RECONSTRUCCIÓN VOLUMÉTRICA Y VISUALIZACIÓN 3D DE ESTRUCTURAS ANATÓMICAS A PARTIR DE IMÁGENES MÉDICAS DE TOMOGRAFÍA.

JOHAN ALBERTO ARZUZA NARVAEZ

UNIVERSIDAD INDUSTRIAL DE SANTANDER

MAESTRÍA EN INGENIERIA.

ÁREA CIENCIAS DE LA COMPUITACIÓN E INFORMÁTICA

FACULTAD DE INGENIERÍAS FISICO-MECÁNICAS

ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

2006

RECONSTRUCCIÓN VOLUMÉTRICA Y VISUALIZACIÓN 3D DE ESTRUCTURAS ANATÓMICAS A PARTIR DE IMÁGENES MÉDICAS DE TOMOGRAFÍA.

JOHAN ALBERTO ARZUZA NARVAEZ

Proyecto de grado para optar al título de
Magíster en informática y ciencias de la computación

Director UIS
ENRIQUE SARMIENTO MORENO

UNIVERSIDAD INDUSTRIAL DE SANTANDER
MAESTRÍA EN INGENIERIA.
ÁREA CIENCIAS DE LA COMPUITACIÓN E INFORMÁTICA
FACULTAD DE INGENIERÍAS FISICO-MECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
2006

AGRADECIMIENTOS

Dedico este trabajo a Dios, por sustentar mi vida.
A mi familia por ser mi apoyo incondicional.
A mi director de proyecto por su confianza, comprensión y apoyo.
A todos los que de una u otra forma me enseñaron cosas en la vida en este periodo de maestría.

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

TITULO: Reconstrucción volumétrica y visualización 3D de estructuras anatómicas a partir de imágenes médicas de tomografía.[1]
AUTOR: Johan Alberto Arzuza Narvaez [2]

PALABRAS CLAVES: Segmentación de imágenes, voxel, computación gráfica, visualización, imágenes médicas.

CONTENIDO:

El estudio de la representación tridimensional aplicado a la medicina se ha expandido considerablemente en los últimos años debido a que puede ser de mucha utilidad para la asistencia de los profesionales en diferentes tipos de procedimientos médicos, como planeaciones de cirugía, ayudas diagnósticas con procedimientos no invasivos, o propósitos de enseñanza de anatomía y patología, partiendo de la aplicación del procesamiento de imágenes de tomografía o Resonancia Magnética.

El objetivo de todas las modalidades de imagen médica es visualizar los órganos internos del cuerpo para obtener información estructural y anatómica, de una manera clara y sin poner en riesgo la integridad del paciente. Existen varias herramientas que ayudan al diagnóstico médico, y aunque la imagen bidimensional es la más frecuente, todas ellas están sujetas a la subjetividad y dependen de la experiencia del especialista, que debe reconstruir mentalmente la estructura 3D a partir de las secuencias de imágenes 2D, para obtener un diagnóstico o proceder a una intervención quirúrgica; por esta razón se proponen nuevas técnicas más objetivas, como la imagen 3D, que permite una visualización realista de las estructuras anatómicas, dando la posibilidad de interactuar con los modelos tridimensionales e incluso utilizarlos con fines de simulación.

La habilidad de detectar estructuras con características determinadas dentro de una imagen, conocido como segmentación de imágenes, constituye un aspecto fundamental para la automatización de la visualización tridimensional de volúmenes de interés. El procesamiento de imágenes, que implica un buen algoritmo de segmentación, es la clave para la posterior representación tridimensional de los objetos a reconstruir, por este motivo, el contenido de la investigación realizada se basa en dos actividades. La primera describe cómo a partir del procesamiento digital de las imágenes de Tomografía Computarizada (TC) se puede obtener otro conjunto de imágenes que contienen un subgrupo de órganos de interés que posteriormente puedan ser representados en un espacio tridimensional; la segunda actividad trata sobre el desarrollo de una herramienta software de Visión Artificial donde se aplican algunos algoritmos clásicos y otros desarrollados por los autores para segmentación y visualización 3D, que permite reconstruir estructuras anatómicas a partir de un conjunto de imágenes 2D, como herramienta de apoyo y ayuda en el diagnóstico médico.

---

[1] Trabajo realizado como parte del proyecto de Investigación correspondiente a los estudios de Maestría en Ingeniería, área informática y ciencias de la computación. Universidad Industrial de Santander, Bucaramanga, Colombia.

[2]Ingeniero de Sistemas, candidato a Magíster en informática y ciencias de la computación. Universidad Industrial de Santander, Bucaramanga, Colombia. (e-mail: johanarzuza@hotmail.com)

TITLE: Volumetric reconstruction and 3D visualization of anatomical structures from medical tomography images[3]
AUTHOR: Johan Alberto Arzuza Narvaez [4]
KEYWORDS: Image Segmentation, voxel, Computer Graphics, Medical visualization, Images.

CONTENT:

Three-dimensional representation studies applied to medicine has expanded considerably in last years because it can be useful for professionals aid in different types from medical procedures, like surgery planning, diagnóstical aids with noninvasive procedures, or intentions of anatomy education and pathology, starting off the application of the image processing from tomography or Magnetic Resonance.

The objective of all the modalities of medical image is to visualize the internal body's organs to obtain structural and anatomical data, since a clear way and without putting in risk the integrity of the patient. Several tools exist that help the medical diagnosis, and although the bidimensional image is most frequent, all of them is subject to the subjectivity and depends on specialist's experience, who must reconstruct the 3D structure mentally since sequences of 2D images, to obtain a diagnosis or behavior to an operation; therefore new more objective techniques set out, like the 3D image, that allows a realistic visualization of the anatomical structures, giving possibility for interacting with the threedimensional models and even use them with simulation aims.

The ability to detect structures with characteristics within an image, known like segmentation of images, constitutes a fundamental aspect for automatization for the three-dimensional volume's visualization of interest. The image processing, that implies a good algorithm of segmentation, is the key for the later three-dimensional representation of the objects to reconstruct, for this reason, the content of this investigation is based on two activities. First it describes how from the digital processing images from Tomography, another set of images can be obtained that contain a sub-group of interest organs which later they can be represented in a three-dimensional space; the second activity treats on the development of a software tool of Artificial Vision where to some classic algorithms and others developed by the authors for segmentation and 3D visualization are applied, that allows to reconstruct anatomical structures from a set of 2D images, like tool of support and aid in the medical diagnosis.

[3] Work made like part of the project of Investigation corresponding to the studies of Masters in Engineering, computer science area and sciences of the computation. Industrial university of Santander, Bucaramanga, Colombia.

[4] Engineer in computer science, candidate to Master in computer science. Industrial university of Santander, Bucaramanga, Colombia (email: johanarzuza@hotmail.com)

1. PLANTEAMIENTO DEL PROBLEMA

INTRODUCCIÓN

El estudio de la representación tridimensional (3D) en medicina se ha expandido considerablemente en los últimos años. En el campo de la visualización en 3D puede ser de mucha utilidad en planeaciones de cirugía, ayudas diagnósticas o propósitos de enseñanza de anatomía y patología, partiendo de estudios de imagenología tales como ultrasonido, tomografía axial computarizada, resonancia magnética nuclear, y medicina nuclear, de los cuales se extrae información anatómica o funcional del órgano en estudio.
Las posibilidades de despliegue que presentan actualmente la mayoría de las estaciones de trabajo han permitido obtener resultados espectaculares en el dominio de la representación 3D. Sin embargo, en la mayoría de los sistemas existentes actualmente, se ha dado énfasis a la parte algorítmica, en detrimento de la calidad de la representación y visualización; esta situación se agrava considerando el caso específico de la aplicación a las imágenes médicas, puesto que el médico se basa en el resultado en 3D para emitir un diagnóstico.

Este proyecto engloba dos actividades. La primera actividad describe cómo las imágenes de Tomografía Computarizada (TC) pueden ser representadas de manera tridimensional; la segunda actividad trata sobre el desarrollo de un subsistema computarizado que permita visualizar esta representación tridimensional. Para cumplir con estos objetivos, se hace una definición básica de los conceptos, métodos y algoritmos del área de Análisis de Imágenes y Visión por computadora. Específicamente, se presentan temas relacionados a la segmentación de imágenes y reconstrucción tridimensional. Como resultado final de este proyecto se pretende obtener un sistema de reconstrucción y visualización de imágenes médicas 3D.

Se pueden definir las diversas etapas que intervienen en la reconstrucción tridimensional de un corte tomográfico, desde su obtención en el equipo correspondiente hasta su despliegue en una computadora.

- **Adquisición.** Normalmente, se obtiene la imagen en dos dimensiones directamente del equipo cuya modalidad nos interesa utilizar. En este proyecto las imágenes son obtenidas de la base de datos de la Librería Nacional de Medicina de Estados Unidos (NLM) de su proyecto *Visible Human Project* (VHP)[5].
- **Registro.** Esta etapa consiste en ubicar en un mismo plano de referencia una serie de imágenes que contienen al órgano de interés. Al considerar imágenes tomográficas, el problema radica en que cada una de éstas tiene un plano de referencia propio, por lo que es indispensable en estos casos, la alineación de las imágenes.
- **Segmentación.** La segmentación consiste en extraer objetos de interés a partir de las imágenes en tonos de gris. Este tema ha sido ampliamente investigado, pero no se ha llegado a obtener un método universal que clasifique automáticamente cualquier estructura anatómica de interés, a partir de cualquier modalidad de imagen, de ahí que se continúe esta investigación según la aplicación.
- **Interpolación.** Los contornos obtenidos en la etapa precedente deben ser alineados en pila y en caso de que la resolución sea insuficiente, se requiere aplicar un algoritmo de interpolación entre datos para las dimensiones deseadas. Dentro de los métodos de interpolación existen aquellos que resaltan características de calidad o de tiempo de ejecución. Es necesario realizar un estudio para determinar el mejor método de interpolación que se adapte a la aplicación que en particular se desee desarrollar.
- **Presentación de resultados en 3D.** La etapa final de la reconstrucción es la que determina su calidad al poder visualizar en un espacio tridimensional las geometrías obtenidas. Consiste en desplegar el volumen del órgano en estudio o de una de sus estructuras en la pantalla de la computadora. Los dos grandes campos en la representación tridimensional de imágenes médicas son la representación en superficie **(surface rendering)** y la representación en volumen **(volume rendering)**. En representación en superficie, la visualización y manejo de datos tridimensionales se basa únicamente en los límites del objeto y el despliegue de la superficie del objeto de interés está aislado del resto del volumen. En representación en volumen, se persigue la representación de múltiples estructuras con diferentes tipos de tejido. A

[5] http://www.nlm.nih.gov/research/visible/visible_human.html

cada voxel[6] se le asocia un nivel de opacidad y se asume que el valor de cada uno está correlacionado con el tipo de material que lo integra. Para cada versión de despliegue se examina cada voxel, lo que hace a esta representación muy costosa computacionalmente. El tema de las ventajas y desventajas de uno u otro tipo de representación se ha convertido en controversia, pero se puede hacer una diferencia drástica entre los dos: la representación en volumen es muy útil cuando las estructuras en la escena son difusas, como puede ser un cáncer dentro de un tejido; por otro lado la representación en superficie pierde la información del material del objeto que lo limita para fines de comparación con otros tejidos. La gran ventaja de tener únicamente información sobre los límites del objeto es que permite una más rápida interacción y manipulación de los datos, además de obtener una aproximación mejor al detalle y textura de la superficie del objeto. Para realizar la representación en 3D se efectúan las transformaciones de movimiento requeridas, la eliminación de áreas ocultas, la asignación de sombreado o transparencia y la proyección desde el ángulo deseado. Actualmente existen paquetes comerciales para cualquier estación de trabajo que realizan funciones de despliegue de estructuras en 3D de una manera automática, principalmente para aplicaciones de imágenes sintéticas. Sin embargo, no cualquier método de sombreado entrega un resultado acorde con lo que se vería en la realidad. Nuevamente, es necesaria una evaluación de los métodos empleados en esta etapa, de acuerdo a criterios cualitativos y cuantitativos del especialista.

En el presente proyecto se pretende obtener un sistema de reconstrucción y visualización de imágenes médicas 3D a partir de un conjunto de imágenes 2D, aplicando los pasos antes descritos.

1.1 DEFINICIÓN DEL PROBLEMA

La computación científica es una rama de la informática que ha venido evolucionando muy rápidamente en los últimos años, ya que ésta avanza a la par de los resultados en las ciencias existentes. La medicina es uno de los campos que se ha visto muy

[6] Voxel: Contracción de «volume element». Es la unidad cúbica que compone un objeto tridimensional. Constituye la unidad mínima procesable de una matriz tridimensional y es el equivalente, en un objeto 3D, del píxel.

beneficiado por la computación en las últimas décadas, y aunque es una ciencia compleja, se ha visto favorecida por el desarrollo de muchas herramientas como ayuda al diagnóstico de enfermedades.

