UNIVERSIDAD NACIONAL AUTÓNOMA
DE MÉXICO

FACULTAD DE CIENCIAS

NEURITE: SISTEMA DE SOFTWARE PARA LA CUANTIFICACIÓN SEMIAUTOMÁTICA DE LA REGENERACIÓN NERVIOSA A PARTIR DE VIDEOS DIGITALES

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

LICENCIADO EN CIENCIAS DE LA COMPUTACIÓN

P R E S E N T A :

ZIAN FANTI GUTIÉRREZ

DRA. MARIA ELÉNA MARTÍNEZ PÉREZ
DR. FRANCISCO FERNÁNDEZ DE MIGUEL

2007

Hoja de Datos del Jurado

1. Datos del alumno:
   Fanti
   Gutiérrez
   Zian
   55 49 19 48
   Universidad Nacional Autónoma de México
   Facultad de Ciencias
   Ciencias de la Computación
   09960060-0

2. Datos del Tutor
   Dra.
   Maria Eléna
   Martínez
   Pérez

3. Datos del Co-tutor
   Dr.
   Francisco
   Fernández
   De Miguel

4. Datos del Sinodal 1
   Dra.
   Catherine
   García
   Reimbert

5. Datos del Sinodal 2
   Dr.
   Edgar
   Gardúno
   Ángeles

6. Datos del Sinodal 3
   M. en C.
   Alejandro
   Aguilar
   Sierra

7. Datos del Trabajo Escrito
   Neurite: Sistema de software para la cuantificación semiautomática de la regeneración nerviosa a partir de videos digitales.
   94p
   2007

Índice general

RESUMEN

El análisis cuantitativo del crecimiento de células nerviosas, es parte esencial de diversas investigaciones en el área de las neurociencias. Actualmente el crecimiento es registrado en secuencias de imágenes digitales las cuales son analisadas manualmente, imagen tras imagen, lo que hace esta tarea imprácticа y tediosa por la necesidad de analizar secuencias de cientos de imágenes. En los últimos años, algunas aplicaciones de software, tanto comercial como libre, han sido desarrolladas con el fin de facilitar ésta cuantificación morfológica. Sin embargo estas herramientas solo sirven para tipos específicos de imágenes y no tienen soporte para secuencias de imágenes o videos.

El presente trabajo consistió en desarrollar una herramienta de software multiplataforma, implementada en Java, con una interfaz gráfica para el usuario , la aplicaciónń utiliza la biblioteca Java Advanced Image (JAI) para la identificación, trazado y medición de neuritas en secuencias de imágenes de una manera semiautomática. Esto permite que el usuario identifique una sola vez en la secuencia las neuritas que desea medir y el software realice el seguimiento y medición de una manera automática imagen por imagen, guardando los resultados en una hoja de cálculo Excel para su análisis posterior. El despliegue gráfico permite al usuario corregir los errores y proseguir de manera automática. Además el programa permite trabajar con imágenes adquiridas mediante diferentes tipos de óptica, como la de contraste de fase, fluorescencia, o bien de microscopía diferencial de contraste de interferencia

(Nomarski). El programa cuenta con un modulo exclusivo para este último tipo de óptica.

El sistema se basa en la extracción de la linea central de las formas tubulares que caracterizan a las neuritas. Tras haber seleccionado algunos puntos de interés en una imagen de la secuencia, un algoritmo de trazado y seguimiento de neuritas segmenta la secuencia de manera automática, obteniendo así la longitud de cada neurita seleccionada en cada imagen de la secuencia. El sistema provee herramientas para un preproceso con el fin de realzar las partes de interés en las imágenes. Entre estas herramientas hay filtros tradicionales de procesamiento digital de imágenes, un sistema de normalización de luz en secuencias, capacidad de eliminación de fondo y detección de movimiento, entre otras.

El rendimiento del método depende del tipo de imágenes a procesar. Éste se probó en tres secuencias de imágenes. Obteniendo que en un 80 % de las imágenes, las neuritas seleccionadas para ser caracterizadas fueron segmentadas y cuantificadas correctamente de una manera automática. La aproximación propuesta al problema del trazado y seguimiento de neuritas es una innovación substancial sobre los métodos tradicionales, ya que permite cuantificar secuencias de 70 imágenes en minutos mientras que a un experto le tomaría horas o incluso días, realizar la misma cuantificación con las herramientas tradicionales.

Capítulo 1

INTRODUCCIÓN

La regeneración nerviosa, es un fenómeno ampliamente estudiado desde principios de siglo XX, entre las distintas ramas de la medicina y biología. En gran medida, estos estudios son realizados debido a que éste fenómeno tiene una relación directa con la vida cotidiana de una parte de la población mundial. Y es que el completo entendimiento de los factores que regulan este fenómeno, podría resultar en la cura de las personas que han sufrido lesiones en el sistema nervioso central entre muchas otras. Es bien sabido que la regeneración nerviosa es casi nula en los mamíferos, mientras que en algunos invertebrados si se presenta.

Cientos de estudios se realizan a diario en el mundo para tratar de comprender mas a fondo este fenómeno. Uno de ellos es el desarrollado en el laboratorio del Dr. Fernández de Miguel del Instituto de Fisiología Celular de la UNAM, en el cual se utilizan neuronas de sanguijuela, para entender los factores que intervienen en el proceso de regeneración. El crecimiento de neuritas, que son expansiones del soma de las neuronas las cuales se convertirán en dendritas o bien en un axón, son los primeros pasos en la regeneración nerviosa. Diferentes estudios siguieren que las interacciones entre neuritas provenientes de la misma célula son cruciales en el establecimiento de los pa-

trones de ramificación. El análisis de la dinámica los patrones de crecimiento y retracción, sugieren que la extensión de algunas neuritas ocurre a expensas de la retracción otras neuritas de la célula. La correcta cuantificación del los patrones de crecimiento y retracción de estas neuritas es parte fundamental en esta investigación.

La regeneración de neuritas depende del tipo de sustrato sobre el cual crecen las neuronas y tiene cambios dinámicos en las tasas de extensión y crecimiento de las neuritas [De Miguel y Vargas, 2002]. Para poder entender el fundamento de estos fenómenos es necesario conocer en detalle la dinámica del crecimiento dendrítico durante el periodo de regeneración, pero con una resolución temporal de minutos o incluso de segundos a lo largo de periodos prolongados. Una manera de poder conocer detalladamente estos cambios, es capturar una secuencia de imágenes del fenómeno, en las cuales se puedan apreciar los cambios presentados. El grupo del Dr. Fernández de Miguel del Instituto de Fisiología Celular de la UNAM, ha desarrollado un sistema de adquisición automática de imágenes que permite hacer secuencias de imágenes de varias neuronas individualmente en un mismo experimento. El resultado de esta adquisición son secuencias de varios cientos de imágenes por célula en un periodo de 24 horas.

La naturaleza de los experimentos, requiere obtener la mayor cantidad de información de los patrones morfológicos para verificar qué variables son las responsables del crecimiento neuronal. Esto es, que para cada variable se requieren varias secuencias de imágenes, las cuales se deberán analizar detalladamente. Este análisis incluye el trazado y medición de las neuritas regeneradas, a lo largo de toda la secuencia. Actualmente el análisis se realiza usando paquetes de programas comerciales que permiten trazar a mano cada neurita en imágenes individuales. Este proceso puede llevar a una persona experimentada hasta varias horas para cada imagen en la secuencia. Haciendo que el análisis de las secuencias completas sea prácticamente imposible y sólo se seleccionen algunas imágenes representativas.

Dicho analisis de las secuencias, ha hecho necesario el contar con una herramienta que pueda realizar el trazado y medición de manera automática o semiautomática. En la actualidad muchos métodos han sido desarrollados para resolver dicho problema de trazado de neuritas y algunos programas dedicados al procesamiento digital de imágenes médicas han incluido herramientas para facilitar el trazado y medición. Algunas son: NIH Image [2007]; ImageJ [2007]; Scion Image [2007]; Metamorph 7 [2007]; Media Cybernetics [2004]. Sin embargo ninguno de ellos cuenta con herramientas para segmentar automática o semi-automáticamente secuencias de imágenes. Image Xpress (Axon Instruments) provee un modulo de análisis para secuencias de imágenes que requiere imágenes de fluorescencia o microscopía de fluorescencia y no identifican cada una de las neuritas en la secuencia [Keenan *et al.*, 2006].

La automatización del trazado de neuritas es en sí mismo útil, y varios procedimientos han sido propuestos, [Keenan *et al.*, 2006; Al-Kofahi *et al.*, 2003; Meijering *et al.*, 2003], pero como antes se mencionó, la mayoría sólo trabajan con una imagen a la vez. Esto implica que el seguimiento de una neurita a lo largo del tiempo tiene que hacerse manualmente.

Al-Kofahi *et al.* [2006] han presentado un método para dar seguimiento y medir las características morfológicas de axones a lo largo de una secuencia de imágenes, siendo el único trabajo de este tipo presentado hasta el momento. Este método tiene la capacidad de poder detectar crecimiento, desaparición, unión o ruptura de axones. Este trabajo está especialmente diseñado para imágenes adquiridas mediante microscopía de contraste de fase e imágenes de fluorescencia.

El objetivo del presente trabajo fue desarrollar una herramienta de software de fácil manejo y ambiente interactivo para la cuantificación de la regeneración nerviosa. Para ello se aplicaron técnicas usadas en el procesamiento

digital de imágenes que contienen formas tubulares o elongadas, como los vasos sanguíneos en la retina [Martinez-Perez *et al.*, 2007] y el análisis de secuencias de imágenes para vigilancia de carreteras [Cheung y Kamath, 2004]. Se hizo una aproximación a la solución del trazado, seguimiento y cuantificación automática de neuritas en secuencias de imágenes tomadas mediante microscopía diferencial de contraste de interferencia o Nomarski (DIC por sus siglas en ingles). La estructura general de nuestro método es la siguiente: las imágenes son procesadas para eliminar ruido y artefactos provenientes de la adquisición. Las imágenes en la secuencia son alineadas una a una de manera automática. Seguidamente las imágenes son deconvolucionadas para eliminar la sombra características de las imágenes DIC, y por último las neuritas de interés son seleccionadas de una manera semiautomática en una imagen, dejando que el programa de una manera automática realice la identificación, trazado y medición a lo largo de la secuencia.

Capítulo 2

ADQUISICIÓN DE IMÁGENES

Capturar eventos es uno de los propósitos de la fotografía. El encuadre, la profundidad de campo, la iluminación, son cosas que un fotógrafo toma en cuenta para poder obtener buenas imágenes. Sea cual sea la fotografía (familiar, de paisaje, deportiva, científica, etc), si la imagen está mal tomada quizá no exista un proceso, función o algoritmo capaz de compensar la diferencia por completo y en el caso de que existiera.

Un buen fotógrafo debe de conocer su equipo y las condiciones óptimas para obtener una buena imagen. En nuestro caso, saber manipular correctamente el equipo y conocer las condiciones óptimas para fotografiar una célula harán la diferencia entre una buena imagen y una mala imagen.

2.1. Sistema de adquisición de imágenes

Aunque la adquisición de imágenes no es el tema principal de estudio de este trabajo, un conocimiento detallado del proceso es de gran utilidad para la obtención de imágenes de buena calidad, con las que se minimice su posterior procesamiento. Se requiere de buena iluminación, enfoque y que

Figura 2.1: Microscopio Nikon DIATHOP-TMD [Nikon, 1996]

.

la información obtenida contenga mínimas cantidades de ruido y error. En nuestro caso el sistema de adquisición consta de varios equipos, cada uno con sus propias características, las cuales afectan la adquisición de la imagen. A continuación se describirán brevemente las características y funcionamiento del sistema de adquisición usado en este estudio.

2.1.1. Microscopio

El microscopio es un *Nikon DIATHOP-TMD*, con la capacidad de trabajar en diferentes configuraciones: iluminación basada en luz blanca, epifluorescencia, ópticas de contraste de fase y de Contraste Diferencial de Interferencia (DIC). Cuenta con una cámara con detector Charge Couple Device (CCD) para el registro de imágenes. Para este estudio la configuración del microscopio usada fue óptica DIC con objetivo de 40X (ver Figura 2.1).

El microscopio tiene montada una platina motorizada con dos grados de libertad, esto es, se puede desplazar en el plano X, Y. La platina puede ser manejada remotamente a través de una computadora o localmente por medio de su controlador, el cual también puede manipular la altura del revolver de objetivos, lo que permite enfocar el microscopio a diferentes alturas en el plano Z.

2.1.2. Cámara CCD

La cámara del equipo de adquisición, modelo *Hamamatsu C2400*, diseñada para observaciones cercanas al infrarrojo, ideal para aplicaciones DIC. El detector es un CCD con transferencia por cuadro en formato de 2/3 de pulgada y una resolución de 768 por 494 pixeles a 8 bits.

Figura 2.2: Cámara CCD y el controlador del equipo de adquisición [Hamamatsu Corp., 2006]

Esta cámara es controlada por el *Hamamatsu C2400-60 Camera Controller* que provee un realce de contraste y brillo así como corrección de sombras. Estas aplicaciones son muy útiles cuando se trabaja en condiciones de poco contraste o de bajas intensidades de luz [Hamamatsu Corp., 2006]. En la figura 2.2 se muestra la cámara CCD y su controlador, mientras que la figura 2.3 muestra el diagrama de conexiones para la cámara y su controlador.

Figura 2.3: Diagrama de conexión entre la cámara y el controlador [Hamamatsu Corp., 2006]

.

2.1.3. Software de adquisición

Las imágenes captadas por la cámara son guardadas en la computadora mediante el software *ImagePro Plus 5.1* [Media Cybernetics, 2004]. Ésta, es una de las más avanzadas y costosas herramientas para el procesamiento digital de imágenes biológicas, el cual también nos permite controlar cámaras y sistemas como los mencionados en las secciones 2.1.1 y 2.1.2.

Mediante un macro para el software *ImagePro Plus 5.1* desarrollado en el laboratorio del Dr. Fernández de Miguel, el software controla todos los movimientos de la platina y algunas funciones de la cámara, con el fin de proveer una automatización completa del sistema de adquisición de imágenes, esto es, una vez que se han seleccionado los parámetros iniciales, la adquisición de imágenes se podrá realizar sin la supervisión del usuario.

2.2. Microscopía diferencial de contraste de interferencia (DIC)

La microscopía diferencial de contraste de interferencia, también es conocida como *Microscopía de Nomarski*. Es usada para observar espécimenes translúcidos. Esta óptica genera imágenes monocromáticas con un característico efecto de sombra, que da la sensación de volumen. Lo anterior es

debido a que el DIC trabaja bajo el principio de interferometría, captando la información de la densidad óptica de la muestra.

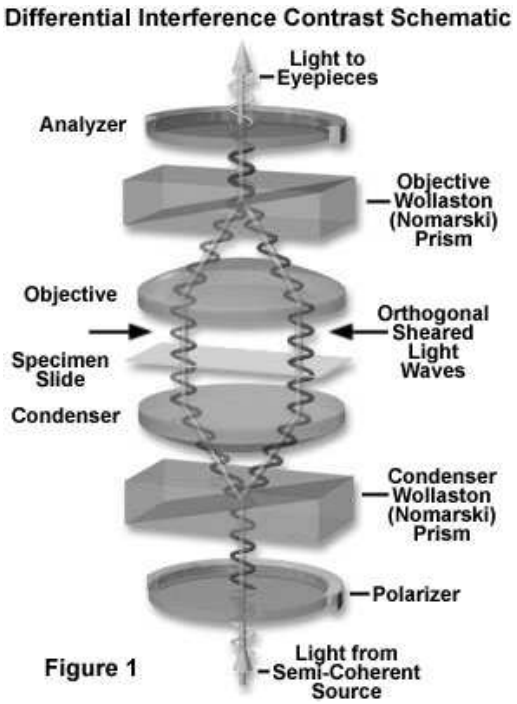


Figura 2.4: Diagrama general del funcionamiento de la microscopía DIC. [Molecular Expressions, 2004]

La técnica DIC trabaja separando un haz de luz polarizada en dos rayos, cada uno con una separación de aproximadamente la mitad de la longitud de onda del rayo de luz original, como se observa en la figura 2.4. Estos atraviesan la muestra y dependiendo del índice de refracción y de el grosor, los rayos incidentes se desfasan de diferente manera, y después son captados por el sistema DIC del microscopio y mezclados para generar la imagen [Padawer, 1968]. En la figura 2.5 se muestra un ejemplo de una imagen adquirida por medio de la microscopía DIC, en la figura se puede apreciar el típico efecto de tridimensionalidad que le da a los objetos translúcidos.

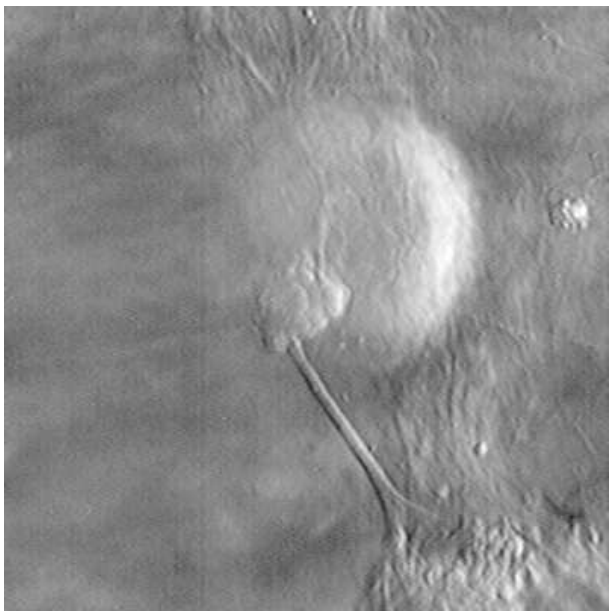


Figura 2.5: Imagen de 400 × 400 pixeles, de una neurona en regeneración obtenida mediante óptica DIC.

2.3. Adquisición de imágenes en el tiempo

El montaje y configuración del equipo descrito en las secciones anteriores, facilita la adquisición de datos de los experimentos en los que se busca registrar patrones de crecimiento celular. La duración de cada experimento puede tomar varias horas o incluso días, de aquí la necesidad de un sistema automático de adquisición de imágenes.

En una muestra típica pueden haber una o más células y el objetivo es registrar el crecimiento de cada una de ellas. Para llevar a cabo esto el sistema realiza los siguientes pasos: a) al usuario establece las coordenadas x, y de la posición de cada una de las células en la platina; b) se seleccionan los puntos de enfoque en los que se desea fotografiar a cada célula sobre el eje z y c) se selecciona el intervalo de tiempo entre cada toma. En cada toma de imágenes, la platina se desplaza en el plano x, y hasta centrar la primera célula de la muestra; entonces se mueve en el eje z tomando una imagen

en cada plano seleccionado, y de ahí se desplaza a las coordenadas x, y que corresponden a la siguiente célula para repetir el proceso, y así sucesivamente hasta abarcar las células en la muestra. El sistema de adquisición se detiene cuando el tiempo de muestreo definido por el usuario concluye. Todas las imágenes adquiridas son guardadas en una computadora de y están listas para su posterior análisis o proceso.

2.4. Problemas con las imágenes adquiridas

Aunque el sistema de adquisición es muy completo y sus elementos de la mejor calidad, no es perfecto. Las imágenes o secuencias de imágenes presentan características que dificultan la extracción de los datos de interés. A continuación describiremos algunos de estos problemas.

Uno de los problemas más visibles y graves es la desalineación de las imágenes en una secuencia, debido a imperfecciones en el posicionamiento de la platina de un ciclo a otro. Este problema hace que al desplegar la serie, pareciera que la célula se traslada, lo que hace imposible la aplicación de algunas técnicas de filtrado.

