



POLITÉCNICA

"Ingeniamos el futuro"

CAMPUS
DE EXCELENCIA
INTERNACIONAL



Graduado en Ingeniería Informática

Universidad Politécnica de Madrid
Escuela Técnica Superior de Ingenieros Informáticos

TRABAJO FIN DE GRADO

Mejora de la segmentación de espinas en imágenes de microscopía confocal

Autor: Andrés Orcajo García

Director: Luis Baumela Molina

MADRID, JUNIO DE 2014

“Todo hombre puede ser, si se lo propone,
escultor de su propio cerebro”,

Santiago Ramón y Cajal

En memoria de mi padre.

Agradecimientos

A mi tutor Luis, por ayudarme y guiarme a lo largo este proyecto.

A Mayte, por su paciencia y su compañía.

A mi madre, por enseñarme y ayudarme incondicionalmente durante todo este tiempo.

A mi hermano, por hacer sus labores como hermano :)

A Julián, por estar siempre ahí, incluso desde la lejanía (¿por qué no hablar por TFG?).

A Daniel, por ser como es y revolucionar por donde pasa.

A Dani, por su positividad y ganas por compartir sus vídeos.

A Jian, por su ayuda en este documento y él lo es más¹ ;)

A Guillermo y Sandra, por compartir esos viajes de vuelta a casa.

A ACM, por compartir grandes momentos, aficiones y conocimientos durante toda la carrera.

A la gente del laboratorio PCR, por compartir sus conocimientos.

A todos aquellos que me ayudaron, que merecen ser mencionados pero que no estoy haciendo.

¹J. Chen Zhang, “Diseño e implementación de un algoritmo para la detección de la negación de textos clínicos en español” Bachelor’s final degree, Universidad Politécnica de Madrid, Escuela Técnica Superior de Ingenieros Informáticos

Índice general

1. INTRODUCCIÓN	1
1.1. Estructura y funcionalidad de la neurona	1
1.2. Microscopio confocal láser de barrido	3
1.3. Planteamiento del problema	5
1.4. Importancia de la solución	6
1.5. Dificultad	6
1.6. Objetivos	7
1.7. Descripción de los apartados	7
2. SOLUCIÓN	9
2.1. Algoritmo de evolución de la curvas	9
2.1.1. Contornos activos	9
2.1.2. Snakes	9
2.1.3. Contornos activos geodésicos	10
2.1.4. Contornos activos morfológicos	11
2.2. Función $g(I)$	14
2.2.1. Nuestra $g(I)$	14
2.2.2. Cálculo del umbral variable	15
2.3. Aplicación del algoritmo de evolución de curvas	16
2.4. Proceso intermedio	16
2.4.1. Parser VRML	17
2.4.2. Extracción de la región de interés	19
2.4.3. Extracción de anillos	20
2.4.4. Voxelización	22
2.4.5. Generación de una nueva malla y suavizado	25
2.4.6. Zoom de imágenes	25
2.5. Detalles de la implementación	26
2.5.1. Detalles	26
2.5.2. Mejora en rendimiento	27
3. EXPERIMENTACIÓN	31
3.1. Introducción	31
3.2. Resultados	31
3.2.1. Espina dendrítica 11	32
3.2.2. Espina dendrítica 12	34
3.2.3. Espina dendrítica 33	36

Índice general

3.2.4. Espina dendrítica 45	39
3.3. Resultados de la optimización	41
4. CONCLUSIONES	43
4.1. Conclusiones del proyecto	43
4.2. Futuras líneas de investigación	43
Anexos	44
A. Soluciones descartadas	45
A.1. Puntos de inflexión	45
A.2. Puntos de inflexión v2	47
A.3. Evolución de curvas 2D	48
A.4. Evolución de curvas 3D	49
B. Ejemplo VRML	50
5. Bibliografía	53

Índice de figuras

1.1.	Esquema de la neurona	1
1.2.	Dendrita y espina dendrítica	2
1.3.	Tipos de espinas dendríticas	3
1.4.	Principio de la microscopía confocal láser	4
1.5.	Comparación entre microscopio óptico y microscopio confocal	4
1.6.	Secciones transversales de las reconstrucciones originales	5
1.7.	Imagen sin procesar obtenida del microscopio	6
2.1.	Resultado de la función $g(I)$	14
2.2.	Autómata del parser VRML.	17
2.3.	Proceso de transformación de un polígonos de cuatro lados a dos de tres.	18
2.4.	Rodaja de la región de interés de una espina dendrítica	19
2.5.	Organización de los puntos en anillos.	20
2.6.	Anillos de una espina dendrítica	23
2.7.	Voxelización por intersección de superficies con rectas.	23
2.8.	Voxelización de una espina dendrítica.	24
2.9.	Comparación del resultado obtenido del algoritmo Marching cubes. .	25
2.10.	Interfaz del visualizador de espinas dendríticas, ROI y anillos.	27
3.1.	Espina dendrítica 11	32
3.2.	Espina dendrítica 11 con anillos	33
3.3.	Secciones transversales de la espina dendrítica 11	34
3.4.	Espina dendrítica 12	35
3.5.	Espina dendrítica 12 con anillos	35
3.6.	Secciones transversales de la espina dendrítica 12	36
3.7.	Espina dendrítica 33	37
3.8.	Espina dendrítica 33 con anillos	37
3.9.	Secciones transversales de la espina dendrítica 33	38
3.10.	Espina dendrítica 45	39
3.11.	Espina dendrítica 45 con anillos	40
3.12.	Secciones transversales de la espina dendrítica 45	40
4.1.	Medición del valor de gris de los radios de un anillo.	46
4.2.	Resultado de la solución por puntos de inflexión.	46
4.3.	Resultado de la solución por puntos de inflexión v2.	47
4.4.	Reconstrucción obtenida por evolución de curvas 2D	48
4.5.	Resultado de la evolución de curvas 3D	49

Índice de tablas

3.1.	Resultados de las medidas locales de la espina 11	33
3.2.	Resultados de las medidas locales de la espina 12	36
3.3.	Resultados de las medidas locales de la espina 33	38
3.4.	Resultados de las medidas locales de la espina 45	41
3.5.	Tiempos para calcular la $g(I)$ en el lenguaje Python.	41
3.6.	Tiempos para calcular la $g(I)$ en el lenguaje Julia.	41
3.7.	Ganancia conseguida tras la optimización en Julia.	41

RESUMEN

La reconstrucción y caracterización de las espinas dendríticas es hoy en día un área de trabajo de gran interés en la investigación neurobiológica. Las dendritas son prolongaciones en forma de ramas de la neurona. Las espinas dendríticas se encuentran a lo largo de las dendritas y son las encargadas de transmitir los impulsos electroquímicos al cuerpo de la neurona.

El objetivo de este trabajo es desarrollar un algoritmo con el objetivo de mejorar las reconstrucciones 3D de las espinas dendríticas. Se ha utilizado un algoritmo de segmentación basado en los contornos activos morfológicos para analizar las imágenes de partida y conseguir nuevas reconstrucciones 3D fieles a estas imágenes. En este documento presentamos todo el desarrollo necesario para llevar a cabo los objetivos del proyecto. Por último también se presentarán los resultados obtenidos con este método comparándolo con las reconstrucciones de partida.

ABSTRACT

The reconstruction and characterization of dendritic spines is a hot topic in modern neurobiology research. Dendrites are the branched ramifications of a neuron. Dendritic spines are found along the dendrites and are responsible for transmitting electrochemical signals to the neuron's main body.

The purpose of this work is to develop an algorithm to improve the 3D reconstruction of dendritic spines. We use a segmentation algorithm based on morphological active contours to analyze the images and get new faithful 3D reconstructions of these images. In this document we present all the development necessary to accomplish the project goals. Finally, we will compare present results obtained by this method with the starting reconstructions.

1. INTRODUCCIÓN

El objetivo de este trabajo es el desarrollo de una solución que genere reconstrucciones 3D de espinas dendríticas más realistas que las que se están generando ahora mismo por el software *Imaris Filament Tracer*. Estas reconstrucciones son de interés para los neuroanatomistas del CTB de la UPM para extraer información de forma y tamaño de las espinas dendríticas.

El trabajo que he realizado está apoyado por las investigaciones realizadas por Miguel Federico Barguear con su tesis de máster [1] y Pablo Márquez-Neila con los trabajos [2] [3] y con su tesis doctoral [4].

1.1. Estructura y funcionalidad de la neurona

Una neurona es una célula excitible eléctricamente que procesa y transmite información mediante señales químicas y eléctricas. Según Purves [5], las neuronas comparten los distintos órganos hallados en todas las células, sin embargo en las neuronas estos órganos son más sobresalientes en distintas regiones de la célula.

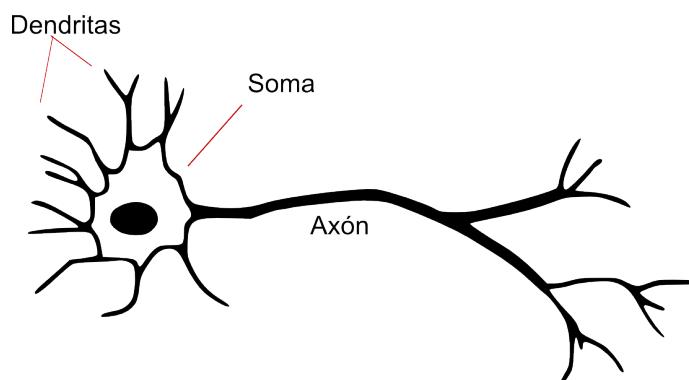


Figura 1.1.: Esquema simplificado de la neurona.

La organización celular básica de las neuronas se asemeja a la de otras células, pero se especializan para la comunicación intercelular. El signo de especialización neuronal para la comunicación a través de señalamiento eléctrico es la ramificación extensa de las neuronas. El aspecto más sobresaliente de esta ramificación en las células

1. INTRODUCCIÓN

nerviosas típicas es la ramificación compleja de dendritas que surgen del cuerpo de la célula neuronal (también llamadas ramas dendríticas o prolongaciones dendríticas).

Las neuronas presentan unas características morfológicas típicas (figura 1.1): cuerpo o soma, una prolongación larga denominada axón y una o varias ramificaciones mencionadas anteriormente como dendritas. El axón es una prolongación de la neurona especializada en la conducción de señales desde el cuerpo hacia el sitio siguiente de la interacción sináptica. Las dendritas son las encargadas de la recepción de las transmisiones sinápticas desde otras neuronas hacia el cuerpo de la neurona.

Las espinas dendríticas son protrusiones pequeñas que aparecen en la membrana de la dendrita y reciben señales de una sola sinapsis de un axón. Se distinguen por la presencia de extremos globulares que se denominan cabezas de las espinas; las sinapsis que inervan las dendritas se forman con estas cabezas. Las cabezas de las espinas están conectadas al tronco principal de las dendritas por nexos estrechos llamados cuellos. Como podemos observar en la figura 1.2, de la dendrita salen muchas espinas dendríticas, de la espina dendrítica señalada en la figura se puede observar que el cuello está unido a la dendrita y a continuación está la cabeza de la espina.

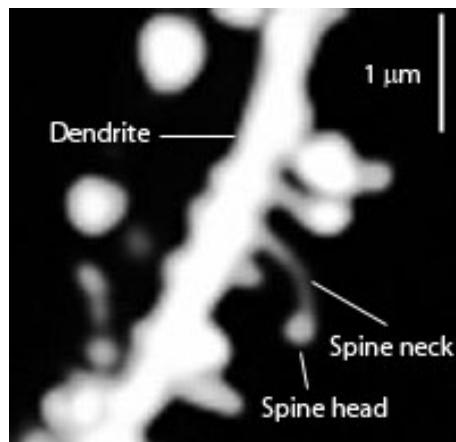


Figura 1.2.: Dendrita y espina dendrítica. En esta imagen tenemos la dendrita (Dendrite), donde se encuentran las espinas dendríticas. También podemos distinguir la cabeza (Spine head) y cuello (Spine neck) de la espina dendrítica. Fuente [6]

Las espinas dendríticas muestran una gran variedad de formas. Se han propuesto varias clasificaciones según su características morfológicas. Pero la más aceptada es la propuesta por Peters and Kaiserman-Abramof [8]. Según esta clasificación se pueden encontrar tres tipos: *finas* (thin), *tipo seta* (mushroom-type) y *cortas y gruesas* (stubby). Las finas son las más comunes y tienen un cuello largo, fino y con una cabeza pequeña. Las espinas en forma de seta tienen una cabeza grande. Y las cortas y gruesas carecen de cuello.

1.2. Microscopio confocal láser de barrido

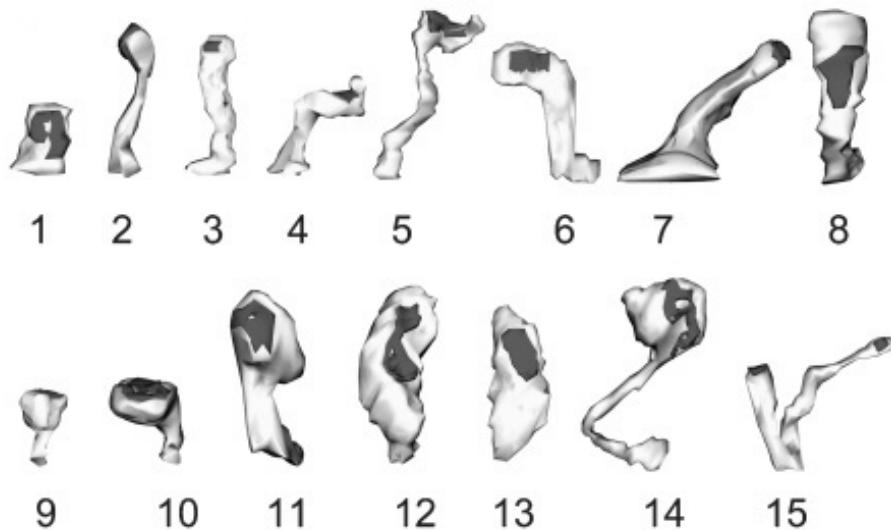


Figura 1.3.: Tipos de espinas dendríticas: cortas y gruesas (1), fina (2), en forma de seta (9-11), intermedias (3-8, 12-14). No entra en la clasificación clásica, pero posteriormente llamada ramificada (15). Fuente Arellano et al [7].

En ocasiones es difícil clasificar algunas espinas al tratarse de formas intermedias de las anteriores. En la figura 1.3 tenemos distintos tipos de espinas dendríticas, tanto las mencionadas anteriormente como formas intermedias.

Una herramienta que se utiliza para poder obtener imágenes de estas estructuras neuronales es el microscopio confocal láser, que explicamos a continuación.

1.2. Microscopio confocal láser de barrido

La microscopía confocal es una técnica óptica de imagen para incrementar la resolución y el contraste de un microscópico. El principio de la microscopía confocal se basa en eliminar la luz reflejada procedente de los planos fuera de foco. Para ello restringe la iluminación una pequeña zona de la muestra y se toma el haz luminoso que proviene del plano focal, eliminándose los haces procedentes de los planos inferiores y superiores. En la figura 1.4 tenemos el principio que sigue esta técnica, donde la luz procedente de los puntos fuera del plano focal (en verde) es eliminada por el diafragma.

