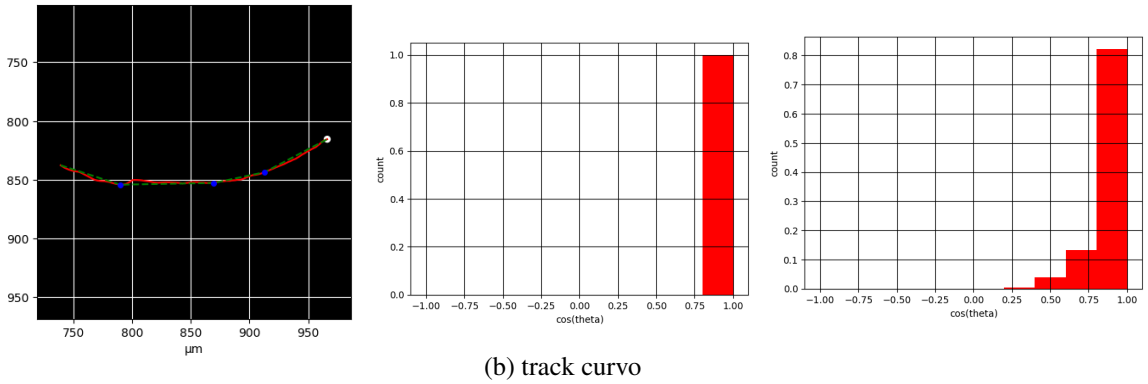
Los tracks simplificados, conformados solamente por los puntos de cambios de dirección, brindan información valiosa para poder realizar análisis macroscópicos. Posiblemente, mediante el ajuste de los valores de ϵ y el número de cuerpos mínimo que se debe desplazar una partícula para considerar una detección como tumbo, se puede obtener una representación más adecuada al movimiento de las partículas lentas, que ayude a tomar decisiones en cuanto a las clasificaciones.

El abordaje en cuanto a la clasificación de manera eficiente de cada uno de los tracks queda exento de este trabajo y requiere de una profunda investigación.

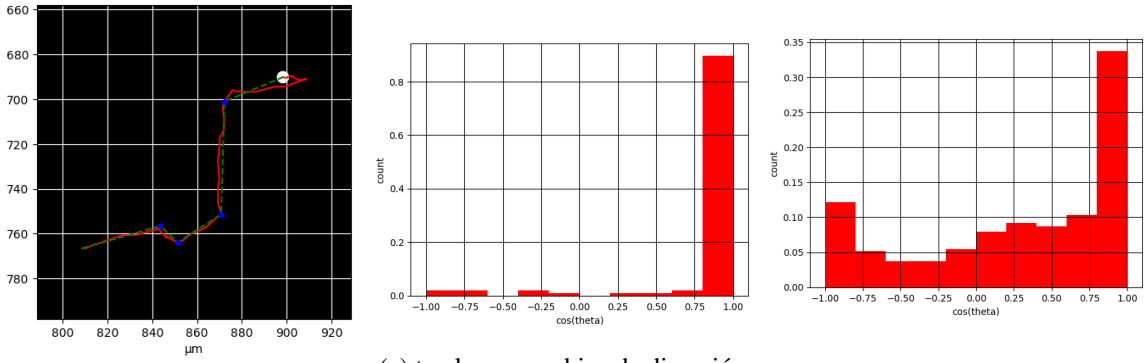
En la Figura 5.5, se muestra un ejemplo de cada tipo de movimiento, las respuestas del algoritmo de detección de cambios de dirección, y los resultados de ambos histogramas.



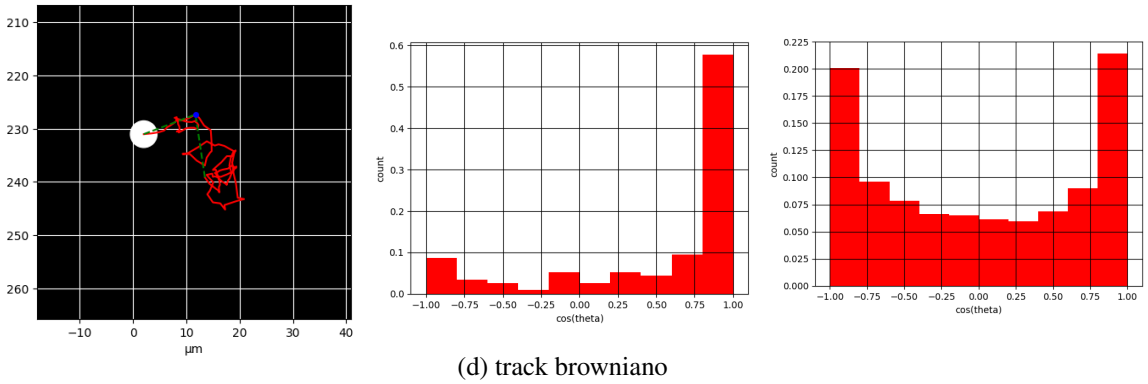
(a) track lineal



(b) track curvo



(c) track con cambios de dirección



(d) track browniano

Figura 5.5: Líneas verdes representan el track simplificado. Puntos azules son los puntos considerados tumbos, con $\epsilon = 10$ y $numero\_cuerpos = 2$. Los histogramas de la izquierda, son los formados por cosenos de ángulos entre vectores de tres puntos consecutivos. Los histogramas de la derecha corresponden a los cosenos de ángulos entre todos los pares de vectores.