La variación de la intensidad de luz de una imagen a otra es otro problema técnico ya que algunas funciones se basan en las intensidades para realizar las mediciones. Este problema es debido a que la intensidad de luz no siempre es constante, y aunque este problema está previsto por los diseñadores de la lámpara del microscopio, en experimentos largos las variaciones eléctricas, térmicas, etc, hacen variar la intensidad de luz.

Las variaciones de luz no solo se presentan de imagen a imagen, sino que también están presentes en una sola imagen, debido a que la luz al atravesar los lentes del microscopio no es captada homogéneamente en toda la imagen

y existen partes de ésta que son más brillantes que otras. Las impurezas de los lentes también presentan problemas ya que si éstos están sucios, se traduce a manchas obscuras que pueden ocultar información valiosa en la imagen.

Debido a los problemas anteriormente expuestos, es necesario realizar una serie de preprocesos a la secuencia de imágenes con el fin de mejorar su calidad visual.

Capítulo 3

PREPROCESO

Como se vio en el Capítulo 2 las secuencias de imágenes presentan características que dificultan la segmentación automática de las partes de interés. Por ello un preproceso mejora el desempeño de las herramientas utilizadas en la fase de segmentación.

Los métodos descritos a continuación, no son todos necesarios para obtener imágenes óptimas. La elección y el orden de aplicación lo decidirá el usuario con base en su experiencia en imágenes y su conocimiento de los procesos implementados.

3.1. Corrección de campo plano

Al adquirir una imagen digital, mediante un microscopio, cámara, o algún sistema óptico, la imagen adquirida probablemente registre, además del objeto de interés, algunos otros artefactos que dificulten el análisis o la correcta visualización del espécimen. Tales artefactos pueden ser la iluminación desigual, ruido térmico, patrones fijos de ruido; rayones, polvo o manchas en las lentes, etc. Además de lo anterior, un ajuste incorrecto de una imagen basal puede generar valores de brillo más allá de sus valores reales, que por consiguiente provocará errores en la medición.

La *corrección de campo plano*, o *Flat Field Correction*, puede eliminar muchos de los artefactos no deseados en la imagen y restaurar la fidelidad de las características originales [Molecular Expressions, 2004]. Es muy recomendable la aplicación de este procedimiento antes de realizar cualquier medición o procesamiento de la imagen. Para poder efectuar el proceso es necesario contar con tres imágenes: I, I_b, I_{ff}.

I : es la imagen que contiene el objeto de interés.

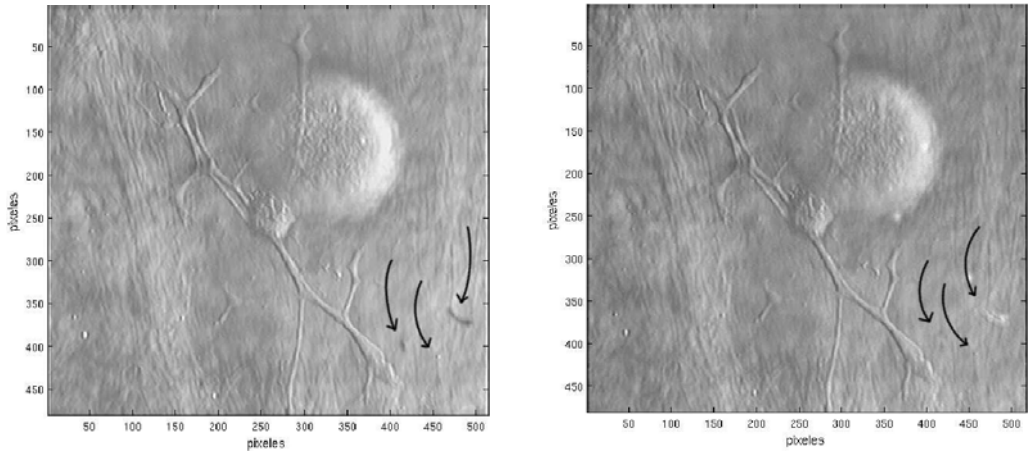
I_b : es una imagen tomada con la luz del microscopio apagada, esto con la finalidad de tener una imagen en la que se capte el ruido causado por la interferencia de señales que afecten al sensor, así como el ruido térmico, ya que el sensor recolecta el valor de cada pixel cuando se expone a una escena

I_{ff} : es una imagen sin espécimen. Ésta se puede adquirir de dos maneras; una vez que se tiene todo listo para tomar la imagen del espécimen, este se retira y se adquiere la imagen o bien, si el espécimen no se puede retirar simplemente se desenfoca la lente de tal manera que este desaparezca completamente de la imagen. Con esta imagen se capturara la luminosidad en el campo de enfoque del microscopio. Se recomienda tomar de 3 a 20 imágenes y promediarlas para de esta forma obtener una imagen sólida y con bajos niveles de ruido.

Una vez obtenidas las imágenes es posible realizar la corrección de acuerdo a la ecuación 3.1.

$$I_c = \frac{(I - I_b) \cdot \mu}{(I_{ff} - I_b)} \tag{3.1}$$

donde μ la media de la imgen resultante de $I_{ff} - I_b$. En la figura 3.1 se muestra un ejemplo de la aplicación del método de corrección de campo plano.



(a) Imagen original (b) Imagen corregida

Figura 3.1: Corrección de campo plano en una imagen. Las flechas marcan correcciónes a los artefactos ópticos en la imagen.

3.2. Filtros

Los filtros son una parte esencial en el procesamiento digital de imágenes. Mediante ellos podemos modificar las imágenes dependiendo de la aplicación. Por ejemplo: podemos resaltar características de interés o eliminar algún tipo de ruido. Existen gran variedad de filtros diseñados para una gran variedad de tareas. En nuestro caso sólo nos serán de utilidad algunos de ellos, en particular los que sirven para eliminar algunos tipos de ruido y los de detección de bordes.

La mayoría de los filtros operan con el *método del vecindario* el cual consiste en definir un punto central (x, y) y realizar una operación que involucra solo a los pixeles en un vecindario predefinido alrededor del punto central, asignándole el resultado final de la operación a dicho punto (x, y). Las operaciones realizadas sobre los pixeles en el vecindario pueden ser lineales o no lineales.

El mecanismo de los filtros lineales consiste en multiplicar cada pixel en el vecindario por un coeficiente y sumar los resultados para obtener la respuesta

para cada punto (x, y). Si el vecindario es de un tamaño de $m \times n$, entonces serán necesarios $n \cdot m$ coeficientes. Estos coeficientes son colocados en una matriz llamada *máscara* ó *kernel de filtro*. La aplicación del filtro en toda la imagen se lleva acabo moviendo el centro del kernel ω a lo largo de cada uno de los pixeles en la imagen. Para cada punto (x, y), la respuesta del filtro en ese punto será la suma del producto de los coeficientes del filtro contenidos en el kernel por el valor de los pixeles en el área cubierta por el tamaño de la vecindario.

Para los filtros no lineales el mecanismo es muy similar, pero como su nombre lo dice, se aplican funciones no lineales a los pixeles que están dentro del vecindario, por ejemplo la respuesta de un filtro sobre pixel (x, y) podría ser el valor máximo de los pixeles contenidos en el vecindario. Una característica importante es que una vez aplicado el filtro no existe un filtro inverso que regrese la imagen a su estado original [Gonzalez y Eddins, 2004].

3.2.1. Gaussiano

Es un filtro lineal usado para eliminar el ruido y los detalles finos de la imagen. El kernel usado representa una forma gaussiana (Figura: 3.2). La distribución circularmente simétrica de una gaussiana en 2D queda definida por la ecuación 3.2.

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{3.2}$$

Donde σ es la desviación estándar, que representa el ancho de la curva, por lo que variando el valor de σ se cambiará la forma de la distribución y por consiguiente la fuerza de aplicación del filtro. El mecanismo del filtro gaussiano consiste en usar alguna distribución gaussiana en 2D para generar el kernel que se usará con el método descrito en la sección 3.2. Al aplicar el filtro de esta manera tendremos una buena aproximación del filtro real y será más eficiente que si se hubiera generado completamente un filtro gaussiano 2D

del tamaño de la imagen y en el espacio de las frecuencias lo hubiéramos multiplicado por la imagen, es decir hubieramos realizado la convolución de la imagen con un filtro gaussiano en el espacio de las frecuencias [Russ, 1998]. En la Figura 3.3 vemos el ejemplo de la aplicación del filtro Gaussiano sobre una sección de imagen.

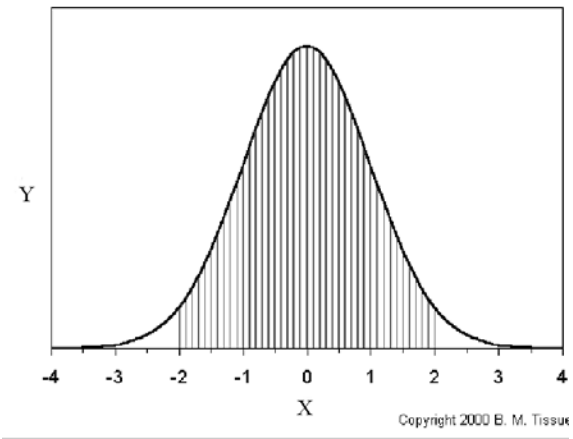
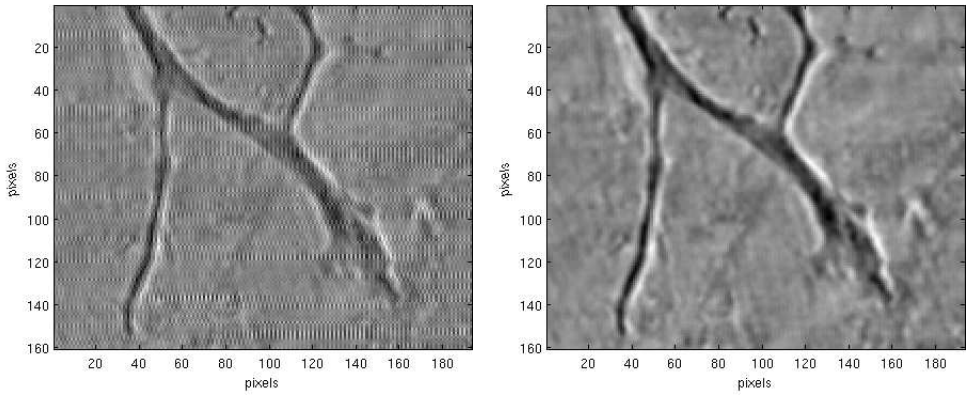


Figura 3.2: Forma clásica de una curva gaussiana en 2D.



(a) Imagen original (b) Aplicación del filtro gaussiano

Figura 3.3: Ejemplo del filtro Gaussiano. En (a) se muestra un segmento de la imagen original. En (b) se tiene la misma imagen después de la aplicación del filtro Gaussiano. Se puede apreciar como el ruido ha desaparecido.

3.2.2. Filtro de énfasis de altas frecuencias, "High Boost"

Es un filtro lineal que puede ser usado cuando se necesita dar énfasis a los detalles finos de la imagen sin perder las formas, ya que este filtro resalta los componentes de altas frecuencias, manteniendo las bajas. Este filtro puede ser visto como la imagen origen menos un filtro paso bajas, como se muestra en la figura 3.4.

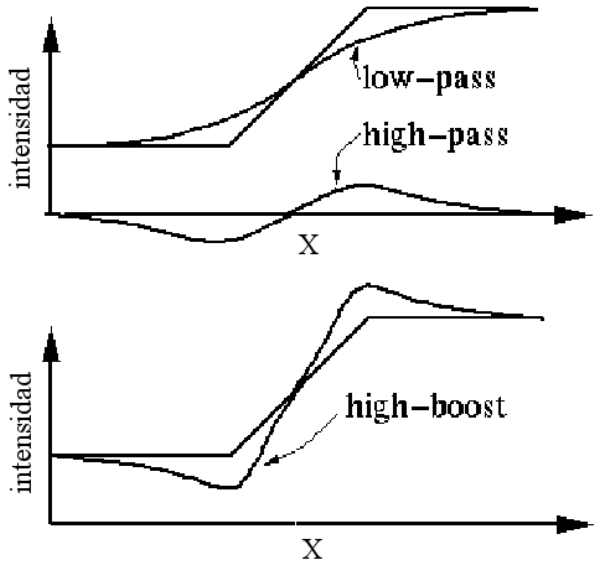


Figura 3.4: Filtro High Boost con respecto a los filtros de frecuencias altas y bajas [Wang, 2004]

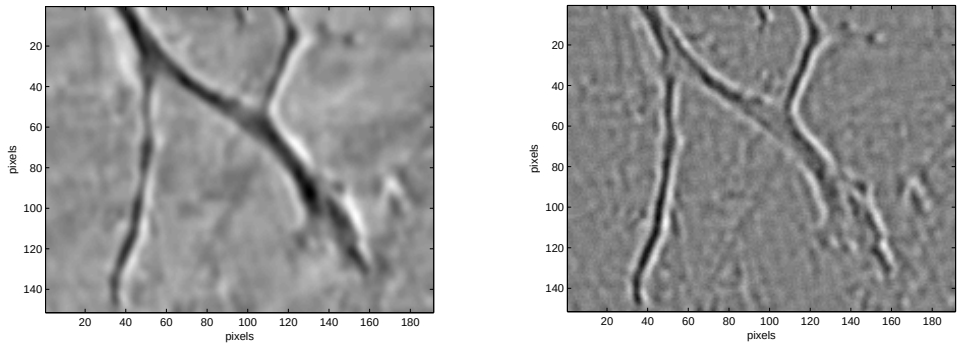
Este filtro en 2D utiliza el siguiente kernel:

$$\mathbf{HB_k} = \begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 + c & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix}$$

donde $c \in \mathbb{N}$.

En la figura 3.5 se puede ver como el filtro enfatizó los bordes, ésto es,

conservó las formas.



(a) Imagen original (b) Aplicación del filtro high boost

Figura 3.5: Ejemplo del filtro High Boost. (a) Imagen original. (b) Se aprecia la alicación del filtro high boost, a la imagen original

3.2.3. Sobel

El operador sobel es un filtro lineal usado para la detección de bordes. Calcula una aproximación del gradiente de la función de intensidad de la imagen. Si es usado en imágenes en escala de grises, el filtro destaca la tasa de cambio en las intensidades de la región definida como vecindario, y a su vez, qué tanto un punto (x, y) en la imagen puede representar un borde [Russ, 1998].

El funcionamiento del operador está basado en el uso de dos kerneles de tamaño (3×3) para calcular la aproximación de la convolución aplicada a la imagen original, ésto es, que un kernel responderá a los cambios en la dirección horizontal mientras que el otro responderá a los cambios en la dirección vertical. Si definimos a I como la imagen original y S_x y S_y los kerneles que representan las aproximaciones de las derivadas de intensidad horizontal y vertical respectivamente, entonces:

$$\mathbf{S_x} = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \otimes \mathbf{I} \tag{3.3}$$

para responder a cambios horizontales y

$$\mathbf{S_y} = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \otimes \mathbf{I} \tag{3.4}$$

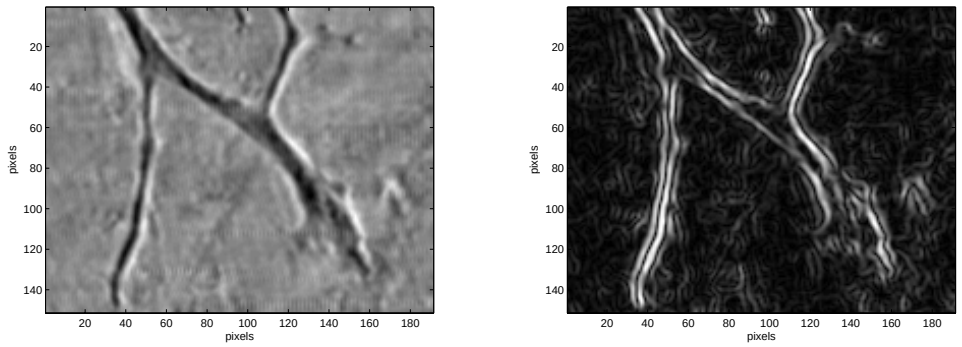
para responder a cambios verticales. Ahora en cada punto de la imagen los gradientes pueden ser combinados para obtener la magnitud del gradiente, mediante:

$$Grad = \sqrt{S_x^2 + S_y^2} \tag{3.5}$$

el cálculo de la dirección tangente en cada punto, esta se puede obtener con la siguiente ecuación:

$$\Theta = \arctan\left(\frac{S_y}{S_x}\right) \tag{3.6}$$

En la figura 3.6 se puede apreciar cómo el filtro respondió a los cambios de intensidad entre pixeles claros y obscuros, obteniendo líneas más anchas cuando los cambios fueron más abruptos y líneas más delgadas cuando los cambios fueron más suaves.



(a) Imagen original (b) Aplicación del filtro sobel

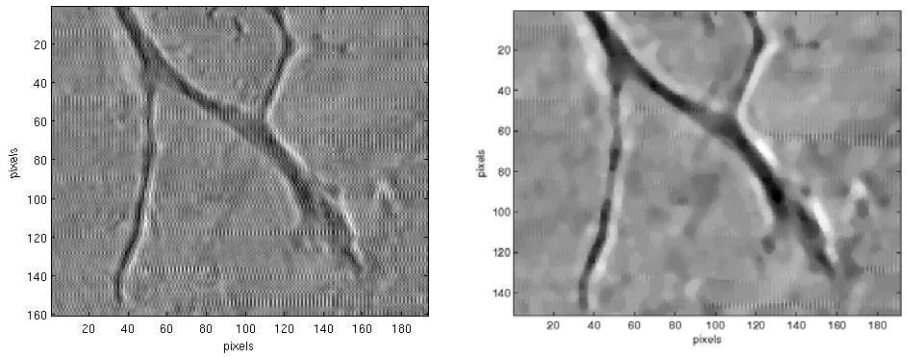
Figura 3.6: Ejemplo de aplicación del filtro Sobel

3.2.4. Mediana

El filtro *mediana* es un operador no lineal, altamente recomendado para la eliminación de ruido *sal y pimienta*. Una de las grandes ventajas de este filtro es que elimina el ruido mientras preserva la buena definición de los bordes, por lo que es una buena práctica aplicar este filtro antes de realizar algún otro proceso de alto nivel. Otras ventajas ofrecidas por este filtro, es la de homogeneizar los pixeles, ésto es que los pixeles situados en una región dada tengan intensidades similares.

El funcionamiento de este operador está basado en el método de vecindario (Sección 3.2) para el cual se toma una vecindad de $(n \times n)$ con $n = 2i + 1$ donde $i \in \mathbb{N}$. Se obtiene la mediana de las intensidades de los pixeles contenidos en la vecindad, asignándole el valor de la media al punto (x, y) en el cual está centrada la vecindad.

En la figura 3.7 se muestra la aplicación del filtro mediana. En la figura original (figura 3.7(a)) se muestra una imagen contaminada con ruido, el cual hace que entre algunos pixeles contiguos exista una gran diferencia de tonalidades. El efecto se ve corregido con la aplicación del filtro (figura 3.7(b)).