El microscopio confocal láser de barrido (Confocal Laser Scanning Microscopy, CLSM) obtiene mejores resultados que el microscopio óptico convencional [10]. El microscopio óptico puede enfocar planos por encima o por debajo de la superficie que se observa, obteniendo imágenes borrosas. Como podemos observar en la figura 1.5,

1. INTRODUCCIÓN

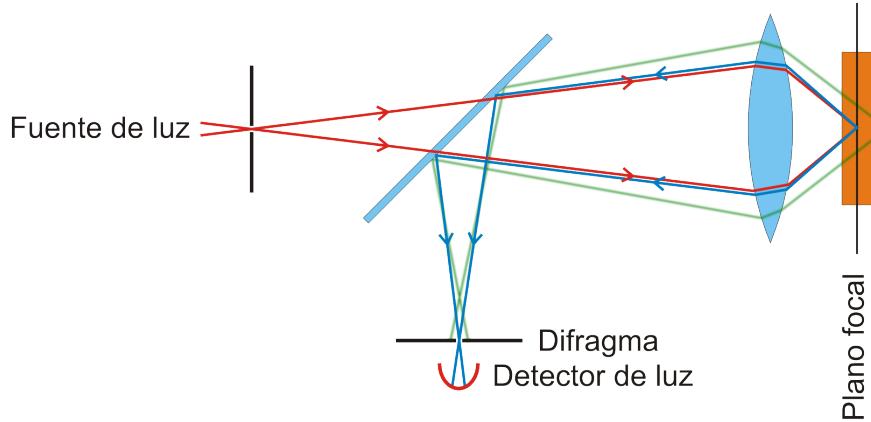


Figura 1.4.: Principio de la microscopía confocal láser. Traducción de [9]

los resultados obtenidos por el microscopio confocal cuentan con mayor nitidez, contraste y resolución.

Se denomina microscopio confocal de barrido porque sólo se ilumina una pequeña zona de la muestra y necesita un barrido que permita muestrear todos los puntos. La luz reflejada o fluorescencia emitida por la muestra (es el caso que vamos a estudiar) es recogida y transformada en una señal eléctrica para ser digitalizada.

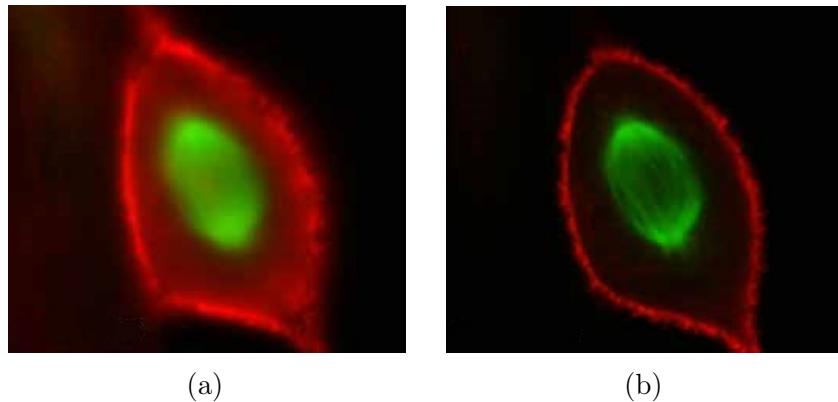


Figura 1.5.: Comparación entre microscopio óptico convencional (a) y microscopio confocal láser (b). Fuente [11]

Además de poder observar un solo plano de la muestra, variando el plano de enfoque el microscopio es capaz de tomar imágenes a diferente profundidad. Lo que permite obtener información de la estructura tridimensional de la muestra. Es decir, es posible obtener rodajas espaciadas de la muestra a lo largo del eje Z. Como resultado obtenemos un conjunto de imágenes que varían en el plano de la coordenada Z.

Para la visualización, los especímenes son tratados con marcadores (proteínas) fluorescentes que hace que las muestras emitan su propia luz al ser excitadas externamente.

1.3. Planteamiento del problema

El marcador absorbe la energía de la luz incidente emitiendo a su vez luz a una longitud de onda mayor. Este tratamiento permite la visualización de la morfología neuronal *in vivo* ya que no afecta la función de la neurona ni presenta toxicidad.

1.3. Planteamiento del problema

En las reconstrucciones tridimensionales de espinas dendríticas que se están relaizando actualmente, se emplea el software comercial *Imaris FilamentTracer*, que ofrece una serie de herramientas para crear reconstrucciones a partir de imágenes obtenidas de la microscopía confocal. Por medio de una serie de pasos, identifica dendrita y espinas dendríticas y permite variar parámetros para obtener de forma semi-atomática reconstrucciones de estas espinas.

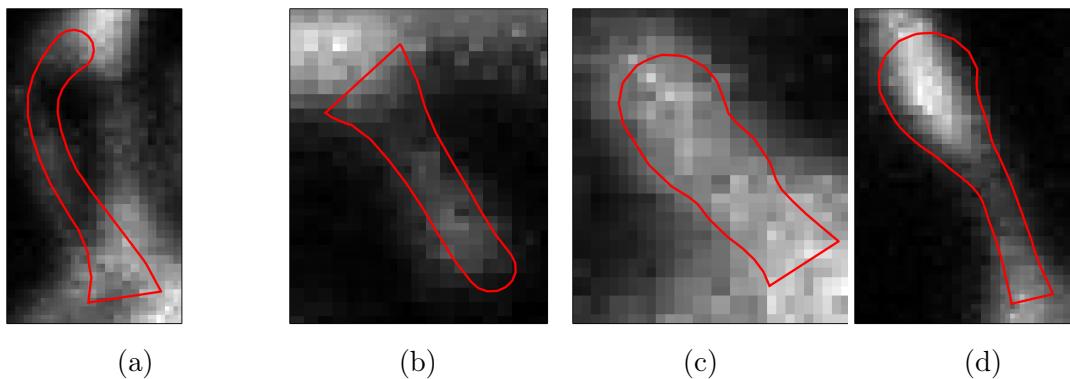


Figura 1.6.: Secciones transversales de las reconstrucciones originales.

Pero la forma en que genera las reconstrucciones no siempre se adapta a la forma original de las espinas dendríticas. La generación de estas sigue un patrón, en el que la cabeza tiene forma esférica que se une al cuerpo de la dendrita a través de un cuello con forma tubular. Por esta simplificación de la forma de las espinas, no siempre llega a ajustarse a la forma original de las imágenes. Como podemos observar en la figura 1.6 las reconstrucciones presentadas no se ajustan completamente a la forma presentada en la imagen. En (a), (b) y (d) las cabezas de las reconstrucciones no se ajustan a la imagen. En (c) el grosor de la reconstrucción es inferior al que debería.

El problema que nos planteamos aquí es mejorar dichas reconstrucciones generando otras que se adapten mejor a las imágenes obtenidas del microscopio.

1. INTRODUCCIÓN

1.4. Importancia de la solución

La reconstrucción de la estructura de la neurona y la caracterización de las espinas dendríticas es hoy en día un área de trabajo de gran interés en la investigación en neurobiología.

La reconstrucción de la estructura de la neurona nos daría información necesaria para el análisis más fiable de la neurona, obteniendo datos reales tanto de superficie, volumen y forma de la misma. Estos datos son de gran interés en neuroanatomía y entre otros, podría sentar las bases para proyectos en los que se intenta simular el comportamiento real de las neuronas y el cerebro.

1.5. Dificultad

El principal problema con el que nos encontramos es la baja resolución de las imágenes obtenidas por el microscopio y el ruido de las mismas, como podemos observar en figura 1.7 o en figura 1.6, provocando que las reconstrucciones sean poco precisas.

Otro problema que dificulta la reconstrucción de las espinas es que la densidad lumínica de las imágenes no es siempre constante, variando con el eje Z de la imagen, de forma que una simple umbralización del nivel de la fluorescencia no proporciona reconstrucciones satisfactorias

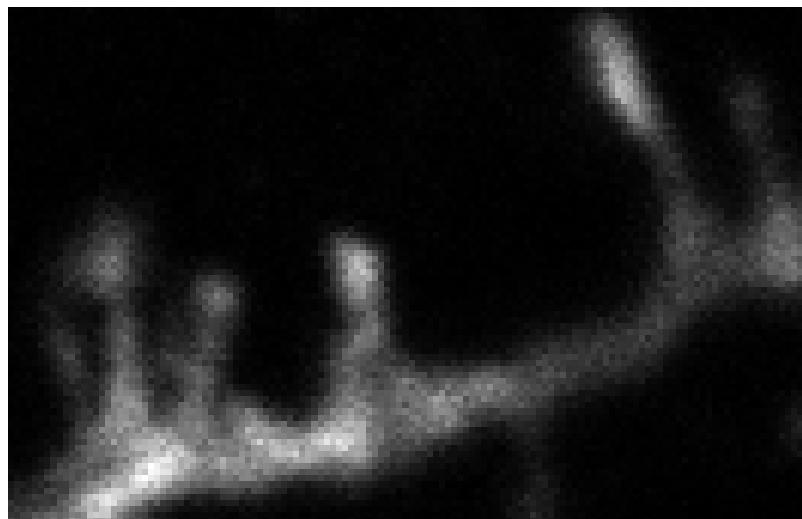


Figura 1.7.: Imagen sin procesar obtenida del microscopio, donde podemos observar el ruido de la imagen.

1.6. Objetivos

Los objetivos definidos para este trabajo se detallan a continuación,

- Lectura e interpretación de las reconstrucciones realizadas por *Imaris FilamentTracer* y de las imágenes generadas por el microscopio.
- Representación de las reconstrucciones realizadas por *Imaris FilamentTracer* en el contexto de la imagen generada por el microscopio.
- Desarrollo de un algoritmo de evolución de curvas que analice tanto las imágenes obtenidas por microscopía confocal de una espina dendrítica y de los resultados obtenidos por el programa *Imaris FilamentTracer*, para obtener segmentaciones más fieles a las imágenes del microscopio.
- Representación gráfica de los resultados.
- Hacer una evaluación del algoritmo desarrollado y analizar los resultados obtenidos.

1.7. Descripción de los apartados

En el capítulo 2, apartado 2.1 se presenta el algoritmo de segmentación que se ha utilizado, mostrando su evolución desde la idea base del algoritmo hasta llegar a la planteamiento actual. En el apartado 2.2 se explica la función $g(I)$ y la necesidad de utilizar un umbral variable en la misma, para dar solución al problema con el que nos encontramos. En el apartado 2.4 se explica todo el proceso que se ha de llevar a cabo para poder realizar las segmentaciones. Y en el apartado 2.5 se explican los detalles de implementación del desarrollo.

Los resultados obtenidos por el algoritmo se presentan en la sección de experimentación (capítulo 3), aplicándolo sobre cuatro espinas para ilustrar los resultados. También se presenta, en el apartado 3.3, los resultados de la mejora de rendimiento conseguida para generación de la $g(I)$.

En el apartado 2.1 se ha descrito el algoritmo final construido, sin embargo, este algoritmo es el resultado de un proceso de ensayo y error en el que se han descartado entre soluciones menos satisfactorias. En el anexo A estas soluciones descartadas.

Las conclusiones y líneas futuras del proyecto se presentan en el capítulo 4.

2. SOLUCIÓN

2.1. Algoritmo de evolución de la curvas

El problema con el que nos encontramos a la hora de generar reconstrucciones 3D de las espinas dendríticas es que es un problema de visión por ordenador. Para ello aplicaremos un algoritmo que sea capaz de segmentar un stack de imágenes ajustándose a la segmentación *subjetiva* de las reconstrucciones originales. Usaremos *contornos activos morfológicos* que está inspirado en los *contornos activos geodésicos* y estos en Snakes.

2.1.1. Contornos activos

Los contornos activos es una técnica de segmentación de imágenes que es robusta ante la presencia de ruido y fondo no uniformes. El tipo de contorno activo en el que nos vamos a centrar es el de snake [12] [13] (serpientes), que propone un modelo elástico de curva continua y flexible que se ajusta a los datos de la imagen.

2.1.2. Snakes

El problema que se intenta resolver consiste en la localización de bordes, líneas y contornos subjetivos. Mediante la solución iterativa de una ecuación de derivadas parciales (EDP), la curva o la serpiente se deforma hasta minimizar las energías internas y externas a lo largo de su contorno. La componente interna mantiene suave la curva mientras la componente externa ajusta la curva a la estructura de la imagen. Este comportamiento hace que la evolución de curvas sea uno de los algoritmos más usados para la segmentación y tracking de objetos.

Sea $\mathcal{C} : [0, 1] \rightarrow \mathbb{R}^2$ una curva parametrizada 2D y $I : \mathbb{R}^2 \rightarrow \mathbb{R}$ una imagen. El en método clásico de contornos activos [13] la energía de una curva para una imagen se obtiene mediante el siguiente funcional de energía:

2. SOLUCIÓN

$$E(\mathcal{C}) = \int_0^1 \underbrace{\alpha |\mathcal{C}_p(p)|^2 + \beta |\mathcal{C}_{pp}(p)|^2}_{\text{Interna}} - \underbrace{\lambda |\nabla I(\mathcal{C}(p))|}_{\text{Externa}} \, dp, \quad (2.1)$$

donde \mathcal{C}_p y \mathcal{C}_{pp} son respectivamente la primera y segunda derivada de \mathcal{C} respecto a p . Y α , β y λ son tres parámetros positivos. Los dos primeros términos en (2.1) son la energía interna de la curva (E_{int}).

$$E_{int}(\mathcal{C}) = \alpha \int_0^1 |\mathcal{C}_p(p)|^2 \, dp + \beta \int_0^1 |\mathcal{C}_{pp}(p)|^2 \, dp \quad (2.2)$$

E_{int} depende de la longitud de la curva (primer término) y en su rigidez (segundo término), por eso cuanto más suave sea la curva más bajo será E_{int} .

El último término en (2.1) es la energía externa. La expresión $|\nabla I(\mathcal{C}(p))|$ es mayor para puntos de bordes de la imagen, por consiguiente, E_{ext} baja en los bordes y crece en las regiones planas.

El modelo de snake pretende encontrar la curva que minimice la función (2.1). Esta curva será suave y seguirá algunos bordes de la imagen. El equilibrio entre suavidad y la atracción por los bordes está controlado por los parámetros α , β y λ . Este método ha sido aplicado extensamente en visión por ordenador, especialmente porque es una buena herramienta para dar solución a problemas de segmentación. Sin embargo depende fuertemente en la parametrización de la curva y de los tres parámetros anteriores.

2.1.3. Contornos activos geodésicos

Para intentar dar solución a uno de los problemas asociados a las serpientes se han planteado los contornos activos geodésicos (Geodesic Active Contour, GAC)[14]. El funcional de energía de los contornos activos geodésicos es:

$$E(\mathcal{C}) = \int_0^{L(\mathcal{C})} g(I)(\mathcal{C}(s)) \, ds = \int_0^1 \underbrace{g(I)(\mathcal{C}(p))}_{\text{atraccion}} \cdot \underbrace{|\mathcal{C}_p|}_{\text{suavizado}} \, dp, \quad (2.3)$$

donde $ds = |\mathcal{C}_p|dp$ es la parametrización euclídea de la longitud del arco de la curva, lo que lleva a modelo geométrico; y $g(I) : \mathbb{R}^2 \rightarrow \mathbb{R}^+$, $x \rightarrow g(I)(x)$ que permite

2.1. Algoritmo de evolución de la curvas

seleccionar qué regiones de la imagen en la que estamos interesados. Normalmente, $g(I)$ podría ser

$$g(I) = \frac{1}{\sqrt{1 + \alpha |\nabla G_\sigma * I|}} \quad (2.4)$$

que es bajo en los bordes de la imagen, o $g(I) = |G_\sigma * I|$, que consigue el mínimo en el centro de las líneas oscuras de la imagen. Pero en nuestro caso utilizaremos otra $g(I)$ variable que detallaremos en el apartado 2.2.