La tomografía axial computarizada es un ejemplo muy claro del avance de la medicina a partir de la tecnología electrónica e informática. Una tomografía consiste en la formación de una imagen con información anatómica obtenida de un corte transversal del cuerpo; cada imagen es producida por la síntesis computarizada de los datos de transmisión radiográfica de muchas direcciones diferentes sobre un plano determinado. La tomografía se realiza a partir de un escáner, el cual es un instrumento que realiza un tipo especial de radiografía, llamada TAC. En ella, en lugar de un sólo haz de rayos a través del cuerpo (como en una radiografía normal), se envían muchos haces simultáneamente desde diferentes ángulos, que son analizados con ayuda de un computador para combinar las diferentes imágenes del área que requiere el análisis en una sola.

El computador utiliza toda esa información para calcular la densidad relativa de los tejidos que son explorados, ofreciendo una imagen en dos dimensiones en blanco y negro, de diferentes cortes transversales del cuerpo, imagen que se imprime sobre película sensible, la misma que se utiliza para imprimir radiografías, que tiene la ventaja de ser semitransparente, por lo que es más fácil de examinar al someterla a una luz desde atrás.

Un TAC ofrece muchos más detalles que una radiografía corriente, y por lo tanto proporciona al médico mucha más información sobre la anatomía interior del cuerpo. Hoy en día, los TAC son fundamentales para localizar tumores y planificar su tratamiento, ya sea con radioterapia, quimioterapia o cirugía. Originalmente el escáner se diseñó para obtener imágenes del cerebro, pero hoy en día la técnica está más avanzada y permite obtener imágenes de cualquier parte del cuerpo. Recientemente se ha probado que la tomografía puede resultar de gran valor para estudiar la estructura y el comportamiento de reconstrucciones y fijaciones de huesos y prótesis.

A pesar de que las imágenes de tomografía son de gran ayuda en la medicina, se necesita mucha experiencia y destreza para poder visualizar y determinar anomalías existentes partiendo sólo de un conjunto de imágenes de secciones de una parte

específica del cuerpo, y a menudo hay muchos detalles que se pasan por alto al visualizar un modelo de varias imágenes bidimensionales en tonos de blanco y negro, especialmente porque cada corte se muestra a escala reducida.

La base de la solución al problema que se plantea consiste en obtener un modelo tridimensional de un conjunto de objetos a partir de información parcial de los mismos dada por imágenes digitalizadas de una secuencia de cortes paralelos, específicamente tomografías. La Figura 1.1 muestra un esquema de la solución propuesta; las ventajas de ésta serían una visualización mucho más cercana a la realidad y la posibilidad de manipular el modelo para acercar, rotar, habilitar o deshabilitar las estructuras que son de interés, aislándolas de las que en algún momento no sean de utilidad en el estudio que se esté realizando.

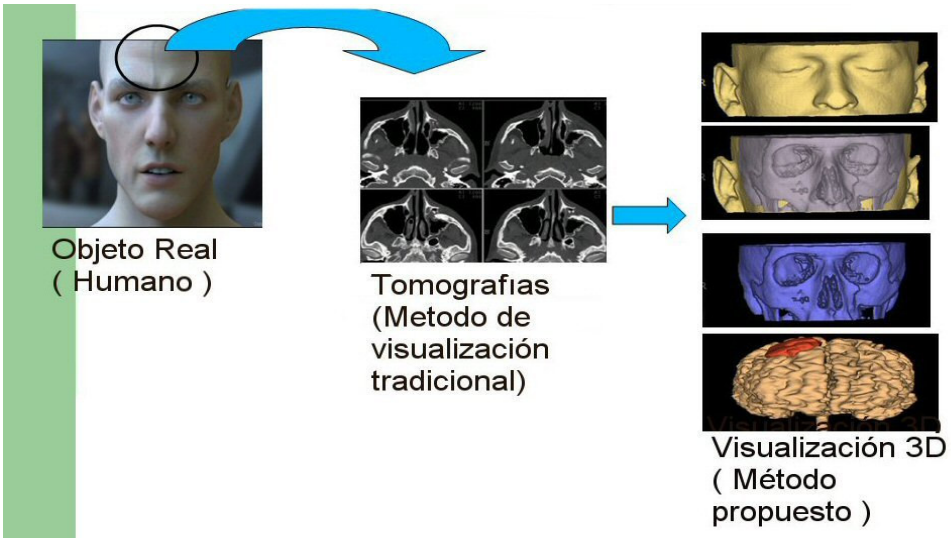


Figura 1. 1 Imágenes médicas de tomografía

La motivación principal de generar un modelo tridimensional, es que la forma y ubicación espacial de los objetos de estudio, no siempre es fácilmente apreciable si se intenta sacar conclusiones solamente de las imágenes, y al obtener un modelo realista y tridimensional del cuerpo en estudio se podrán visualizar de mejor forma las estructuras anatómicas del paciente, facilitando al médico la visualización de posibles anomalías estructurales.

Disponer de un modelo 3D reconstruido a partir de los datos de un determinado paciente como el que se visualiza en la Figura 1.1, puede tener distintas aplicaciones como la ayuda en el diagnóstico médico, la realización de simulaciones en la planificación de operaciones, el adiestramiento de los estudiantes de medicina, entre otros. En la creación de este modelo 3D a partir de los datos volumétricos, en muchos casos se ha optado por un proceso manual utilizando software de procesamiento de imágenes. Este proceso puede ser muy lento además de requerir conocimientos médicos y mucha experiencia.

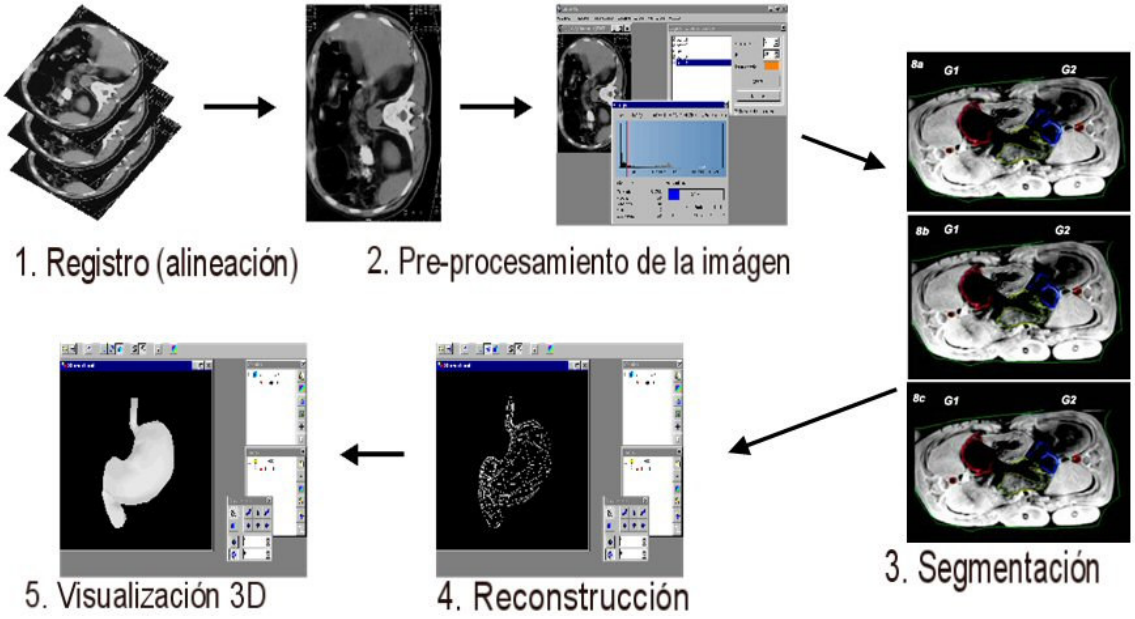


Figura 1. 2 Proceso de reconstrucción 3D a partir de imágenes de tomografía

En la figura 1.2 se visualiza un esquema de los pasos a seguir en el proceso de reconstrucción para una visualización 3D a partir de imágenes de tomografía. Los cuales consisten en una alineación de las imágenes, para luego hacer un pre-procesamiento que puede ser desde un proceso de manipulación de histograma hasta un proceso de interpolación para generar más imágenes a partir de la información que se tiene. Luego se debe realizar el proceso de segmentación que consiste en clasificar y separar unas estructuras anatómicas de otras, lo cual es de utilidad en el proceso de reconstrucción;

por último se toma los datos de la reconstrucción para producir una imagen que se pueda visualizar en un espacio tridimensional.

Debido a que el diagnóstico médico a partir de tomografías es un proceso utilizado con frecuencia, si se pretende cambiar este método tradicional por uno de visualización 3D, es preferible contar con un proceso de Reconstrucción automático que permita la generación del modelo 3D de forma rápida y con la mayor precisión. Por este motivo se plantea en este trabajo de investigación buscar e implementar métodos que faciliten la construcción de modelos tridimensionales que representen los volúmenes de las estructuras anatómicas partiendo solamente de la información contenida en las imágenes de tomografías. La solución al problema se plantea a partir de la creación de una herramienta informática que permita realizar automáticamente reconstrucción 3D a partir de una secuencia imágenes bidimensionales que representen cortes paralelos.

1.2. OBJETIVO GENERAL

Investigar, estudiar e implementar técnicas y algoritmos que permitan reconstruir volúmenes correspondientes a estructuras anatómicas a partir de un conjunto de imágenes obtenidas de tomografías y obtener una visualización tridimensional (3D).

1.2.1. Objetivos específicos

- Analizar diversos métodos de segmentación de imágenes y aplicar los que sean necesarios en la implementación de un módulo de caracterización y clasificación de estructuras orgánicas a partir del conjunto de imágenes dadas, haciendo un estudio comparativo para determinar los algoritmos más eficientes para casos generales.
- Estudiar e implementar métodos de interpolación de tal forma que se apliquen a datos en todas las dimensiones deseadas, en caso de que la resolución o cantidad de imágenes sea insuficiente.
- Desarrollar un entorno computacional que permita la visualización 3D tanto de superficies como de volúmenes, permitiendo al usuario una total interacción a través de acercamientos, rotación, traslación, manejo de transparencias y vista de árbol donde se asocie los diferentes órganos, de tal forma que se puedan habilitar o deshabilitar las diferentes estructuras.

1.3. JUSTIFICACIÓN

En muchos campos de la ciencia es necesario poder visualizar y manipular la información de forma gráfica con tal de brindar un apoyo en la toma de decisiones. En la medicina, durante el ciclo diagnóstico-terapéutico es preciso modelar los procesos del organismo a nivel fisiológico y geométrico, es por ello que la reconstrucción y la visualización 3D de estructuras anatómicas aportan información muy estimable en la medicina. Aunque los especialistas médicos bien entrenados tienen la habilidad de reconstruir mentalmente imágenes 3D a partir de las imágenes 2D de los cortes axiales, esta reconstrucción "mental" resulta demasiado fatigosa y subjetiva La solución ideal estaría en disponer de sistemas computacionales capaces de realizar esta operación de forma más eficiente y presentar el resultado haciendo uso de las capacidades gráficas avanzadas que proporciona hoy la informática, tanto a nivel de visualización como de interacción. Los sistemas de análisis de imagen 3D requieren equipamientos informáticos específicos y personal entrenado en su uso.

Aunque actualmente en el mercado hay diferentes tipos de sistemas que ofrecen buena calidad de representación 3D, tales recursos resultan costosos hoy en día, además de ser sistemas descentralizados. Afortunadamente, estos recursos informáticos son susceptibles de ser centralizados y ofrecer sus servicios a distancia en un entorno de Telemedicina, con lo cual sólo es preciso un servidor potente al que se acceda desde clientes convencionales. En ese servidor podrían instalarse las herramientas computacionales avanzadas para la reconstrucción de las estructuras anatómicas de interés y estar supervisadas por un mínimo equipo de personal cualificado. Desde los clientes remotos se podría ejecutar la reconstrucción (o encargársela al personal cualificado en caso de no disponer de los conocimientos necesarios para el manejo de la herramienta), realizar las correcciones que fuesen necesarias, visualizar la reconstrucción 3D de las estructuras de interés, y obtener valores de parámetros anatomo-quirúrgicos relevantes.

Desde el punto de vista médico, durante el diagnóstico y planificación terapéutica de pacientes, resulta imprescindible conocer la morfología de las estructuras y sus posibles anomalías (por ejemplo un tumor, con sus dimensiones, su localización y las estructuras a

las que toca). Por ello resulta de gran interés una aplicación informática que permita la segmentación automática de estas estructuras y su visualización 3D haciendo uso de las capacidades gráficas avanzadas que hoy en día ofrecen los computadores convencionales.