(a) Imagen original (b) Aplicación del filtro mediana

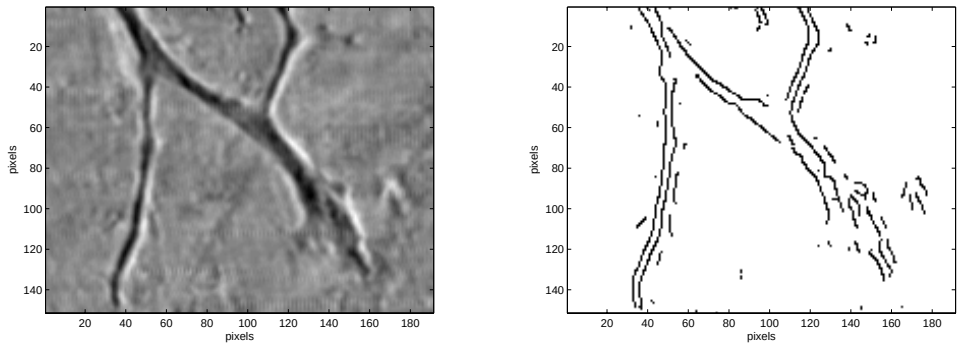
Figura 3.7: Ejemplo del filtro Mediana

3.2.5. Canny

El método Canny es una forma de obtener los bordes de los elementos de una imagen de manera óptima. Este método se basa en el cálculo de variaciones. El detector óptimo de bordes se describe como la suma de términos exponenciales, mientras que una muy buena aproximación se basa en la primera derivada de una gaussiana.

El funcionamiento del método sigue los siguientes pasos. Primero se le aplica un filtro gaussiano a la imagen (sección 3.2.1) ya que pequeñas variaciones de intensidad pueden representar bordes falsos. Después se encuentra el gradiente de intensidad de la imagen. Para ésto se utilizan 4 kerneles que detectan los bordes en las direcciones, horizontal, vertical y las dos diagonales. De los resultados de aplicar los 4 kerneles se obtienen los máximos valores para cada pixel. El último paso es trazar los bordes encontrados a lo largo de la imagen [Canny, 1986].

En la figura 3.8 se muestra el resultado de aplicar el método Canny para localizar los bordes en una imagen.



(a) Imagen original (b) Aplicación del método Canny

Figura 3.8: Ejemplo de la aplicación del método Canny de obtención de bordes

3.3. Umbralización por Otsu mejorado

Otsu es un método basado en el análisis discriminatorio el cual divide los pixeles de una imagen en escala de grises, en 2 clases o conjuntos C_1 y C_2 los cuales se pueden clasificar como objetos de interés y fondo. Para poder llevar a cabo ésta clasificación es necesario dar un valor de umbral K tal que, si $P(x, y)$ es un pixel en la imagen a segmentar, entonces C_1 y C_2 quedan definidos de la siguiente manera:

$$C_1 = \{P(x, y)|P(x, y) \geq K\} \tag{3.7}$$

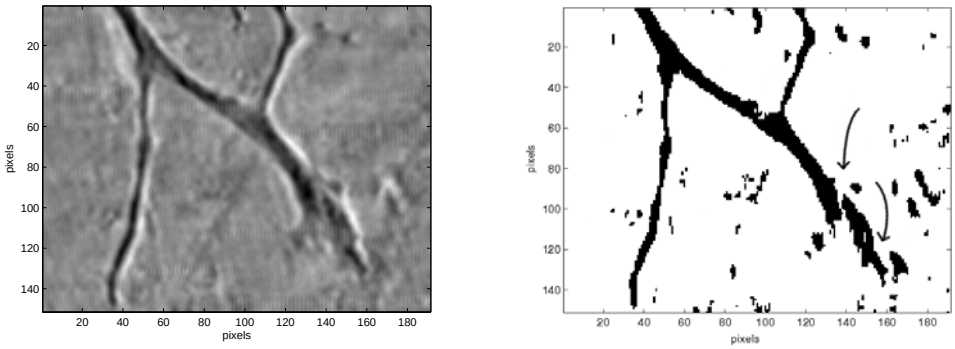
$$C_2 = \{P(x, y)|P(x, y) \leq K\} \tag{3.8}$$

donde $0 \leq K \leq 255$ para imágenes de 8 bits. Si el valor de K es tal, que el valor de la varianza σ_B de los conjuntos es el mayor entre todos los posibles valores, entonces K tiene un valor óptimo y es justamente éste valor el que obtiene el método. Mientras que el algoritmo Otsu tradicional tiene que encontrar las varianzas de los conjuntos para cada escala de gris (255

valores para imágenes de 8 bits), el algoritmo Otsu mejorado propuesto por [Li-Sheng *et al.*, 2005] solo tiene que hacer de 4 a 6 iteraciones para encontrar el valor deseado. Cabe mencionar que el valor obtenido mediante el algoritmo mejorado es una aproximación del valor óptimo.

Este método es muy usado para binarizar una imagen, ya que al separar las intensidades de los pixeles en dos conjuntos, a uno se le asignará el valor cero mientras que al otro el valor uno. El óptimo desempeño de esta función se da cuando se tiene un alto contraste entre lo que llamamos objeto de interés y el fondo, de lo contrario tendremos muchos pixeles erróneamente clasificados, esto es, pixeles que debieran de ser fondo son tomados como objeto de interés o viceversa.

En la figura 3.9(b) se muestra la clasificación de pixeles en dos conjuntos, de la imagen en la figura 3.9(a). A éste método también se le conoce como *binarización*. En este caso el conjunto de los objetos interés se le asignó el valor de cero mientras que al fondo se le asignó el valor de uno.



(a) Imagen original (b) Imagen Binarizada

Figura 3.9: Binarización de la imagen (a) con un valor obtenido mediante el método Otsu. En (b) se puede apreciar que el objeto de interés está segmentado casi en su totalidad, aunque hay algunas discontinuidades (flechas) y algunos pixeles del fondo quedaron clasificados como objetos de interés.

3.4. Normalización de la luz en secuencias

Parte importante del preproceso es tener una iluminación homogénea en toda la secuencia de imágenes. Esto permite que las operaciones sobre la secuencia no se vean afectadas por las variaciones de intensidad lumínica.

Para homogeneizar la intensidad de luz en la secuencia en este método, es necesario elegir una imagen, que servirá como referencia para igualar la intensidad de luz en todas las imágenes restantes. Esto es que todas las imágenes en la secuencia tendrán la misma intensidad de luz que la imagen que se eligió como referencia. La forma para igualar la intensidad de luz de las imágenes es normalizando el valor de intensidad de los pixeles en una imagen de tal manera que el valor de la media y la varianza sean iguales que en los pixeles de la imagen que se tomó como referencia. [Radke y Andra, 2005].

La ecuación 3.9 describe el proceso para normalizar la intensidad de luz en una secuencia de imágenes.

$$\bar{I}_i = \frac{\sigma_i}{\sigma_b}(I_i - \mu_b) + \mu_i \tag{3.9}$$

donde I_i es la *i-ésima* imagen en la secuencia y $\bar{I}_i$ es su correspondiente imagen normalizada. Las variables μ_i y σ_i son la media y la desviación estándar de I_i, mientras que μ_b y σ_b son la media y la desviación estándar de la imagen que se tomó como referencia.

3.5. Alineación

La adquisición de secuencias de imágenes de neuronas usando ciclos de adquisición combinando los planos x y y (Capítulo 2) es muy sensible a desajustes en el plano (x, y). El desajuste no tiene que ser muy grande para que las imágenes estén desalineadas; por ejemplo, con un objetivo de 40X,

basta con que la platina este corrida un par de micras con respecto a la imagen anterior para que el efecto sea visible en las secuencias.

La correcta alineación de las imágenes es indispensable en nuestra aplicación, ya que una desalineación provocará mediciones erróneas al tratar de medir cambios en el tiempo. Además para funciones como la eliminación del fondo (sección 4.2) o la corrección del campo plano (sección 3.1) es indispensable que las imágenes en una secuencia estén perfectamente alineadas.

El programa de adquisición de imágenes *ImagePro* descrito en la sección 2.1.3 cuenta con una función para alinear secuencias de imágenes que no sirvió para nuestra aplicación, ya que el método se basa en intensidades de luz y formas geométricas. Algunas razones de estas fallas son: variaciones de luz no corregidas, y la principal, la neurona crece y cambia de forma con el tiempo, lo que hace que su forma geométrica cambie y el algoritmo usado por el programa no puede compensar ese aspecto.

Thévenaz *et al.* [1998] proponen un método de alineación basado en una mejora del algoritmo Marquardt-Levenberg [Marquardt, 1963]. Éste método fue originalmente desarrollado para alinear imágenes obtenidas de PET *(Positron Emission Tomography)* y hacer reconstrucciones 3D. Este método tiene ventajas sobre otros muchos métodos de alineación, ya que es completamente automático; otros métodos necesitan que el usuario introduzca puntos de referencia en cada imagen para poder hacer la alineación. Otra ventaja es que al alinear las secuencias de imágenes se resuelve el problema de los cambios de geometría en la secuencia, ya que primero alinean dos imágenes, y a partir de la tercera imagen la alineación es con respecto a la de las dos anteriores y así sucesivamente, para todas las imágenes en la secuencia. Este algoritmo está implementado en un *plug-in* para el programa de procesamiento de imágenes ImageJ [ImageJ, 2007]. y utilizo para alinear las imágenes en el presente trabajo.

Para alinear nuestras secuencias se utilizó la transformación de "translación" en la cual una línea recta es mapeada como una línea recta con la misma orientación y conservando la misma distancia entre cualesquiera par de puntos. Se eligió este tipo de alienación ya que la única fuente de error de nuestras imágenes aparece cuando la platina del microscopio no regresa al sitio exacto. Cabe mencionar que el método implementa también otras transformaciones como rotaciones, escalamientos, etc.

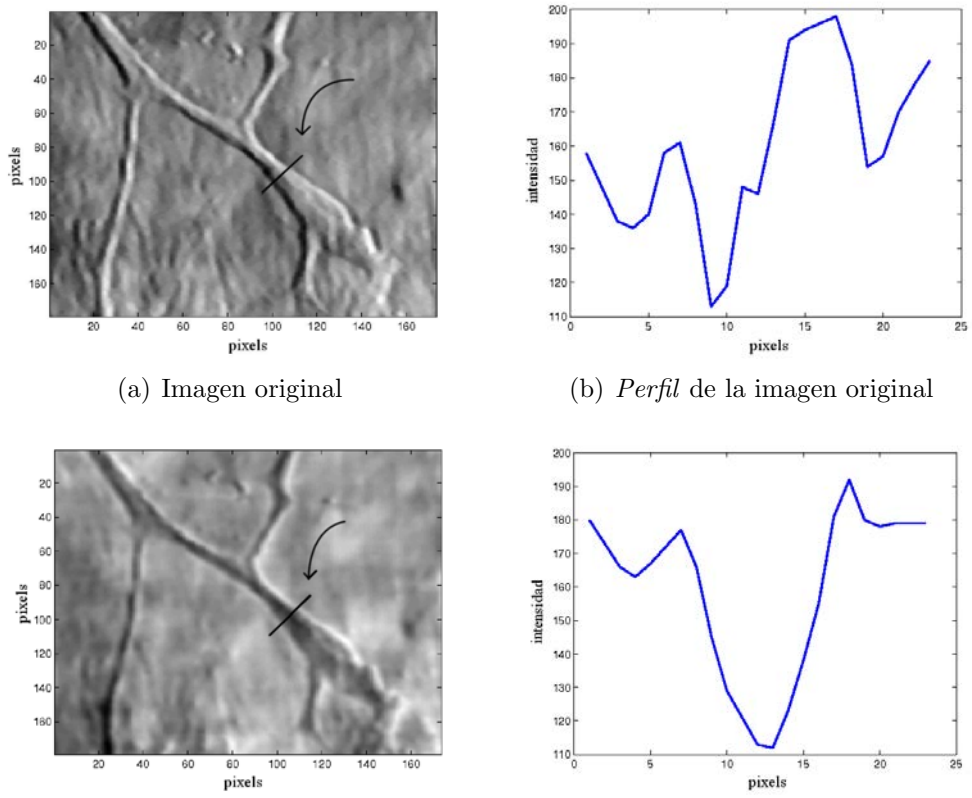
Capítulo 4

DETECCIÓN Y CARACTERIZACIÓN DE NEURITAS

Para estudiar a las neuritas en crecimiento se toman secuencias de imágenes a intervalos fijos a lo largo de un periodo en el cual la célula regenera [Capítulo 2]. Si la secuencia es reproducida a manera de video, se puede apreciar la dinámica del crecimiento de las neuritas.

4.1. Corrección DIC

Como se mencionó en la Sección 2.2, se utiliza microscopía diferencial de contraste de interferencia. Debido a que las técnicas tradicionales de procesamiento de imágenes y reconstrucción de formas tubulares [Sección 4.3] no pueden ser directamente aplicadas a las imágenes DIC, una corrección del típico efecto de luz y sombra es necesaria antes de localizar las regiones de interés.



(a) Imagen original

(b) *Perfil* de la imagen original

(c) Imagen corregida

(d) *Perfil* de la imagen corregida

Figura 4.1: Aplicación del proceso de corrección DIC, (a) imagen original con el *perfil* señalado, (b) gráfica, pixeles *vs* intensidad de un *perfil* transversal a una sección de neurita de la imagen original.(c) aplicación de la reconstrucción de fase con el mismo perfil señalado. (d) gráfica, pixeles vs intensidad de un *perfil* transversal a una sección de neurita de la imagen reconstruida.

Para explicar lo anterior tomemos un *perfil* de una imagen que corte transversalmente a una neurita, [Fig. 4.1(a)]; éste tendrá una forma similar a un ciclo de una onda senoidal con un cruce por cero al ser graficada en el plano cartesiano; debido a la sombra característica que la microscopía DIC añade al espécimen. Por lo tanto el, la forma del *perfil* transversal a la neurita (Figura 4.1(b)) hace que el método de localización [Sección 4.3] no

trabaje adecuadamente, ya que identifica crestas o valles, pero no ambos a la vez. Lo que se busca es un *perfil* como el que se muestra en la figura 4.1(d).

Para lograr que las neuritas tengan una intensidad homogénea a lo largo de un *perfil*, trabajos previos [Arnison *et al.*, 2000; Van Munster *et al.*, 1997] proponen métodos de corrección de este fenómeno. Arnison *et al.* [2000] realiza un análisis de la imagen en el espacio de la frecuencia y posteriormente aplica la transformada de Hillbert. Debido a que una de las propiedades características de esta transformada es cambiar el signo de las componentes de las frecuencias negativas, conservando a las componentes positivas, esto hace que la iluminación sea simétrica al eliminar la sombra de los objetos. [Figura 4.1(c)].

Van Munster *et al.* [1997], basa su método en suponer que la sombra característica de la microscopía DIC es luz fuera de foco. En este caso la luminosidad de la imagen es modelada usando los principios matemáticos de la microscopía DIC, con el fin de eliminar la luz fuera de foco. Este método es el aplicado en este trabajo ya que presenta mejores resultados al corregir la luz en estructuras tubulares. Por ello es descrito brevemente a continuación.

Primero la imagen original es transformada pixel por pixel a diferencias de fase mediante:

$$\Delta\varphi(x,y) = \arccos\left[2 \cdot \frac{I(x,y) - I_{min}(x,y)}{I_{max}(x,y) - I_{min}(x,y)} - 1\right] - \frac{\pi}{2} \qquad (4.1)$$

donde $I(x,y)$ es la intensidad del pixel en la imagen original en la posición (x,y), mientras que $I_{min}(x,y)$ e $I_{max}(x,y)$ son las intensidades residuales cuando hay interferencia destructiva y constructiva respectivamente. Esta información proviene de imágenes son tomadas directamente en el microscopio. La forma de adquirirlas varía dependiendo del equipo. Es importante

verificar que todos los valores $\Delta\varphi(x, y)$ estén dentro del intervalo $[-\pi/2, \pi/2]$, caracteristico de la iluminación DIC.

Conocer el corrimiento lateral que induce el sistema DIC del microscopio, es de suma importancia para éste método, y es preciso calcularlo ya que varía dependiendo del sistema DIC con el que fueron obtenidas las imágenes. Esto se hace aplicando la ecuación 4.1 a una imagen de algún objeto conocido y bien definido como puede ser una esfera de látex, de las cuales se conocen sus dimensiones y se sabe que aparecen como un círculo bien definido, cuando el microscopio está correctamente calibrado. Después de aplicarle la ecuación 4.1 a la imagen se obtienen los valores Δx y Δy, que son el corrimiento de la luz en los ejes x, y respectivamente.

Una vez estimado el corrimiento de la luz se genera una imagen de valores complejos del mismo tamaño que la original, usando los valores de corrimiento de la luz Δx y Δy de acuerdo con la siguiente ecuación:

$$G(u, v) = 2j \sin(\pi \cdot (u\Delta x + v\Delta y)) \tag{4.2}$$

donde (u, v) son las frecuencias espaciales dentro del intervalo $[-1/2, 1/2]$. De lo anterior se genera la relación *señal-ruido* mediante una aproximación de una curva gaussiana en dos dimensiones, centrada en $(u = 0, v = 0)$, mediante de la ecuación:

$$SN(u, v) = s \cdot exp[-2\pi^2\sigma^2(u^2 + v^2)] \tag{4.3}$$

donde s corresponde al máximo de la relación *señal-ruido* y σ describe la disminución hacia frecuencias espaciales más altas. Cabe mencionar que los valores usados para estos parámetros fueron obtenidos de igual manera que Van Munster *et al.* [1997], quien encontró los valores óptimos a dichos parámetros mediante de ensayo y error, mientras que los valores imaginarios de $SN(u, v)$

fueron cambiados a 0.

El siguiente paso consiste en deconvolucionar la imagen generada, con la imagen correspondiente a la distribución *señal-ruido*. Este paso se realiza mediante el método de deconvolución de "Winer" el cual minimiza el ruido introducido por el espectro del radio de distribución de la función *señal-ruido* [Jansson, 1996] descrito por la siguiente ecuación:

$$W(u,v) = \frac{G^*(u,v)}{|G(u,v)|^2 + [SN(u,v)^{-1}]} \tag{4.4}$$

donde $G*$ es el complemento conjugado de G.

Para concluir la deconvolución se transforma la imagen $\Delta\varphi(x,y)$ al espacio de las frecuencias mediante la transformada de Fourier en 2D de $\Delta\varphi(x,y)$ para así obtener $\Delta\Phi(u,v)$ y poder encontrar $\Phi(u,v)$ mediante la siguiente ecuación:

$$\Phi(u,v) = W(u,v) \cdot \Delta\Phi(u,v) \tag{4.5}$$

donde $\Phi(u,v)$ es el resultado de la deconvolución, la cual necesitamos regresar al espacio de las coordenadas de la imagen mediante la inversa de la transformada de Fourier en dos dimensiones. La imagen obtenida es el resultado final del procedimiento de corrección de fase. En la figura 4.1(c) se puede apreciar el funcionamiento correcto del método, mientras que en la figura 4.1(d) se puede observar que el *perfil* resultante cumple con las características necesarias para realizar la detección.

Los valores utilizados fueron: $\Delta x = 1{,}645$, $\Delta y = 0$ para el corrimiento del microscopio y para el filtro Winer se uso $s = 1 \times 10^{15}$ y $\sigma = 1{,}88$. Todos ellos fueron estimados empíricamante.

4.2. Eliminación del Fondo

El sustrato sobre el cual son cultivadas las células, hace que las imágenes adquiridas sean ruidosas, esto es, que las características que deseamos extraer en este caso las neuritas, se confundan con elementos del fondo que no son de nuestro interés, dificultando la separación entre la parte de interés y el fondo. El objetivo de eliminar el fondo será clasificar los pixeles de la imagen en dos clases: los que son fondo, esto es, que permanecen estáticos a lo largo de la secuencia y los de interés, como las neuritas en crecimiento.

Teniendo en cuenta que el sustrato permanece casi inmóvil a lo largo de la secuencia y que los cambios en la imagen, son principalmente el crecimiento de las neuritas, una aproximación para eliminar el fondo es utilizar métodos para detectar movimiento en secuencias de imágenes, los cuales destacan los cambios entre las imágenes de una secuencia. Esta área esta ampliamente desarrollada, debido a las aplicaciones que tiene en temas de visión computacional, por ejemplo seguridad en la vía pública, detección de automóviles, etc. Éstos sistemas ofrecen una gran variedad de algoritmos correctamente desarrollados y clasificados, de los cuales podemos escoger el que mejor se ajuste a nuestras necesidades. Trabajos correspondientes a la eliminación del fondo en secuencias de imágenes están descritos y analizados por Cheung y Kamath [2004].