La ecuación (2.3) minimiza el peso de la longitud de la curva, es decir, la curva tiende a las áreas objetivo en la imagen y, además, será suave y robusta contra el ruido. La curva va evolucionando según dos fuerzas. La regularizadora o de suavizado, que reduce su curvatura; y la fuerza atractiva, que lleva a la curva a zonas interesantes de la imagen.

2.1.4. Contornos activos morfológicos

Los contornos activos morfológicos son una solución propuesta en [3], [2]. Es un método de evolución de curvas que resuelve la EDP a través de operadores morfológicos binarios [15]. Con estos operadores se evita los problemas de velocidad y convergencia asociados a los algoritmos numéricos. Al usar una combinación de operadores morfológicos binarios, cuyo comportamiento es el mismo comportamiento a nivel infinitesimal, se puede aproximar la EDP de los GAC.

Si expresamos el funcional de energía en términos de una implementación de conjuntos de nivel [16], obtenemos

$$\frac{\partial u}{\partial t} = \underbrace{g(I)|\nabla u|\operatorname{div}\left(\frac{\nabla u}{|\nabla u|}\right)}_{\text{suavizado}} + \underbrace{g(I)|\nabla u|\nu}_{\text{globo}} + \underbrace{\nabla g(I)\nabla u}_{\text{atracción}} \quad (2.5)$$

El primer término es la fuerza regularizadora, o energía interna, que hace que la curva se mantenga suave. El segundo término es la fuerza de globo. Es una fuerza que hace que la curva siga evolucionando en zonas donde el gradiente de energía se mantiene constante y donde la fuerza atractiva no es suficientemente fuerte. Esta fuerza actúa *hinchando* la curva. El tercer término es la fuerza de atracción de la imagen, o energía externa, que fuerza que la curva se ajuste a las regiones interesantes

2. SOLUCIÓN

de la imagen.

Fuerza del globo

La fuerza de globo de (2.5) se expresa por la siguiente ecuación:

$$\frac{\partial u}{\partial t} = g(I) \cdot \nu \cdot |\nabla u| \quad (2.6)$$

El factor $g(I)$ controla la fuerza del globo. Cuando el valor de $g(I)$ es grande, el fragmento correspondiente está lejos de la región objetivo y la fuerza del globo tiene que ser fuerte. En el otro caso, el valor de $g(I)$ es pequeño y la curva se está acercando a la región objetivo donde la fuerza del globo ya no es necesaria. El efecto del factor $g(I)$ en (2.6) se puede discretizar con un umbral θ : cuando $g(I)$ sea mayor de θ entonces se aplicará la fuerza del globo, en otro caso no se aplicará. Dependiendo del signo de ν se aplicará la fuerza dilatadora o erosiva de la EDP. Dado el estado de la iteración n , $u^n : \mathbb{Z}^d \rightarrow \{0, 1\}$, la fuerza del globo aplicada sobre u^n se puede aproximar usando la siguiente aproximación morfológica:

$$u^{n+1}(x) = \begin{cases} (D_d u^n)(x) & \text{si } g(I)(x) > \theta \text{ y } \nu > 0 \\ (E_d u^n)(x) & \text{si } g(I)(x) > \theta \text{ y } \nu < 0 \\ u^n(x) & \text{en otro caso} \end{cases}$$

D_d y E_d son las versiones discretas de la dilatación y erosión [2].

Fuerza regularizadora

La fuerza de regularizadora o de suavizado de (2.5) se expresa por la siguiente ecuación:

$$\frac{\partial u}{\partial t} = g(I) \cdot |\nabla u| \cdot \operatorname{div} \left(\frac{\nabla u}{|\nabla u|} \right) \quad (2.7)$$

El factor $g(I)$ actúa como peso que controla la fuerza de suavizado en cada punto. Como los operadores morfológicos binarios $S I_d \circ I S_d$ se comportan igual a nivel

2.1. Algoritmo de evolución de la curvas

infinitesimal que al equivalente GAC, se puede aproximar por la siguiente ecuación morfológica:

$$u^{n+1}(x) = \left((S_d I \circ I_d S)^\mu u^n \right) (x) \quad (2.8)$$

El número de las sucesivas aplicaciones del operador de suavizado controlan la fuerza de suavizado. Este número está indicado por el parámetro $\mu \in \mathbb{N}$.

Resolviendo la EDP del GAC

Como hemos visto en la ecuación del contorno activo (2.5) está compuesta por tres componentes diferentes: una fuerza regularizadora, una fuerza globo y una fuerza de atracción. Dos de ellas se pueden resolver por operadores morfológicos. La tercera componente tiene una directa versión discreta, como podremos ver más abajo.

En la EDP la combinación de los tres componentes se hace con la suma de ellos. Esta solución los combina por composición: en cada iteración se va a aplicar, globo morfológico (2.6), suavizado morfológico (2.7) y la fuerza de atracción discretizada sobre la función de conjunto de nivel u . Dado el estado del snake en la iteración n , u^n , obtenemos u^{n+1} usando:

$$\begin{aligned} u^{n+\frac{1}{3}}(x) &= \begin{cases} (D_d u^n)(x) & \text{si } g(I)(x) > \theta \text{ y } \nu > 0 \\ (E_d u^n)(x) & \text{si } g(I)(x) > \theta \text{ y } \nu < 0 \\ u^n(x) & \text{en otro caso} \end{cases} \\ u^{n+\frac{2}{3}}(x) &= \begin{cases} 1 & \text{si } \nabla u^{n+\frac{1}{3}} \nabla g(I)(x) > 0 \\ 0 & \text{si } \nabla u^{n+\frac{1}{3}} \nabla g(I)(x) < 0 \\ u^{n+\frac{1}{3}} & \text{si } \nabla u^{n+\frac{1}{3}} \nabla g(I)(x) = 0 \end{cases} \\ u^{n+1}(x) &= \left((S_d I \circ I_d S)^\mu u^{n+\frac{2}{3}} \right) (x) \end{aligned} \quad (2.9)$$

que es la implementación morfológica de la EDP del contorno activo geodésico.

2. SOLUCIÓN

2.2. Función $g(I)$

La función $g(I)$ es un componente esencial del GAC. Es quien determina el criterio de parada en la evolución del snake. Esta evolución siempre tiende a expandirse hacia gradientes de *energía* inferiores, es decir, la parada de la evolución se va a producir donde $g(I)$ se haga 0. Por esto se define un umbral subjetivo (que depende de la reconstrucción original) que determina las fronteras de la espina dendrítica.

2.2.1. Nuestra $g(I)$

La $g(I)$ que vamos a emplear es,

$$g(I) = |I - u(I)|, \quad (2.10)$$

que representa la diferencia entre el nivel de gris I de la imagen y un valor umbral u . Como veremos en la sección siguiente, este umbral dependerá del pixel a analizar I , $u(I)$, y nos asegura que el punto mínimo, es decir donde $g(I)$ se hace 0, va a ser la frontera de la espina dendrítica.

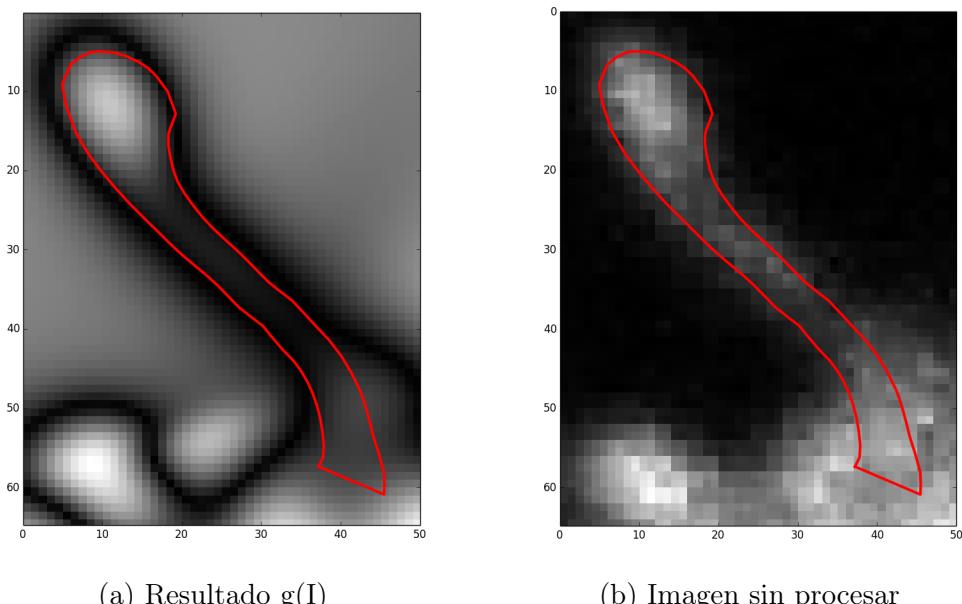


Figura 2.1.: Resultado de la función $g(I)$ sobre la imagen. En rojo tenemos la segmentación original.

Como resultado de aplicar la $g(I)$ sobre la matriz de la imágenes, obtenemos una

2.2. Función $g(I)$

matriz de mismos dimensiones pero con las fronteras de la espina en los puntos donde la $g(I)$ se anula, como podemos ver en la figura 2.1.

2.2.2. Cálculo del umbral variable

La densidad lumínica procedente de las imágenes del microscopio no es siempre constante. Definir un umbral constante que determine la frontera de la espina haría que en algunos casos la espina creciese demasiado, o por el contrario, que el cuello de la espina llegase a desaparecer. Por ello es necesario establecer un umbral variable basado en la malla original, que determina un umbral de partida subjetivo.

Para calcular el umbral de un punto \mathbf{x} , tendremos en cuenta los umbrales de toda la estructura original,

$$u(\mathbf{x}) = \left(\sum_{i=1}^n n_i e^{-\frac{d_i^2}{\lambda^2}} \right) \cdot \frac{1}{\sum_{i=1}^n e^{-\frac{d_i^2}{\lambda^2}}} \quad (2.11)$$

donde n es el número de anillos, en el apartado 2.4.3 se detallará qué es un anillo; d_i es la distancia del punto \mathbf{x} al centro del anillo i ; n_i es el umbral de cada anillo que se calcula con el mínimo valor de gris (de la imagen) que le corresponde a cada punto en el anillo i ; λ es una constante que se calcula:

$$\lambda = \frac{\bar{d}}{k} \quad (2.12)$$

donde \bar{d} es la media de las distancias de los centros de los anillos al punto; $k \in \mathbb{N}^*$.

Con la ecuación (2.11) se obtiene un umbral promediado según la distancia de todos los centros de los anillos respecto al punto. El valor de k en (2.12) determina el peso de las distancias de este promediado, es decir, cuanto más alto sea k , más se ajustará el resultado a los umbrales de los anillos más cercanos. En el caso de que k sea bajo, se tendrá a promediar el umbral con todos los anillos de la espina. La implementación del cálculo del umbral variable se puede encontrar en el apartado 2.5.2.

2. SOLUCIÓN

2.3. Aplicación del algoritmo de evolución de curvas

Una vez calculada la $g(I)$ se puede aplicar el algoritmo explicado en el apartado 2.1.4. El resultado la función $g(I)$ hará que el algoritmo de evolución de curvas expanda o reduzca el volumen hacia potenciales de energía inferiores, es decir, hacia las fronteras que hemos obtenido de ecuación (2.11).

El proceso tiene como punto de partida la superficie inicial, que en nuestro caso será la versión *voxelizada* de la reconstrucción original, que se detallará en el apartado 2.4.4. Y también tendrá la matriz generada a partir de la $g(I)$ mencionada anteriormente. Con estos datos iniciales se puede empezar a aplicar el algoritmo iterativamente.

Con cada iteración irá evolucionando la superficie para adecuarla a la $g(I)$, hasta que llegue el punto en el que no evolucione más la superficie y esto será el punto de parada del algoritmo. El criterio que se sigue para decidir que la evolución se ha quedado parada es que las diferencias entre superficies de iteraciones anteriores sea mínima.

Como resultado obtenemos una matriz de volumen que será necesario procesar para obtener un resultado suave y posteriormente generar una malla 3D.

2.4. Proceso intermedio

Las secciones anteriores explican el algoritmo de evolución de curvas utilizado, la generación de la $g(I)$ que es indispensable para el proceso de la evolución y la aplicación del algoritmo. Pero para llegar a estos resultados ha sido necesario procesar los datos que se nos han sido proporcionado.

Antes de poder aplicar la generación de la $g(I)$ (apartado 2.2) y el algoritmo de segmentación (apartado 2.1.4) es necesario procesar los datos proporcionados para el uso de estos algoritmos. Estos datos se ha extraído de mallas 3D (apartado 2.4.1), subdivisiones de estas las mallas (apartado 2.4.3), regiones de interés de la imagen original de cada una de las espinas(apartado 2.4.2), y el voxelizado de las mallas (apartado 2.4.4). Con estos procesos se tiene lo necesario para poder segmentar las imágenes. Una vez obtenida la nueva segmentación será necesario, generar nuevamente una malla 3D (apartado 2.4.5) para su visualización y evaluación de los resultados.

Todos estos procesos se detallan a continuación.

2.4.1. Parser VRML

La información de las reconstrucciones se obtienen a partir del software *Imaris FilamentTracer* que exporta los resultados en un archivo con formato VRML¹. Este archivo no sólo contiene información de estructuras 3D, sino también de textura, iluminación de la escena, etc. que en este caso no aporta información útil al proceso de reconstrucción posterior.

Este archivo también contiene información de la forma de la dendrita y todas las espinas dendríticas reconstruidas por *Imaris FilamentTracer*. Éstas están representadas en forma de puntos y de uniones entre esos puntos formando superficies o caras que les dan forma.

Al no ser un fichero muy utilizado, no había una biblioteca externa que cumpliese el objetivo de extraer puntos y vértices de una forma directa, por ello fue necesario hacer un parser² diseñado como una autómata de estados finitos específico (figura 2.2) para este problema y poder extraer puntos y vértices del fichero VRML, obteniendo mallas 3D de la dendrita y espinas dendríticas para un posterior procesado.

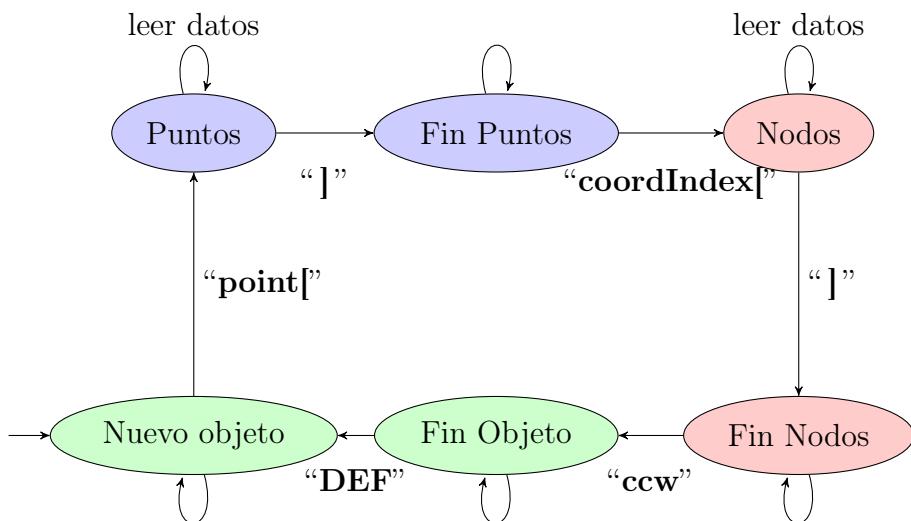


Figura 2.2.: Autómata del parser VRML.