Por otra parte, los médicos identifican la facilidad de interacción como uno de los requerimientos indispensables para la aplicación, pues les permite realizar las correcciones que consideren oportunas sobre los resultados de la segmentación automática. Las aplicaciones totalmente automáticas están lejos de hacerse realidad en un campo tan complejo como lo es el de la imagen médica 3D, por lo que la interacción manual aporta robustez y aumenta la utilidad de la aplicación.

Teniendo en cuenta el gran valor de la representación tridimensional de estructuras anatómicas, es de gran utilidad elaborar un sistema interactivo de reconstrucción 3D a partir de imágenes 2D, en el cual se pueda visualizar el conjunto de órganos e interactuar con el modelo a través de acercamientos, rotaciones, traslaciones, haciendo uso de transparencias y la posibilidad de habilitar deshabilitar volúmenes para brindar un apoyo en el momento del diagnóstico a través de la visualización científica.

1.4 ANTECEDENTES Y ESTADO DEL ARTE

1.4.1 Perspectiva mundial

La tomografía computarizada ha evolucionado en gran manera desde que se inició como método de diagnostico medico a principio de la década de 1970. Desde esa fecha uno de los avances más importantes ha sido el desarrollo de la tomografía de volumen, también llamada tomografía en espiral o helicoidal.

La técnica, creada en 1967 por el ingeniero electrónico británico Godfrey Hounsfield, ha revolucionado el diagnóstico en la medicina. Hounsfield conectó sensores de rayos X a una computadora e ideó una técnica matemática llamada reconstrucción algebraica para armar las imágenes a partir de los datos transmitidos. En 1973 la Mayo Clinic comenzó a usar la primera máquina de TAC en los Estados Unidos. Las primeras máquinas daban

imágenes digitales con una claridad más de 100 veces mayor que las radiografías corrientes. Posteriormente, la velocidad y exactitud de las máquinas ha aumentado mucho más. Los TAC muestran hueso y tejidos blandos, incluyendo órganos, músculos y tumores. Los tonos de las imágenes se pueden ajustar para hacer resaltar los tejidos de densidad semejante y, mediante un software de imágenes, es posible formar imágenes tridimensionales a partir de varios cortes transversales. El TAC ayuda en el diagnóstico, la cirugía y otros tratamientos, incluyendo la radioterapia, en los cuales la dosis efectiva depende en gran medida de la densidad, el tamaño, y la ubicación exacta del tumor.
Con la evolución de la computación gráfica y al rápido avance de los computadores, muchos centros de investigación han dedicado sus esfuerzos a poder reconstruir las estructuras tridimensionales o volumétricas partiendo de imágenes bidimensionales, llevando este avance al campo de la medicina.

Debido a este rápido avance a nivel científico la Biblioteca Nacional de Medicina de Estados Unidos (NLM) ha visto un uso creciente de imágenes electrónicas para la medicina clínica y la investigación biomédica, por lo que en el año de 1989 estableció el Proyecto Humano Visible[7] para construir una biblioteca de la imagen digital de datos volumétricos reconstruidos a partir de tomografías axiales, imágenes de resonancia magnética e imágenes de crioseccionamiento. El crioseccionamiento consiste en una técnica de cortes físicos paralelos de cadáveres con el fin de obtener imágenes de los órganos anatómicos seccionados cada cierta distancia.

El objetivo a corto plazo del proyecto es la de recolección de imágenes transversales por TAC, IRM y criosección, de cadáveres de hombres y mujeres, con un intervalo de 1 milímetro, con el fin de crear una completa representación anatómica detallada, en 3 dimensiones, del cuerpo humano de un hombre y una mujer. El objetivo a largo plazo del Proyecto es producir un sistema de estructuras de conocimiento que de forma transparente enlazarán formas de conocimiento visuales con formatos de conocimiento simbólico, como los nombres de las partes del cuerpo.

[7] Visible Human Project: http://www.nlm.nih.gov/research/visible/visible_human.html

Aunque dicho proyecto no avanza con fines de ayudas diagnósticas a la medicina, ha sido de gran utilidad en el avance de la investigación en reconstrucción 3D a partir de tomografías, ya que el proyecto facilita mucha información anatómica, imágenes de tomografía y en ayuda general para quienes quieran beneficiarse del proyecto.
Actualmente hay una gran cantidad de proyectos basados en la información que durante muchos años ha sido seleccionada por la NLM.

1.4.2 Perspectiva nacional y local

En Colombia se ha tenido la experiencia de trabajar con la tomografía convencional desde 1985 y con tomografía helicoidal con VOXEL para reconstrucciones multiplanares en los diferentes centros de salud a nivel nacional.

Sin embargo, debido al gran costo de las estaciones de trabajo que brindan gráficos sofisticados, se ha usado la técnica convencional de imágenes de TAC, dejando descuidado la investigación en el campo de la reconstrucción volumétrica de estructuras anatómicas a partir de tomografías.

En el año 2003 en la Universidad Nacional de Colombia se presenta el proyecto de grado titulado: “Prototipo de reconstrucción volumétrica de órganos de la cabeza”, cuyo autor Jorge Enrique Monsegny Parra fue distinguido con el primer lugar en el concurso a mejor trabajo de grado Otto de Greiff en los resultados 2003-2004. Este proyecto ha sido pionero en el campo de la reconstrucción 3D a partir de imágenes como apoyo a la medicina.

En otro ámbito de la reconstrucción 3D se ha trabajado en la universidad Industrial de Santander proyectos como:

- Reconstrucción digital 3D y visualización de superficies a partir de un conjunto de muestras obtenidas de un terreno”, cuyo autor es Johan Alberto Arzuza Narváez, ingeniero de sistemas. (2004)

- “Interpolación tridimensional (3D) a partir de imágenes de microscopía”, autor, Julio Daniel Gil Cano, Ingeniero de Sistemas.(1999)

1.5 APLICACIÓN DE LA VISUALIZACIÓN 3D

Diariamente se producen una gran cantidad de datos alrededor del mundo. Los Satélites, sistemas de digitalización láser, sistemas de adquisición de información digitales, adquieren, generan y trasmiten información a velocidades muy altas. Los satélites en órbita transmiten Tbytes de información a diario. Supercomputadoras generan modelos climáticos a nivel mundial. Diariamente se realizan estudios médicos (CT, MRI, etc.) que producen grandes volúmenes de datos. La visualización es una herramienta que permite interpretar esta información aplicada por medio de las computadoras.

Sin técnicas apropiadas de visualización la mayoría de estos datos se perderían o permanecerían guardados en los discos de una computadora sin ser utilizados. Como el volumen de información es grande, el cerebro humano no puede procesarla sin ayuda. La visualización permite extraer la información importante oculta en todos estos datos, aprovechando las habilidades naturales del sistema de visión humano.

Algunas de las ventajas que ofrece la visualización tridimensional es que el ser humano está habituado a vivir, moverse, interactuar y llevar a cabo su vida en un entorno 3D. Esto hace que sea más natural el uso de una herramienta con tales características. Con las imágenes 3D se hace más fácil identificar formas y volúmenes. La gran explosión tecnológica que se ha producido en los últimos tiempos ha provocado grandes avances en la calidad y velocidad de los gráficos en tres dimensiones, por lo tanto las herramientas que se utilizan para ver imágenes 3D avanzan rápidamente.

Este tipo de ventajas pueden ser aprovechadas, para proveer al usuario con sistemas que le faciliten su labor. Las imágenes tridimensionales pueden ser utilizadas para diferentes tipos de tareas:

- **Representación de objetos tridimensionales:** en aplicaciones de ingeniería, ya sea para planeamiento como para prueba de estructuras existentes.
- **Reconstrucción de terrenos:** para proyección de crecimiento de ciudades, evaluar consecuencias ante posibles desastres naturales, evaluar potencial para explotación de grandes áreas (explotación agrícola, minera, etc.)
- **Construcción de modelos para simulación:** los modelos en 3D son muy útiles para realizar pruebas sobre prototipos o en situaciones que pongan en peligro la vida humana o maquinaria.
- **Visualización de grandes volúmenes de datos:** utilizando técnicas de visualización en tres dimensiones se pueden interpretar grandes cantidades de información. Esta información es difícil de ser comprendida por un individuo sin una herramienta que lo asista.
- **Reconstrucción de volúmenes a partir de imágenes digitales:** a partir de información extraída de datos originados por estudios como Tomografía Computada (CT), Resonancia Magnética (MRI), Ultrasonido, entre otros. Esta técnica puede ser utilizada en el campo de la medicina para el diagnóstico de enfermedades basado en imágenes, ó en ingeniería de materiales para detectar la composición interna y detectar fallas en la fabricación. Este uso en particular es el que se desarrolla en la presente investigación.

El área de reconstrucción de superficies a partir de imágenes digitales, es tal vez uno de los más usados teniendo en cuenta la gran cantidad de aplicaciones que tiene. Un ejemplo de esto es la medicina, donde las imágenes digitales (en general son resultado de estudios realizados a pacientes) pueden ser obtenidas de una diversidad de maneras, siendo las más comunes la tomografía computada (CT) y la resonancia magnética (MRI).

1.5.1 Reconstrucción tridimensional aplicada a la medicina

La aplicación de esta tecnología provee una gran cantidad de ventajas, por ejemplo en la utilización en el diagnóstico, en el tratamiento de diferentes enfermedades y el entrenamiento de profesionales. Aplicando técnicas de segmentación y reconstrucción de superficies sobre los datos de los estudios de los pacientes, pueden generarse los volúmenes que representan los diferentes tejidos y regiones dentro del cuerpo del mismo.

Normalmente el médico utiliza un conjunto de imágenes 2D para llevar a cabo el diagnóstico. El médico debe reconstruir mentalmente el modelo tridimensional representado por una serie de imágenes en dos dimensiones. El objetivo de la segmentación es dividir (segmentar) y separar la información original para generar nuevas imágenes que representen las estructuras internas del paciente. Esta técnica, combinada con la reconstrucción de superficies, facilitan la labor del médico al presentar esta información de una manera que le sea fácil de interpretar y visualizar permitiéndole realizar un diagnóstico mas preciso del paciente.

Un médico utiliza regularmente unas 20 a 50 imágenes impresas. Una herramienta de manejo 3D permite al especialista navegar por un entorno tridimensional (virtual), mover los diferentes objetos, girarlos, acercarse y alejarse de ellos aprovechando la información completa que brindan los estudios (generalmente unas 100 imágenes).

Ventajas. Algunas de las ventajas de la aplicación de este tipo de herramientas son:

- Conveniencia las herramientas que permiten generar imágenes digitales trabajan en tiempo real, permitiendo al especialista operar dinámicamente con los estudios del paciente.
- Flexibilidad las imágenes pueden ser almacenadas de una variedad de maneras. Una vez generadas pueden ser enviadas a otros especialistas para realizar interconsultas o simplemente para obtener segundas opiniones. Los resultados pueden utilizarse para generar estadísticas y ser almacenados en una base de datos.
- Comunicación por medio del uso de herramientas tridimensionales los médicos pueden explicar de manera más sencilla al paciente cual es la razón de su malestar o su dolencia logrando una comunicación mas clara entre el paciente y el doctor.

Precisión la calidad de las imágenes obtenidas a partir de la reconstrucción de superficies solo se ve limitada por las imágenes que se proveen como entrada.

1.5.2 Segmentación de imágenes

La segmentación de imagen es un procesado digital que consiste en reconocer de forma automática o semiautomática los objetos de una escena con la menor intervención previa

por parte del usuario. Existen distintos niveles de automatización dependiendo del método que se utilice para segmentar. Este automatismo pone a prueba los algoritmos utilizados por los sistemas para tal fin, ya que los objetos obtenidos deben corresponderse lo más posible con el contenido semántico de la imagen.

Las aplicaciones de este tipo de tecnología son innumerables. La más inmediata es la selección automática de objetos para la generación de máscaras y trazados. A continuación, se enumeran algunas de las aplicaciones más relevantes de la segmentación de imagen:

- **Medicina:** detección de tejidos; localización de fracturas o tumores, análisis de ecografías, CT, MRI e imágenes de rayos X en traumatología.
- **Industria:** supervisión automática de procesos industriales, como control de calidad, vigilancia o detección de fallos.
- **Análisis de fotografías aéreas o por satélite:** clasificación de diferentes tipos de vegetación para el estudio de su degradación; clasificación y control de cultivos en agricultura; detección de objetos.

Básicamente existen dos formas de dividir la imagen en regiones:
1. Mediante contornos.
2. Mediante regiones "uniformes".

Ambas son complementarias. En una imagen "ideal" bastaría con una, ya que la otra da la misma información. Sin embargo en imágenes "reales", en las que hay ruido, atributos faltantes, etc. , utilizar estas técnicas de manera combinada puede dar gran robustez al sistema dando resultados mas precisos. De esta manera se obtiene, a partir de imágenes digitales, objetos que tienen una semántica y significado propio.