Para el correcto funcionamiento de estos métodos, las imágenes de las secuencias deben estar perfectamente alineadas, esto es, que el fondo a eliminar en una imagen debe estar alineado con el fondo en cualquier otra imagen de la secuencia. También deberán evitarse las variaciones de luminosidad [1]. La mayoría de estos métodos requieren de dos pasos. El primero: consiste en generar una secuencia de imágenes que represente el fondo de cada imagen

[1] Los métodos mas avanzados tienen un buen funcionamiento aún si las intensidades de luz varían a lo largo del tiempo.

original, y segundo: comparar la secuencia fondo con la original para clasificar los pixeles que son parte del fondo.

4.2.1. Generación de secuencias fondo

La generación de las secuencias fondo, es la parte más importante en la aplicación de los métodos para su eliminación cuando se trabaja en secuencias de imágenes, ya que dependiendo de la manera como es construido el fondo será el apego del fondo generado al fondo real.

De los trabajos referidos, la mayoría se enfocan en lidiar con problemas característicos de las ciudades, y las mejoras propuestas en trabajos recientes no presentan mejores resultados para nuestra aplicación, así que se decidió implementar un algoritmo sencillo y de procesamiento ágil, basado en uno de los métodos pospuestos por Radke y Andra [2005], el cual se describe a continuación.

Método "Running Average"

Una manera simple de generar un fondo para una imagen en una secuencia, es calcular el promedio de todas las imágenes anteriores a ésta, esto es, sí I_k se refiere a la *k-ésima* imagen en una secuencia entonces, su imagen fondo correspondiente será $\frac{1}{n}\sum_{i=1}^{n} I_i$ donde n es el número total de imágenes en la secuencia. Este procedimiento hará que todos los pixeles que no cambian de intensidad a lo largo de la secuencia, se mantengan al realizar el promedio en el tiempo.

El método "Running Average" es una aproximación para obtener el promedio de una serie de imágenes [Lai y Yung, 1998]. En este método el promedio de un pixel en la posición (x, y) de la imagen I_k quedará determinado por:

$$B_k(x,y) = \frac{1}{k}\left[(k-1)\cdot B_{k-1}(x,y) + I_k(x,y)\right] \tag{4.6}$$

donde $B_0(x,y) = I_0(x,y)$. De esta manera solo se tienen que efectuar dos operaciones y guardar en memoria la imagen promedio B_{k-1}. De otra manera para obtener la imagen promedio B_k en una secuencia, se tendrían que realizar $k+1$ operaciones por cada pixel en la imagen además de tener que guardar en memoria $k-1$ imágenes. Esto trae como resultado, que obtener el promedio de una secuencia de imágenes de una manera tradicional sea lento y el consumo de memoria y tiempo de procesador de la máquina sea elevado.

4.2.2. Eliminación del fondo

Una vez obtenida la secuencia fondo, toca el turno de compararla con la secuencia original. Con esto se busca resaltar aquellas características que cambiaron de posición o forma a lo largo del tiempo. Así se puede asignar un valor el cual todos los pixeles que no lo sobrepasen pertenecerán a una clase y los que sean mayores serán clasificados en otra clase, o bien, obtener una imagen resultante de la comparación en la cual se podrán apreciar las características buscadas de mejor manera que en las secuencias originales.

Los métodos para comparar entre la secuencia original y la secuencia fondo implementados en nuestra aplicación se detallan a continuación.

Resta simple

La resta simple es el método mas sencillo y por lo tanto el de procesamiento mas rápido. Este consiste en restar uno a uno los pixeles de la imagen fondo B_k de la imagen original I_k, osea , para un pixel en la posición (x,y) el resultado de la resta es definido por:

$$\hat{I}_k(x,y) = I_k(x,y) - B_k(x,y) \tag{4.7}$$

donde n el número total de imágenes en la secuencia, y $1 \leq k \leq n$. Dada su simpleza, el método presenta muchos defectos en el resultado final, como pueden ser la sensibilidad a los cambios de iluminación entre las imágenes $I_k(x, y)$ y $B_k(x, y)$; si el valor del pixel resultante tiene valores negativos la información se pierde. Este procedimiento es efectivo cuando las imágenes son muy limpias, homogéneas entre si, y los cambios en las imágenes de la secuencia original son notorios (Figura 4.2(c)).

Resta absoluta

La resta absoluta, es otro método sencillo y rápido de ejecutar, ya que sólo involucra el valor absoluto de la resta. Entonces para un pixel (x, y)

$$\hat{I}_k(x, y) = |I_k(x, y) - B_k(x, y)| \tag{4.8}$$

Algunas ventajas del método es que conserva la información aún si el valor obtenido para el pixel es negativo. Las magnitudes entre los pixeles resultantes son más distantes entre los pixeles fondo y los que no lo son, lo que facilita encontrar un valor para separar los pixeles de la imagen restada en dos clases, o bien si se utiliza un método automático de clasificación como el Otsu [Sección 3.3] tendrá un mejor desempeño. A pesar de las mejoras con respecto a la resta simple, la resta absoluta es muy sensible a los cambios de iluminación [Figura 4.2(d)].

Resta normalizada

Otro método para substraer el fondo de la secuencia original, es la resta normalizada, también conocida como diferencia relativa [Fuentes y Velastin, 2001]. Con este método se puede enfatizar el contraste en áreas obscuras de la imagen, por ejemplo si la imagen contiene zonas de mucha luz. Con los métodos anteriores [Secciones 4.2.2 o 4.2.2] sería casi imposible encontrar un

valor que separe óptimamente los pixeles en dos clases, mientras que este método permite homogeneizar la imagen, para obtener un único valor que la divida en dos clases. Así métodos como el Otsu [Sección 3.3] trabajan mejor. En este método la resta normalizada para un pixel en la posición (x, y) queda definido por:

$$\hat{I}_k(x, y) = \frac{|I_k(x, y) - B_k(x, y)|}{B_k(x, y)} \tag{4.9}$$

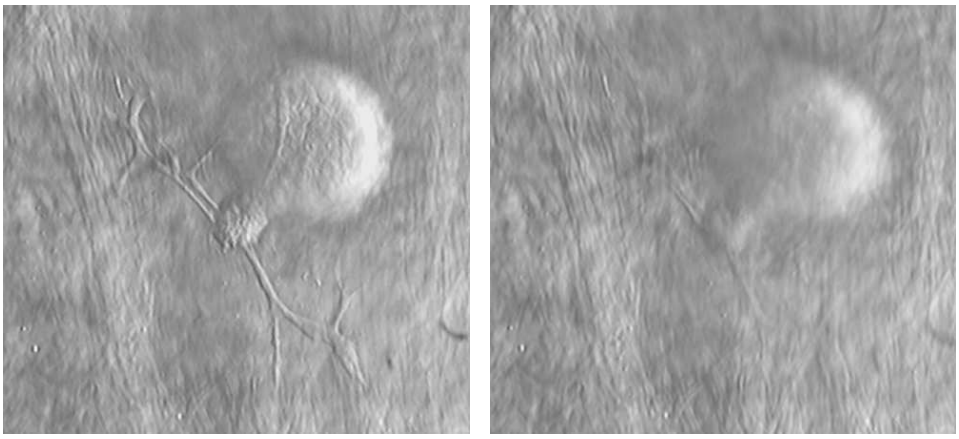
Esta técnica puede no funcionar si la imagen es muy luminosa (Figura 4.2(e)).

Diferencia estadística normalizada

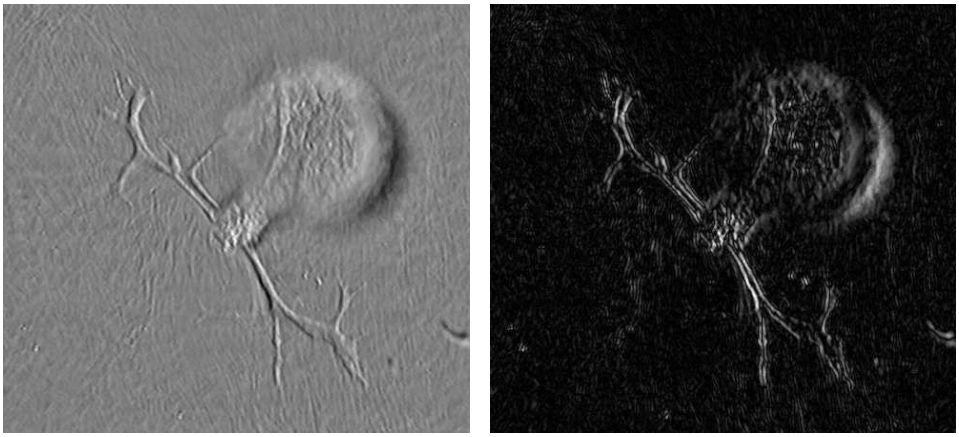
Este método es de los más usados para comparar la secuencia original y la secuencia fondo ya que es sumamente robusto ante cambios de iluminación, o bien, cuando las imágenes contienen áreas obscuras que pueden ser causadas por sombras u otros artefactos. Este método se basa en igualar la media de la resta entre las secuencias (original y fondo) y la imagen resultante. Así, las diferencias de intensidad no marcan cambios drásticos en la imagen final [Cheung y Kamath, 2005]. Esto es que para un pixel en la posición (x, y) la comparación está dada por:

$$\hat{I}_k(x, y) = \frac{D(x, y) - \mu_d}{\sigma_d} \tag{4.10}$$

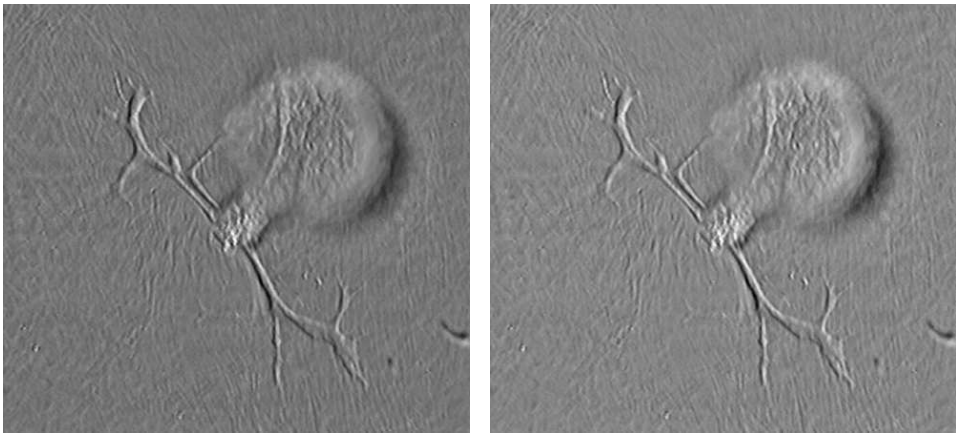
donde $D(x, y) = I_k(x, y) - B_k(x, y)$, μ_d y σ_d son la media y la desviación estándar respectivamente de $D(x, y)$. La utilización de este método debe de considerarse con cuidado ya que encontrar la media de la imagen y la desviación estándar de la imagen resta involucra un gran número de operaciones. En la Figura 4.2(f) podemos observar el resultado al aplicar éste método.



(a) Imagen original (b) Imagen fondo



(c) Resta simple de (a)-(b) (d) Resta absoluta de (a)-(b)



(e) Resta normalizada de (a)-(b) (f) Resta estadística normalizada (a)-(b)

Figura 4.2: Resultados de los diferentes tipos de comparación entre la secuencia original y la secuencia fondo, para imágenes de $495(x) \times 450(y)$ pixeles.

4.3. Obtención de ridges

La detección de formas tubulares o elongadas ha sido recientemente estudiada debido a sus diferentes aplicaciones médicas, como la detección y segmentación de vasos sanguíneos para su modelado, en 2D o 3D [Aylward, 2002; Sofka, 2006; Martinez-Perez y Espinosa-Romero, 2004]. El modelado y el conocimiento exacto de características especiales de éstas estructuras como longitud, grosor, tortuosidad, entre otras, sirve a los médicos especialistas para tener una mejor visualización y entendimiento de los vasos sanguíneos y con ella elaborar un mejor diagnóstico en algunos tipos de padecimientos [Martinez-Perez *et al.*, 2007]. Este tipo de mediciones en vasos sanguíneos se puede extrapolar a otras estructuras tubulares como las neuritas o axones de neuronas en crecimiento. Meijering *et al.* [2003] utilizó la idea de extracción de vasos sanguíneos para la medición de axones de una manera semiautomática en imágenes obtenidas por medio de microscopía confocal.

Los trabajos mencionados basan la extracción de las estructuras tubulares mediante la detección y seguimiento de *ridges*, ya que éstos coinciden con la linea central a lo largo de estas estructuras. Varias definiciones de *ridge* han sido propuestas dependiendo del contexto o aplicación en que se utilice. La definición que más se apega a nuestro trabajo es: un *ridge* es un punto en la imagen cuya intensidad es un máximo o un mínimo local en la dirección de la máxima curvatura o gradiente de la superficie (máxima concavidad) [Lindeberg, 1996].

El objetivo en la detección de ridges es encontrar el mayor eje de simetría en objetos de forma tubular, en contraste con los métodos de detección de bordes que se enfocan en resaltar el contorno de los objetos. Algunos de éstos métodos como el *Sobel* y *Canny* (ver secciones: 3.2.3, 3.2.5 y figura 4.3) fueron probados en nuestro estudio dando como resultado, detecciones muy deficientesde de los objetos de interés, con gran número de discontinuidades y fueron muy susceptibles al ruido de la imagen, como puede verse en la Figura 4.3.

Es importante destacar que para el tipo de imágenes obtenidas mediante óptica DIC, la segmentación de neuritas por medio de la aproximación de *ridges* es infructuosa si las imágenes no son previamente procesadas con el fin de corregir el desplazamiento de fase generado por el DIC (ver secciones: 2.2 y 4.1).

(a) Imagen Original (b) Aplicación del filtro *Sobel* a (a) (c) Aplicación del filtro *Canny* a (a)

Figura 4.3: Comparación entre los diferentes métodos probados para la detección de neuritas, los cuales dieron malos resultados.

La extracción de puntos centrales en las neuritas o *ridges* puede llevarse a cabo aplicando filtros diseñados para resaltar elementos tubulares. Estos filtros están basados en una combinación de los eigenvalores de la matriz Hessiana cuyos elementos son las derivadas parciales de segundo orden de las intensidades de cada uno de los pixeles en la imagen. De ésta manera la matriz Hessiana describe las variaciones de intensidad alrededor de cada punto en la imagen. Las estructuras de tipo curvilíneo son separadas de otras, como pueden ser puntos o estructuras homogéneas. Los eigenvectores correspondientes proveen la dirección del eje sobre las neuritas.

4.3.1. Representación Multiescala

Comenzaremos por definir el espacio de las escalas en el cual el método queda definido. El definir un espacio de escalas se traduce en detectar neuritas

con diferentes diámetros en una misma imagen, si el método de obtención de *ridges* se implementara para una sola escala, este solo detectaría un tipo de neurita, discriminando a otras que muy probablemente fueran también de nuestro interés. Lindeberg [1996] provee una definición para el espacio de las escalas $L(x, y; s)$ de la siguiente manera:

$$L(x, y; s) = G(x, y; s) \otimes I(x, y) \tag{4.11}$$

donde $I(x, y)$ es un pixel de la imagen en la posición (x, y) y $G(x, y; s)$ es un filtro Gaussiano con varianza s^2 definido como:

$$G(x, y; s) = \frac{1}{2\pi s^2} \exp^{-\frac{x^2+y^2}{2s^2}} \tag{4.12}$$

Teniendo claro el espacio de las escalas sobre cual se está trabajando, al aplicar a la imagen un filtro gaussiano con una escala s, las estructuras con diámetros menores a s serán descartadas por el método, siendo ésta una forma de aislar estructuras con diferentes tamaños [Martinez-Perez *et al.*, 2007].

4.3.2. Extracción de la línea central o *ridge*

La información que provee la segunda derivada de una imagen, permite distinguir entre crestas y valles en un *perfil* de la imagen. La matriz Hessiana H captura la información de la segunda derivada en cada punto de la imagen. Con esta información podemos calcular la dirección de la máxima curvatura, dada por el eigenvector $\vec{v}$ correspondiente al mayor eigenvalor λ, ambos obtenidos de la matriz de segundas derivadas en cada punto de la imagen (ver Figura 4.4). El signo de λ determina que un máximo ($\lambda > 0$) o mínimo ($\lambda < 0$) local ha sido encontrado [Eberly, 1996]. La matriz de segundas derivadas o matriz Hessiana queda definida de la siguiente manera:

$$H(x,y) = \begin{bmatrix} \frac{\partial^2 I}{\partial x^2} & \frac{\partial^2 I}{\partial x \partial y} \\ \frac{\partial^2 I}{\partial y \partial x} & \frac{\partial^2 I}{\partial y^2} \end{bmatrix} = \begin{bmatrix} L_{xx} & L_{xy} \\ L_{yx} & L_{yy} \end{bmatrix} \tag{4.13}$$

donde $L_{xy} = L_{yx}$. Esta igualdad implica que la matriz Hessiana es simétrica con respecto a los eigenvalores reales y ortogonal a los eigenvectores, esto la hace invariante a rotaciones. Una aproximación al cálculo de las derivadas parciales de segundo orden necesarias para obtener la matriz Hessiana, es aplicar un filtro gaussiano derivativo a la imagen I. La finalidad de realizar una aproximación en lugar de obtener el valor exacto es reducir el costo de procesamiento de la imagen.

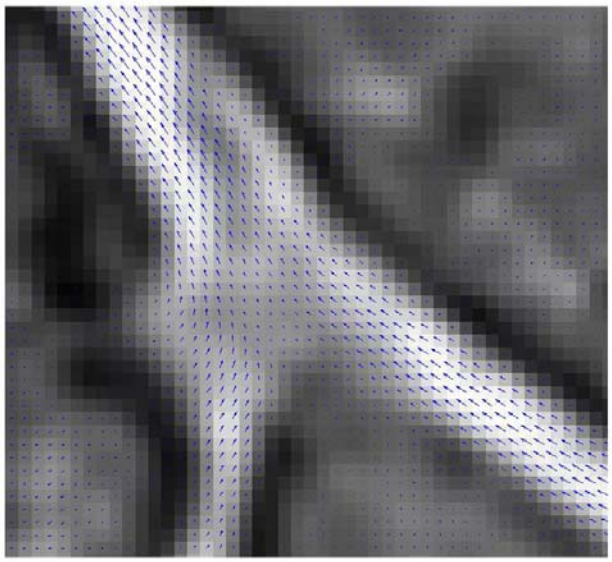
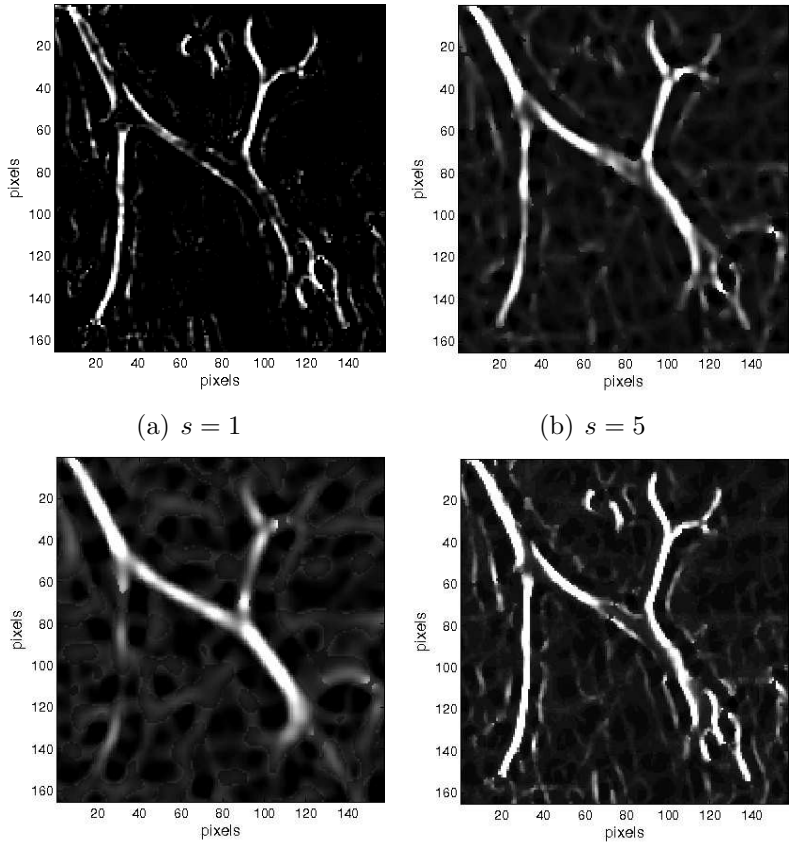


Figura 4.4: Se muestra el eigenvector $\vec{v}(x,y)$ asociado a cada uno de los pixeles en la imagen. También se puede apreciar que la dirección de los vectores es tangente a las neuritas.