Dentro del fichero VRML podemos encontrar la información en dos grupos, dendrita y espinas dendríticas. Dentro de estos dos grupos está la definición por separado de cada una de las mallas. En la definición de cada geometría podemos encontrar tanto la lista de puntos, la lista de la unión de estos puntos (caras), como otros elementos

¹VRML (Virtual Reality Modeling Language): formato que fue diseñado por el Consorcio Web3D para la representación de escenas y de objetos 3D en la web.

²Parser o analizador sintáctico, convierte el texto de entrada en otras estructuras para el análisis posterior.

2. SOLUCIÓN

que no aportan información al objeto que definen color o textura. La información de los puntos se muestra en una lista en el que cada elemento está separado por comas y está compuesto por tres números reales separados por espacios. La lista de unión de estos puntos es una lista de números naturales que simbolizan la posición del punto en la lista de puntos anterior. Cada cara está formada por cuatro puntos y la separación entre caras se hace por el número -1. Un ejemplo de la disposición de esta información en el fichero VRML se puede encontrar en el anexo B.

Después de la extracción, el formato en el que vienen las caras (formadas por vértices) es una lista de puntos agrupados de cuatro en cuatro puntos, es decir, la malla 3D está definida por polígonos de cuatro lados que conforman la superficie. Dado que esta representación es inconveniente para su posterior manejo, es necesario el procesamiento de las caras transformando estos polígonos de cuatro lados a polígonos de tres.

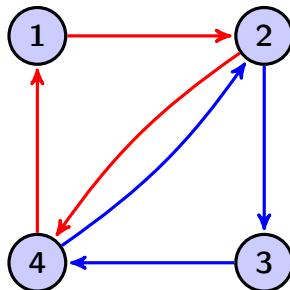


Figura 2.3.: Proceso de transformación de un polígonos de cuatro lados a dos de tres.

Mediante este paso se obtiene dos polígonos de tres lados a partir de un polígono de cuatro lados, como podemos observar en la Figura 2.3, podemos pasar de una cara formada por los puntos 1, 2, 3 y 4 a dos caras formados por los puntos 1, 2 y 4 (en rojo) y por los puntos 2, 3 y 4 (en azul).

Con esta transformación obtenemos la misma superficie y manteniendo la dirección de la normal, pero dividida en dos caras triangulares, que más tarde será más fácil de tratar.

Tras la extracción de los datos del fichero VRML y de la transformación de sus caras obtenemos para cada espina, puntos y uniones entre esos puntos (caras o superficies), que en conjunto hacen una malla 3D. Estas mallas se guardan por separado, obteniendo tantos ficheros como espinas y dendritas, para posteriores pasos del proceso.

2.4.2. Extracción de la región de interés

Con esta información ya extraída del fichero vrml, obtenemos unos puntos que tienen un origen de coordenadas y una escala distintas al stack de imágenes, ya que la escala de los puntos extraídos es la escala real de la dendrita y las espinas dendríticas. Por lo hay que adaptar los datos para que coincidan con el stack de imágenes que nos proporcionan con el fichero vrml.

El proceso necesita de unos parámetros que se obtienen del software *Imaris FilamentTracer*, que definen dónde se encuentra el origen de coordenadas en la imagen y la escala real de la imagen, es decir, nos dan los valores del tamaño real de un voxel de la imagen, con el que obtenemos un cambio de coordenadas y de escala para todos los puntos la malla 3D.

Este proceso es necesario para definir la zona de las imágenes donde se encuentra la espina que se va a procesar. A esta zona la llamaremos región de interés (ROI, Region Of Interest), que es una subregión de la imagen original donde se encuentra la espina. En posteriores pasos, la extracción de la ROI nos permitirá analizar sólo una porción de la imagen y así evitando tener que procesar todo el stack de imágenes.

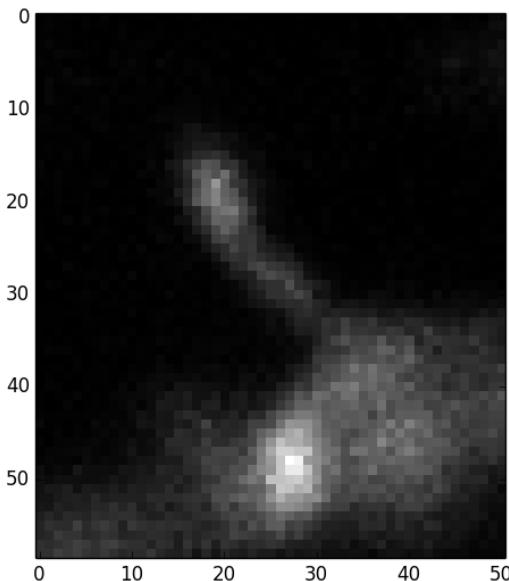


Figura 2.4.: Rodaja de la región de interés de una espina dendrítica donde podemos observar que en el centro de la imagen tenemos la espina dendrítica y en la parte inferior unida a la espina está la dendrita.

La ROI está delimitada por las coordenadas máximas y mínimas de los ejes de la malla 3D original. Pero las reconstrucciones que se obtienen de *Imaris FilamentTracer*

2. SOLUCIÓN

pueden ser incorrectas, como podemos ver en la figura 1.6, si se ciñese sólo a las coordenadas máximas y mínimas de la malla, podríamos dejar parte de la imagen que nos interesa fuera de la ROI y obtener una reconstrucción poco fiel a la imagen. Por esto la región de interés que extraemos tiene un margen de error, ampliando la delimitación original.

Este proceso se aplica a todos los ficheros obtenidos del proceso anterior, creando nuevos ficheros con el cambio de coordenadas y de escala en los puntos originales e incluyendo en este fichero nuevo, la región de interés y el resto de datos extraídos que no sufren modificaciones.

2.4.3. Extracción de anillos

Las mallas 3D que genera el software *Imaris FilamentTracer* se caracterizan por un patrón que forma anillos o elipses a lo largo de toda la malla 3D, como podemos ver en la figura 2.6. Estos anillos están formados por una serie de puntos que se pueden considerar parte del mismo conjunto porque están en el mismo plano y al unirlos tienen forma de elipse.

Todos los puntos de la malla siguen esta estructura y ningún punto está fuera de un anillo o elipse. También una característica de estos anillos es que siempre cuentan con el mismo número de puntos, que en nuestro caso son de 16. Separar los puntos de esta forma nos permite agrupar los anillos en niveles distintos de la espina, también nos proporciona información del nivel medio de gris de la imagen en ese anillo, que nos será útil para la reconstrucción posterior.

Para encontrar qué puntos pertenecen al mismo anillo necesitamos ver cada punto como un nodo que está unido a otros por medio de conexiones que forman las caras.

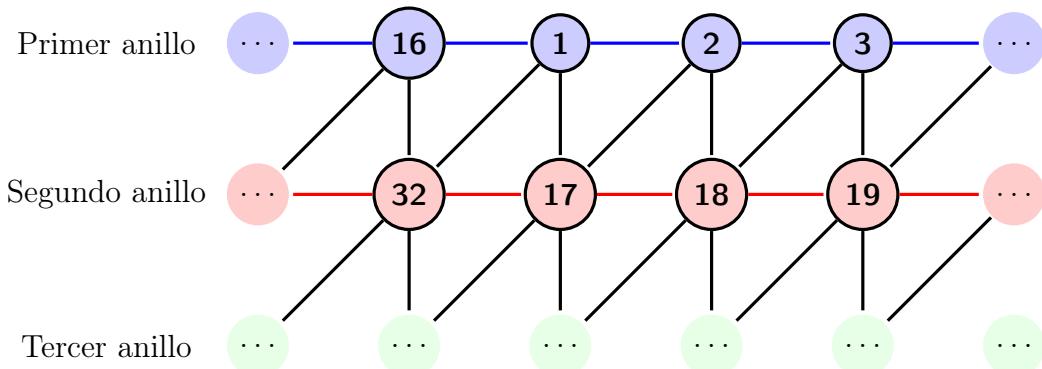


Figura 2.5.: Organización de los puntos en anillos.

Para empezar calculamos el número de conexiones de todos los nodos de la estructura y cogiendo los nodos con menor número de conexiones nos dará un conjunto de

2.4. Proceso intermedio

puntos entre los que se encuentran los puntos del primer y último anillo. Cogemos un punto y buscamos los vecinos del mismo que estén en el conjunto anterior, como resultado dará un conjunto de 16 puntos que llamaremos anillo. Este primer anillo será nuestro punto de partida para encontrar los siguientes anillos.

En la figura 2.5 podemos ver que los nodos con menores conexiones son los de la primera fila, cada nodo tiene 8 conexiones (hay que contar con la conexión doble de la figura 2.3) y si buscamos los vecinos de este con el menor número de conexiones obtendremos la primera fila de nodos y este conjunto será el primer anillo. En el algoritmo 1 cogemos un nodo con 5 conexiones y a partir de él generamos un conjunto de nodos que tengan 5 conexiones y que sean adyacentes entre sí.

```

1 continuar ← true;
2 nodo ← 0;
3 anillo ← set();
4 while continuar do
5   if conexiones(nodo) = 5 then
6     // Encontrado nodo del anillo inicial o final
7     anillo.add(nodo);
8     // Buscamos los nodos adyacentes al nodo que no estén en
      el conjunto anillo
9     adyac_nodo ← adyacentes(nodo) \ anillo;
10    continuar ← false;
11    for i ← adyac_nodo do
12      // De todos los adyacentes buscamos los del anillo.
13      if conexiones(i) = 5 then
14        anillo.add(i);
15        nodo ← i;
16        continuar ← true
17      end
18    end
19  else
20    // Seguimos buscando un nodo del comienzo o del final
21    nodo ← nodo + 1;
22  end
23 end

```

Algoritmo 1: Extracción del primer anillo

Con los puntos del primer anillo en un mismo conjunto, sacamos todos sus vecinos que no estén en el conjunto del primer anillo y obtendremos todos los puntos del siguiente anillo. En la figura 2.5 podemos ver que las conexiones en negro del primer anillo son conexiones con todos los puntos del segundo anillo.

Repetiendo el paso anterior pero como punto de partida el segundo anillo, todos

2. SOLUCIÓN

```
1 continuar ← true;
2 anillos ← [anillo,];
3 visitados = setanillo;
4 while continuar do
5     anillo_nuevo ← set();
6     // Cogemos el último anillo y recorremos sus nodos
7     for i ← anillos[-1] do
8         // Nodos adyacentes del nodo i que no estén en otros
9         // anillos.
10        nodos ← adyacentes(i) \ visitados;
11        anillo_nuevo.add(nodos);
12        visitado.add(nodos)
13    end
14    if length(visitados) = length(vértices) then
15        // Se han visitado todos los anillos de la malla 3D
16        continuar ← false;
17    end
18 end
```

Algoritmo 2: Extracción del resto de anillos

los vecinos del segundo anillo que no estén en el conjunto de puntos ya clasificados, pertenecerán al siguiente anillo. Este paso se repetirá sucesivamente hasta tener todos los puntos de la malla clasificados en anillos. En el algoritmo 2 está el proceso para obtener el resto de los anillos a partir del primer anillo.

Los conjuntos de anillos junto a toda la información que se ha generado antes la guardamos en un fichero para su posterior procesado.

2.4.4. Voxelización

Para poder aplicar el algoritmo de evolución de curvas es necesario tener un volumen de partida sobre el que poder evolucionar la superficie y adaptarse a la $g(I)$. Para ello es necesario pasar de una la estructura de puntos y caras (malla 3D) a una matriz de volumen por medio de la voxelización.

En la matriz de volumen cada voxel³ será binario, 0 o 1, significando 0 como que no hay volumen en ese voxel y de 1 como lo contrario. O dicho de otra forma, 1 si está dentro de la estructura 3D voxelizada y 0 si está fuera de esta estructura.

El proceso para voxelizar una estructura es el de definir el tamaño de la matriz y del

³Pixel 3D o unidad mínima procesable de una matriz tridimensional.

2.4. Proceso intermedio

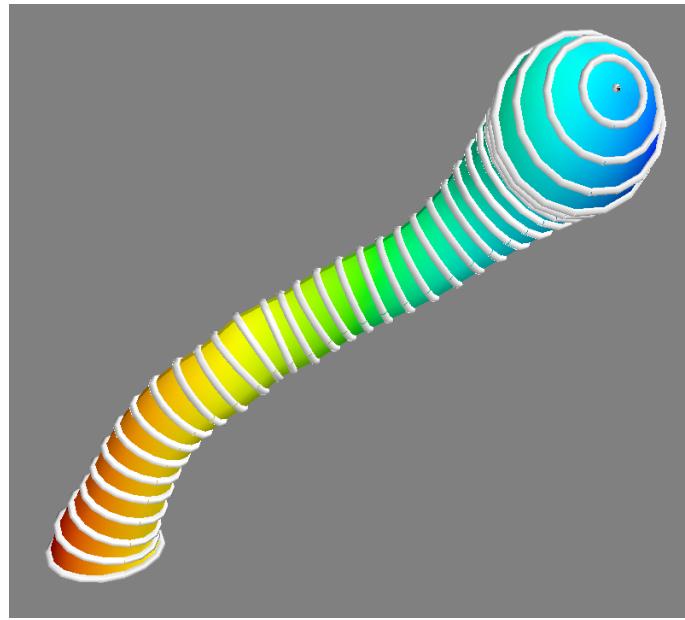


Figura 2.6.: Anillos de una espina dendrítica: en color blanco sobre la estructura de la espina.

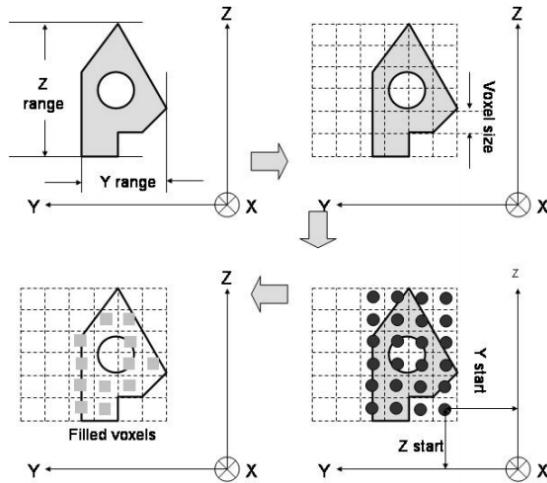


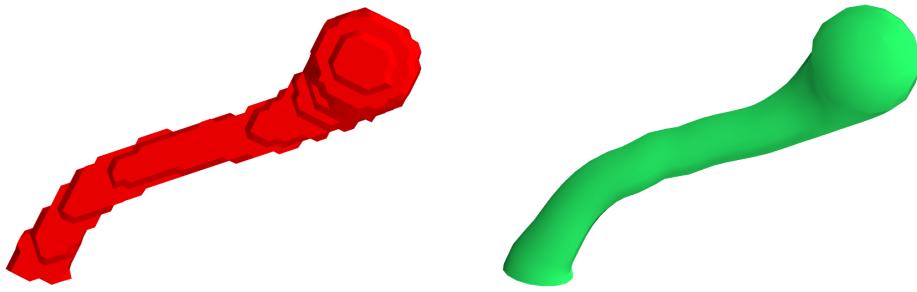
Figura 2.7.: Voxelización por intersección de superficies con rectas. Imagen obtenida de [17]

voxel, que en este caso será el mismo que el de la matriz de la imagen, paso 1 de la figura 2.7. Después hay que calcular la coordenadas máximas y mínimas que toma el modelo 3D para luego calcular intersecciones sólo en el rango de estas dimensiones, paso 2 de la figura 2.7. En el siguiente paso se trazan rectas sobre un eje y sólo en las dimensiones antes calculadas, paso 3 de la figura 2.7.