¿Qué representa para el futuro de la imagen digital la segmentación? La respuesta es una nueva generación de sistemas interactivos. El ser humano no aspira a interaccionar con entidades abstractas, como el pixel, sino con entidades con significado propio que sean parte de la escena, por ejemplo un tejido, una región del cuerpo, etc. La segmentación de

imágenes es la herramienta que permite pasar del pixel al objeto. Y no hay otro camino. Más aún, el éxito de todo un sistema de análisis de imagen de alto nivel puede depender del éxito o fracaso de esta primera etapa. En definitiva, segmentar con fiabilidad los objetos de una escena es el primer paso para la toma de decisiones precisas.

En la figura1.3 se presentan algunos ejemplos de imágenes 2D segmentadas.

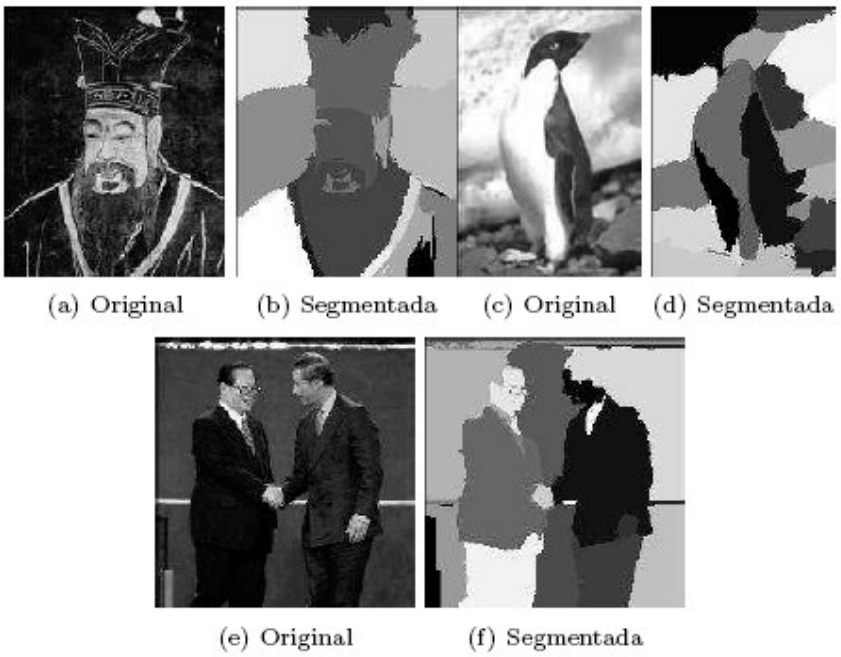


Figura 1. 3 Ejemplos de imágenes segmentadas.

Las imágenes anteriores son algunos ejemplos de imágenes 2D en escala de grises segmentadas. En estas se pueden distinguir las diferentes regiones de la imagen original. Al tener esta habilidad se puede dar semántica a los elementos de la imagen en lugar de operar con elementos individuales como un pixel, se posee el potencial de operar con elementos concretos, como ser persona, auto, fondo, en lugar de pixels.

La segmentación de imágenes tiene su origen en numerosos estudios psicológicos que indican la preferencia de los humanos por agrupar regiones visuales en términos de proximidad, similitud y continuidad, para construir un conjunto de unidades significativas. Existe cierta confusión en torno al concepto de segmentación ya que si bien algunos autores consideran suficiente marcar los puntos de la imagen (pixels) con un valor

indicativo de su pertenencia a determinada región o clase, otros indican que además es necesario proveer un mecanismo que permita una representación simbólica de las relaciones topológicas existentes entre las distintas unidades. Esta discrepancia se debe fundamentalmente al nivel de abstracción al cual se asocia el proceso de segmentación. Si se asocia la segmentación a un nivel de abstracción medio, ésta deberá de producir la representación simbólica anteriormente indicada, correspondiendo el marcaje a un proceso previo sito en un nivel bajo de abstracción y encaminado al objetivo de segmentar. Si bien se ha adoptado esta segunda postura, cabe reconocer que existen situaciones o aplicaciones donde no merece la pena soportar la carga computacional que supone la obtención de dicha representación simbólica. Esto suele ser cierto solamente en casos donde la tarea a realizar o el entorno están muy delimitados y controlados. El tipo de aplicaciones que se desean abordar, sin embargo, corresponden a escenarios o tareas variables, al menos dentro de una clase. Los métodos de segmentación pueden ser clasificados según la forma y técnica que utilizan para detectar las diferentes regiones. Entre los objetivos principales de este trabajo se encuentran: segmentación de imágenes, reconstrucción de superficies y optimización de las superficies generadas.

En el siguiente capítulo se presenta una breve descripción de algunas de las técnicas existentes mas comunes de segmentación como así de reconstrucción y post procesado de superficies.

2. MARCO TEÓRICO

2.1 IMÁGENES MÉDICAS

Para presentar las nuevas perspectivas que se abren para la imagen médica en el futuro es conveniente establecer algunos conceptos y definiciones, que serán necesarios para estudiar las diferentes técnicas de imagen médica. Definiremos **imagen médica** como una Representación de la distribución espacial de una o más propiedades físicas o químicas dentro del cuerpo humano. El objetivo de todas las modalidades de imagen médica es visualizar los órganos internos del cuerpo para obtener información estructural y anatómica, como en la tomografía computarizada, de una manera clara y sin poner en riesgo la integridad del paciente.

El principio de la reconstrucción de imagen en todas las modalidades de tomografía es que un objeto se puede reproducir exactamente a partir de un conjunto de sus proyecciones tomadas desde diversos ángulos. En una aplicación práctica, en cualquier modalidad de tomografía computarizada se puede obtener solamente una estimación de la imagen real del objeto bajo estudio. La fidelidad de la reconstrucción en cada caso dependerá de las respuestas a una serie de preguntas sobre la adquisición y el proceso previo de los datos adquiridos, de la implementación numérica de las fórmulas matemáticas de reconstrucción, y del post-procesado de las imágenes reconstruidas.

Dos parámetros de la imagen van a ser de especial interés: el contraste y la resolución. El **contraste** determina qué es lo que se ve en la imagen; técnicamente se define como diferencia de intensidad entre dos áreas, medida en valor absoluto o relativo (contraste relativo). Lo más importante, en el contexto que nos ocupa, es tener claro el origen de dicho contraste, esto es, saber qué parámetro físico o químico es el que está siendo representado en forma de intensidad luminosa. Por su parte, el concepto de **resolución espacial** nos ayuda a caracterizar la imagen desde el punto de vista de su capacidad para discernir detalles. Es frecuente definir resolución como la distancia mínima que la imagen es capaz de resolver o separar, medida en unidades longitud del mundo físico

(milímetros o micras en biología, a lo mejor años luz en imágenes astronómicas). Este modo de medir la resolución es el más intuitivo, pero no el único utilizado: según el tipo de imagen se puede expresar la resolución en puntos por pulgada (dpi, 'dots per inch'), pares de líneas por centímetro, etc.

En resumen, el contraste de una imagen determina qué podemos ver en ella, y su resolución espacial nos indica el grado de detalle con que se representará dicha información.

Otro parámetro de interés es la denominada **resolución temporal**, que determina la capacidad del sistema de imagen para 'congelar' situaciones en el tiempo, estando muy relacionado con la velocidad de adquisición de las imágenes.

Las imágenes contienen no sólo información de interés, que representa fielmente los objetos en que estamos interesados, sino también datos espurios o erróneos que los distorsionan.

Esta información superpuesta no deseada se llama **ruido** si tiene carácter aleatorio o **artefacto** en caso contrario. Estos últimos suelen depender de problemas en la técnica de imagen, se podrían considerar errores sistemáticos, frente a los errores aleatorios que introduce el ruido.

Últimamente se viene dando en denominar **modalidades de imagen** a las diferentes técnicas de obtención de imagen médica. El término 'modalidad' es un anglicismo de muy dudosa ortodoxia académica, pero cuyo uso se impone desde la literatura anglosajona, como es tan habitual. El elemento básico que define las diferentes modalidades es el tipo de energía utilizada. Como en casi todo proceso de medida, la obtención de imágenes médicas implica irradiar la muestra (el paciente, en este caso) con algún tipo de energía. El carácter de la misma va a definir el contraste de la imagen y también dará nombre a la modalidad correspondiente. Como veremos más detalladamente a continuación, las modalidades fundamentales de imagen médica son: Radiología (radiación electromagnética: rayos X), Ecografía (energía ultrasónica), Medicina Nuclear (radiación electromagnética: radiación gamma) y Resonancia Magnética (radiación

electromagnética: ondas de radio). Es interesante resaltar que el tipo de energía utilizada determina el tipo de interacción bioquímica que se produce en los tejidos biológicos y, por tanto, en qué medida puede ser nociva para el organismo. Se denominan **radiaciones ionizantes** aquéllas que por su alta energía son capaces de inducir directamente reacciones químicas, a través fundamentalmente de la ionización de diferentes moléculas. Las **radiaciones no ionizantes** se limitan a producir calentamiento que, en principio, no presenta efectos biológicos relevantes si es ligero.

Es muy frecuente clasificar las modalidades en **morfológicas** (o estructurales) y **funcionales**. Las primeras se caracterizan por producir imágenes de muy buena resolución, que permiten una representación muy detallada de la anatomía del paciente. Las segundas, en cambio, se caracterizan por aportar información sobre el funcionamiento de los diferentes órganos o sistemas: algún rasgo de su metabolismo, su perfusión sanguínea, su capacidad para acumular ciertas sustancias, etc. El origen del contraste es lo que distingue esencialmente las modalidades morfológicas de las funcionales; además es frecuente que la resolución de las primeras sea mucho mejor.

Otra característica importante de las imágenes médicas deriva de su capacidad para separar objetos que se hallan a diferentes profundidades. Se llaman **imágenes proyectivas** aquéllas que representan la suma de todas las estructuras del objeto, proyectadas sobre una superficie bidimensional. Por el contrario, cuando el método de imagen es capaz de separar diferentes planos –'cortar la muestra en rodajas'-, cada uno de los cuales se representa en una imagen bidimensional, se denomina **tomográfico**. La imagen topográfica crea menos problemas de superposición de objetos, facilitando notablemente su interpretación.

La presente investigación se basa en Reconstrucción imágenes médicas de tomografía.

2.1.1 Realidad virtual e imágenes 3D en medicina

La realidad virtual puede ser definida como un grupo de hardware y programas de software, que permite hacer simulaciones realistas de la interacción con objetos virtuales, los cuales son modelos computarizados de objetos reales. Frecuentemente se toma un

objeto ya existente (objeto real) y a partir de éste se crea una versión virtual. Los datos son adquiridos desde la realidad a través de digitalización procesado y segmentación de imágenes. La realidad virtual tiene una gran gama de aplicaciones en las más diversas áreas del conocimiento humano, en especial en el área médica. Frecuentemente los médicos hacen su diagnóstico con base en la evolución de lesiones, tumores o análisis de estructuras anatómicas, fundamentándose en el análisis de imágenes del cuerpo humano.

En particular, suelen utilizarse imágenes obtenidas por rayos-X, tomografía computarizada, resonancia magnética, endoscopía, ultrasonografía, etc, que permiten obtener información tridimensional de las estructuras anatómicas. El análisis de las estructuras internas del cuerpo sólo ha sido posible durante el último siglo, con el descubrimiento de los rayos-X. Otra revolución llegó no hace más de 30 años con la utilización de las tomografías computarizadas y resonancias magnéticas que permiten el estudio y análisis en tres dimensiones del cuerpo humano. No obstante, como todas las técnicas, los rayos-X tienen limitaciones en cuanto a tipos de tejidos, textura y profundidad que pueden estudiar. En este contexto, las técnicas médicas de ultrasonido se han revelado de gran utilidad. Los ultrasonidos son particularmente utilizados en especialidades como obstetricia, ginecología y cardiología, demás del diagnóstico de una infinidad de enfermedades. Gracias a las diferentes técnicas de obtención de datos anatómicos, hoy ya se puede trabajar con imágenes digitalizadas en la red *world wide web (www)*.

Existen varias herramientas que ayudan al diagnóstico médico, aunque la imagen bidimensional es la más frecuente. Sin embargo, todas ellas están sujetas a la subjetividad y dependen de la experiencia del especialista a la hora de reconstruir mentalmente la estructura 3D a partir de las secuencias de imágenes 2D, para obtener un diagnóstico o proceder a una intervención. Por esta razón se proponen nuevas técnicas más objetivas, como la imagen 3D que va ganando terreno en varias especialidades. La imagen 3D se puede hacer utilizando instrumentación 3D, que en general es muy cara, o a partir de la secuencia de imágenes 2D utilizando un visualizador que simula modelos 3D. La base de este trabajo es la simulación 3D a partir de la secuencia de imágenes 2D, centrándose en la adquisición y reconstrucción de imágenes.