Una vez obtenida la matriz Hessiana faltará obtener el eigenvalor y su eigenvector asociado para cada punto en la imagen y verificar en cada punto cual de ellos cumple con la condición de ser un punto *ridge*.

El cálculo de los eigenvectores y eigenvalores de la matriz Hessiana también conocido como eigen-descomposición se lleva a cabo de la siguiente manera: sea H la matriz Hessiana y λ sus eigenvalores asociados. Si resolvemos para λ la expresión $det(H - \lambda I) = 0$ encontraremos el conjunto de eigenvalores de H, donde I es la matriz identidad. Una vez obtenidos los eigenvalores, los correspondientes eigenvectores $\vec{v}$ asociados a λ se obtienen al resolver $(H - \lambda I)v = 0$ para v. Este método también es conocido como encontrar el espacio nulo. Una vez encontrados los eigenvectores éstos representan la dirección de la máxima curvatura perpendicular al *ridge*. Para obtener la máxima curvatura en la dirección tangente, los eigenvectores se deben rotar 90 grados, cuidando de ser consistentes para todos ellos. En la figura 4.4 se puede apreciar el resultado del cálculo de la eigen-descomposición para cada punto o pixel de un segmento de imagen conteniendo una neurita (en blanco). En ésta también se puede apreciar cómo en el centro la magnitud de los eigenvectores es máxima en la dirección tangente.

Para encontrar el eigenvalor máximo a lo largo de las escalas seleccionadas, es necesario generar la matriz Hessiana y realizar la eigen-descomposición para cada escala seleccionada. Una vez obtenidos todos los posibles eigenvalores para un punto en la imagen, éstos son comparados, y se conserva únicamente el de mayor magnitud a lo largo de las escalas, asociándole su correspondiente eigenvector. En la figura 4.5 se puede apreciar la diferencia de calcular los eigenvalores para diferentes escalas, conservando únicamente los máximos eigenvalores. A lo largo de las escalas se puede hacer un seguimiento de éstos en la dirección de su correspondiente eigenvector y así obtener la línea central a lo largo de las neuritas.



(a) $s = 1$

(b) $s = 5$

(c) $s = 12$

(d) Máximo eigenvalor a lo largo de las escalas (1, 12)

Figura 4.5: Figura comparativa de los gráficos de los eigenvalores encontrados para diferentes valores de s indicado debajo de cada gráfica(a), (b) y (c). En (d) apreciamos el resultado de conservar sólo el máximo eigenvalor a lo largo de las escalas (1-12 pixeles). Las imágenes están mostradas en en escala [0, 255] para tener mejor resolución

Para comenzar el seguimiento o *tracking* de los *ridges* y así extraer la linea central de la neurita seleccionada, es necesario contar con los eigenvalores $\lambda(x, y)$ y sus eigenvectores $\vec{v}(x, y)$ asociados para cada punto de la imagen (ver Figura 4.4). Así mismo se necesitará un punto semilla $p(x, y)$

definido por el usuario sobre la neurita de interés y una condición de paro, que en nuestro caso será un valor asignado para cada imagen. Este valor dependerá del contraste entre las estructuras a segmentar y el fondo de la imagen.

Dado el punto semilla $p(x, y)$ realizamos una búsqueda lineal dentro de una ventana de tamaño predefinido centrada en $p(x, y)$ para obtener el punto $x_1(x, y)$, el cual es máximo local en la dirección del gradiente (ver Figura 4.6). Una vez obtenido $x_1(x, y)$ determinamos la dirección de su eigenvector y dependiendo de ella, se escogerá la posición del siguiente posible punto ridge $x_2(x, y)$. Si en esa posición el eigenvalor no cumple la condición de paro, se añade al ridge, y se analiza la dirección de su eigenvector para obtener la siguiente posición del posible punto *ridge* $x_3(x, y)$, y así sucesivamente, hasta que en algún punto el eigenvalor cumpla la condición de paro. Entonces todos los puntos previamente seleccionados $x_i(x, y), x_{i-1}(x, y), \ldots, x_2(x, y), x_1(x, y)$ corresponderán a la linea central de la neurita seleccionada.

La manera de elegir la posición del posible nuevo punto ridge $x_{i+1}(x, y)$ dependerá de la dirección del eigenvector $\vec{v_i}$ actual. Esta dirección determinará 3 de las 8 localidades adyacentes a la posición de $\vec{v_i}$ que servirán para buscar el siguiente máximo local en la dirección del gradiente. Las localidades quedan determinadas por el ángulo de $\vec{v_i}$. En la tabla 4.1 están definidas las posiciones de búsqueda, dependiendo del ángulo dado. Por ejemplo, si $\vec{v_i}$ está en la posición (x, y) y tiene un ángulo de 46 grados, entonces según la tabla 4.1, las posiciones adyacentes en las cuales buscar el siguiente punto *ridge* serán: $(x+1, y), (x+1, y-1), (x, y-1)$, ya que la dirección de 46 grados esta dentro del intervalo de $(5, 85]$ grados.

Algunos problemas con la imagen, o bien problemas intrínsecos del crecimiento neuronal, hacen que regiones de neuritas que el usuario sabe que

Dirección del Eigenvector	**Localidades de busqueda**
$(355, 5]$	$(x+1, y+1)$ $(x+1, y)$ $(x+1, y-1)$
$(5, 85]$	$(x+1, y)$ $(x+1, y-1)$ $(x, y-1)$
$(85, 95]$	$(x+1, y-1)$ $(x, y-1)$ $(x-1, y-1)$
$(95, 175]$	$(x, y-1)$ $(x-1, y-1)$ $(x-1, y)$
$(175, 185]$	$(x-1, y-1)$ $(x-1, y)$ $(x-1, y+1)$
$(185, 265]$	$(x-1, y)$ $(x-1, y+1)$ $(x, y+1)$
$(265, 275]$	$(x-1, y+1)$ $(x, y+1)$ $(x+1, y+1)$
$(275, 355]$	$(x, y+1)$ $(x+1, y+1)$ $(x+1, y)$

Cuadro 4.1: Posiciones en las que se buscara el siguiente eigenvector máximo dependiendo de la dirección en grados del eigenvector $\vec{v}(x, y)$

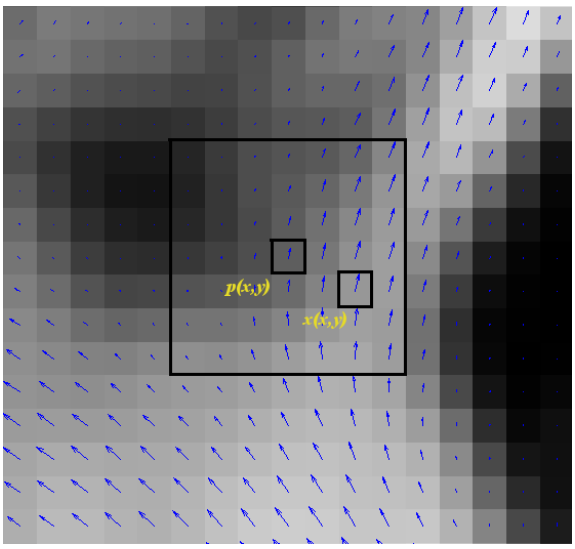


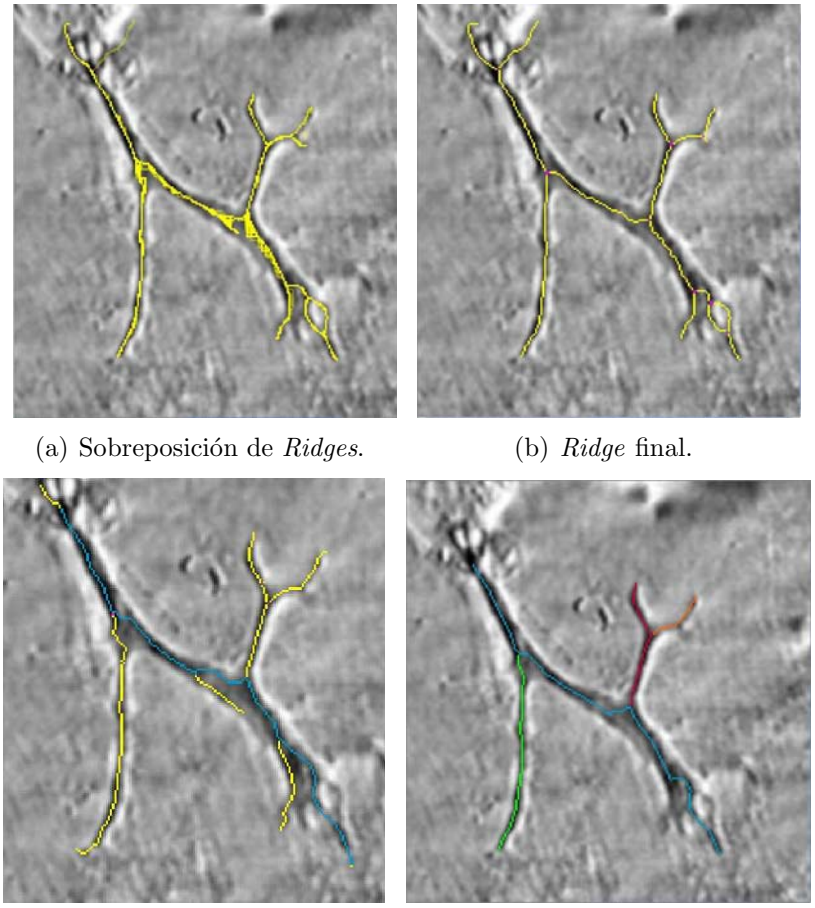
Figura 4.6: Muestra al punto semilla $p(x, y)$ dado por el usuario, y el punto $x_1(x, y)$ que es máximo local en la dirección gradiente dentro de una ventana de tamaño 7.

existen, no cumpla n la condición para pertenecer al *ridge*. En estos casos el usuario puede marcar manualmente las secciones que el método no pudo obtener.

4.4. Extracción de neuritas

Una vez que se ha obtenido la linea central de las neuritas de interés [Sección 4.3.2], son necesarios dos puntos definidos por el usuario, que determinan el inicio y el final de cada neurita, para así poder asociar un segmento de *ridge* a una neurita en específico y así concluir la segmentación. Las condiciones que deben cumplir estos puntos son: estar sobre el *ridge* y que exista un camino continuo que los una. Otro requerimiento es que el *ridge* sobre el cual se vaya a seleccionar la neurita debe tener un grosor de un pixel. Esto se logra aplicando un filtro de *adelgazamiento* o *thining* antes de seleccionar el segmento deseado. En la figura 4.7 se muestra éste procedimiento, primero

encontrando la linea central para las neuritas de interés (ver Figura 4.7(a)). Después la aplicación del filtro de adelgazamiento al ridge, y por último la asociación de cada neurita a diferentes secciones del ridge (ver Figuras 4.7(c) y 4.7(d)).



(a) Sobreposición de *Ridges*.

(b) *Ridge* final.

(c) Selección de la neurita primaria

(d) Neuritas de interés

Figura 4.7: Proceso final de extracción de neuritas una vez obtenidos los *ridges* necesarios. (a) *Ridges* encontrados tras utilizar un gran número de puntos semilla. (b) *Ridge* final después de haber aplicado un filtro de adelgazamiento al *ridge* de la Figura (a). (c) Neurita primaria seleccionada (color obscuro) sobre el *ridge* (color claro) mostrado en la Figura (b). (c) Asociación de un segmento del *ridge* a cada una de las neuritas de interés

Capítulo 5

DETECCIÓN EN SECUENCIAS Y CUANTIFICACIÓN

La detección y caracterización semiautomáticas de neuronas en regeneración registradas mediante secuencias de imágenes es el enfoque principal de este trabajo. De los trabajos revisados tomados como referencia para la detección de neuritas en el capítulo 4, todos realizan una detección de los objetos de interés para una sola imagen en particular. En nuestro caso tenemos imágenes del mismo evento adquiridas en diferentes tiempos. Al-Kofahi *et al.* [2006] presentan hasta el momento el único trabajo publicado para detectar de manera semiautomática los axones de neuronas en secuencias de imágenes. En el trabajo de Kofahi se utilizan imágenes de fluorescencia adquiridas por medio de un microscopio confocal.

Nuestro método de detección y caracterización de neuritas en secuencias de imágenes, consta principalmente de dos pasos: el primero consiste en caracterizar las neuritas [Capítulo 4], y el segundo consiste en propagar la detección a las siguientes imágenes y la medición de las partes de interés

en cada una de ellas. Logramos que este procedimiento se haga de manera automática en el mejor de los casos. ¿Qué tan automático será el proceso? Dependerá de la calidad de las imágenes, considerando que las imágenes de mejor calidad son aquellas que cuentan con buen contraste entre las neuritas y el fondo, que contienen pocas cantidades de ruido, y patrones de crecimiento simples. Aún con imágenes que no presentan estas características el método funciona de manera aceptable, pero requerirá de mayor intervención por parte del usuario. De cualquier manera el método requiere de una inspección detallada de cada secuencia de imágenes, para que a partir de ella se haga la toma de decisiones iniciales.

5.1. Correlación de neuritas en el tiempo

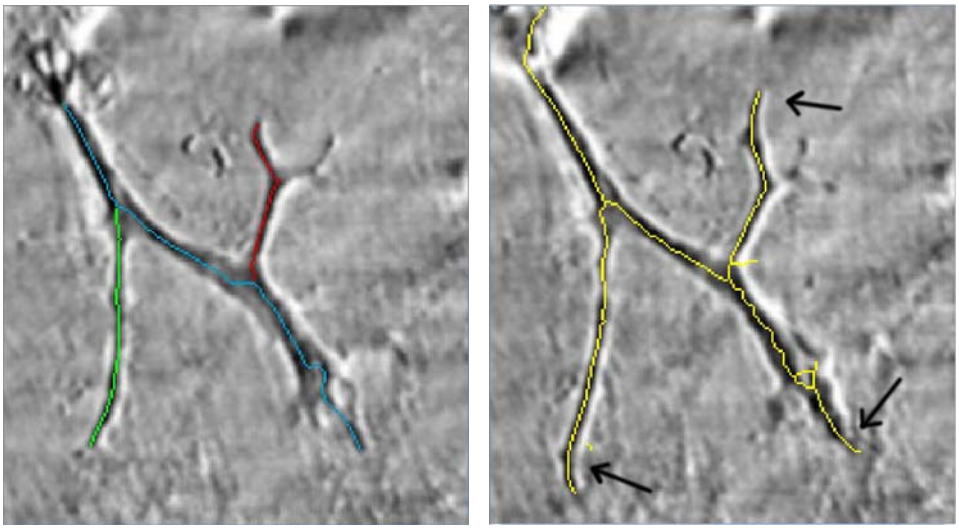
Consideremos que una neurita N_k^j pertenece al conjunto de segmentos o neuritas detectadas en la imagen I_k donde $1 < k \leq n$ con n el número total de imágenes en la secuencia y $1 < j \leq m$, donde m es el número de total de neuritas caracterizadas en una imagen. El siguiente objetivo será identificar el cambio de N_k^j en la imagen I_{k+1} y en la imagen I_{k-1} (en caso de iniciar la segmentación en una imagen intermedia en la secuencia). Para ello necesitaremos algunos datos iniciales como son: a) los pixeles que conforman a N_k^j que deberán de estar expresados como un arreglo de coordenadas (x, y), donde cada coordenada representa la posición de un punto de la neurita. Este arreglo de puntos debe de cumplir con la condición de ser continuo. b) Además de la información de la neurita, se deberán conocer los datos que fueron utilizados para obtener dicha detección como son: el rango de escalas sobre las cuales se obtuvieron los eigenvalores y la condición de paro al extraer la línea central. Conociendo estos datos, el primer paso es encontrar la matriz Hessiana y los correspondientes eigenvalores y eigenvectores para

cada punto en la imagen[1] I_{k+1} en la secuencia [Sección 4.3.2]. Segundo, las posiciones de algunos puntos de N_k^j se utilizarán como semillas para extraer la línea central o *ridge* de la neurita N_{k+1}^j, como se explicó en la sección 4.3. Por último se le asociará a N_{k+1}^j un segmento del *ridge* y se relacionará dicho segmento con N_k^j, ya que N_k^j y N_{k+1}^j son la misma neurita pero segmentada en tiempos diferentes (ver Figura 5.1).

5.1.1. Extracción de la línea central en secuencias

Una vez obtenidos los eigenvalores y eigenvectores de la imagen I_{k+1} se procederá a extraer el *ridge* que mas se asemeje al *ridge* obtenido para la neurita N_k^j. Esto implica que la detección primaria para la imagen I_k debe ser lo más precisa posible, ya que de ésta se desprende todo el análisis posterior. El nuevo *ridge* se obtendrá según lo descrito en la Sección 4.3. Como se menciono en dicha sección, son necesarios uno o varios puntos semilla $p(x, y)$ dados por el usuario. Para este paso, los puntos semilla son generados automáticamente y éstos serán un subconjunto de puntos de N_k^j. La cantidad de elementos del subconjunto, dependerá de la calidad de la imagen. Si se eligen muchos puntos probablemente se seleccionarán en repetidas ocasiones puntos ya seleccionados en el *ridge*; por el contrario si se eligen muy pocos, probablemente algunas partes de neurita que están sobre el *ridge* no serán seleccionadas. En la práctica se ha visto que tomar un décimo de los puntos totales de N_k^j da buenos resultados sin redundar en puntos previamente seleccionados. Es importante destacar que los puntos semilla seleccionados deben de estar uniformemente distribuidos a lo largo de N_k^j. En la Figura 5.1 se muestra el resultado de la obtención automática del *ridge* para la imagen I_{k+1} dadas 3 neuritas seleccionadas en la imagen I_k.

[1]Sin perdida de generalidad nos referiremos a la imagen I_{k+1} o I_{k-1} como la imagen I_{k+1} sin hacer distinción entre ellas.