Para el último paso hay que calcular las intersecciones de las rectas, anteriormente trazadas, con las caras de todo el modelo. Si una recta no tiene ninguna intersección

2. SOLUCIÓN

significa que no hay volumen en esa región de la matriz. Si hay intersecciones, desde la primera intersección hasta la siguiente es la región donde habrá volumen y se llenará con unos. Si hay más intersecciones se hará lo mismo, la siguiente intersección marcará el comienzo del interior de la malla hasta la siguiente intersección. Este proceso se puede ver con el pseudocódigo del algoritmo 3.



(a) Voxelizado

(b) Sin voxelizar

Figura 2.8.: Voxelización de una espina dendrítica.

```

1 for i ← min eje X to max eje X do
2   for j ← min eje Y to max eje Y do
3     // Se calculan las intersecciones del modelo y el rayo
      trazado
4     intersecciones ← intersección(recta(i,j), estructura3D);
5     if hay intersecciones then
6       // Se pone en formato de unos y ceros
7       volumen[i,j] ← formato_binario(intersecciones);
8     end
9   end
10 end

```

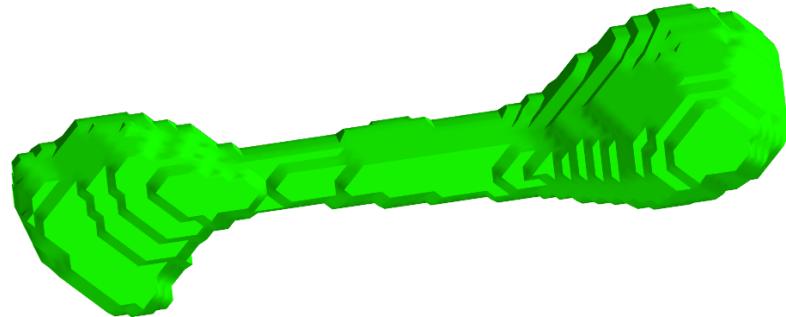
Algoritmo 3: Calcular la matriz de volumen.

Para realizar este proceso es necesario que el modelo sea una malla cerrada, es decir que no haya huecos que podría hacer que hubiera un número impar de intersecciones, obteniendo resultados incorrectos. Para ello, antes de voxelizar el modelo se procede a cerrar la malla en sus extremos, que son los únicos puntos en los que la mallas están abiertas. En el caso de la cabeza, hay un último anillo de pequeña dimensión y se puede convertir en un único punto sin alterar la forma de la malla. En el otro extremo se genera un punto central en el anillo y se crean caras que unen ese punto con el resto de puntos del anillo.

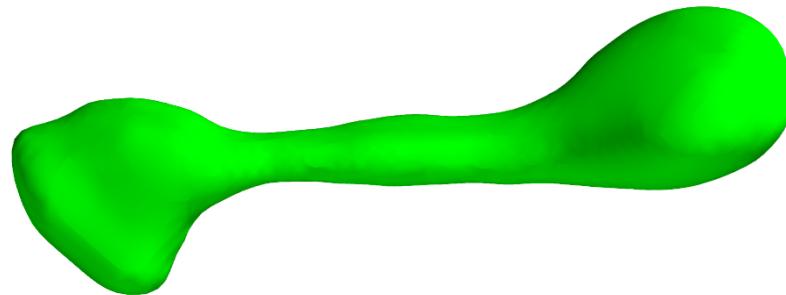
Al terminar se guarda la matriz de volumen en un fichero para su la evolución posterior.

2.4.5. Generación de una nueva malla y suavizado

Al final del proceso de evolución de curvas se obtiene una matriz de volumen, que es útil para el proceso de evolución pero es difícil de tratar para su visualización. Por esta razón se utiliza el algoritmo de Marching cubes [18] para extraer las superficies del volumen de la segmentación. El proceso hace que la malla resultante tenga *curvas* escalonadas, como se puede observar en la figura 2.9a, esto es debido al propio carácter de la matriz y al tamaño del voxel de dicha matriz. Este resultado a parte de no ser visualmente suave también puede llevar a un error en el cálculo del área de las espinas. Para evitar este efecto se tiene que suavizar la superficie y al mismo tiempo ceñirnos a la forma resultante de la segmentación. Para ello se aplica al volumen de la segmentación un algoritmo de suavizado iterativo como el expuesto en [19]. Una vez aplicado el algoritmo de suavizado se obtiene del algoritmo de Marching cubes una malla *suave* y sin variar en forma.



(a) sin suavizar



(b) suavizado

Figura 2.9.: Comparación del resultado obtenido del algoritmo Marching cubes.

2.4.6. Zoom de imágenes

Dado que las imágenes de partida son de baja resolución, el algoritmo de evolución de curvas puede no llegar a evolucionar bien en ciertas partes de reducido tamaño,

2. SOLUCIÓN

como el cuello de la dendrita. Por ello es necesario obtener una imagen de mayor resolución de la imagen de partida. El zoom de la imagen se hace con una función de la biblioteca de SymPy, que utiliza una interpolación polinómica de grado 3. Este zoom se aplica sobre las imágenes previamente filtradas para reducir el ruido de las mismas.

Según los experimentos realizados, los mejores resultados se obtienen con un factor de zoom de 4 veces el tamaño de la imagen original. Los detalles de los resultados se mostrarán en el apartado 3.2.

2.5. Detalles de la implementación

2.5.1. Detalles

El desarrollo del casi todo el proyecto se ha realizado en Python. Se ha elegido este por su rapidez en el desarrollo, aportando una gran agilidad a la hora de probar nuevos algoritmos. Otra razón es por las excelentes bibliotecas matemáticas disponibles, como NumPy y SymPy, que ofrecen una extensa cantidad de herramientas para el cálculo numérico.

Para la utilización del algoritmo sin parte visual será necesario el uso de este software

- Python (versión 2.7) – <http://www.python.org>
- NumPy – <http://www.numpy.org>
- SciPy – <http://scipy.org>
- Morphological Snakes – <https://github.com/pmneila/morphsnakes>
- pylibtiff3d – <https://github.com/joe-jordan/pylibtiff3d>

También se ha diseñado una herramienta software para la visualización de las espinas dendríticas con su región de interés y anillos, figura 2.10. Esta herramienta hace uso de la biblioteca gráfica QT, con la versión de Python PyQt, para la interfaz de usuario. Para la visualización 3D de las espinas se usa la biblioteca Mayavi.

Para esta herramienta mencionada y para las demás visualizaciones durante el proceso de evolución de curvas serán necesarios tener:

2.5. Detalles de la implementación

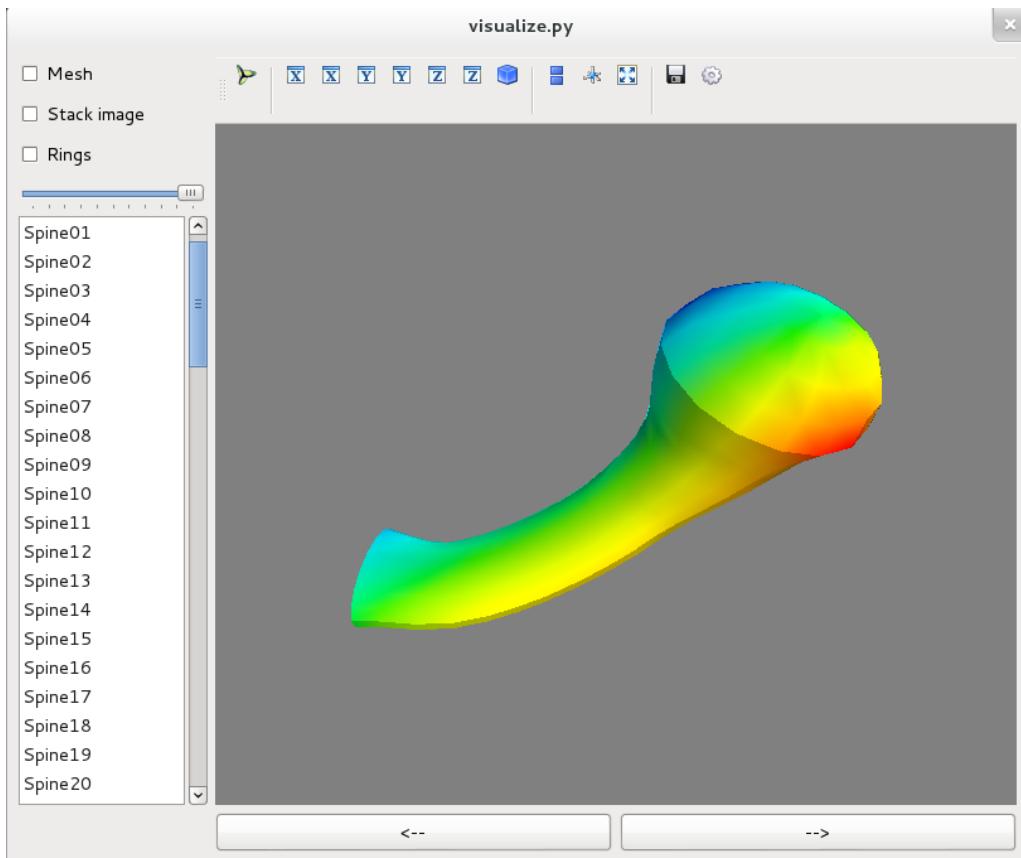


Figura 2.10.: Interfaz del visualizador de espinas dendríticas, ROI y anillos.

- QT (versión 4) – <http://qt-project.org>
- PyQt (misma versión QT) – <http://www.riverbankcomputing.co.uk/software/pyqt/>
- Matplotlib – <http://matplotlib.org>
- Mayavi – <http://docs.enthought.com/mayavi/mayavi/>

2.5.2. Mejora en rendimiento

La mejora que he realizado se ha centrado en incrementar la velocidad del algoritmo de generación de la $g(I)$. Esta optimización ha consistido en reimplementar la operación que más tiempo de cómputo llevaba hacer. Esta reimplementación se ha hecho en otro lenguaje de programación, Julia, al que se puede hacer una traducción casi directa del algoritmo implementado en Python, obteniendo mejores resultados.

2. SOLUCIÓN

Julia es un lenguaje de programación de reciente creación (2012), que fue diseñado para hacer operaciones numéricas con un alto rendimiento. No sólo se limita al cálculo numérico, si no que también fue pensado para que pudiese ser usado como lenguaje de programación de propósito general. Una de las ventajas de Julia es que fue diseñado para el paralelismo y la computación distribuida.

La elección de Julia frente a otras alternativas fue por la alta compatibilidad entre el lenguaje de programación principal del proyecto, Python, porque ambos lenguajes pueden interoperar sin problemas. Y por la velocidad de procesamiento que ofrece tras la traducción del núcleo del algoritmo.

A la hora de calcular la $g(I)$, el proceso de filtrado y ampliación de las imágenes sigue estando en Python, dado que Julia no cuenta aún con los elementos necesarios para realizarlos, tales como bibliotecas de interpolación en stack imágenes. Por esto, la carga de datos en el programa la hace Python, que los filtra y procesa. Después de preparar los datos, mediante el uso de una biblioteca Python que actúa como puente entre los dos, se ejecuta la función de Julia de generación de $g(I)$ con argumento los datos. El resultado se devuelve al entorno Python que se estaba ejecutando y se guarda la matriz resultado para el algoritmo de evolución de curvas.

Los resultados de la optimización se pueden ver en el apartado 3.3 de resultados.

Para la parte de optimización será necesario tener este software:

- Julia – <http://julialang.org>
- Distance – <https://github.com/JuliaStats/Distance.jl>

2.5. Detalles de la implementación

Como ejemplo, se puede ver que el código que se ha utilizado tanto en la versión Python y la versión Julia, no difieren mucho, pero los resultados de tiempos se mejoran significativamente (apartado 3.3).

Código Python

```
1 def th_centers(point, planes, points, thrings):
2     dist = [dist_p2p(plane, point) for plane in planes]
3     l = np.mean(dist)/4
4
5     gv = 0
6     norm = 0
7
8     for i,n in enumerate(dist):
9         ex = np.exp(-(n**2)/(l**2))
10        gv += thrings[i] * ex
11        norm += ex
12
13    return gv/norm
```

Código Julia

```
1 function th_centers(point, centers, points, thrings)
2     dist = pairwise(Euclidean(), transpose(centers),
3                      transpose(point))
4     l = mean(dist)/4
5
6     gv = 0
7     norm = 0
8
9     for (i, n) in enumerate(dist)
10        ex = exp(-(n^2)/(l^2))
11        gv += thrings[i] * ex
12        norm += ex
13    end
14
15    return gv/norm
end
```


3. EXPERIMENTACIÓN

3.1. Introducción

En esta sección se van a presentar los resultados del algoritmo propuesto en la sección anterior. La medida de éxito de la solución es en parte cualitativa, ya que las fronteras de las espinas dendríticas son difusas. Uno de los objetivos de este trabajo se basa en la subjetividad de la forma inicial, es decir que las reconstrucciones originales no coinciden con las imágenes de la fluorescencia. Las pruebas objetivas se basan en comprobar la centralidad del resultado obtenido comparado con la imagen de partida y con el modelo original.

3.2. Resultados

Los resultados los presentamos en dos apartados: medias globales y medidas locales. La medida global se hace por la comparación visual de las reconstrucciones en la imagen de la fluorescencia. Las medidas locales se hacen sobre rodajas transversales al eje central de la espina. Estas medidas se dividen en dos: la distancia entre centros y la simetría de la reconstrucción. La distancia entre centros consiste en calcular la distancia entre el punto de la imagen con mayor nivel de gris y los centros geométricos de la reconstrucciones. El punto con mayor nivel de gris lo consideremos como el punto en el que debería estar centrada la reconstrucción y en las figuras que vamos a presentar se representa como un punto azul. Los centros geométricos de las reconstrucciones se hacen sobre la intersección de la malla 3D y el plano elegido para su estudio. Esta medición nos indica el grado de centrado de los modelos respecto a la imagen y está medida en píxeles.

Otra medida de simetría es la desviación típica (σ) de los valores de gris de las reconstrucciones de un anillo. Este valor lo obtenemos a partir del nivel de gris que tiene la intersección de las mallas con la imagen. Esto nos muestra el grado de simetría de la reconstrucción respecto a la imagen, o dispersión de los niveles de gris a lo largo de la reconstrucción ya que todos los puntos de un anillo deberían tener el mismo nivel de gris. Esta medida se expresa en niveles de gris, con el rango de la

3. EXPERIMENTACIÓN

propia imagen, que toma valores entre 0 y 2^{16} .