El empleo de imágenes 3D digitales son muy comunes en medicina, siendo los mejores ejemplos las tomografías computadas (CT) y las de resonancia magnética (MRI). Las mismas pueden ser descritas como un arreglo de $N_x x N_y x N_z$ voxels (pixels tridimensionales) con un color definido (normalmente se utiliza una escala de grises con 8, 12 o 16 bits de información que corresponden a 256, 4096 o 64K niveles de grises respectivamente) en correspondencia a la densidad del tejido en ese voxel. La figura 1 muestra, a modo de ejemplo, algunas capas correspondientes a una tomografía computada del cerebro.

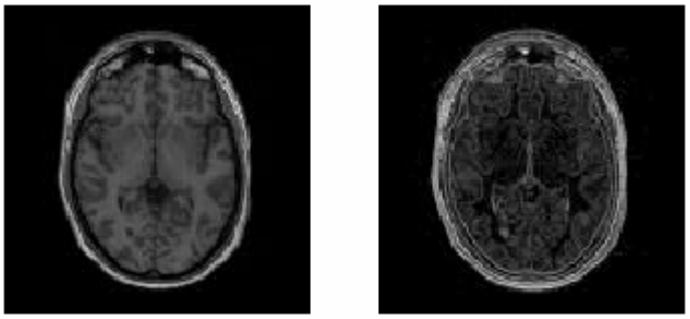


Figura 2. 1 Algunos cortes de una CT correspondiente al cerebro humano.

Esta forma de definir una geometría presenta varias dificultades para ser utilizada en la generación de una malla. En general, los métodos de generación requieren la descripción de la superficie del dominio y toman la información como superficies definidas a través de una triangulación y contornos. En las imágenes digitales, esta información no está directamente disponible y de hecho la superficie del dominio suele no estar netamente definida. Otra dificultad de estas imágenes son los bordes escalonados, producto de la pobre discretización de éstas.

Una posibilidad para solucionar estos problemas que da lugar a una metodología que promete un razonable grado de automatización en este proceso, presenta cuatro pasos bien diferenciados:

2.1.2 Procesamiento de la imagen tomográfica

Estas imágenes en general no se encuentran bien definidas, presentando ruidos y pobre definición de algunos tejidos, y en las mismas aparecerán otros cuerpos que no son de

interés para la simulación. En esta etapa se deben eliminar tanto las imperfecciones como los demás componentes. Tanto para la eliminación de los ruidos como para enfatizar las fronteras entre el cuerpo de interés y los demás, resultará de gran utilidad el empleo de algunos filtros globales. Estos filtros afectan el color de un voxel independientemente del color de sus vecinos con base en una dada función de transformación. La figura 2 muestra una imagen de tomografía y una función para transformar la imagen convirtiéndola en otra perfectamente contrastada.

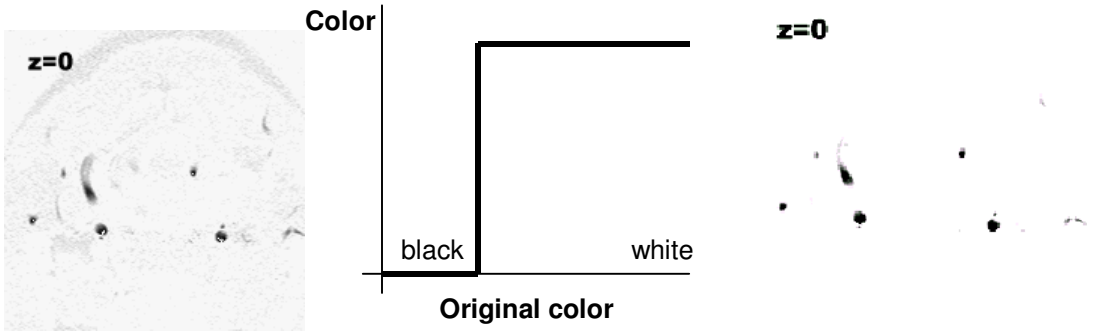


Figura 2. 2 Imagen original, función de transformación de colores e imagen resultante.

Para mejorar la calidad o el contraste de la imagen se puede recurrir a otras funciones de transformación, donde el rango de colores de interés se define con base en un histograma de la imagen (ver figura 3 para un ejemplo). Debe notarse cuáles diferencias en la función de transformación utilizada tendrán un impacto en la extensión del cuerpo de interés.

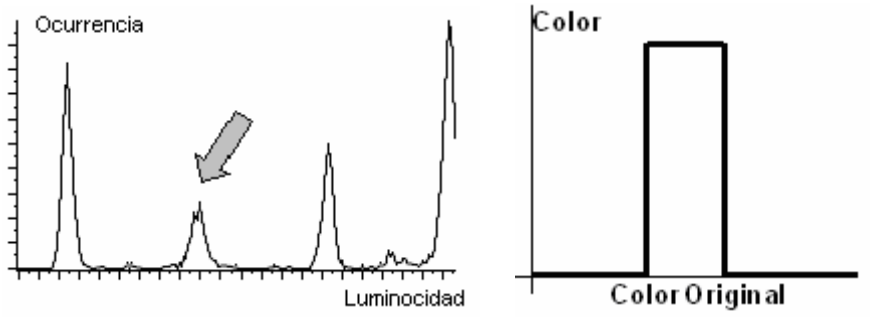


Figura 2. 3 Función de transformación de colores para seleccionar un componente dado.

Una vez aplicado este filtro se puede seleccionar una componente conexa y eliminar todo lo demás. Para ello se utiliza un algoritmo de llenado: partiendo de un dado voxel que

pertenece al dominio de interés, se lo selecciona y se marcan sus voxeles vecinos para analizarlos. En este análisis los voxels serán seleccionados si su color no difiere en más de una tolerancia dada con la del voxel original, y en este caso se marcarán sus vecinos aún no visitados para ser también analizados. Este algoritmo recursivo es sumamente simple y efectivo.

La figura 4 nos muestra de manera esquemática cómo el procesamiento de imágenes nos permite resaltar las partes del organismo que deseamos reconstruir tridimensionalmente.

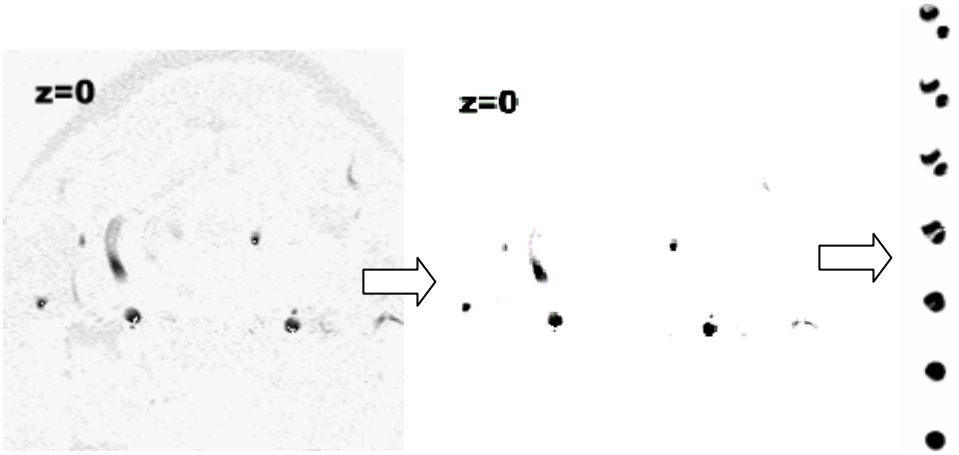


Figura 2. 4 Estructura que se desea reconstruir extraída a través del procesamiento de imágenes.

2.1.3 Extracción de la superficie del cuerpo

Una vez procesada la tomografía, es necesario detectar la superficie que delimita el cuerpo de interés. Se trata quizás de la etapa más delicada desde el punto de vista de obtener un algoritmo robusto y automático. La idea más simple para extraer esta superficie es definirla como la superficie de los voxels que tienen un color dado (densidad), es decir el conjunto de voxels que forman el componente seleccionado según se explicó en el punto anterior. Lamentablemente la geometría que así se obtiene no es en general aceptable a causa de los bordes escalonados. Una mejor alternativa es calcular la superficie de nivel donde el color (densidad) tiene un valor dado. Para ello es preciso primero tener una definición continua de los colores y no discreta como se la manejó hasta ahora. Es decir, en lugar de tener un color por voxel, se debe tener un color por vértice (cada voxel está formado por ocho vértices). Finalmente, el color a ser

atribuido a cualquier punto interior del voxel es obtenido por interpolación. Posteriormente, cada voxel es particionado en tetraedros y en estos tetraedros se determina la intersección con la superficie de nivel escogida.

2.1.4 Generación de la malla de superficie

Una vez clasificadas las estructuras que se van a tener en cuenta en el proceso, se procede a hacer el enmallado necesario para representar la superficie en un espacio tridimensional. Para realizar la representación en 3D, se efectúan las transformaciones de movimiento requeridas, la eliminación de áreas ocultas, la asignación de sombreado o transparencia y la proyección desde el ángulo deseado.

2.2 PROCESO DE RECONSTRUCCIÓN 3D A PARTIR DE IMÁGENES

En el proceso de reconstrucción 3D a partir de imágenes de tomografía se pueden definir varias etapas, desde su obtención en el equipo correspondiente hasta su despliegue en una computadora.

2.2.1 Adquisición

Se obtiene el conjunto de imágenes en dos dimensiones directamente del equipo cuya modalidad nos interesa utilizar, en este caso el tomógrafo o TAC.

El dispositivo de TAC emite un haz muy fino de rayos X. Este haz incide sobre el objeto que se estudia y parte de la radiación del haz lo atraviesa. La radiación que no ha sido absorbida por el objeto, en forma de espectro, es recogida por los detectores. Luego el emisor del haz, que tenía una orientación determinada, (por ejemplo, estrictamente vertical a 90º) cambia su orientación (por ejemplo, haz oblicuo a 95º). Este espectro también es recogido por los detectores. El ordenador hace un proceso de 'suma' de las imágenes, promediándolas. Nuevamente, el emisor cambia su orientación (según el ejemplo, a 100º de inclinación). Los detectores recogen este nuevo espectro, lo 'suman' a los anteriores y 'promedian' los datos. Esto se repite hasta que el tubo de rayos y los detectores han dado una vuelta completa, momento en el que se dispone de una imagen tomográfica definitiva y fiable de un primer corte.

Una vez que ha sido digitalizado el primer corte, la mesa donde el objeto reposa avanza (o retrocede) una unidad de medida (hasta menos de un milímetro) y el ciclo vuelve a empezar. Así se obtiene un segundo corte (es decir, una segunda imagen tomográfica) que corresponde a un plano situado a una unidad de medida del corte anterior.

A partir de todas esas imágenes transversales (axiales) un computador reconstruye una imagen bidimensional que permite ver secciones del objeto de estudio desde cualquier ángulo.

2.2.2 Registro

Esta etapa consiste en ubicar en un mismo plano de referencia una serie de imágenes que contienen al órgano de interés. Al considerar imágenes multimodales, el problema radica en que cada una de éstas tiene un plano de referencia propio, por lo que es indispensable en estos casos, la alineación de las imágenes.

En la implementación actual, la alineación se realiza de forma interactiva, permitiendo al usuario seleccionar los pares equivalentes, desplazar, girar y escalar una superficie hasta hacerla coincidir con la otra. También se puede hacer de forma semi-automática, señalando cuatro puntos característicos de las dos superficies para realizar un ajuste por mínimos cuadrados. Finalmente, el cálculo automático del centroide y los ejes principales de las superficies permiten una alineación totalmente automática.

2.2.3 Segmentación

La segmentación consiste en extraer objetos de interés a partir de las imágenes en tonos de gris. Una de las principales etapas del análisis de imagen y del reconocimiento es la segmentación de la imagen en regiones y el cómputo de varias propiedades de las relaciones entre estas regiones. Sin embargo, las regiones no siempre están claramente definidas; a veces es más apropiado mirarlas como subconjuntos difusos de la imagen.

En esencia, la segmentación consiste en hallar la estructura interna del conjunto de descripciones de los objetos (píxeles) en el espacio de representación (imagen). Esta estructura interna evidentemente depende, en una primera instancia, de la selección de

los píxeles y de la forma como éstos se comparan, es decir, del concepto de similitud que se utilice y de la forma como éste se emplee.

Se puede decir que en un problema de segmentación (*o clasificación sin aprendizaje*) los tres elementos esenciales son: *el espacio de representación de los objetos, la medida de similitud* (función de semejanza, no necesariamente una distancia) y el criterio de agrupamiento, es decir, la manera como será utilizada la similitud para la solución del problema planteado.

Los algoritmos de agrupamiento pueden ser categorizados, tanto por una base axiomática: *determinístico*, *estadístico*, *difuso* y *posibilístico*, como por el criterio agrupacional; existen tres tipos de criterios para cada clase axiomática: *función objetivo, teoría de grafos y jerárquicos.*

Este tema ha sido ampliamente investigado, pero no se ha llegado a obtener un método universal que clasifique automáticamente cualquier estructura anatómica de interés, a partir de cualquier modalidad de imagen, de ahí que se continúe esta investigación según la aplicación.