(a) Neuritas seleccionadas. Imagen k

(b) Obtención automática del *ridge*. Imagen $k+1$

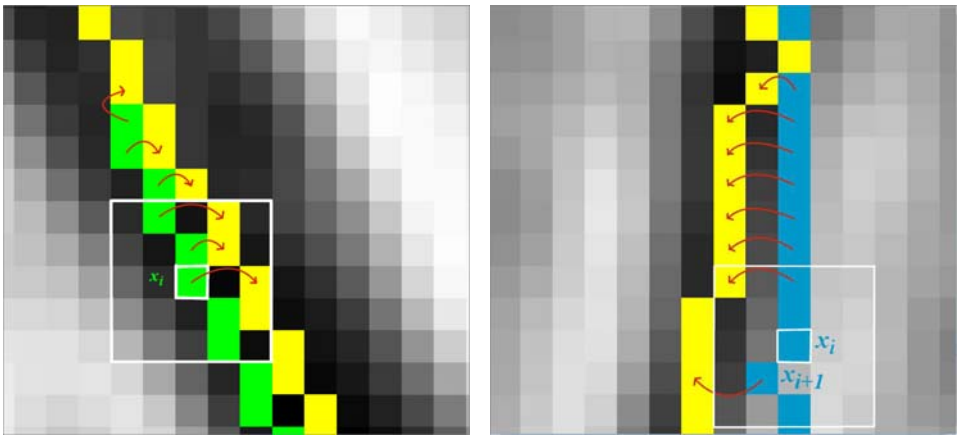
Figura 5.1: Obtención automática de la línea central para la imagen I_{k+1} dadas las neuritas seleccionadas en la imagen I_k . (a) Imagen I_k con tres neuritas seleccionadas (N_k^1 azul, N_k^2 verde, N_k^3 rojo). (b) Resultado de obtener automáticamente el *ridge* en la imagen I_{k+1}, correspondiente a las neuritas seleccionadas en (a). Las flechas indican en las partes en las que hubo un cambio considerable, sinembargo el *ridge* se detecto correctamente.

5.1.2. Correlación

Se entiende como correlación a la asociación de un segmento del *ridge* obtenido para I_{k+1} con la neurita previamente extraída N_k^j. A este nuevo segmento se le llamará N_{k+1}^j, ya que corresponde a la neurita N_k^j pero en la imagen I_{k+1}.

Para este proceso hay que tener en cuenta el crecimiento de la célula. En la nueva imagen, la neurita N_k^j pudo haber crecido, se pudo haber retraído, movido, cambiado de forma o bien una combinación de las opciones anteriores. Dado que las imágenes en la secuencia fueron tomadas en intervalos de tiempo cortos, suponemos que el cambio que sufrió N_{k+1}^j con respecto a la que sería N_k^j no es muy drástico y que en el proceso anterior [Sección 5.1.1] se

incluyeron estos cambios al obtener el *ridge*. Ahora para cada punto $x_i \in N_k^j$ se fija una ventana de un tamaño predefinido y se busca el punto del nuevo *ridge* más cercano a x_i dentro de esa ventana. El punto mas cercano en el nuevo *ridge* se liga con x_i y se agrega al conjunto de puntos que conformarán a N_{k+1}^j (ver Figura 5.2(a)). En caso de que el punto más cercano ya este ligado a otro punto de N_k^j, éste se ligara con el punto más cercano que este libre dentro de la ventana. En caso de que no exista un punto con el cual ligar a x_i, éste quedará libre y se continuará con el siguiente punto $x_{i+1} \in N_k^j$ y así sucesivamente con todos los puntos de N_k^j (ver Figura 5.2(b)).



(a) Asociación del nuevo *ridge*. (b) Asociación del nuevo *ridge*.

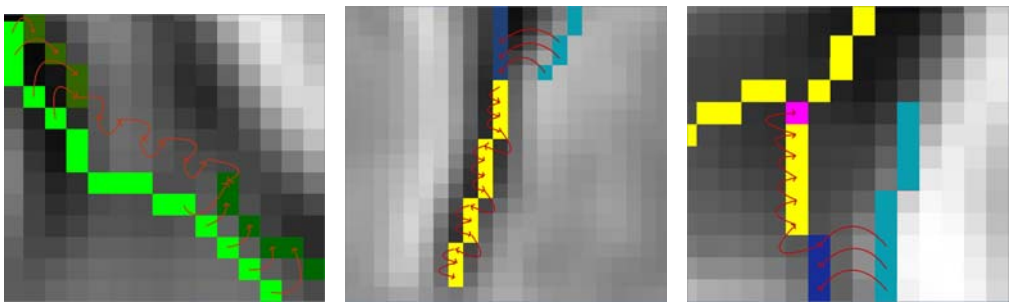
Figura 5.2: Muestra el primer paso de correlación de la neurita N_k^j de la imagen I_k con la neurita N_{k+1}^j en la imagen I_{k+1}. (a) Asociación de cada uno de los puntos x_i de la neurita N_k^j (verde) con el *ridge* (amarillo) obtenido para la imagen I_{k+1} con una ventana de tamaño 5×5. (b) Mismo proceso que en (a), pero en este caso para el punto x_i no hubo ningún punto del *ridge* que estuviera dentro de la ventana.

Al finalizar el proceso anterior se obtienen algunos puntos correspondientes a la nueva neurita N_{k+1}^j. Como no hay seguridad que el nuevo arreglo de puntos sea continuo, lo siguiente será buscar y reparar las discontinuidades. Para ésto se hace un seguimiento de los puntos recién agregados de N_{k+1}^j, buscando cual es punto final. Si un punto final es encontrado y aún quedan

puntos en el arreglo sin verificar, este punto es unido con el siguiente punto por medio de una línea recta, agregando los puntos de esa línea al arreglo de N_{k+1}^j. Así se continua hasta terminar de recorrer todos los puntos de N_{k+1}^j. De esta manera cubrimos todos los cambios que pudo haber sufrido N_k^j exceptuando los asociados al crecimiento. En la Figura 5.3(a) se muestran los pixeles ligados entre N_k^j (verde claro) y N_{k+1}^j (verde obscuro). Se muestra una discontinuidad de los puntos de N_{k+1}^j, y el camino marcado con flechas rojas contiene los puntos agregados que corrigen esa discontinuidad.

Si la neurita N_k^j creció entre las imágenes I_k y I_{k+1} probablemente los puntos correspondientes al crecimiento no estarán incluidos en N_{k+1}^j pero si estén en el *ridge*, esto es, el punto final de N_{k+1}^j debe de estar sobre el *ridge* así que seguimos éste hasta encontrar un punto final, añadiendo todos los puntos recorridos a N_{k+1}^j. En la Figura 5.3(c) se muestran los pixeles ligados entre N_k^j (azul claro) y N_{k+1}^j (azul obscuro), así como los puntos que están sobre el *ridge* (amarillo) que fueron agregados debido al crecimiento de la neurita.

Para la elección de los puntos iniciales de cada neurita el usuario deberá de especificar si el inicio de N_{k+1}^j se asociará con el punto más cercano sobre el nuevo *ridge* al punto inicial $x_1 \in N_k^j$, o bien si el punto inicial de N_{k+1}^j se ajustará a la intersección de *ridges* más cercana a $x_1 \in N_k^j$. Para el segundo caso, si la intersección más cercana a x_1 ya está incluida en los puntos que conforman a N_{k+1}^j, los puntos anteriores al punto intersección se eliminarán y la intersección será el punto inicial. Si la intersección no está en los puntos de N_{k+1}^j, entonces se seguirá el ridge hasta encontrar la intersección, agregando todos los puntos recorridos a N_{k+1}^j. La Figura 5.3(c) muestra los puntos agregados a N_{k+1}^j (azul obscuro) cuando se seleccionó ajustar el punto inicial a la intersección más cercana (morado) y ésta no estaba contenida en los puntos que forman a N_{k+1}^j. También se muestran los puntos ligados entre N_k^j y N_{k+1}^j para dar referencia de por qué esos puntos del *ridge* (amarillo) no se agregaron a los puntos de N_{k+1}^j.



(a) Corrección de discontinuidades (b) Agregación de puntos faltantes (c) Agregación de puntos faltantes al inicio

Figura 5.3: Corrección de diferentes problemas en la correlación entre N_k^j y N_{k+1}^j. (a) Corrección de discontinuidades, en verde claro N_k^j y en verde obscuro los puntos asociados que contienen una discontinuidad, y marcado con flechas el camino de puntos que corrige la discontinuidad. (b) Puntos agregados del *ridge* (amarillo) ante el crecimiento de N_k^j (verde claro) y N_{k+1}^j (verde obscuro). (c) Puntos agregados si se escogió como punto inicial una intersección y ésta no se encontraba ya en los puntos de N_{k+1}^j (azul obscuro).

Si la neurita segmentada N_{k+1}^j no cumple con los requerimientos del usuario, éste siempre podrá modificar la segmentación de la neurita manualmente siguiendo los pasos descritos en el Capítulo 4.

5.2. Cuantificación

La cuantificación, consiste en dar una medida a los segmentos de línea obtenidos.

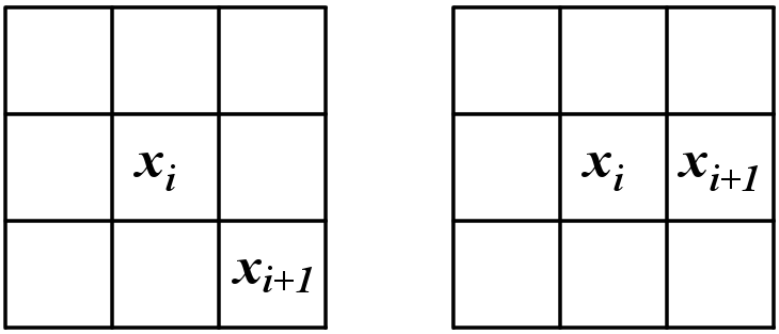
Estos segmentos representan la línea central que recorre a las neuritas de principio a fin (ver Figura 5.1(a)). Aunque muchas propiedades de estas neuritas son de interés [Capítulo 1] tales como longitud, grosor, bifurcaciones, etc, este trabajo está enfocado únicamente a la medición de longitudes. Los métodos de detección utilizados, harán fácil la implementación de otras mediciones.

El enfoque principal del método, consiste en medir la longitud de una neurita N_k^j, al tiempo k y de ahí tener el seguimiento de esta misma neurita a lo largo de toda la secuencia. Cabe mencionar que la aplicación no se limita a seguir una sola neurita a la vez, sino que se pueden seleccionar varias de ellas y hacer el seguimiento simultáneo para todas.

La manera en la que representamos la línea central de cada neurita es mediante un arreglo ordenado de coordenadas (x, y) localizadas en el espacio de la imagen. Con esta información se puede recorrer ordenadamente el arreglo e ir calculando la longitud en cada coordenada, hasta llegar al final. Así tendremos la longitud total del segmento de línea, que representa una aproximación de la longitud real de la neurita.

El cálculo de la longitud de la neurita N_k^j se realiza de la siguiente manera: se toma el primer punto (pixel) x_1 en N_k^j y se centra en ella una ventana de 3×3. Al ser el segmento de línea un continuo en un espacio discreto, la siguiente coordenada en el arreglo debe de estar contenida en dicha ventana. En una métrica euclideana, si esta coordenada se encuentra en las posiciones diagonales con respecto al centro, la distancia entre coordenadas será de $\sqrt{2}$ pixeles; de otra manera la distancia sera de 1 pixel. Esto se puede apreciar con mayor claridad en la figura 5.4. Una vez tomada esa medida, se toma el siguiente punto de N_k^j, y se repite la operación. Cada medida se suma a la distancia anterior hasta llegar al último punto de N_k^j.

Así podremos conocer la longitud euclideana aproximada de la neurita N_k^j para una imagen I_k, correspondiente a un tiempo t. Al obtener las longitudes de todas las neuritas seleccionadas a lo largo de la secuencia de imágenes se obtiene una cuantificación del patrón de crecimiento de la célula. Las longitudes de las neuritas seleccionadas de una neurona se muestran en la figura 5.5(b). La tabla de la figura 5.5(b) incorpora la medición del segmento en tres imágenes de una secuencia de 90 imágenes. Además muestra de manera gráfica el segmento de linea correspondiente a la línea central de la neurita (Fig. 5.5(a)).



(a) El siguente punto está en la diagonal

(b) El siguiente punto está en la transversal

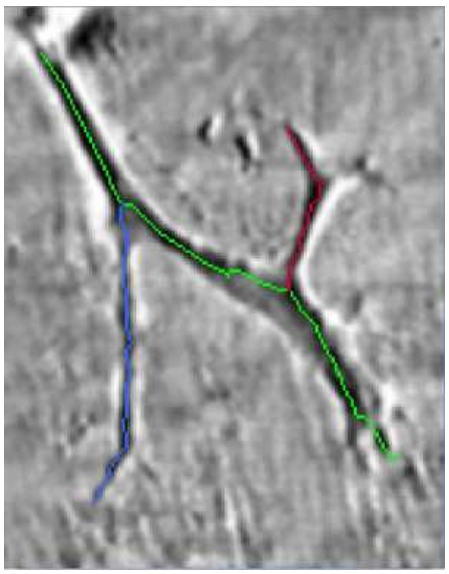
Figura 5.4: Cuantificación de la distancia entre la coordenada actual y la siguiente dependiendo de su posición. (a) Si la coordenada siguiente está en la diagonal, la distancia entre los puntos será de $\sqrt{2}$ pixeles. (b) Si la coordenada siguiente está en posición horizontal o vertical, la distancia entre puntos será de 1 pixel

Las longitudes son medidas en pixeles, sin embargo el usuario puede calibrar las unidades en micras o milímetros. Todas las medidas de la secuencia pueden ser exportadas en un formato *Excel* para su análisis.

El valor de calibración puede variar dependiendo de la resolución de la cámara digital y el objetivo con el que fueron adquiridas las imágenes. La medida de longitud de una neurita y en general cualquier medida que se realice sobre las imágenes, puede afectarse si a la imagen original se le aplican transformaciones que modifiquen la cantidad de pixeles en la imagen.

5.2.1. Resultados

Los métodos de detección y caracterización de neuritas descritos en los capítulos anteriores, fueron probados con dos tipos distintos de secuencias.



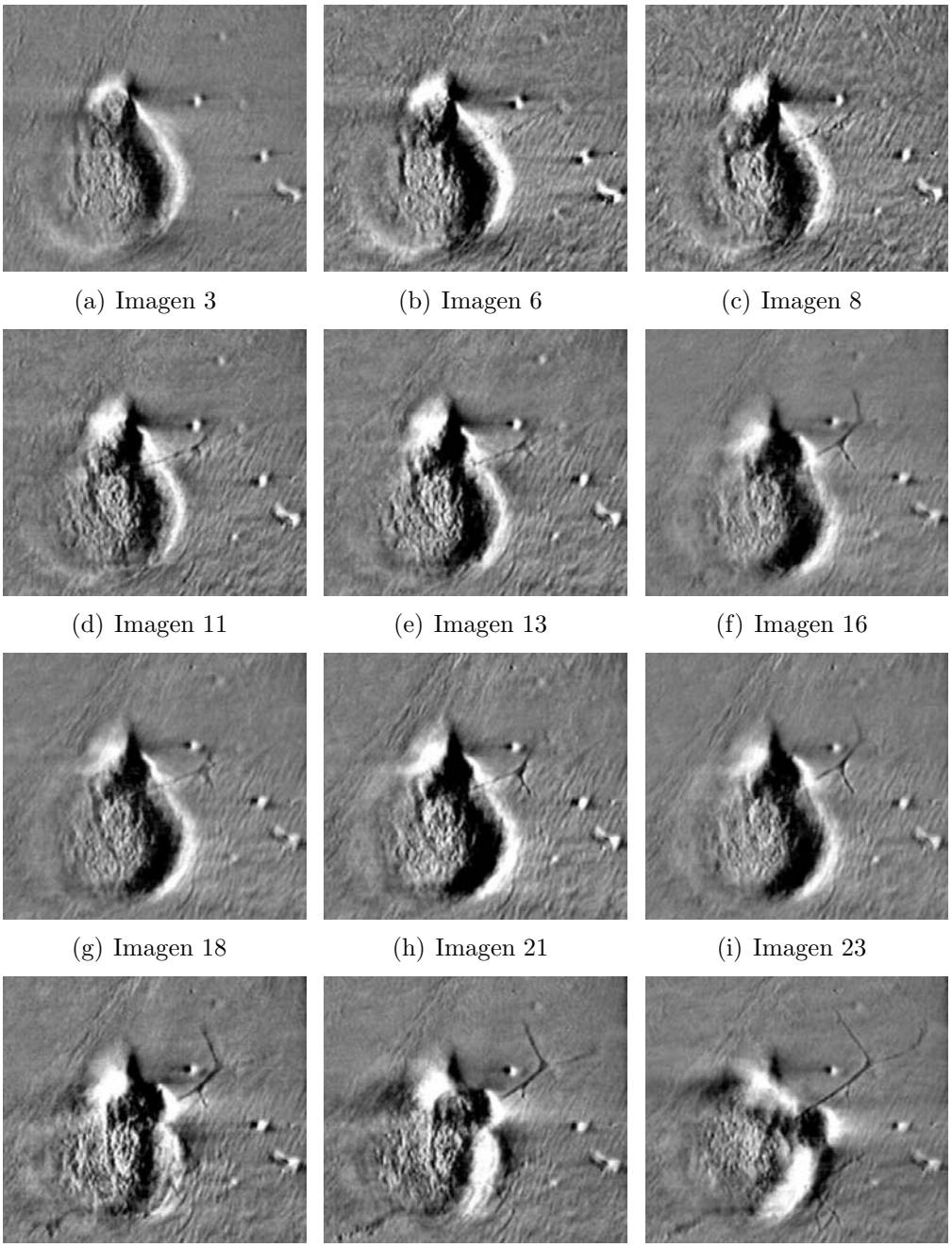
(a)

Name	Frame	Type	Length	Color
primary	6	1st Grade	219.17871...	
secondary_01	6	2nd Grade	90.970562...	
secondary_02	6	2nd Grade	56.041630...	
primary	7	1st Grade	215.35028...	
secondary_01	7	2nd Grade	101.62741...	
secondary_02	7	2nd Grade	67.698484...	
primary	8	1st Grade	223.00714...	
secondary_01	8	2nd Grade	118.35533...	
secondary_02	8	2nd Grade	69.941125...	

(b)

Figura 5.5: (a) Neuritas detectadas y caracterizadas automáticamente. (b) Resultados de la caracterización y cuantificación automática, con longitud en pixeles, de las neuritas mostradas en (a).

La primera, fue una secuencia de 30 imágenes, a lo largo de la cual la célula desarrollo un patrón de regeneración simple caracterizado por dos neuritas con ramificaciones (ver Figura 5.6) y la segunda una secuencia en la cual la célula desarrollo un patrón bipolar, de nuevo, con neuritas primarias y secundarias.



(a) Imagen 3 (b) Imagen 6 (c) Imagen 8

(d) Imagen 11 (e) Imagen 13 (f) Imagen 16

(g) Imagen 18 (h) Imagen 21 (i) Imagen 23

(j) Imagen 26 (k) Imagen 28 (l) Imagen 31

Figura 5.6: Imágenes muestra de una secuencia considerada con un patrón de regeneración simple.

En la primera secuencia de 30 imágenes, se caracterizaron 5 neuritas. El tiempo aproximado que tomó la cuantificación del crecimiento una vez que la secuencia fue preprocesada, fue de alrededor de 30 minutos. Con métodos tradicionales, consistentes en la segmentación manual usando el *ratón* y el programa *NeuronJ* la cuantificación de la misma secuencia se realizó en aproximadamente 4 horas. En la figura 5.7 se muestra la gráfica crecimiento vs tiempo de las medidas obtenidas por medio de **Neurite**, al caracterizar dicha secuencia.