Todas las espinas dendríticas presentadas a continuación se han evolucionado con un factor de zoom de 4 y las imágenes están tratadas con un filtro gaussiano con una desviación típica de 2, para quitar el ruido inicial. Para la representación de los anillos en las medidas locales, se ha decidido proyectar los anillos sobre la imagen. Esto permite ver la posición de los mismos respecto a la segmentación original. A continuación mostramos los resultados del proceso descrito para un conjunto de 4 espinas, seleccionadas por ser representativas de los 3 tipos de espinas mencionadas en el apartado 1.1

3.2.1. Espina dendrítica 11

Esta espina dendrítica es de tipo fina, con cuello largo y una cabeza no demasiado grande. La evolución original se queda corta en la cabeza siendo más corta de lo que debería. También la transición de la cabeza al cuello no es tan suave como lo que debería en las imágenes. En el cuello de esta espina, la reconstrucción original no respeta la forma de las imágenes siendo totalmente distinta,

Medidas globales

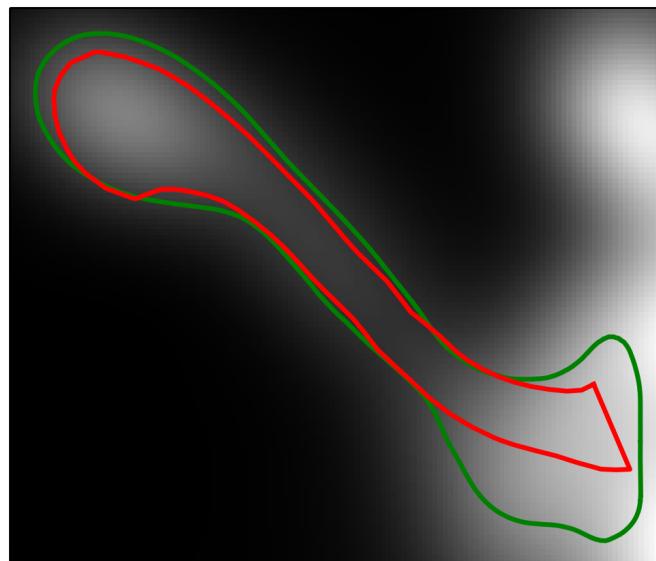


Figura 3.1.: Espina dendrítica 11. En rojo tenemos el corte de la reconstrucción original con el plano seleccionado. En verde tenemos la sección de la reconstrucción que hemos obtenido.

3.2. Resultados

Como se puede observar en la figura 3.1 la reconstrucción obtenida, en verde, se ajusta mejor a la imagen que la obtenida con el software Imaris.

Medidas locales

Se han elegido 4 planos para estudiar la reconstrucción obtenida y la de partida.

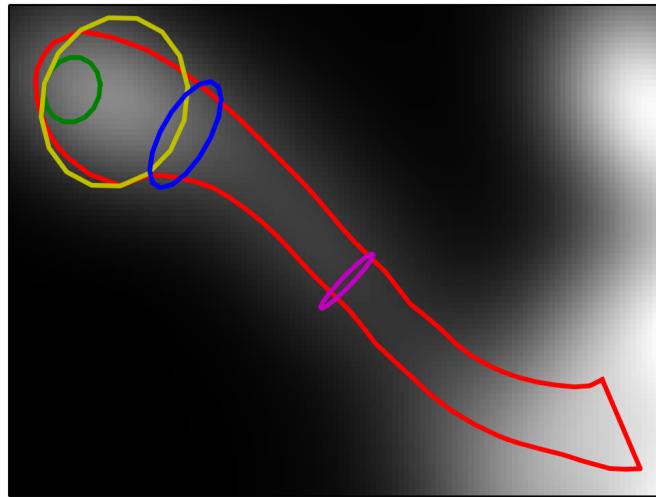


Figura 3.2.: Espina dendrítica 11. En rojo tenemos el corte de la reconstrucción original con el plano seleccionado. El resto de colores son las proyecciones sobre el plano de los anillos elegidos para su estudio.

Las medidas locales que vamos a estudiar son de los anillos representados en la figura 3.2. Los anillos han sido escogidos por las siguientes razones: espina 32, en verde, por ser el primer anillo de la espina (empezando desde la cabeza); el anillo 29, en amarillo, por representar la sección de máximo diámetro de la cabeza; anillo 25, en azul, el punto de cambio de cabeza a cuello; anillo 15, en magenta, un anillo del centro del cuello.

Anillo	Original		Resultado	
	Distancia	σ	Distancia	σ
32	5.39	1183.02	2.11	305.16
29	1.87	1326.44	2.59	284.05
25	3.52	975.72	0.55	107.30
15	0.87	276.47	0.73	79.10

Tabla 3.1.: Resultados de las medidas locales de la espina 11

Los resultados obtenidos en la tabla 3.1 muestran que se obtiene una mejora en cuanto al centrado de la espina dendrítica, excepto para el anillo 29 que no es un mal resultado si lo comparamos en la figura 3.3(b). Los resultados de la desviación

3. EXPERIMENTACIÓN

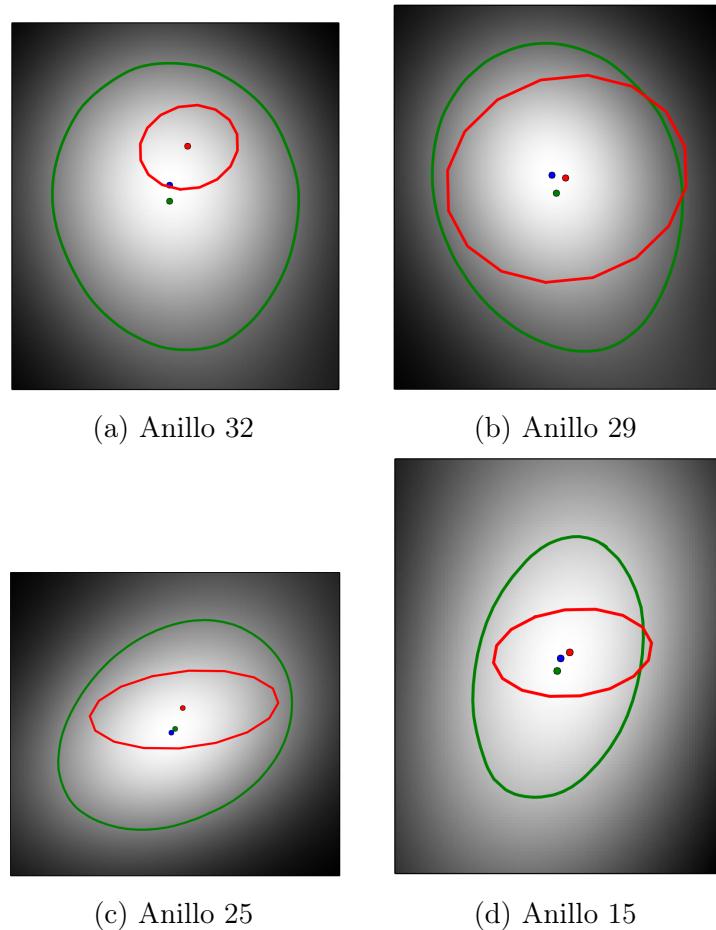


Figura 3.3.: Secciones transversales de la espina dendrítica 11. En rojo la reconstrucción original, en verde la reconstrucción obtenida. El punto azul es el nivel de gris más alto en la figura.

típica del nivel de gris mejoran a los de partida y como podemos observar en las figuras 3.1 y 3.3, las reconstrucciones se ajustan mejor a la forma de la espina.

3.2.2. Espina dendrítica 12

Esta espina dendrítica es de tipo corta y gruesa. Esta espina es interesante porque la reconstrucción original difiere mucho de la obtenida por la reconstrucción de nuestro método.

Medidas globales

Como se puede observar en la figura 3.4 la reconstrucción obtenida, en verde,

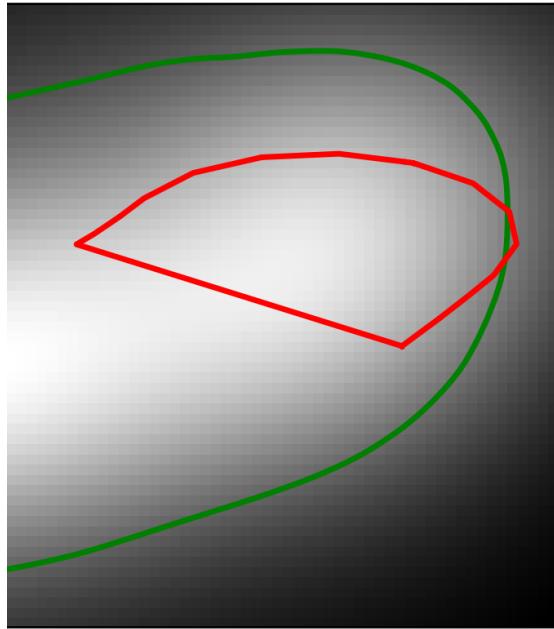


Figura 3.4.: Espina dendrítica 12. En rojo tenemos el corte de la reconstrucción original con el plano seleccionado. En verde tenemos la sección de la reconstrucción que hemos obtenido.

mejora la forma de la espina respecto a la obtenida con *Imaris FilamentTracer*. La reconstrucción obtenida representa mejor la estructura de la espina.

Medidas locales

Se han elegido sólo dos planos debido a su reducida dimensión, para estudiar la reconstrucción obtenida y la de partida.

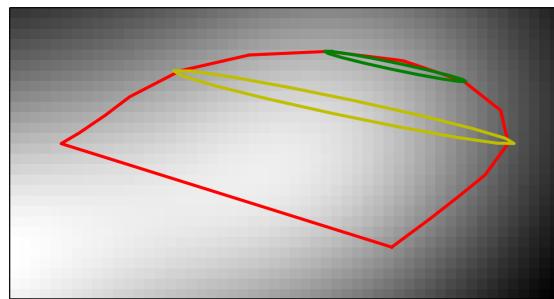


Figura 3.5.: Espina dendrítica 12. En rojo tenemos el corte de la reconstrucción original con el plano seleccionado. El resto de colores son las proyecciones sobre el plano de los anillos elegidos para su estudio.

Las medidas locales que vamos a presentar son de los anillos representados en la

3. EXPERIMENTACIÓN

figura 3.5. Se han escogido por las siguientes razones: anillo 6, en verde, por ser el primer anillo de la espina (empezando desde la cabeza); el anillo 4, en amarillo, por representar la sección central.

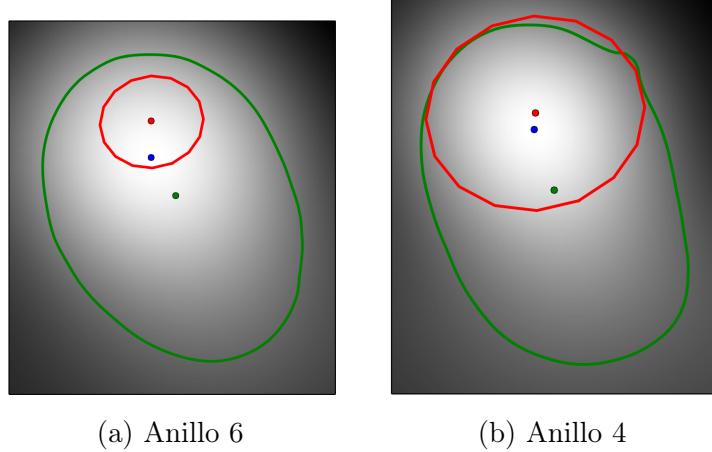


Figura 3.6.: Secciones transversales de la espina dendrítica 12. En rojo la reconstrucción original, en verde la reconstrucción obtenida. El punto azul es el nivel de gris más alto en la figura.

Anillo	Original		Resultado	
	Distancia	σ	Distancia	σ
6	5.49	2024.24	6.71	2222.85
4	2.85	3095.78	10.99	2219.79

Tabla 3.2.: Resultados de las medidas locales de la espina 12

Los resultados obtenidos en la tabla 3.2 muestran que se obtiene una mejora en la desviación típica. En el corte del anillo 4 se observa que la medida de la distancia obtiene un resultado peor que el de la reconstrucción original, pero como podemos observar la figura 3.6 la distribución del nivel de gris no es simétrico y el valor de la distancia en este caso no representa la centralidad del modelo. El valor de σ o en la figura 3.6 (b) representa mejor la centralidad en este caso y vemos que la reconstrucción se ajusta mejor.

3.2.3. Espina dendrítica 33

Esta espina dendrítica es de tipo fina. La reconstrucción se queda corta en la cabeza y no llega hasta el final de la espina. En el cuello, la reconstrucción no respeta el la forma que tiene la espina como podemos observar en el último corte de la figura 3.9.

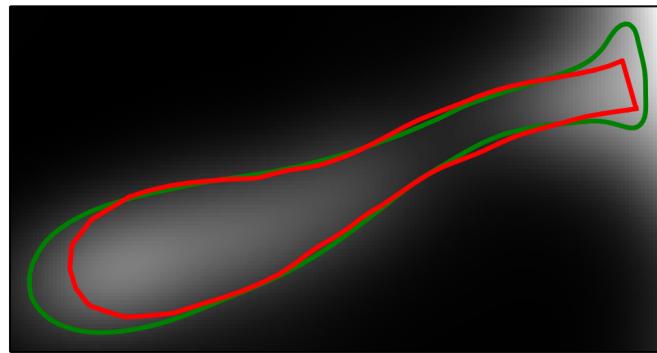


Figura 3.7.: Espina dendrítica 33. En rojo tenemos el corte de la reconstrucción original con el plano seleccionado. En verde tenemos la sección de la reconstrucción que hemos obtenido.

Medidas globales

Como se puede observar en la figura 3.7 la reconstrucción obtenida, en verde, la cabeza de la espina se ajusta mejor. Pero no sólo la cabeza, que es lo apreciable con el corte de la figura 3.7, sino con las secciones obtenidas y mostradas a continuación.

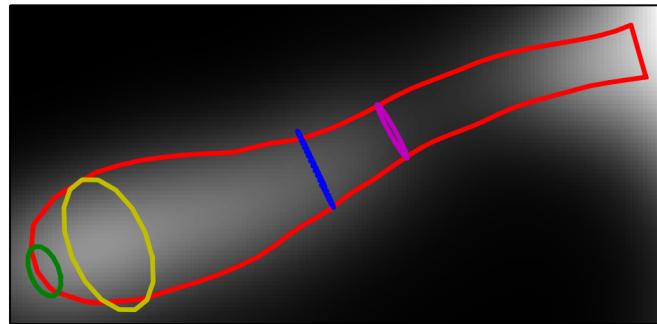


Figura 3.8.: Espina dendrítica 33. En rojo tenemos el corte de la reconstrucción original con el plano seleccionado. El resto de colores son las proyecciones sobre el plano de los anillos elegidos para su estudio.

Medidas locales

Se han elegido 4 planos para estudiar la reconstrucción obtenida y la de partida.

Las medidas locales que vamos a presentar son de los anillos representados en la figura 3.8. Se han escogido por las siguientes razones: espina 37, en verde, por ser el primero de la espina (empezando desde la cabeza); el anillo 33, en amarillo, por representar la sección de la cabeza; anillo 20, en azul, punto entre la cabeza y el cuello; anillo 15, en magenta, un anillo del cuello.