2.2.4 Interpolación

Los contornos obtenidos en la etapa precedente deben ser alineados en pila y en caso de que la resolución sea insuficiente, se requiere aplicar un algoritmo de interpolación entre datos para todas las dimensiones deseadas. Dentro de los métodos de interpolación existen aquellos que resaltan características de calidad o de tiempo de ejecución. Es necesario realizar un estudio para determinar el mejor método de interpolación que se adapte a la aplicación que en particular se desee desarrollar.

2.2.5 Presentación de resultados en 3D

La etapa final de la reconstrucción es la que determina su calidad al poder visualizar en un espacio tridimensional las geometrías obtenidas. Consiste en desplegar el volumen del órgano en estudio o de una de sus estructuras en la pantalla de la computadora.

La presentación de los resultados en 3D implica la reconstrucción de una superficie aproximada que une los contornos en secciones contiguas. Esta etapa se puede descomponer en varios problemas. Primero se resuelve el problema de la correspondencia que consiste en determinar cuáles de los contornos de cada par de secciones contiguas deben estar unidos entre sí por una superficie. Luego surgen nuevas interrogantes relacionadas con la diferencia de forma de las secciones que se analizan. Esta diferencia está originada por la naturaleza discreta de los datos tratados, ya que entre cada par de secciones contiguas existe una distancia de separación (por lo general constante) y se desconoce la información intermedia. Esta condición hace que todos los problemas aquí enunciados sean abiertos.

Cuando la correspondencia entre dos contornos es biunívoca, entonces sólo resta resolver el problema del enlosado, que consiste en unir los contornos relacionados por una superficie formada por polígonos adyacentes. Sin embargo, en la mayoría de los objetos ocurren protuberancias y hendiduras que provocan ramificaciones en la superficie de interés. Al obtener las secciones de estos objetos ocurre que uno o varios contornos de una sección se corresponden con varios contornos de la sección contigua y surge el problema de la ramificación que consiste en determinar de qué modo están unidos por una superficie.

Debido a la naturaleza abierta de los problemas, algunos criterios de verificación de los resultados responden a la valoración subjetiva de simples usuarios o, incluso, de expertos. En este trabajo se tienen en cuenta dos criterios básicos que han sido utilizados por muchos autores. La solución propuesta debe obtener como resultado una superficie topológicamente correcta (en general, cerrada y que no se intercepte consigo misma) y segundo, un nuevo muestreo de la misma en el lugar que ocupaban las secciones originales, debe producir los datos originales.

Dentro de la representación 3D de imágenes médicas existen dos campos: la representación en superficie **(*surface rendering*)** y la representación en volumen **(*volume rendering*)**. En representación en superficie, la visualización y manejo de datos multidimensionales se basa únicamente en los límites del objeto, y el despliegue de la superficie del objeto de interés está aislado del resto del volumen. En representación en

volumen, se persigue la representación de múltiples estructuras con diferentes tipos de tejido. A cada voxel se le asocia un nivel de opacidad y se asume que el valor de cada uno está correlacionado con el tipo de material que lo integra. Para cada versión de despliegue se examina cada voxel, lo que hace a esta representación muy costosa computacionalmente. El tema de las ventajas y desventajas de uno u otro tipo de representación se ha convertido en controversia, pero se puede hacer una diferencia drástica entre los dos: la representación en volumen es muy útil cuando las estructuras en la escena son difusas, como puede ser un cáncer dentro de un tejido; por otro lado la representación en superficie pierde la información del material del objeto que lo limita para fines de comparación con otros tejidos. La gran ventaja de tener únicamente información sobre los límites del objeto es que permite una más rápida interacción y manipulación de los datos, además de obtener una aproximación mejor al detalle y textura de la superficie del objeto.

Para realizar la representación en 3D, se efectúan las transformaciones de movimiento requeridas, la eliminación de áreas ocultas, la asignación de sombreado o transparencia y la proyección desde el ángulo deseado. Actualmente existen paquetes comerciales para cualquier estación de trabajo que realizan funciones de despliegue de estructuras en 3D de una manera automática, principalmente para aplicaciones de imágenes sintéticas. Sin embargo, no cualquier método de sombreado entrega un resultado acorde con lo que se vería en la realidad. Nuevamente, es necesaria una evaluación de los métodos empleados en esta etapa, de acuerdo a criterios cualitativos y cuantitativos del especialista.

2.3 MÉTODOS DE REGISTRO DE IMAGEN

El registro de imagen es el proceso de cálculo de la correspondencia entre dos o más imágenes de una misma escena tomadas generalmente en tiempos diferentes, desde distintas orientaciones y/o captadas por varios tipos de sensores. Esta es una operación realmente necesaria en muchas tareas de procesado y análisis de imágenes. Especial atención merece el registro de imágenes médicas donde las informaciones específicas de cada tipo de imagen (modalidad) se combina y fusionan en una sola aumentado así la

capacidad, por parte del clínico, de mejorar la diagnosis. Por ejemplo, un buen registro y fusión de una imagen de tomografía computarizada (CT, ”Computed Tomography”) con una imagen de tomografía por emisión de positrones (PET, ”positron emission tomography”) se traduce en una aportación de información adicional de una modalidad respecto a la otra; en este caso, a la información de tipo funcional que aporta la imagen PET, se le añade la información anatómica detallada disponible en la imagen CT. En el registro de imagen, la precisión de los resultados depende de la medida de similitud empleada. Gracias a ésta, se realizan las correspondencias entre las imágenes completas o entre bloques de ellas.

El análisis de imágenes médicas es una tarea compleja ya que normalmente estas imágenes constituyen grandes volúmenes de datos y, además, presentan ruido y otras imperfecciones de la imagen como los cambios de iluminación. Sin embargo, la inclusión de información a priori en el análisis de estas imágenes puede facilitar mucho su estudio. La información a priori está normalmente representada por las imágenes de referencia o atlas que determinan un espacio concreto en el cual se describe la anatomía, por ejemplo, del cerebro humano. Actualmente los atlas del cerebro (creados a partir de secciones criogénicas o de imágenes in vivo, basados en un solo sujeto o en toda una población) proporcionan imágenes digitales muy detalladas que pueden ser examinadas interactiva y fácilmente en el diagnostico de tratamientos y planificación de los mismos por ordenador. En consecuencia, para poder combinar de manera eficiente toda la información contenida en los distintos tipos de imágenes médicas surgen las técnicas de registro* que proporcionan las transformaciones geométricas que sitúan dos imágenes en correspondencia anatómica voxel a voxel.

Existen diferentes métodos para realizar registro de imágenes, cada uno de los cuales se comporta de una mejor o peor manera según el tipo de imágenes que se estén estudiando. Los métodos de registro de imágenes pueden clasificarse por ejemplo como lo hace *Maeset al*. [2]. Se distinguen los métodos basados en *marcos estereotácticos*, en *puntos característicos* (*landmark registration*), en *superficie* o en *voxels*. Cada uno de ellos cuenta con diferentes características que los hacen idóneos según los requisitos de la aplicación que se tenga. En el caso de los *marcos estereotácticos*, se cuenta con

algoritmos de precisión muy elevada, pero inconveniente por la necesidad de fijar el marco sobre estructuras rígidas, usualmente el hueso. Esto no ocurre con los procedimientos basados en la segunda clase, donde la mayor limitante se encuentra en la incertidumbre producida al detectar puntos característicos en el físico del paciente para realizar el registrado. En los métodos basados en *superficie* se requiere delinear las superficies en cada una de las imágenes y la precisión está en dependencia directa con los datos y con la modalidad de imágenes, ya que en estudios que muestran funcionalidad como en PET la segmentación se torna compleja. Las estrategias de registro basadas en *voxels* optimizan la alineación geométrica midiendo la similitud en los niveles de gris en un par (o más) de imágenes, lo que permite que su precisión no se vea limitada por errores en la segmentación

Entre los algoritmos más destacados para el registro de imágenes generales se encuentran los planteados por Szymon Marek Rusinkiewicz [16], quien dividió el proceso de registro en 6 etapas y analizó los métodos existentes para cada una, además de ofrecer su propio método basado en sus observaciones.

Para poder hablar de registro de imágenes es necesario tener claro el significado de lo que son las imágenes de rango, y en qué se diferencian de las imágenes tradicionales.

2.3.1 Imágenes de Rango

Una imagen de rango es una representación de una escena similar a las imágenes de intensidad, pero que a diferencia de ellas tiene información de profundidad asociada a cada pixel. La codificación más utilizada para la información de profundidad es la escala de grises, donde la intensidad de gris en cada pixel representa la distancia en línea directa del objeto al sensor. Éstas imágenes pueden expresarse en espacio tridimensional, relacionando la información de profundidad a un eje coordenado que generalmente tiene como origen la posición del sensor.

De la anterior definición podemos concluir que el conjunto de todas las imágenes de tomografía obtenidas de un estudio de un paciente se constituyen en imágenes de rango ya que poseen una profundidad asociada a cada píxel.

2.3.2 Problema de alineación

Debido a que en las imágenes de rango se trabaja con un nivel de profundidad de píxeles, que es a lo que llamamos voxels, hay que tener en cuenta que esta profundidad está definida por la intensidad de un píxel en cada imagen del conjunto, por lo tanto éstos se deben corresponder de la manera más precisa posible, caso que no siempre es posible en primera instancia. En términos generales, el problema reside en encontrar la rotación y la translación que debe aplicársele al conjunto de datos de una imagen de rango, para que traslape una zona común con otra imagen de rango y de esta manera poderlas asociar a un mismo espacio de coordenadas.

El problema general puede separarse en problemas puntuales ubicados en el desarrollo del algoritmo. Primero, se debe aplicar un tipo de muestreo que permita contar con un subconjunto de datos que contenga información de las características principales de la imagen. Luego, asegurar la similitud entre los puntos de diferentes muestras sin incluir demasiada carga computacional al proceso. Finalmente encontrar el equilibrio entre la minimización del error, la velocidad de convergencia y la sensibilidad del algoritmo a los mínimos locales.

2.3.3 Aplicación del registro de imágenes

Existen aplicaciones generales para el registro de imágenes de rango en campos como:

- ***Inspección*:** cuando una pieza industrial es construida, a menudo existe su modelo CAD. Si esta pieza falla sería posible construir un modelo 3D de la pieza defectuosa para compararla con el diseño original y así encontrar las posibles causas del problema.

- ***Ingeniería Inversa*:** a menudo encontramos objetos artificiales únicos, sin molde o modelo CAD, con el modelado 3D a partir de elementos reales es posible digitalizar una pieza única con un alto grado de exactitud para su posterior reproducción, en caso de falla o deterioro.

- ***Entretenimiento*:** La animación 3D ha cobrado fuerza gracias al avance tecnológico de la computación, los simuladores de vuelo virtuales ya no son deuso

exclusivo militar y los juegos de video de última generación cada veztienden más a los gráficos 3D, y que mejor que recrear todos estos mundos virtuales a partir de objetos reales con el modelado 3D.

- ***Medicina*:** mediante los diferentes scanners médicos es posible modelar órganos del cuerpo para su análisis, en busca de anomalías o realizar modelos para protesis y reconstrucciones.

2.3.4 Registro de imágenes de rango

Puesto que en el proceso de adquisición de imágenes de rango, como las de la Figura 2.5, generalmente solo se realiza una toma a la vez y desde una sola dirección, se hace necesario realizar un proceso de alineación de las muestras que inicialmente están referenciadas a la posición de la cámara, mediante transformaciones geométricas con el fin de poder conformar un modelo 3D representativo de objeto real. Este proceso se conoce como registro de imágenes de rango.

Figura 2. 5 Ejemplo de imágenes de rango.

El problema de Registro de imágenes de rango se puede definir de la siguiente manera:
Tenemos una superficie M de la cual se realizan N tomas Fi de diferentes posiciones donde cada una corresponde a una imagen de rango, se deben encontrar N tranformaciones geométricas Ti que acoplen las imágenes de rango Fi para que se puedan expresar en un mismo espacio coordenado S(x,y,z) y así poder construir un modelo digital del objeto.

2.3.5 Algoritmo de Registro Multimodal

El método para registro de imagen multimodal y fusión de imagen se divide en dos etapas. Primero se calcula la relación entre las intensidades de pıxel de las imágenes a registrar con un método de estimación no lineal basado en funciones de base radial mediante el histograma conjunto de niveles de gris. Esta relación se utiliza para modificar los niveles de gris de ambas imágenes. La segunda etapa realiza el registro de imagen mediante el método de estimación de movimiento no rígido basado en modelos difusos.