6. Conclusión y trabajos futuros

Dentro de este trabajo se analizaron diferentes inconvenientes conocidos dentro de la literatura a la hora de abordar el problema de reconstrucción de trayectorias de partículas mediante secuencias de video obtenidas con técnicas de microscopía óptica. Se divide el problema en tres, una parte de preprocesamiento, una de detección, y una de enlace. Donde las etapas de preprocesamiento y de detección pueden unirse en una sola.

Dentro de la etapa de detección se analizan diferentes tipos de ruido, en especial el que mayor afecta a las muestras, que es el ruido blanco Gaussiano. Se proponen diferentes técnicas para poder limpiar cada una de las imágenes. Las cuales incluyen la sustracción de la imagen de fondo, que elimina agentes externos (suciedad), y la eliminación del ruido, entre los que se encuentran el redimensionado (que principalmente sirve para reducir tiempos de cómputo, pero al aplicarlo se realiza un suavizado), el algoritmo de vecinos no locales (non-local means) y el suavizado Gaussiano. Este último el más importante en cuanto a la eliminación del ruido. Luego de tener imágenes limpias, se realiza la detección de las partículas al aplicar un operador Laplaciano para destacar las mismas y posteriormente obtener las coordenadas utilizando detección de máximos locales.

Este algoritmo no tiene en cuenta algunas propiedades características de algunas muestras, como por ejemplo las partículas con flagelo. Éstas, pueden llegar a generar falsos positivos, y encontrar un umbral de separación a la hora de detectar máximos locales puede ser difícil si el fluido donde se mueven las partículas es muy profundo, ya que los flagelos pueden generar valores superiores a los que generan las partículas inmersas. Contemplar estas situaciones en trabajos futuros servirá a obtener un algoritmo más integral.

Otra situación que se suele dar en este tipo de videos es que las partículas forman halos al moverse dentro del fluido. Considerar esta información, ayudaría a obtener mejores datos de las partículas. Ya que por ejemplo, la velocidad calculada en este trabajo se realiza sobre dos dimensiones, y tener en cuenta los tamaños de los halos ayudaría a obtener información acerca de las velocidades a través de la profundidad del fluido.

Finalmente, el objetivo principal en trabajos futuros, debería ser que el algoritmo contemple la detección de diferentes tamaños de partículas dentro de una imagen. Para ello, se plantea aplicar los filtros mencionados con diferentes valores de σ, y obtener máximos locales sobre tres dimensiones.

La tarea de enlace se realizó siguiendo un algoritmo que utiliza filtros de Kalman. El mismo, se podría mejorar para que luego de obtener los tracks, los mismos se analicen para corroborar buenos

enlaces. Un ejemplo, sería recorrer los tracks, y para cada detección calcular estimaciones hacia adelante y hacia atrás, y comparar ambos resultados.

En cuanto al análisis de las trayectorias, para caracterizar el movimiento de un grupo de organismos choanoflagelados, se sugieren algunas métodos para poder discernir sobre algunos categorías y sobre algunas situaciones. A su vez se plantean interrogantes sobre los análisis basados en los datos utilizados en este tipo de estudios, y se discuten algunas falencias de los mismos.

La caracterización de tipos de movimientos requiere una investigación más profunda. Las herramientas desarrolladas dentro de este trabajo sirven para considerar poblaciones grandes, obteniendo las trayectorias de manera automática, sin la necesidad de realizar anotaciones a mano. Una tarea, que es prácticamente imposible de llevar a cabo de forma manual con grandes densidades y tiempos prolongados. Se espera que la investigación y las herramientas presentadas dentro de este trabajo, sirvan como base para futuras investigaciones científicas.

Bibliografía

[1] QiaoQiao Ruan Valeria Levi and Enrico Gratton. 3-d particle tracking in a two-photon microscope: Application to the study of molecular dynamics in cells. *Biophysical Journal*, 88:2919–2928, 2005.

[2] Michelle Dawson Junghae Suh and Justin Hanes. Real-time multiple-particle tracking: applications to drug and gene delivery. *Advanced Drug Delivery Reviews*, 57:1551, 2005.

[3] Michael J. Saxton and Ken Jacobson. Single-particle tracking:applications to membrane dynamics. *Annual Review of Biophysics and Biomolecular Structure*, 26:373–399, 1997.