3. EXPERIMENTACIÓN

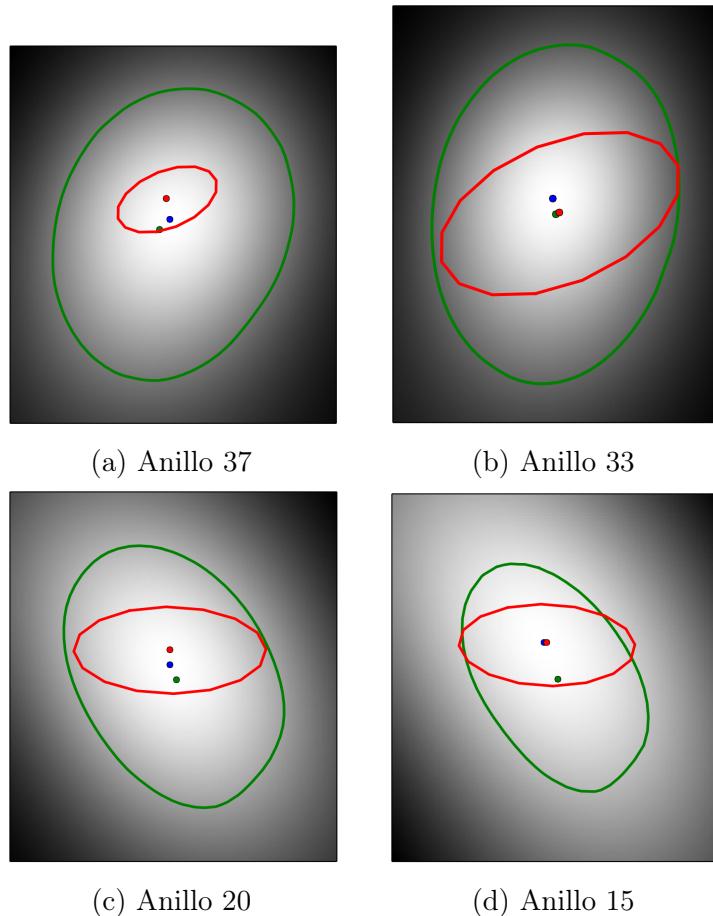


Figura 3.9.: Secciones transversales de la espina dendrítica 33. En rojo la reconstrucción original, en verde la reconstrucción obtenida. El punto azul es el nivel de gris más alto en la figura.

Anillo	Original		Resultado	
	Distancia	σ	Distancia	σ
37	2.86	421.10	1.75	117.88
33	2.32	1549.28	2.30	153.93
20	1.45	517.11	1.44	156.95
15	0.16	181.70	2.85	175.82

Tabla 3.3.: Resultados de las medidas locales de la espina 33

Los resultados obtenidos en la tabla 3.3 muestran que se obtiene una mejora en la cabeza de la espina dendrítica. Otra mejora respecto a la reconstrucción original es el grosor de la espina, como podemos observar en la figura 3.9, se ve incrementado ajustándose a la imagen que la original.

3.2.4. Espina dendrítica 45

Esta espina dendrítica es de tipo seta, tiene los mismos problemas que las anteriores espinas: la cabeza no llega hasta donde debería y el cuello es más estrecho de lo que debería.

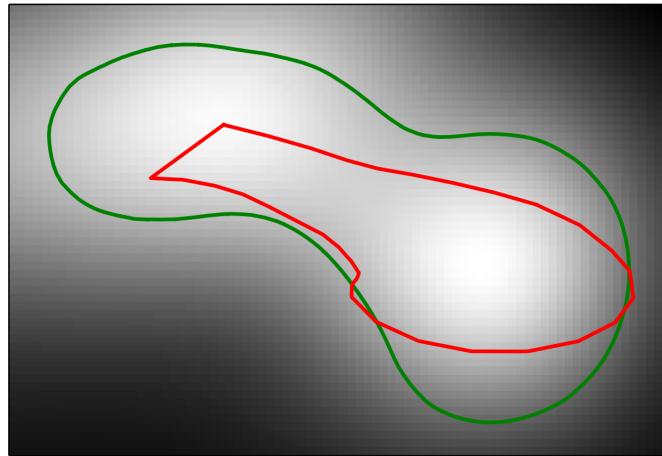


Figura 3.10.: Espina dendrítica 45. En rojo tenemos el corte de la reconstrucción original con el plano seleccionado. En verde tenemos la sección de la reconstrucción que hemos obtenido.

Medidas globales

Como se puede observar en la figura 3.10, la reconstrucción obtenida, en verde, se ajusta mejor a la imagen que la proporcionada por el software *Imaris FilamentTracer*.

Medidas locales

Se han elegido 4 planos, para estudiar la reconstrucción obtenida y la de partida.

Las medidas locales que vamos a presentar son de los anillos representados en la figura 3.8. Se han escogido por las siguientes razones: anillo 15, en verde, por ser el primer anillo de la espina (empezando desde la cabeza); el anillo 13 y 11, en amarillo y azul, por representar la sección de la cabeza; anillo 5, en magenta, un anillo del cuello.

Los resultados obtenidos en la tabla 3.4 muestran que se obtiene una mejora en el comienzo de la cabeza de la espina dendrítica. En el resto de la cabeza se ve que los

3. EXPERIMENTACIÓN

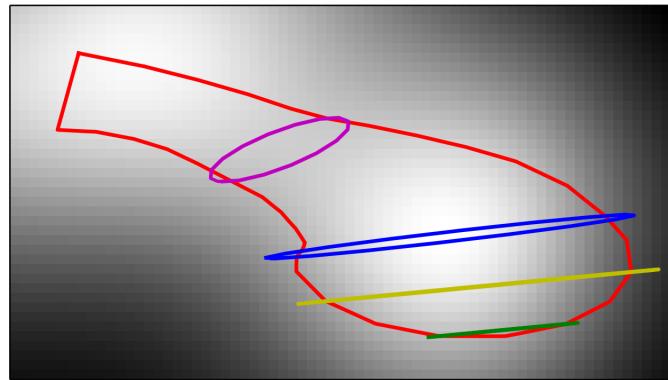


Figura 3.11.: Espina dendrítica 45. En rojo tenemos el corte de la reconstrucción original con el plano seleccionado. El resto de colores son las proyecciones sobre el plano de los anillos elegidos para su estudio.

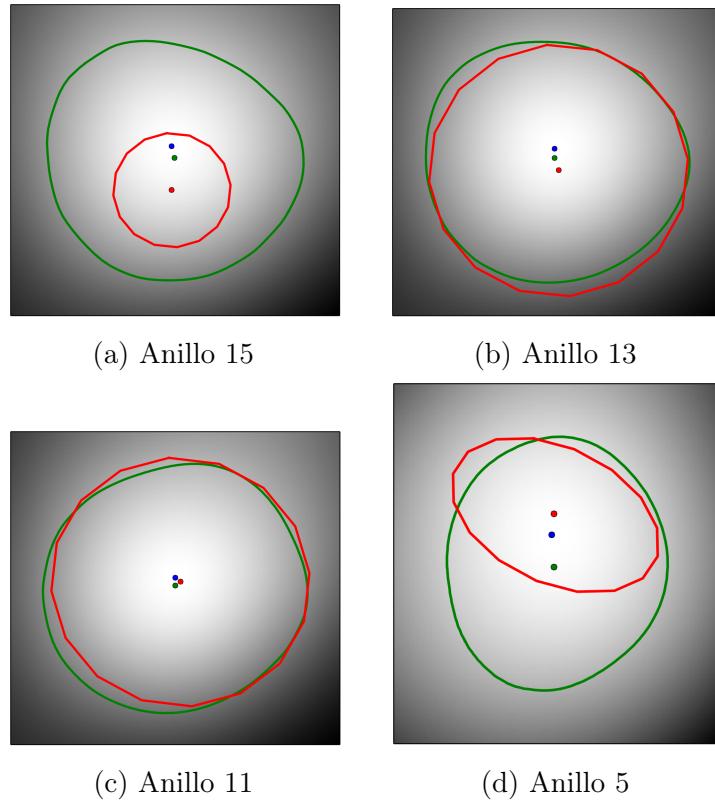


Figura 3.12.: Secciones transversales de la espina dendrítica 45. En rojo la reconstrucción original, en verde la reconstrucción obtenida. El punto azul es el nivel de gris más alto en la figura.

resultados son muy parecidos. En el cuello, en la figura 3.12 (d) las reconstrucciones se ajustan mejor a la forma de la imagen.

3.3. Resultados de la optimización

Anillo	Original		Resultado	
	Distancia	σ	Distancia	σ
15	5.89	1659.32	1.58	1385.67
13	3.20	2586.30	1.36	1888.04
11	0.87	1752.20	1.18	2067.33
5	2.18	607.16	3.33	654.08

Tabla 3.4.: Resultados de las medidas locales de la espina 45

3.3. Resultados de la optimización

Aquí se van a presentar los resultados de la optimización presentada en la sección 2.5.2, para las espinas de la sección anterior y para distintos niveles de zoom.

Python	x1	x2	x4
Espina 11	43.35 s	338.65 s	2740.90 s
Espina 12	9.49 s	74.81 s	604.79 s
Espina 33	48.60 s	383.39 s	3100.09 s
Espina 45	18.03 s	142.62 s	1152.31 s

Tabla 3.5.: Tiempos para calcular la $g(I)$ en el lenguaje Python.

Julia	x1	x2	x4
Espina 11	1.92 s	14.64 s	118.71 s
Espina 12	0.37 s	3.17 s	24.42 s
Espina 33	2.10 s	17.24 s	133.96 s
Espina 45	0.74 s	5.95 s	46.42 s

Tabla 3.6.: Tiempos para calcular la $g(I)$ en el lenguaje Julia.

Python/Julia	x1	x2	x4
Espina 11	22.57	23.13	23.08
Espina 12	25.64	23.59	24.76
Espina 33	23.14	22.25	23.14
Espina 45	24.36	23.96	24.82

Tabla 3.7.: Ganancia conseguida tras la optimización en Julia.

En las tablas 3.5 y 3.6 se muestra, en segundos, el tiempo de proceso de la función $g(I)$ respectivamente en Python y en Julia. En la tabla 3.7 se muestra la ganancia que se obtiene y las cifras muestran el número de veces que es más rápida la implementación en Julia frente a Python.

Las medidas se han obtenido ejecutando tres veces los algoritmos y cogiendo el mejor valor de los tres.

4. CONCLUSIONES

4.1. Conclusiones del proyecto

En este documento se ha presentado un algoritmo que permite mejorar las segmentaciones de las imágenes obtenidas en microscopio confocal a partir de los datos proporcionados por el software *Imaris FilamentTracer*, para obtener unas reconstrucciones precisas de la estructura 3D de las espinas dendríticas. También se ha presentado todo el proceso para llevar esto a cabo, tanto la lectura de los datos como la generación de estructuras necesarias para el algoritmo principal.

También se han creado herramientas para visualización y análisis, tanto de los datos de partida como de los resultados obtenidos de la segmentación realizada por nuestro algoritmo. Con estas herramientas hemos podido hacer un análisis detallado de los resultados para validarlos.

Por tanto, los objetivos que nos planteamos en un principio se han conseguido con éxito.

4.2. Futuras líneas de investigación

Ahora mismo el proceso de evolución depende fuertemente de unos parámetros que se especifican manualmente. Una futura línea de investigación se puede centrar en la detección automática de los parámetros que mejor se adapten a cada espina dendrítica.

El trabajo realizado toma como punto de partida la detección de las espinas realizadas por un software externo. El desarrollo de un algoritmo propio de detección de espinas dendríticas daría a este trabajo independencia de este software. Esta detección ha sido estudiada en [20] y la unión de ambos trabajos sería otra futura línea de investigación.

La última línea de trabajo intentaría mejorar el rendimiento de nuestro algoritmo.

4. CONCLUSIONES

Para ello mejoraríamos la implementación del algoritmo de evolución de curvas por operadores morfológicos. Una de las opciones sería mejorar el algoritmo actual y traducirlo a otro lenguaje compilado , en vez de la implementación Python actual.

Anexos

A. Soluciones descartadas

En esta apartado se van a explicar las distintas soluciones por la que ha pasado la investigación para llegar a la solución actual. Se van a presentar los estados por los que se ha pasado y los motivos para descartar cada una de las soluciones planteadas.

En cada apartado se va a utilizar la espina dendrítica estudiada en el apartado 3.2.1.

A.1. Puntos de inflexión

El primer intento de solución se basa en modificar la estructura 3D de la reconstrucción original. Se modifica cada anillo expandiendo o contrayendo individualmente cada punto en el plano del anillo. Se hace teniendo en cuenta el nivel de gris de la recta que une el centro del anillo con el punto, formando radios. Para cada punto de esa recta se mide el valor de gris de la imagen, desde el dentro del anillo pasando por el punto que se quiere modificar y continuando hasta tener suficientes valores. El resultado es una curva que se asemeja a una distribución normal y el punto de parada que definimos fue los puntos de inflexión de la curva y para ello usamos la primera derivada.

En la figura 4.1, cada línea en rojo representa el valor de gris de cada radio en un anillo, las líneas en magenta representan la primera derivada. La medida de distancia está normalizada, donde 1 es la distancia del centro al punto. Esto quiere decir, que de 0 a 1 es el valor de gris que está dentro de la reconstrucción y a partir de la distancia 1 son los valores que están fuera. En el caso de la figura 4.1, el anillo se va expandir ligeramente porque los puntos de inflexión de las curvas sobrepasan ligeramente la distancia 1.

El resultado obtenido por este método se puede observar en la figura 4.2. Como se puede apreciar no se acerca a la solución que se le quería dar, obteniendo una malla

4. CONCLUSIONES

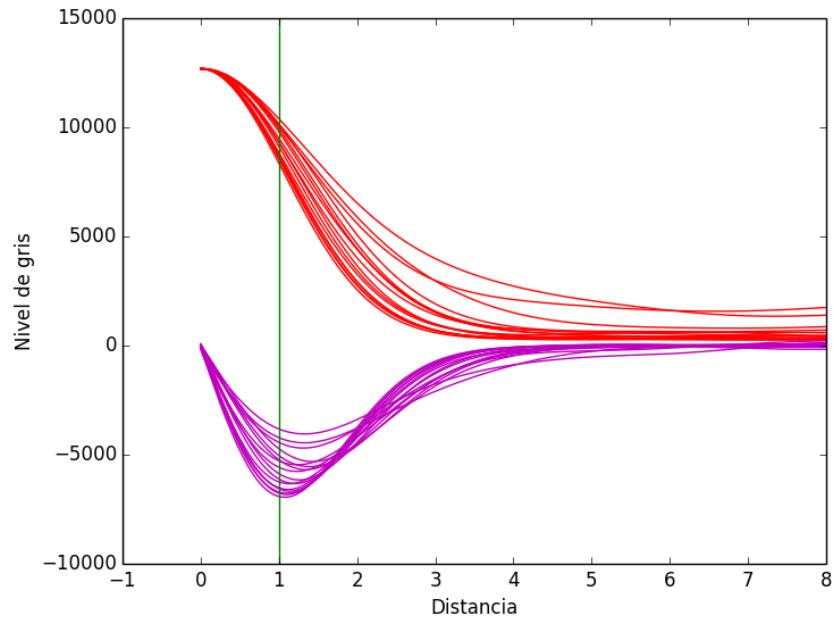


Figura 4.1.: Medición del valor de gris de los radios de un anillo.

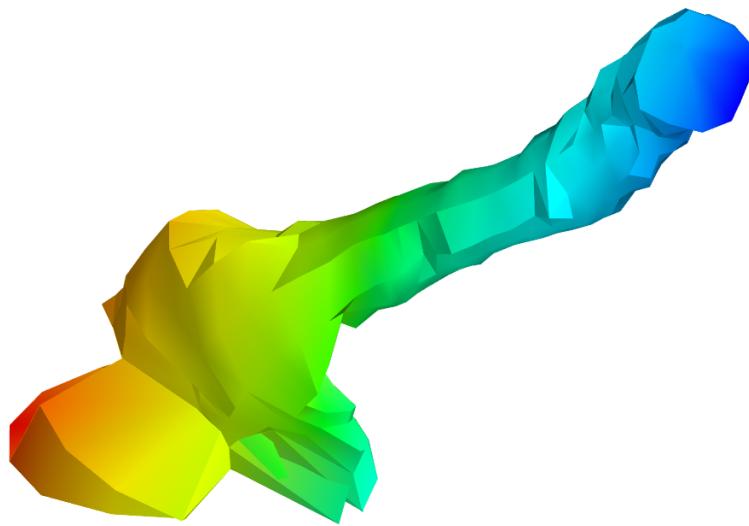


Figura 4.2.: Resultado de la solución por puntos de inflexión.

irregular en forma que no se adapta a la forma de la espina. La cabeza de la espina está mal reconstruida.