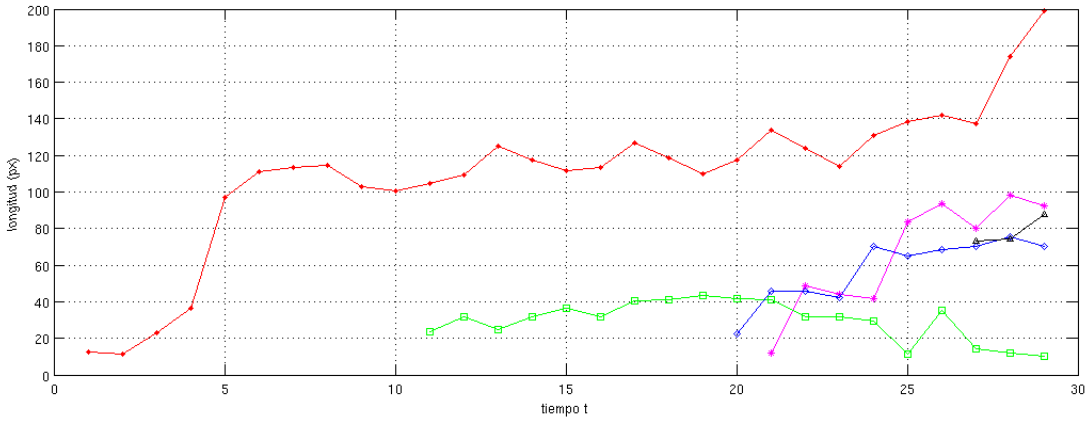
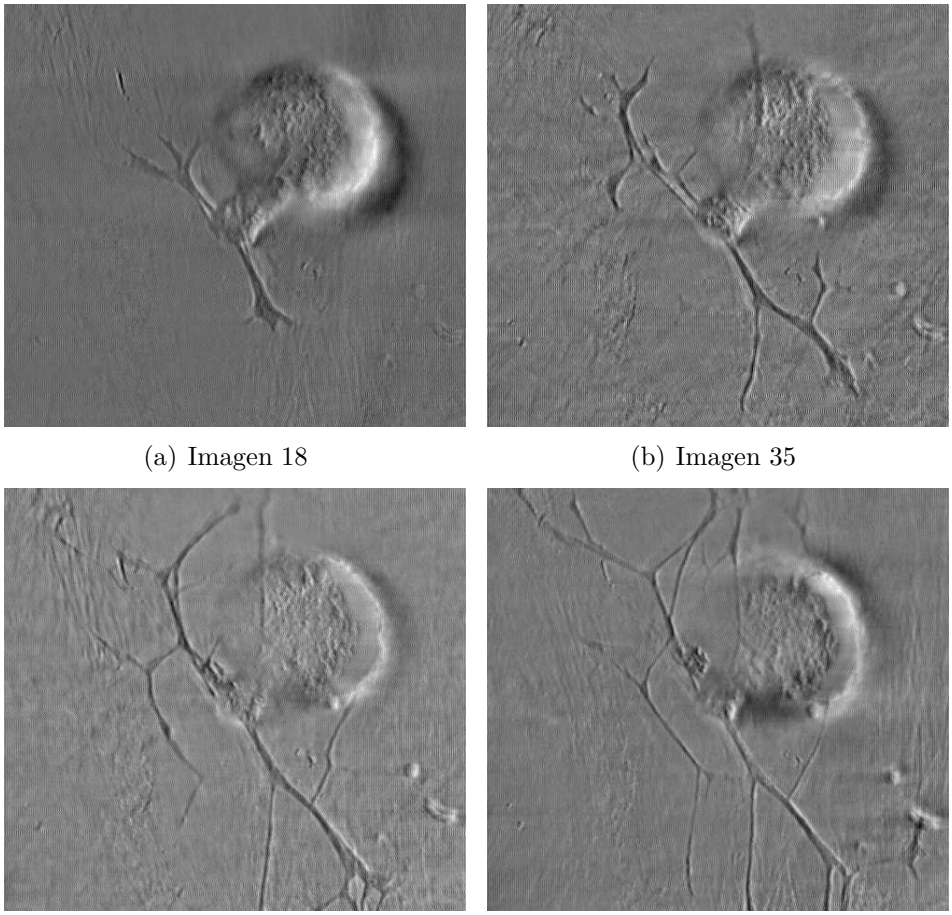


Figura 5.7: Gráfica de longitud contra tiempo de neuritas de la neurona 1, obtenidas a partir del programa **Neurite**. $(\cdot)$ en rojo la neurita primaria 1; $(*)$ en magenta la neurita primaria 2, las neuritas secundarias están como $(\square)$ en verde, $(\diamond)$ en azul y $(\triangle)$ en negro.

La cuantificación de la segunda secuencia, incluyó 70 imágenes. La neurona presentó un patrón de crecimiento difícil de cuantificar automáticamente. Debido a que la mayoría de las neuritas se enciman, haciendo imposible su detección por separado. Otro problema, es el gran número de neuritas a cuantificar, y los cruces entre neuritas. Estos problemas hicieron que la detección de esta secuencia se hiciera en gran medida de forma manual. Aún así la detección y cuantificación hecha con el programa *Neurite*, se realizó más rápido que con el programa *NeuronJ*, tomando aproximadamente 4 horas con *Neu-*

rite y poco mas de 10 horas con *NeuronJ*. En la figura 5.8 se muestran 4 imágenes representativas de la secuencia para las cuales se realizó la cuantificación. En éstas se puede apreciar como el patrón de crecimiento se vuelve más complicado con el paso del tiempo, en comparación con la secuencia mostrada en la Figura 5.6. La figura 5.9 muestra la longitud contra el tiempo, de las diferentes neuritas de interés presentes en la segunda secuencia.



(a) Imagen 18 (b) Imagen 35

(c) Imagen 52 (d) Imagen 70

Figura 5.8: Secuencia de imágenes de una neurona la cual desarrollo un patrón de crecimiento complejo y además diversos tipos de problemas para realizar la cuantificación automática.

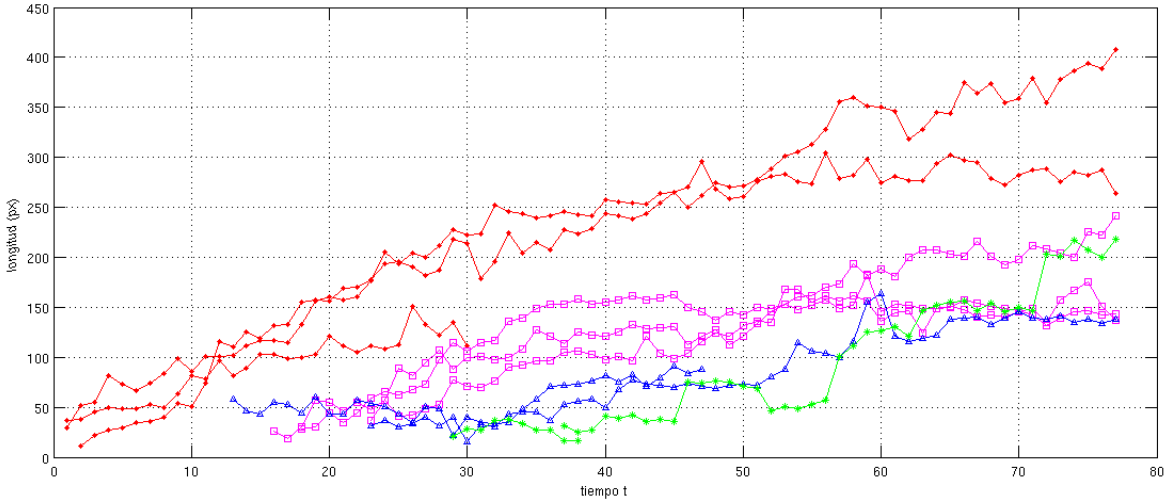


Figura 5.9: Gráfica obtenida con los datos obtenidos mediante el programa *Neurite* que expresa el crecimiento de cada una de las neuritas de la secuencia. ($\cdot$) en rojo - los trazos de 3 neuritas primarias. ($\square$) en magenta - los tres trazos de las neuritas secundarias provenientes de una misma primaria. ($*$) en verde, 2 trazos de neuritas terciarias provenientes de la misma secundaria marcada en magenta ($\triangle$) en azul 2 neuritas secundarias provenientes de la misma primaria (diferentes a la magenta). Además pueden verse eventos de extención y retracción de neuritas.

En la tabla 5.1 se comparan medidas realizadas con el programa *NeuronJ* y **Neurite** a una misma secuencia. Ahí se puede apreciar que las variaciones en longitud estimadas con ambos métodos son similares.

5.2.2. Errores

Toda medida física tiene un error implícito. En nuestro caso, hemos detectado al menos dos medidas de error: la primera, derivada de la adquisición de las imágenes, afecta el cálculo del valor de equivalencia de pixeles a unidades de longitud. Esta medida se puede conocer dependiendo de los valores de

Primaria 1			Secundaria 1.2		
Frame	**Neurite**	**NeuronJ**	**Frame**	**Neurite**	**NeuronJ**
20	155.75	121.53	20	30.80	50.04
21	160.51	124.78	21	56.63	38.23
22	157.44	138.36	22	54.46	41.22
23	160.44	136.22	23	46.80	37.65
24	176.41	130.73	24	55.21	40.97
25	205.38	137.69	25	47.63	47.57
26	193.87	160.88	26	57.11	63.09
27	203.87	174.12	27	89.04	84.95
28	199.79	164.06	28	81.11	89.21
29	211.45	170.04	29	94.53	94.33
30	227.76	193.67	30	106.87	99.73

Cuadro 5.1: Muestra las comparaciones de medidas de longitud en pixeles para dos diferentes neuritas de una misma célula, cuantificadas con los programas **Neurite** y *NeuronJ*.

error que marquen los fabricantes del equipo de adquisición de imágenes. La segunda fuente de error se deriva del método usado para realizar la detección del objeto de interés y la manera de definir los puntos inicial y final de cada segmento de línea.

Las mediciones hechas con **Neurite** buenas aproximaciones a las medidas reales, y permiten estudiar en detalle y de manera cuantitativa los eventos que ocurren durante la regeneración neurítica. Existen otras razones vinculadas al método y a la adquisición por las cuales las medidas pueden ser inexactas. En primer lugar tenemos que al discretizar el espacio continuo, asignándole a una región del espacio un valor en una escala de grises, o sea, que a toda una región se le asigna un solo valor, esto se traduce como pérdida de información. La cantidad de información perdida depende fundamentalmente de la resolución de la cámara CCD. Por otro lado, los problemas con el método de cuantificación pueden ser varios. Uno de estos es que no siempre se extrae la línea central de las neuritas, y ésto tiene sus mayores consecuen-

cias cuando las neuritas son muy anchas (más de 10 pixeles de grosor), y su forma no es una línea recta. Entonces no será el mismo valor de longitud si se toma la parte interna de una curva que si se toma la parte externa. Otro error del método, derivado directamente de la forma en que regeneran las neuronas, es que no siempre es posible saber acertadamente el punto de inicio de una neurita, y la selección de éste dependerá en mayor medida del usuario final. Sin embargo, la fuente de error puede ser constante si se definen y mantienen fijos los criterios en la adquisición, detección y caracterización de las imágenes.

Debido a estos errores presentados entre otros, las medidas efectuadas por nuestro método deberán de tomarse como medidas que representen el fenómeno desde un punto de vista general y no tomarse como medidas exactas del fenómeno.

Capítulo 6

CONCLUSIONES Y TRABAJO A FUTURO

Hemos desarrollado un método para cuantificar neuritas en secuencias de imágenes de manera semiautomática. Todos los algoritmos y métodos aquí descritos fueron implementados e incorporados en un sistema de software al cual nombramos **Neurite**. Los métodos cuantificación fueron optimizados para trabajar con imágenes adquiridas por medio de una óptica DIC, pero la aplicación no se restringe a procesar únicamente este tipo de imágenes. Cualquier secuencia de imágenes de neuritas en regeneración pueden ser detectadas y caracterizadas con el programa **Neurite** siempre y cuando las neuritas sean de una intensidad homogénea.

La versión actual de **Neurite** cumple con su primer objetivo, que es facilitar el trazado de neuritas y a partir de ello hacer estudios cuantitativos que permitan entender los mecanismos finos del crecimiento y retracción neurítica durante el fenómeno de regeneración a nivel neuronal. Esto comparado con los métodos tradicionales de cuantificación (trazado a mano), ya que aporta ventajas significativas. Dependiendo de la complejidad del patrón de crecimiento, el tiempo de análisis de la secuencia variará. Este tiempo también se ve afectado por el ruido de la imagen ya que imágenes más limpias

permitirán resultados más precisos y la intervención del usuario para corregir errores será menor. El método ha sido probado en varias secuencias (todas con óptica DIC), cada una con distintas dificultades ópticas, teniendo resultados favorables en todas ellas. Logramos analizar las secuencias en una fracción del tiempo que usualmente se requiere para ello [J.Vargas y F.Fernández de Miguel (comunicación personal)]. Además la certeza con la que las neuritas son medidas no se ve afectada al comparar el resultado con otros métodos. El tiempo de análisis de una secuencia de 30 imágenes conteniendo un patrón de crecimiento simple, llevo un tiempo de análisis de aproximadamente 30 minutos, mientras que con los métodos tradicionales, un análisis equivalente tomo aproximadamente 4 horas utilizando la herramienta *NeuronJ*.

Neurite no realiza las mediciones de una manera autónoma. Tampoco está diseñado para introducirle datos iniciales y dejarlo trabajar solo, ya que un error surgido en una imagen se propagara a lo largo de la secuencia. El método requiere que el usuario verifique constantemente que el trazado propuesto por el programa sea el deseado para así poder continuar con la secuencia.

Neurite dista mucho de ser un método definitivo, y algunas mejoras son necesarias para un mejor funcionamiento. Una mejora inmediata es la optimización de algunos algoritmos para el seguimiento de neuritas a lo largo de la secuencia, ya que los actuales tienen que recorrer varias veces la imagen para poder encontrar la neurita correspondiente en una nueva imagen. Se requieren también interfaces más amigables al usuario y depuraciones del programa ya que algunos errores sólo se podrán localizar después de su uso extenso.

Además de las mejoras al método en sí, existen muchas otras medidas morfológicas que pueden añadirse. El grosor de las neuritas es una medida de gran importancia que puede ser extraída mediante los métodos utilizados para la extracción de longitudes. El área superficial de la arborización de la célula, puede estimarse del grosor de las neuritas. Otro tipo de medición más

preciso puede ser realizado en en tres dimensiones, ya que las neuronas no crecen únicamente en dos planos. El poder realizar reconstrucciones en tres dimensiones proveería de toda la información necesaria para poder tener un entendimiento completo del fenómeno. Por otra parte, los eigenvectores obtenidos permitirán hacer modelos. Esto es esencial ya que se pueden extraer las componentes fundamentales del patrón y estudiar las variaciones producidas experimentalmente.

Apéndice A

IMPLEMENTACIÓN

Neurite fue implementado completamente en JAVA, utilizando la librería Java Advanced Image (JAI) que incluye funciones básicas para el procesamiento digital de imágenes. Además en la mayoría de las herramientas proporcionadas por **Neurite** se implementan nuevas funciones mas complejas, las cuales, en su gran mayoría hacen uso de la librería JAI para ayudarse en funciones básicas.

A.1. Java

JAVA es un lenguaje de programación orientado a objetos desarrollado en 1990 por Sun Microsystems, que basa su funcionamiento en la instalación de una maquina virtual la cual entiende el código Java y lo traduce al código de máquina sobre la cual esta montado dicha maquina virtual. Desde su creación el lenguaje Java se dio a conocer rápidamente y se hizo de gran popularidad. Actualmente es uno de los lenguajes mas usados y la mayoría de sus aplicaciones están enfocadas principalmente al diseño de portales web, manejo de bases de datos y muy recientemente en el diseño de aplicaciones para dispositivos móviles. Entre las aplicaciones científicas existe el software para hacer reconstrucciones 3D y modelos a partir de tomografías realiza-

das mediante microscopía electrónica [McMahan U. J. Laboratory, 2007], o el software de navegación para el "Mars Exploration Rover", entre muchos otros.

Una de las grandes ventajas de el lenguaje JAVA es la amplia disponibilidad de la documentación de funciones y librerías, lo cual nos permite un fácil entendimiento y uso de las mismas, sin tener que recurrir a libros especializados en el tema. JAVA provee una serie de funciones de muy fácil uso para la creación de interfaces gráficas (GUI por sus siglas en ingles) conocidas como Java Swing. Por último una de las características más importantes de JAVA es, que es un lenguaje multiplataforma, esto es, que cualquier aplicación Java correrá en cualquier plataforma que tenga soporte para la maquina virtual de Java (JVM), entre estas plataformas estas las mas usadas actualmente como son Microsoft Windows, Mac OS, Linux, Solaris, Free BSD, etc.

A.2. Java Advanced Imaging (JAI)

Java Advanced Imaging provee un conjunto de interfaces orientadas a objetos que soportan un modelo simple de programación de alto nivel que permiten la manipulación de imágenes de una forma sencilla dentro de aplicaciones Java. El API de JAI va mas allá de la funcionalidad tradicional que proveen los API's para el proceso digital de imágenes. La librería JAI permite tener una estructura de trabajo de alto rendimiento, independiente de la plataforma usada y extensible a nuevas operaciones.

El uso de JAI, facilitó el manejo de muchas de las operaciones básicas sobre imágenes, que son necesarias para la serie de procedimientos mas complejos implementados en **Neurite**. Muchos de estos procedimientos se hicieron utilizado las opciones de expansibilidad que provee el API de JAI, haciendo que el uso de las nuevas funciones sea mucho mas sencillo. Al ser JAI una extensión directa del proyecto Java original, embona a la perfección en nuestra aplicación.

A.3. Neurite

A grandes rasgos, en la implementación del sistema Neurite se usaron las versiones JDK 6 Update 2 de java y la versión JAI1.1.2 para el Java Advanced Imaging, usando los compiladores provistos por Sun Microsystems.

La aplicación esta implementada siguiendo las normas del patrón *modelo vista controlador* el cual separa los datos de la aplicación, la interfaz de usuario y la lógica de control en tres componentes distintos. El *modelo* representa la información con la cual el sistema opera, esto es, todas las funciones que realizan el procesamiento de las imágenes las cuales regresan los datos de las transformaciones en crudo. A la interfaz gráfica, que se encarga de mostrar el resultado de las operaciones, y las ventanas de interacción con el usuario le llamamos la *vista*, mientras que la parte que se encarga de gestionar la interacción con el usuario, esto es, decir que función del modelo se ejecutará al presionar tal botón de la vista, le llamamos *controlador*.

Neurite se maneja completamente mediante una interfaz gráfica, ésta trata de ser lo más intuitiva y simple. La interfaz esta completamente construida con la biblioteca Java Swing y Java 2D. Esta última se usa para realizar todo el despliegue de las imágenes y del trazado de neuritas.

Apéndice B

MANUAL DE USUARIO

El programa **Neurite**, es un sistema de interfaz gráfica modular, que consta de una ventana principal, desde la cual se pueden abrir varias ventanas dependiendo de la aplicación que se requiera. La figura B.1 muestra la ventana principal, que cuenta con una barra de menús en la parte superior; en el centro una barra de botones de acceso rápido a las funciones más usadas, y en la parte inferior con un panel de mensajes de estado del programa.

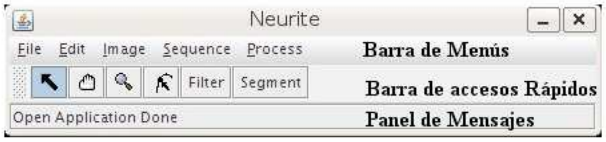


Figura B.1: Ventana principal de la aplicación **Neurite**

En la barra de Menú contamos con diferentes opciones dependiendo del proceso que se desea realizar.