A. Ajuste no lineal de la relación de intensidades de pixel

Dado que la fuente de las imágenes a registrar por lo general es diferente, se necesita conocer su relación entre las intensidades de pixel para poder encontrar la correspondencia espacial entre ambas imágenes. Para obtener esta relación se efectúa un ajuste no lineal del histograma conjunto de niveles de gris. El ajuste se realiza mediante funciones de base radial. El primer paso consiste en obtener el histograma conjunto de niveles de gris, contabilizando para ello el número de ocurrencias de cada pareja de niveles de gris. Este histograma se umbraliza posteriormente, y tratándolo como si de una imagen se tratase, se procesa mediante operadores morfológicos de dilatación y erosión realizando una operación de cierre. El objetivo de este procesado es eliminar las combinaciones con un número escaso de ocurrencias, que no proporcionan información significativa, para así, tener en cuenta sólo los niveles de gris de mayor probabilidad. Entonces se etiqueta cada una de las regiones presentes en el histograma procesado.

Después, para cada región etiquetada por separado, se procede a promediar los valores en vertical para cada uno de los valores en horizontal, obteniendo una curva única y equivalente al histograma conjunto procesado. Para obtener la relación matemática de intensidades de pixel, se aplica el método no lineal basado en funciones de base radial. Se obtienen dos nuevas imágenes con las intensidades de píxel calculadas. Cada nueva imagen corresponde a la primera y segunda imagen con las intensidades de pixel evaluada de acuerdo a la relación de intensidades de pixel final.

B. Algoritmo de estimación de movimiento no rígido

El algoritmo de estimación de movimiento no rígido se presenta en [4], [5]. El objetivo de este método es estimar la correspondencia espacial entre pares de imágenes que

contienen movimiento no rígido. El método se puede descomponer en dos etapas de procesado:

1) Estimación del emparejamiento difuso. El objetivo de esta fase es calcular y parametrizar para cada pixel de una rejilla regular de la primera imagen la zona de la segunda imagen que es más similar. Se necesita definir el tamaño del bloque en la imagen original y el tamaño de la región de búsqueda en la segunda imagen (imagen con movimiento no rígido) para obtener el mapa de similitud. Posteriormente el campo de movimiento con la similitud más alta se parametriza como modelos punto, línea curva o indeterminado.

2) Regularización mediante modelos deformables: esta fase convierte la información difusa de cada pıxel en vectores de correspondencia numéricos. Esto se consigue mediante interpolación no uniforme y aplicación de restricciones, un proceso iterativo que consiste en un filtrado no lineal discreto de mínima energía restringido a los modelos paramétricos de cada pıxel.

2.3 MÉTODOS DE SEGMENTACIÓN DE IMÁGENES

El primer paso en cualquier proceso de análisis de imagen es la segmentación. Mediante la segmentación vamos a dividir la imagen en las partes u objetos que la forman. El nivel al que se realiza esta subdivisión depende de la aplicación en particular, es decir, la segmentación terminará cuando se hayan detectado todos los objetos de interés para la aplicación. En general, la segmentación automática es una de las tareas más complicadas dentro del procesado de imagen. La segmentación va a dar lugar en última instancia al éxito o fallo el proceso de análisis. En la mayor parte de los casos, una buena segmentación dará lugar a una solución correcta, por lo que, se debe poner todo el esfuerzo posible en la etapa de segmentación.

Los algoritmos de segmentación de imagen generalmente se basan en dos propiedades básicas de los niveles de gris de la imagen: discontinuidad y similitud. Dentro de la primera categoría se intenta dividir la imagen basándonos en los cambios bruscos en el nivel de gris. Las áreas de interés en esta categoría son la detección de puntos, de líneas

y de bordes en la imagen. Las áreas dentro de la segunda categoría están basadas en las técnicas de umbrales, crecimiento de regiones, y técnicas de división y fusión.

2.3.1 Detección de Discontinuidades

Generalidades

En esta sección vamos a presentar varias técnicas para detectar varios tipos de discontinuidades: puntos, líneas y bordes. El método más común de buscar discontinuidades es la correlación de la imagen con una máscara. En la Figura 2.6 se puede ver un caso general de máscara de 3 x 3. En este procedimiento se realiza el producto de los elementos de la máscara por el valor de gris correspondiente a los pixels de la imagen encerrados por la máscara. La respuesta a la máscara de cualquier pixel de la imagen viene dado por

$$R = \sum_{i=1}^{9} w_i z_i$$

Ecuación 2. 1

donde zi es el nivel de gris asociado al pixel de la imagen con coeficiente de la máscara wi. Como suele ser habitual, la respuesta de la máscara viene referida a su posición central. Cuando la máscara esté centrada en un pixel de borde de la imagen, la respuesta se determina empleando el vecindario parcial apropiado.

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

Figura 2. 6 Máscara 3x3

−1	−1	−1
−1	8	−1
−1	−1	−1

Figura 2. 7 Máscara usada para la detección de puntos aislados.

Detección de Puntos

La detección de puntos aislados es inmediata. Empleando la máscara de la Figura 2.7, vamos a decir que se ha detectado un punto en la posición en la cual está centrada la máscara si

$$|R| > T$$

Ecuación 2. 2

donde T es un umbral.

Básicamente se mide la diferencia entre el pixel central y sus vecinos, puesto que un pixel será un punto aislado siempre que sea suficientemente distinto de sus vecinos. Solamente se considerarán puntos aislados aquellos cuya diferencia con respecto a sus vecinos sea significativa.

Detección de Líneas

En este caso se consideran las máscaras de la Figura 2.8. Si pasamos la primera de las máscaras a lo largo de la imagen, tendrá mayor respuesta para líneas de ancho un pixel orientadas horizontalmente. Siempre que el fondo sea uniforme, la respuesta será máxima cuando la línea pase a lo largo de la segunda fila de la máscara.

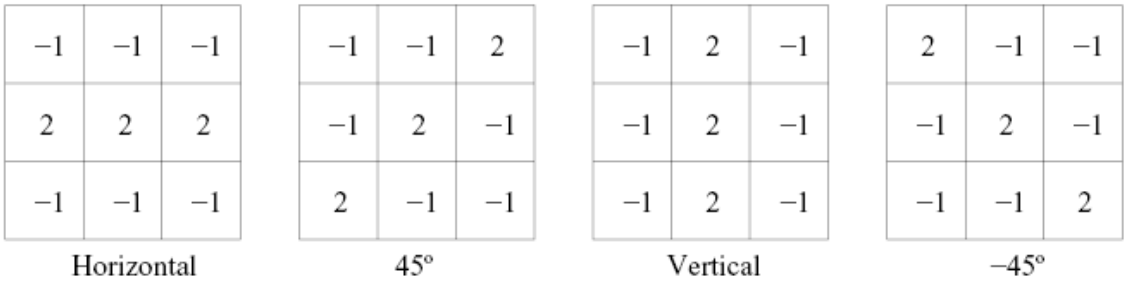


Figura 2. 8 Máscaras de Línea

La segunda máscara de la Figura 2.8 responderá mejor a líneas orientadas a 45º; la tercera máscara a líneas verticales; y la última a líneas orientadas a - 45º. Estas direcciones se puede establecer observando que para la dirección de interés las máscaras presentan valores mayores que para otras posibles direcciones. Si denotamos con R1, R2, R3 y R4 las respuestas de las cuatro máscaras de la Figura 2.8 para un pixel en particular, entonces si se cumple que:

$$|R_i| > |R_j| \text{ con } j \neq i,$$

Ecuación 2. 3

será más probable que dicho pixel esté asociado a la dirección correspondiente a la máscara i.

Detección de Bordes

Preliminares

La detección de bordes es el procedimiento empleado más habitualmente para la detección de discontinuidades. Un borde se define como la frontera entre dos regiones con nivel de gris relativamente diferente. Vamos a suponer a partir de ahora que las regiones de interés son suficientemente homogéneas de modo que la transición entre dichas regiones se puede determinar empleando exclusivamente las discontinuidades en el nivel de gris.

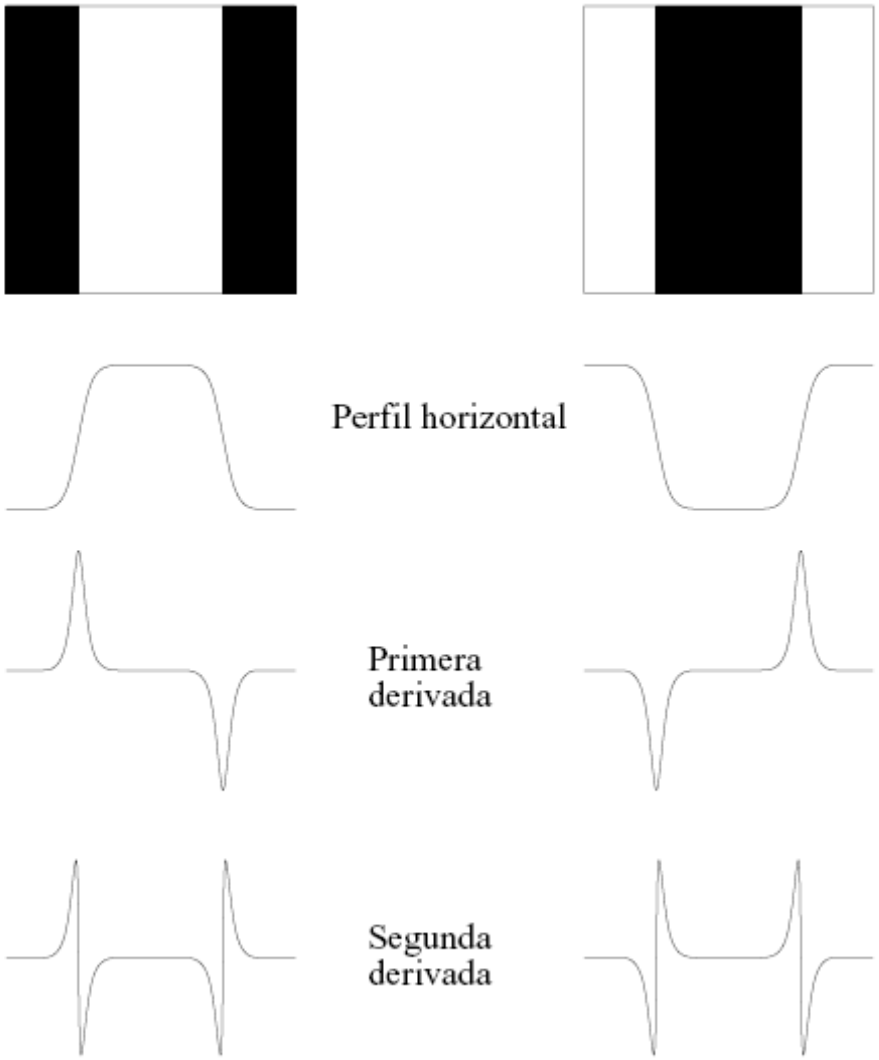


Figura 2. 9 Detección de bordes empleando operadores de derivación. La segunda derivada tiene un cruce por cero en la posición de cada borde.

La idea básica detrás de cualquier detector de bordes es el cálculo de un operador local de derivación. En la Figura 2.9 se puede ver este concepto. En la parte derecha se puede ver una imagen de una banda clara sobre un fondo oscuro, el perfil a lo largo de una línea horizontal y la primera y segunda derivada de dicho perfil. Se puede observar que el perfil del borde se ha modelado como una discontinuidad suave. Esto tiene en cuenta el hecho de que en las imágenes reales los bordes están ligeramente desenfocados.

Como se puede observar en la Figura 2.9 la primera derivada es positiva para cambio a nivel de gris más claro, negativa en caso contrario y cero en aquellas zonas con nivel de gris uniforme. La segunda derivada presenta valor positivo en la zona oscura de cada borde, valor negativo en la zona clara de cada borde y valor cero en las zonas de valor de gris constante y justo en la posición de los bordes. El valor de la magnitud de la primera derivada nos sirve para detectar la presencia de bordes, mientras que el signo de la segunda derivada nos indica si el pixel pertenece a la zona clara o a la zona oscura. Además la segunda derivada presenta siempre un cruce por cero en el punto medio de la transición. Esto puede ser muy útil para localizar bordes en una imagen.