A. Soluciones descartadas

A.2. Puntos de inflexión v2

Una de las razones por la que el modelo anterior quedaba tan irregular era porque se modificaba cada punto de forma independiente con el valor de nivel de gris propio. Otro problema era que la cabeza de la espina siempre se quedaba corta y acaba teniendo una forma plana.

Para dar solución a estos problemas en vez de modificar los puntos individualmente se promedió la distancia de expansión o contracción de todos los puntos del anillo, haciendo que el anillo se modificase por igual a todos los puntos.

Para arreglar el problema de la cabeza lo que se hizo fue añadir anillos nuevos. Con esta solución se evitaba el corte que se obtenía en la cabeza de la reconstrucción anterior.

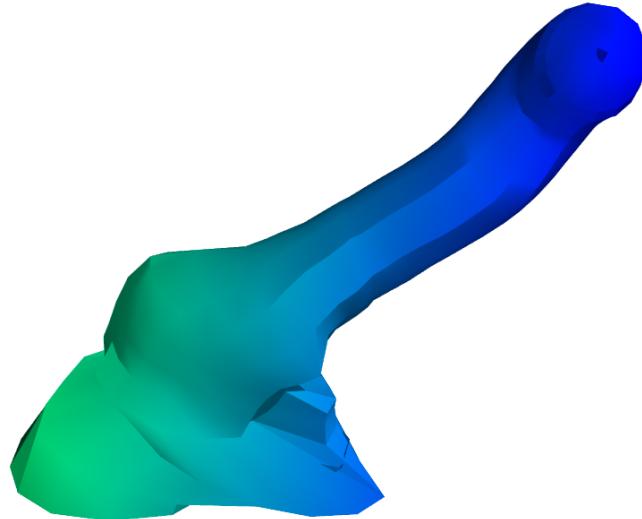


Figura 4.3.: Resultado de la solución por puntos de inflexión v2.

En la figura 4.3 se puede observar el resultado obtenido por esta nueva modificación. Se puede observar claramente que ahora la reconstrucción obtenida es más suave que la anterior y la cabeza no se corta. Pero el problema de esta reconstrucción es que la forma de la espina no es la que debería, la cabeza se reduce mientras que el cuello aumenta.

4. CONCLUSIONES

A.3. Evolución de curvas 2D

La reconstrucción anterior plantea que el método de segmentación que se estaba utilizando no era bueno y motivó la búsqueda de nuevos métodos más complejos para dar solución al problema. La siguiente solución se basa en la evolución de curvas presentada en este documento. La única diferencia es que la evolución se hace localmente sobre el plano 2D de cada anillo.

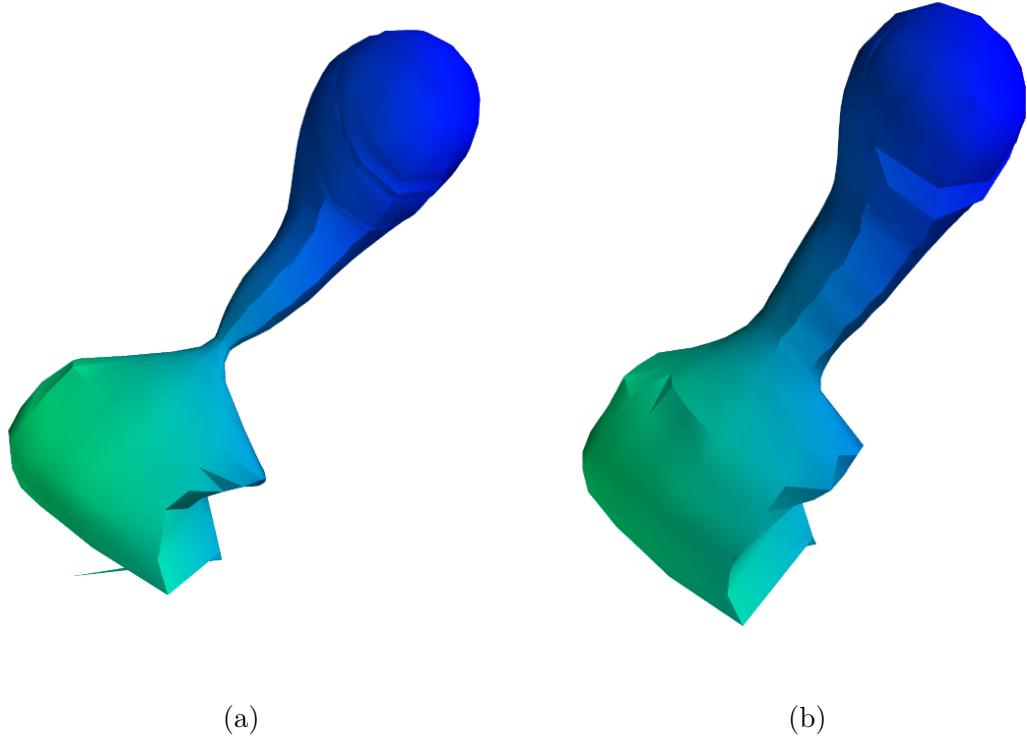


Figura 4.4.: Reconstrucción obtenida por evolución de curvas 2D. (a) tiene un mayor nivel de umbral que (b).

Este método mejora considerablemente la solución anterior, pero se basa en el mismo método, modificar los anillos de la estructura original para adaptarlos a la evolución 2D. Y por este motivo sigue teniendo el mismo problema, la falta de anillos en la cabeza, por lo que se tiene que añadir anillos nuevos en esta parte de la estructura para solucionarlo. Otro problema es la fuerte dependencia del nivel de umbral para la evolución. La elección de este valor se hacía manualmente y como ejemplo de una mala elección del umbral se puede observar en la figura 4.4 (a).

Esta solución es un puente entre la anterior y la evolución 3D, ya que utiliza la modificación del anillo pero con el algoritmo de evolución de curvas.

A.4. Evolución de curvas 3D

Para obtener mejores resultados el siguiente paso fue dejar de evolucionar sobre cada plano del anillo para evolucionar en toda el stack de imágenes y específicamente con el método de evolución de curvas 3D. Este paso da solución a la fuerte dependencia de la segmentación original, pero sigue siendo dependiente de un umbral para segmentar que es definido manualmente.

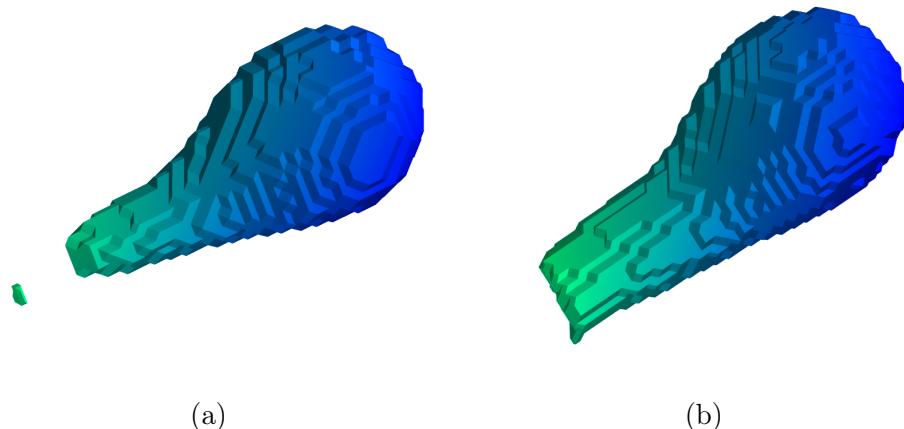


Figura 4.5.: Resultado de la evolución de curvas 3D. (a) tiene un nivel de umbral de 5000 frente a 4000 de (b).

Como podemos observar en la figura 4.5, la evolución 3D añade una mejora en la forma de la espina dendrítica pero en el primer caso (a) se estrecha demasiado en el cuello y en el segundo (b) se corta antes de llegar a la dendrita y esta reconstrucción también tiene un aspecto general demasiado expandido.

Encontrar un umbral fijo para estas evoluciones en algunos casos era totalmente imposible. Para umbrales bajos haría que la reconstrucción de la espina se partiese en el cuello, y para umbrales altos haría que la reconstrucción de la cabeza fuese demasiado grande, evolucionando más de lo que debería.

Estos resultados arrojaban la necesidad de hacer la evolución con niveles de umbral variables basados en la estructura original. De esta forma se obtienen buenas reconstrucciones tanto del cuello como de la cabeza de la espina.

4. CONCLUSIONES

B. Ejemplo VRML

```

1 DEF FilamentSegment700000000      Group {
2   children [
3     Group {
4
5       geometry
6         IndexedFaceSet {
7           coord
8             Coordinate {
9               point [ -21.059978 25.052097 4.0095067,
10                  -21.100483 25.043221 4.0359025,
11                  -21.127548 25.034418 4.0759792,
12                  -21.137051 25.027029 4.1236362,
13
14                  -20.631077 24.161316 3.6663611,
15                  -20.631134 24.161295 3.6663427,
16                  -20.631184 24.161259 3.6663411 ]
17
18   }
19
20   coordIndex [ 0, 1, 18, 17, -1, 1, 2, 19,
21                 18, -1, 2, 3, 20, 19, -1, 3,
22                 4, 21, 20, -1, 4, 5, 22, 21,
23                 -1, 5, 6, 23, 22, -1, 6, 7,
24
25                 301, 318, 317, -1, 301, 302, 319, 318,
26                 -1, 302, 303, 320, 319, -1, 303, 304,
27                 321, 320, -1, 304, 305, 322, 321, -1 ]
28
29
30   ccw TRUE
31   solid FALSE
32   convex TRUE
33   creaseAngle 0
34
35 }
36 }
37 ]
38 },
39
40 DEF FilamentSegment700000001      Group {

```

A pesar de comenzar en la línea 1, no es el comienzo del fichero VRML pero para ilustrar mejor el ejemplo y simplificarlo, numeramos a partir de ese punto.

B. Ejemplo VRML

Los fragmentos anteriores son los puntos más relevantes de la definición de una espina dendrítica contenida en el fichero VRML. El primer fragmento es el comienzo de la espina, que empieza dándole un nombre, *FilamentSegment700000000*, y agrupa los datos de la malla de esta espina. El segundo fragmento es el comienzo de la información que se extrae sobre la estructura 3D, es donde empieza la lista de puntos (línea 763) que finaliza en el tercer fragmento. En el cuarto y quinto fragmento es donde está la información de la unión entre los puntos anteriormente descritos, es decir, los vértices. El último fragmento marca el fin de la definición de la espina y el comienzo de la siguiente.

5. Bibliografía

- [1] M. F. Baguear, “Morphological study of dendritic spines,” Master’s thesis, Universidad Politécnica de Madrid, Facultad de Informática, 2011.
- [2] P. Marquez-Neila, L. Baumela, and L. Alvarez, “A morphological approach to curvature-based evolution of curves and surfaces,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 1, pp. 2–17, 2014.
- [3] L. Álvarez, L. Baumela, P. Henríquez, and P. Márquez-Neila, “Morphological snakes,” pp. 2197–2202, June 2010.
- [4] P. Márquez Neila, “Higher-order regularization and morphological techniques for image segmentation,” Ph.D. dissertation, ETSI _Informatica, 2014.
- [5] D. Purves, *Neurociencia / Neuroscience*. Editorial Medica Panamericana Sa de, 2007. [Online]. Available: <http://books.google.es/books?id=wjIhNQAAQAAJ>
- [6] Wikipedia, “Dendritic spine — wikipedia, the free encyclopedia,” 2014, [Online; visitado 17-Junio-2014]. [Online]. Available: http://en.wikipedia.org/wiki/Dendritic_spine
- [7] J. I. Arellano, A. Espinosa, A. Fairén, R. Yuste, and J. DeFelipe, “Non-synaptic dendritic spines in neocortex,” *Neuroscience*, vol. 145, no. 2, pp. 464–469, 2007.
- [8] A. Peters and I. R. Kaiserman-Abramof, “The small pyramidal neuron of the rat cerebral cortex. the perikaryon, dendrites and spines,” *American Journal of Anatomy*, vol. 127, no. 4, pp. 321–355, 1970.
- [9] Wikipedia, “Confocal microscopy — wikipedia, the free encyclopedia,” 2014, [Online; visitado 17-Junio-2014]. [Online]. Available: http://en.wikipedia.org/wiki/Confocal_laser_scanning_microscopy
- [10] J. G. White, W. B. Amos, and M. Fordham, “An evaluation of confocal versus conventional imaging of biological structures by fluorescence light microscopy.” *The Journal of Cell Biology*, vol. 105, no. 1, pp. 41–48, 1987. [Online]. Available: <http://jcb.rupress.org/content/105/1/41.abstract>

5. Bibliografía

- [11] B. I. for Advanced Science and Technology, “Laser scanning confocal microscopy — comparison of wide-field versus confocal microscopy methods.” 2014, [Online; visitado 19-Junio-2014]. [Online]. Available: https://itg.beckman.illinois.edu/technology_development/web_atlas/microscopy/confocal/
- [12] A. Blake and M. Isard, *Active Contours: The Application of Techniques from Graphics, Vision, Control Theory and Statistics to Visual Tracking of Shapes in Motion*, 1st ed. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1998.
- [13] M. Kass, A. Witkin, and D. Terzopoulos, “Snakes: Active contour models,” *International journal of computer vision*, vol. 1, no. 4, pp. 321–331, 1988.
- [14] V. Caselles, R. Kimmel, and G. Sapiro, “Geodesic active contours,” *International journal of computer vision*, vol. 22, no. 1, pp. 61–79, 1997.
- [15] R. Van den Boomgaard and A. Smeulders, “The morphological structure of images: the differential equations of morphological scale-space,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 16, no. 11, pp. 1101–1113, Nov 1994.
- [16] S. Osher and J. A. Sethian, “Fronts propagating with curvature-dependent speed: algorithms based on hamilton-jacobi formulations,” *Journal of computational physics*, vol. 79, no. 1, pp. 12–49, 1988.
- [17] S. Patil and B. Ravi, “Voxel-based representation, display and thickness analysis of intricate shapes,” *Ninth International Conference on Computer Aided Design and Computer Graphics (CAD-CG'05)*, vol. 0, pp. 415–422, 2005.
- [18] W. E. Lorensen and H. E. Cline, “Marching cubes: A high resolution 3d surface construction algorithm,” in *ACM Siggraph Computer Graphics*, vol. 21, no. 4. ACM, 1987, pp. 163–169.
- [19] V. Lempitsky, “Surface extraction from binary volumes with higher-order smoothness,” in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, June 2010, pp. 1197–1204.
- [20] J. M. Suárez Alvarado, “Automated multi-scale detection and segmentation of dendrites and spines from three-dimensional confocal microscopy images,” Master’s thesis, Universidad Politécnica de Madrid, Facultad de Informática, 2012.

Este documento esta firmado por

	Firmante	CN=tfgm.fi.upm.es, OU=CCFI, O=Facultad de Informatica - UPM, C=ES
	Fecha/Hora	Thu Jun 26 23:47:07 CEST 2014
	Emisor del Certificado	EMAILADDRESS=camanager@fi.upm.es, CN=CA Facultad de Informatica, O=Facultad de Informatica - UPM, C=ES
	Numero de Serie	630
	Metodo	urn:adobe.com:Adobe.PPKLite:adbe.pkcs7.sha1 (Adobe Signature)