[4] Jani Tuoriniemi Julian A.Gallego-Urrea and Martin Hassellö. Applications of particle-tracking analysis to the determination of size distributions and concentrations of nanoparticles in environmental, biological and food samples. *TrAC Trends in Analytical Chemistry*, 30:473–483, 2011.

[5] Denis Wirtz. Particle-tracking microrheology of living cells: Principles and applications. *Annual Review of Biophysics*, 38:301–326, 2009.

[6] D.J. Stephens and V.J. Allan. Light microscopy techniques for live cell imaging. *Science*, 300:82–86, 2003.

[7] Ian Parker Michael J. Sanderson, Ian Smith and Martin D. Bootman. Fluorescence microscopy. *Cold Spring Harb Protoc.*, 10, 2014.

[8] David B. Williams and C. Barry Carter. *The transmission electron microscope*. Transmission electron microscopy, Springer, Boston, MA, 1996.

[9] ImageJ. https://imagej.net/welcome.

[10] Amira 3D/4D Software for Cell Biology. https://www.fei.com/software/amira-for-cell-biology.

[11] Track Manager Icy tools like Motion Profiler, Spot Tracking. https://omictools.com/cell-tracking-category.

[12] Tracking tools. https://omictools.com/cell-tracking-category.

[13] Marconi Verónica I. Sánchez Jorge A., Pury Pedro A. Un algoritmo modular para el seguimiento de partículas en videos de microscopía. *Mecánica Computacional*, 34(51):3443–3450, 2016.

[14] Nicole King Roman Stocker Gastón L. Miño, M. A. R. Koehl. Finding patches in a heterogeneous aquatic environment: ph-taxis by the dispersal stage of choanoflagellates. *Limnology and Oceanography Letters 2*, 2:37–46, 2017.

[15] A.J. Banchio J. Sparacino, G.L. Miño and V.I. Marconi. Solitary choanoflagellate dynamics and micro-confined directed transport. *PReprint*, 2017.

[16] Jean-Michel Morel Antoni Buades, Bartomeu Coll. Nonlocal image and movie denoising. *International Journal of Computer Vision*, 76(2):123–139, 2008.

[17] Jean-Michel Morel Antoni Buades, Bartomeu Coll. Non-local means denoising. *Image Processing On Line*, 2011.

[18] Sánchez J. A. Evaluación de operadores diferenciales γ-normalizados para selección automática de escalas. *Mecánica Computacional*, 26:2129–2144, 2007.

[19] Neusser K. *State-Space Models and the Kalman Filter. In: Time Series Econometrics*. Springer Texts in Business and Economics. Springer, Cham, 2016.

[20] Kalman Rudolf E. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

[21] Lihhua Xie Frank L. Lewis and Dan Popa. Optimal and robust estimation: With an introduction to stochastic control theory. 2:1–50, 2008.

[22] R. G. Brown and P. Y. C. Hwang. *Introduction to Random Signals and Applied Kalman Filtering: with MATLAB Exercises and Solutions (Third ed.)*. Wiley and Sons Inc., 1996.

[23] Ribeiro Maria Isabel. *Kalman and Extended Kalman Filters: Concept, Derivation and Properties*. Institute for Systems and Robotics, Instituto Superior Técnico, Av. Rovisco Pais, 1 1049-001 Lisboa Portugal, 2004.

[24] Smal I. Meijering E., Dzyubachyk O. Methods for cell and particle tracking. *Methods Enzymology*, 504:183–200, 2012.

[25] William F. Walker Michael K. Cheezum and William H. Guilford. Quantitative comparison of algorithms for tracking single fluorescent particles. *Biophysical Journal*, 81:2378–2388, 2001.

[26] Shubeita G.T. Carter B.C. and Gross S.P. Tracking single particles: a user-friendly quantitative evaluation. *Physical Biology*, 2:60–72, 2005.

[27] Niessen Wiro Smal Ihor, Loog Marco and Meijering Erik. Quantitative comparison of spot detection methods in fluorescence microscopy. *IEEE Transactions on Medical Imaging*, 29(2):282–301, 2010.

[28] De Chaumont F. Maška M. Sbalzarini I.F. Gong Y. Cardinale J. Carthel C. Coraluppi S. Winter M. et al. Chenouard N., Smal I. Objective comparison of particle tracking methods. *Nature methods*, 11(3):281–289, 2014.

[29] Urs Ramer. An iterative procedure for the polygonal approximation of plane curves. *Computer Graphics and Image Processing*, 1(3):244–256, 1972.

[30] David Douglas and Thomas Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *The Canadian Cartographer*, 10(2):112–122, 1973.

Los abajo firmantes, miembros del Tribunal de Evaluación de tesis, damos Fe que el presente ejemplar impreso, se corresponde con el aprobado por éste Tribunal