Menú File

New File: Crea una nueva imagen de 8 bits con una tonalidad asignada entre 0 y 255. Abre una ventana emergente [ver Figura: B.2] en la cual solicita el nombre del nuevo archivo, las dimensiones que tendrá y el valor de 0 a 255.

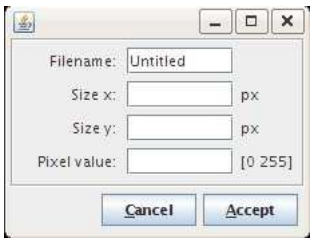


Figura B.2: Ventana de emergente, para crear una nueva imagen.

Open File: Abre una imagen. Si en el selector de archivos se seleccionan más de una imagen, el programa abrirá un ventana por cada imagen seleccionada.

Open Sequence: Abre una secuencia de imágenes con todas las imágenes seleccionadas en el selector de archivos.

Save: Guarda en disco una sola imagen, sobreescribiendo el archivo original.

Save As: Guarda en disco el archivo que se encuentre seleccionado asignándole el nombre propuesto por el usuario.

Save Sequence As: Guarda en disco toda una secuencia de imágenes. El programa abre una ventana emergente en la cual pregunta el nombre que tendrán las imágenes, el formato, el valor de inicio para numerar las imágenes. En seguida el selector de archivos preguntará por una carpeta en la cual guardar las imágenes

Exit: Sale de la aplicación, cerrando todas las ventanas.

Menú Edit

Trim Borders: Elimina automáticamente franjas o bordes negros alrededor de una imagen o secuencia.

Menú Image

Sub-Menu Type

Grayscale 8 Bits: Convierte una imagen o secuencia del tipo RGB o de tres capas, a una imagen o secuencia de una sola capa en escala de grises con pixeles de 8 bits.

RGB 8 Bits: Convierte una imagen o secuencia de una sola capa, a una imagen o secuencia en RGB de 3 capas

Byte: Cambia la escala de los pixeles de una imagen o secuencia, a pixeles en escala de 8 bits.

Enhance Image: Realza automáticamente el brillo y contraste de una imagen o secuencia.

Invert: Obtiene el negativo de una imagen o secuencia.

Zoom: Aplica un zoom a una imagen o secuencia. El zoom puede ser para acercarse, alejarse o bien regresar la imagen a su tamaño original. Estas funciones tienen la posibilidad de realizarse con comandos: $Ctrl + 1$ hace un acercamiento y $Ctrl + 2$ hace un alejamiento. Esto "shortcuts" funcionan siempre y cuando la ventana principal esté seleccionada.

Menú Sequence

Extract Frame: Extrae la imagen actual mostrada en la secuencia que se encuentre abierta en una ventana independiente.

Normalize Sequence Light: Normaliza u homogeniza la intensidad de luz en todas las imágenes de la secuencia, con respecto a la imagen actual.

Sub-Menu Background generation

Kalman Filter: Genera una secuencia fondo para la secuencia actual utilizando en filtro Kalman, el cual requiere de 2 parámetros para realizar la predicción del fondo.

Average Background: Genera una secuencia fondo para la secuencia actual, haciendo para cada imagen el promedio de todas las anteriores.

Step Average Background: Genera una secuencia fondo haciendo para cada imagen el promedio de las 5, 10, 15, 20, 30 o 50 imágenes anteriores, según se seleccione.

Menú Process

Sub-Menu Filters

Filter Manager: Abre la ventana del manejador de filtros.[Figura: B.3]. Esta ventana cuenta con tres secciones: *Filter*, *Preview*, *Filter Options*. Con esta ventana se pueden aplicar a una imagen o secuencia de imágenes los filtros mencionados en la sección *Filter*. Dependiendo del filtro seleccionado se habilitaran o deshabilitaran las opciones en la parte de *Filter Options*. Las opciones de tamaño del kernel son *3x3, 5x5, 7x7*, para cada uno de los filtros. La casilla de *Threshold* dependerá del filtro usado, la casilla *Strength* es la intensidad con la que se aplicará el filtro, mientras que la casilla *Passes* indica las veces que se aplicará el filtro.

Median: Aplica a la imagen o secuencia un filtro mediana.

High Boost Aplica a la imagen o secuencia un filtro High Boost.

Sub-Menu Edge Detection

Canny: Aplica a la imagen o secuencia un filtro Canny.

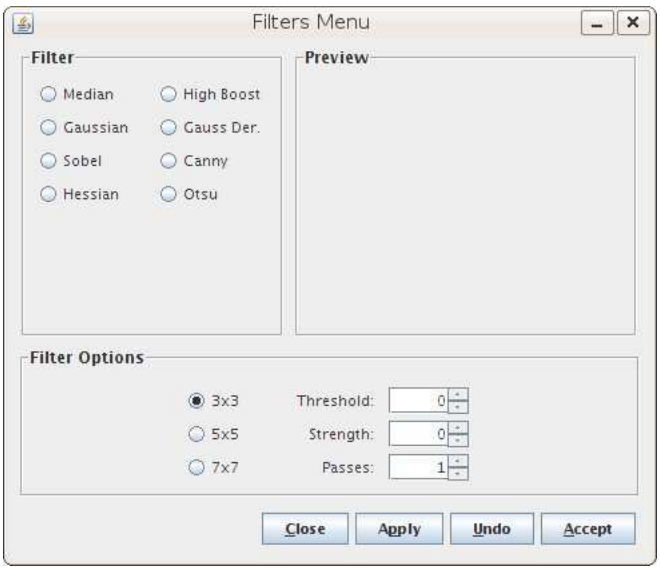


Figura B.3: Ventana del manejador de filtros.

Sobel: Aplica a la imagen o secuencia un filtro Sobel.

Sub Menu Arithmetic Operation

Add: Suma dos imágenes seleccionadas, suma una a una las imágenes si se seleccionaron dos secuencias, o bien suma una imagen a todas las imágenes de una secuencia.

Multiply: Multiplica dos imágenes seleccionadas, multiplica una a una las imágenes si se seleccionaron dos secuencias, o bien Multiplica una imagen a todas las imágenes de una secuencia.

Average: Obtiene la imagen promedio de dos imágenes seleccionadas. Obtiene el promedio una a una de las imágenes si se seleccionaron dos secuencias, o bien obtiene el promedio de una imagen con cada una de las imágenes de una secuencia, si se selecciono un imagen y una secuencia.

Subtract: Resta una imagen a otra; resta una a una las imágenes

si se seleccionaron dos secuencias, o bien resta una imagen a todas las imágenes de una secuencia.

Absolute Subtraction: Aplica la resta absoluta a dos imágenes seleccionadas. Aplica la resta absoluta una a una las imágenes si se seleccionaron dos secuencias, o bien aplica la resta absoluta una imagen a todas las imágenes de una secuencia.

Normalized Subtraction: Aplica la resta normalizada de una a otra imágenes seleccionadas. Aplica la resta normalizada una a una a las imágenes si se seleccionaron dos secuencias, o bien aplica la resta normalizada de una imagen a todas las imágenes de una secuencia.

Statistical Subtraction Aplica la resta estadística a dos imágenes seleccionadas. Aplica la resta estadística una a una a las imágenes si se seleccionaron dos secuencias, o bien aplica la resta estadística una imagen a todas las imágenes de una secuencia.

Sub Menu Logical Operation

And Aplica la función lógica *and* a dos imágenes seleccionadas. Aplica la función lógica *and* una a una a las imágenes si se seleccionaron dos secuencias, o bien Aplica la función lógica *and* a una imagen a todas las imágenes de una secuencia.

Xor Aplica la función lógica Xor a dos imágenes seleccionadas. Aplica la función lógica Xor una a una a las imágenes si se seleccionaron dos secuencias, o bien Aplica la función lógica Xor a una imagen a todas las imágenes de una secuencia.

Sub-Menú Special Operations

Dic Phase Correction Abre la ventana que se encarga de realizar la corrección de fase de óptica DIC (Fig. B.4). Para realizar

la corrección DIC son necesarias tres imágenes (Sección: 2.2), las cuales tienes que estar abiertas, y se deben seleccionar en las casillas: *Source image, Min image, Max image*. En las casillas *Delta X* y *Delta Y* se debe introducir el corrimiento de fase X y Y, en la óptica DIC del microscopio.

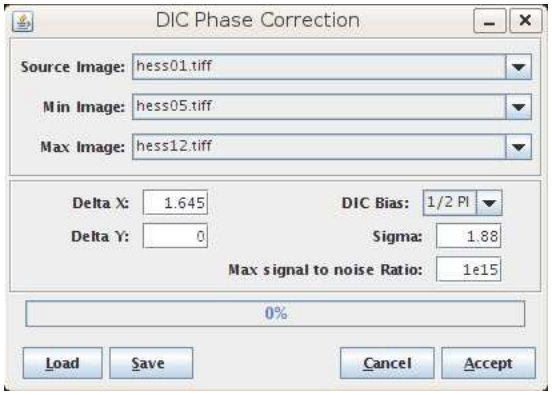


Figura B.4: Ventana para realizar la corrección de fase de la luz para imágenes DIC

Flat Field Correction Abre un selector de imágenes, en el cual se seleccionan las imágenes necesarias para realizar esta operación.

Segment Abre la ventana encargada de realizar la segmentación y cuantificación de neuritas (Fig. B.5).

Para realizar la cuantificación, se deberán seguir los siguientes pasos: oprimir el botón *Start*, una vez concluido el proceso, oprimir el botón *Select*, y comenzar a seleccionar las neuritas de interés. Si hay discontinuidades y partes de las neuritas no fueron seleccionadas, se debe presionar el botón *Draw* y dibujar esa sección a mano. Una vez seleccionadas todas las neuritas presionar el botón *Ready* para seleccionar la linea central de las neuritas. Una vez hecho lo anterior se pueden comenzar a añadir neuritas. Se presiona el botón el cual abrirá una nueva ventana donde se deben de seleccionar las propiedades princi-

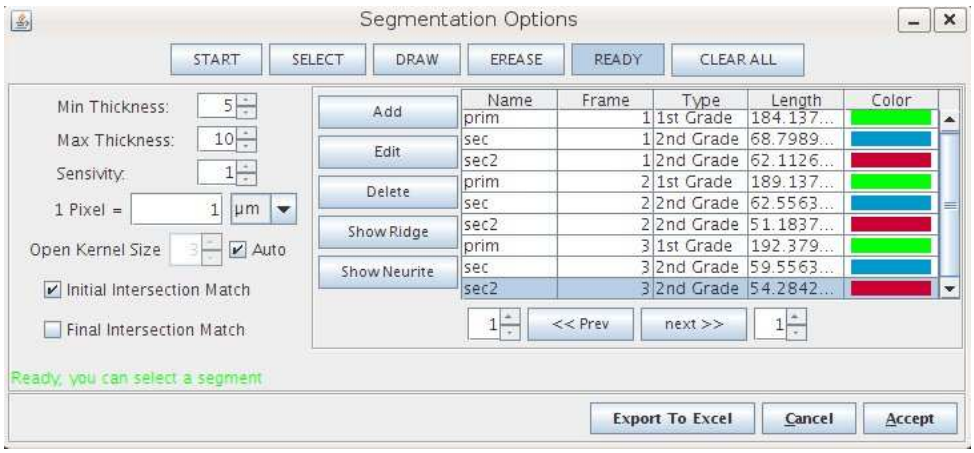


Figura B.5: Ventana para realizar la cuantificación de las neuritas

pales, además de los puntos inicial y final de una neurita. El proceso de añadir neuritas se repetirá cada vez que se quiera añadir una nueva neurita. Una vez caracterizadas todas las neuritas en dicha imagen se deberá de presionar el botón *Next* para realizar automáticamente el proceso de caracterización de neuritas en las siguientes imágenes.

Min Thickness: es el grosor en pixeles de las neuritas mas delgadas que se quieran detectar.

Max Thickness: es el grosor en pixeles de las neuritas mas gruesas que se quieran detectar.

Sensitivity: el valor de sensibilidad de selección automática de neuritas. Este valor dependerá del contraste de las neuritas con el fondo de la imagen.

1 Pixel: equivalencia de un pixel en medidas de longitud. Dependiendo de este valor será el valor de la longitud de las neuritas.

Initial Intersection Match: se deberá seleccionar si se desea que el punto inicial de las neuritas, sea la intersección mas cercana al punto inicial seleccionado, para cada una de las neuritas

agregadas.

Final Intersection Match: se deberá seleccionar si se desea que el punto final de las neuritas, sea la intersección mas cercana al punto final seleccionado, para cada una de las neuritas agregadas.

Add: Abre la ventana para agregar neuritas.

Edit: Abre la ventana para editar neuritas. Para editar una neurita primero se debe de seleccionar de la lista de neuritas. Si la neurita a editar no corresponde al número de imagen mostrada se deberá seleccionar además, la imagen correcta con el fin de evitar la sobreposición de neuritas en diferentes tiempos.

Delete: elimina la neurita seleccionada.

Show Ridge: Muestra el *ridge* encontrado para las las neuritas en proceso. Si se vuelve a presionar el botón, se esconde el *ridge*.

Show Neurites: Muestra la caracterización de las neuritas agregadas en la imagen actual. Si se vuelve a presionar el botón, se esconden las caracterizaciones de las neuritas mostradas.

Prev: Encuentra y caracteriza automáticamente las neuritas seleccionadas para las n imágenes anteriores. El número n de imágenes se deberá de seleccionar en el recuadro del lado izquierdo al botón *Prev*.

Next: Encuentra y caracteriza automáticamente las neuritas seleccionadas para las n imágenes siguientes. El número n de imágenes se deberá de seleccionar en el recuadro del lado derecho al botón *Next*.

Clear All: Elimina toda la información calculada para la imagen actual..

Bibliografía

Al-Kofahi, O., Can, A., Lasek, A., y Szarowski, D. (2003). Median-based robust algorithms for tracing neurons from noisy confocal microscope images. *Information Technology in Biomedicine, IEEE Transactios on*, **7**(4), 302–317.

Al-Kofahi, O., Radke, R. J., Roysam, B., y G., B. (2006). Automated semantic analysis of changes in image sequences of neurons in culture. *Biomedical Engineering, IEEE Transactions on*, **53**(6), 1109–1123. IEEExplore.

Arnison, M. R., Cogswell, C. J., Smith, N. I., Fekete, P. W., y Larkin, K. G. (2000). Using the hilbert transform for 3d visualization of differential interference contrast microscope images. *Journal of Microscopy*, **119**(1), 79–84.

Aylward, S. R. (2002). Initilization, noise, singularities, and scale in height ridge traversal for tubular object centerline extraction. *IEEE Transactions on Medical Imaging*, **21**(2), 61–75.

Canny, J. (1986). A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, **PAMI-8**(6), 679–698.

Cheung, S.-c. S. y Kamath, C. (2004). Robust techniques for background subtraction in urban traffic video. In S. Panchanathan y B. Vasudev, editors, *Visual Communications and Image Processing 2004*, volume 5308, pages 881–892. Proceedings of the SPIE.

Cheung, S.-C. S. y Kamath, C. (2005). Robust background subtraction with foreground validation for urban traffic video. *Journal on Applied Signal Processing*, pages 2330–2340.

De Miguel, F. y Vargas, J. Arias, C. E. C. (2002). Extracellular matrix glycoproteins inhibit neurite production by cultured neurons. *Journal of Comparative Neurology*, pages 401–411.

Eberly, D. (1996). *Ridges in Image and Data Analysis*. Kluwer Academic Publishers.

Fuentes, L. M. y Velastin, S. M. (2001). Foreground segmentation using luminance contrast. In *Proceedings of the WSES/IEEE Conference on Signal XXXX and Image Processing*.

Gonzalez, W. y Eddins (2004). *Digital Image Processing Using MATLAB*. Prentice Hall.

Hamamatsu Corp. (2006). http://sales.hamamatsu.com/en/home.php.

ImageJ (1997-2007). U. S. National Institutes of Health, Bethesda, Maryland, USA. http://rsb.info.nih.gov/ij/. Open Source Image Processing Software.

Jansson, P. A. (1996). *Traditional Linear Deconvolution Methods*. Academic Press, Inc.

Keenan, T. M., Hooker, A., Spilker, M. E., Li, N., Boggy, G. J., Vicini, P., y A., F. (2006). Automated identification of axonal growth cones in time-lapse image sequences. *Journal of Neuroscience Methods*, **151**(2), 232–238.

Lai, A. y Yung, N. (1998). A fast and accurate scoreboard algorithm for estimating stationary backgrounds in an image sequence. In *Circuits and Systems, 1998. ISCAS '98. Proceedings of the 1998 IEEE International Symposium on*, volume 4, pages 241–244.

Li-Sheng, Tian Lei, Woang Rong-ben, Guo Lie, y Chu Jinag-wei (2005). An Improved Otsu Image Segmentation algorithm for Path Mark Detection under Variable Illumination. In *IntelligentVehicles Symposium, 2005. Proceedings IEEE*, pages 840–844.

Lindeberg, T. (1996). Edge detection and ridge detection with automatic scale selection. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 465 – 470.

Marquardt, D. (1963). An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal on Applied Mathematics*, **11**, 431–441.

Martinez-Perez, M. y Espinosa-Romero, A. (2004). Retinal blood vessel 3d reconstruction from two views. In *4th Indian Conference on Computer Vision and Graphics*, pages 258–263.

Martinez-Perez, M., Hughes, A., Thom, S., Bharath, A., y Parker, K. (2007). Segmentation of blood vessels from red-free and fluorescein retinal images. *Medical Image Analisis*, **11**(1), 47–61.

McMahan U. J. Laboratory (2003-2007). EM3D: Software for Electron Microscope Tomography. http://em3d.stanford.edu/.

Media Cybernetics (2004). http://www.mediacy.com/. ImagePro Plus 5.1.

Meijering, E., Jacob, M., Sarria, J.-C., y Unser, M. (2003). A novel approach to neurite tracing in fluorescence microscopy images. In M. Hamza, editor, *Proceedings of the Fifth IASTED International Conference on Signal and Image Processing (SIP'03)*, pages 491 – 495, Honolulu HI, USA. ACTA Press, Calgary.

Metamorph 7 (2007). http://www.moleculardevices.com/pages/software/metamorph.html. Comercial Image Processing Software.

Molecular Expressions (2004). http://www.microscopy.fsu.edu/primer/digitalimaging/imageprocessingintro.html. Flat Field Correction.

NIH Image (2007). http://rsb.info.nih.gov/nih-image/. Image Processing Software, Freeware.

Nikon (1996). User manual, Microscope Nikon DIATHOP-TMD.

Padawer, J. (1968). The nomarski interference-contrast microscope. An experimental basis for image interpretation. *Journal of Royal Microscopical Society*, **88**, 305–349.

Radke, R. J. y Andra, S. E. a. (2005). Image change detection algorithms: A systematic survey. *IEEE Trans. Image Processing*, **14**(3), 294–307.

Russ, J. C. (1998). *Image Processing Handbook*. CRC Press.

Scion Image (2007). http://www.scioncorp.com/. Image Processing Software.

Sofka, M.; Stewart, C. (2006). Retinal vessel centerline extraction using multiscale matched filters, confidence and edge measures. *Medical Imaging, IEEE Transactions on*, **25**(12), 1531–1546.

Thévenaz, P., U.E., R., y M., U. (1998). A pyramid approach to subpixel registration based on intensity. *Image Processing, IEEE Transactions on*.

Van Munster, E., Van Vliet, L., y Aten, J. (1997). Reconstruction of optical pathlength distributions from images obtained by a wide-field differential interference contrast microscope. *Journal of Microscopy*, **188**(2), 149–157.

Wang, R. (2004). http://fourier.eng.hmc.edu/e161/lectures/gradient/node2.html.