Aunque lo que llevamos dicho se refiere a perfiles unidimensionales, la extensión a dos dimensiones es inmediata. Simplemente se define el perfill en la dirección perpendicular a la dirección del borde y la interpretación anterior seguirá siendo válida. La primera derivada en cualquier punto de la imagen vendrá dada por la magnitud del gradiente, mientras que la segunda derivada vendrá dada por el operador Laplaciano.

a. Gradiente

El gradiente de una imagen *I*(x; y) en la posición (x; y) viene dado por el vector

$$\nabla I = \begin{bmatrix} I_x \\ \\ I_y \end{bmatrix} = \begin{bmatrix} \frac{\partial I}{\partial x} \\ \\ \frac{\partial I}{\partial y} \end{bmatrix}.$$

Ecuación 2. 4

El vector gradiente siempre apunta en la dirección de la máxima variación de la imagen *I* en el punto (x; y). En la detección de bordes es muy importante la magnitud de este vector, denominado simplemente como gradiente de la imagen, denotado por

$$\nabla I$$

y dado por

$$\nabla I = ||\nabla I|| = \sqrt{I_x^2 + I_y^2}.$$

Ecuación 2. 5

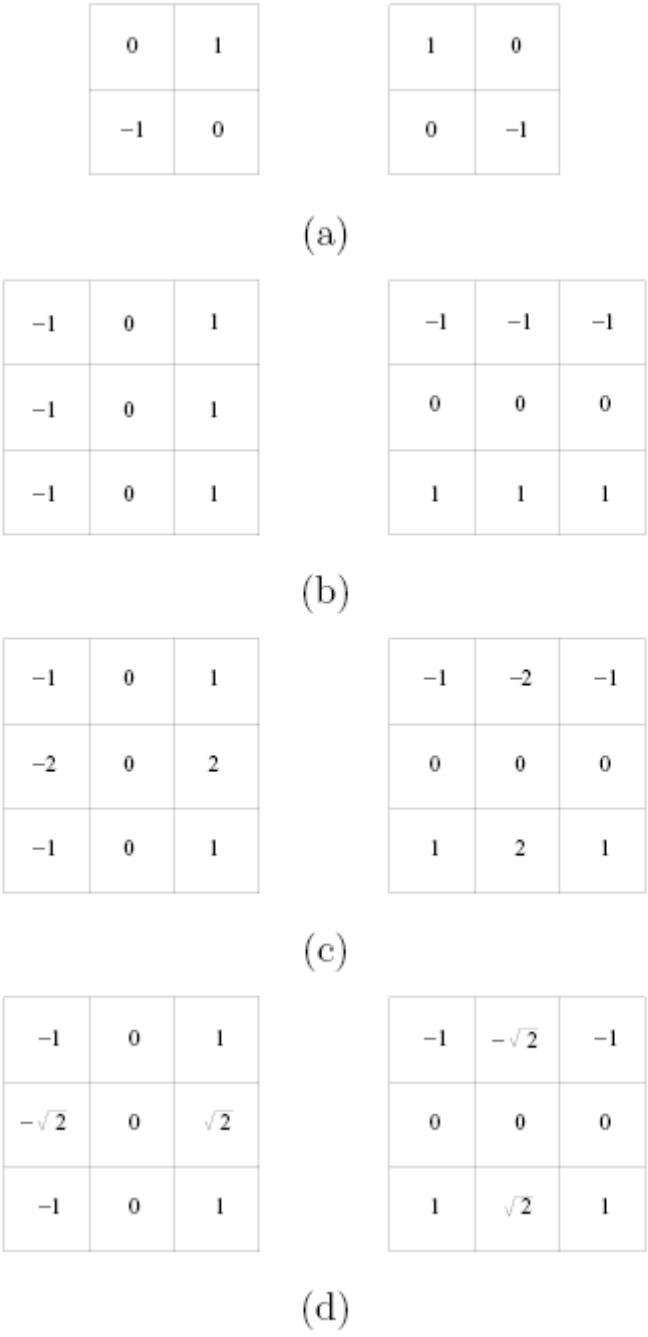


Figura 2. 10 Operadores de derivación: (a) de Roberts, (b) de Prewitt, (c) de Sobel y (d) de Frei-Chen.

Esta cantidad representa la variación de la imagen *I*(x; y) por unidad de distancia en la dirección del vector ∇I. En general, el gradiente se suele aproximar mediante la expresión

$$\nabla I \approx |I_x| + |I_y|,$$

Ecuación 2. 6

que es mucho más simple de implementar en la práctica. La dirección del vector gradiente también es una cantidad importante. Sea $\alpha(x, y)$ el ángulo del vector ∇I en el punto (x; y). Entonces se tiene que

$$\alpha(x, y) = \tan^{-1} \frac{I_y(x, y)}{I_x(x, y)}$$

Ecuación 2. 7

donde los ángulos se miden con respecto al eje de abscisas.

El cálculo del gradiente se basa en obtener las derivadas parciales para cada pixel. Las derivadas se pueden implementar digitalmente de varias formas.

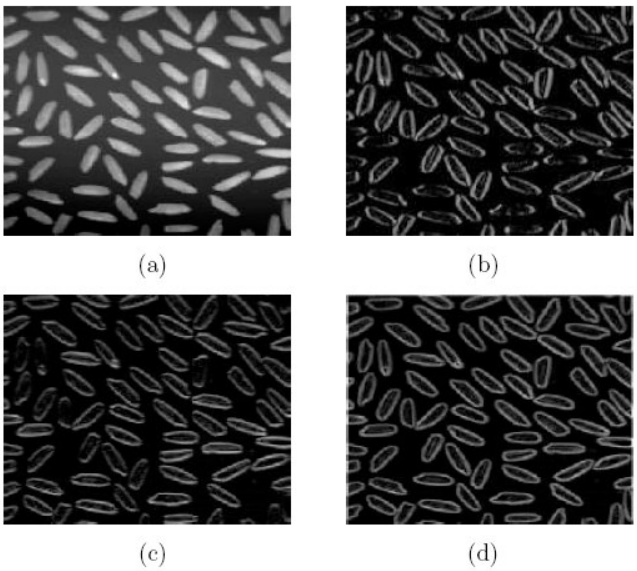


Figura 2. 11 (a) Imagen original. (b) Derivación horizontal usando Sobel. (c) Derivación vertical usando Sobel. (d) Imagen gradiente usando Sobel.

En la Figura 2.10 se pueden ver los operadores de Roberts, Prewitt, Sobel y FreiChen [17] para determinar las derivadas parciales. Sin embargo, los operadores de Sobel y de FreiChen tienen la ventaja de que proporcionan un suavizado además del efecto de derivación. Ya que la derivación acentúa el ruido, el efecto de suavizado es particularmente interesante, puesto que elimina parte del ruido. El requisito básico de un operador de derivación es que la suma de los coeficientes de la máscara sea nula, para que la derivada de una zona uniforme de la imagen sea cero.

Según la Figura 2.10 las derivadas según el operador de Sobel vienen dadas por

$$R_x = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

$$R_y = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$

Ecuación 2. 8

En la Figura 2.11 podemos ver el resultado de aplicar las máscaras de Sobel a una imagen que contiene granos de arroz.

b. Laplaciano

El Laplaciano de una imagen I(x; y) es una derivada de orden dos definida por

$$\nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial x^2}.$$

Ecuación 2. 9

En general se suele tomar el valor negativo del Laplaciano. Al igual que en el caso del gradiente se puede implementar en forma digital de varias formas.

0	−1	0
−1	4	−1
0	−1	0

Figura 2. 12 Máscara utilizada para calcular el Laplaciano.

Puesto que el Laplaciano es un operador de derivación la suma de los coeficientes debe ser cero. Además, el coeficiente asociado con el pixel central debe ser positivo y los demás coeficientes negativos o ceros. En la Figura 2.12 podemos ver una máscara para el Laplaciano. En este caso la expresión para determinar el Laplaciano viene dada por

$$4z_5 - (z_2 + z_4 + z_6 + z_8).$$

Aunque el Laplaciano responde a transiciones en la intensidad de la imagen, se emplea en pocas ocasiones en la práctica. Debido a que es un operador de segunda derivada es sensible en exceso a la presencia de ruido. Además, el Laplaciano da lugar a bordes dobles, como se puede apreciar en la Figura 2.9 y no permite determinar direcciones. En general juega un papel secundario en la detección de bordes para determinar si un pixel está en la zona clara o en la zona oscura del borde a través del signo del Laplaciano.

Un uso más generalizado del Laplaciano es a través de la propiedad de localización de los bordes usando la propiedad de los cruces por cero (ver Figura 2.9). Este concepto se basa en correlar la imagen con el Laplaciano de una función Gaussiana de la forma

$$h_\sigma(x,y) = \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right),$$

Ecuación 2. 10

donde σ es la desviación estándar. Si $r^2 = x^2+y^2$, el Laplaciano de una función h(r) se puede expresar como

$$\nabla^2 h = \frac{\partial^2 h}{\partial r^2} + \frac{1}{r}\frac{\partial h}{\partial r},$$

Ecuación 2. 11

entonces el Laplaciano de $h_\sigma(r)$ (h_σ no depende de la variable angular) se puede poner como

$$\nabla^2 h_\sigma = \left(\frac{r^2-2\sigma^2}{\sigma^4}\right)\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right).$$

Ecuación 2. 12

El negativo de esta función es una función suave que presenta cruces por cero para $\pm\sigma$, valor positivo en el origen y negativo a partir de $\pm\sigma$, como puede verse en la Figura 8. Se puede comprobar que el valor medio del operador $-\nabla^2 h_\sigma$ es cero, al igual que el valor medio de la imagen que se obtiene al emplear este operador. Correlaciondo una imagen con este operador, se suaviza o desenfoca por un factor proporcional al valor de σ empleado. Aunque éste da lugar a una reducción del ruido, su utilidad fundamental es debido a los cruces por cero.

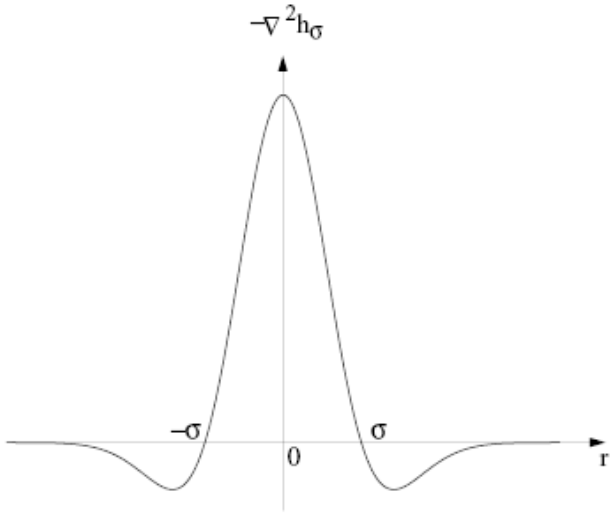


Figura 2. 13 Corte por el origen de $-\nabla^2 h_\sigma$.

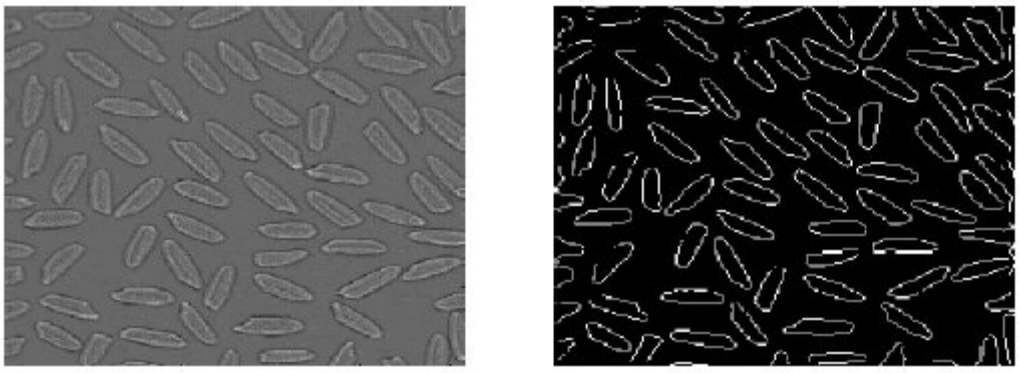


Figura 2. 14 (a) Resultado tras el operador $-\nabla^2 h_\sigma$. (b) Cruces por cero.

En la Figura 2.14(a) se puede ver el resultado de aplicar el operador $-\nabla^2 h_\sigma$ a la imagen de la Figura 2.11(a). En la Figura 2.14(b) se pueden observar los cruces por cero.

2.3.2 Enlazado de Bordes.

Necesidad

Las técnicas de detección de discontinuidades idealmente, proporcionarán los pixels correspondientes a los contornos o fronteras entre las regiones de la imagen. En la práctica este conjunto de pixels no suele caracterizar completamente a esos contornos

debido a la presencia de ruido, ruptura en los propios contornos debido a la iluminación no uniforme y otros efectos que introducen espúreos en las discontinuidades de la intensidad. En general, después de los procedimientos de detección de bordes se suele emplear técnicas de enlazado u otras técnicas de detección de contornos designadas para unir los pixels de bordes en contornos significativos.

Procesado Local

Uno de los procedimientos más simples para enlazar puntos de bordes es analizar las características de los pixels en un vecindario pequeño (ventana de 3 X 3) alrededor de cada punto (x; y) donde se ha detectado la presencia de un borde.

Todos los puntos que son similares se enlazan, dando lugar a un contorno o frontera de pixels que comparten ciertas propiedades en común.

Las dos propiedades que se suelen utilizar son la magnitud del operador de gradiente que ha dado lugar al pixel de borde y la dirección del gradiente en ese punto. Un pixel de borde con coordenadas (x', y') en el vecindario predefinido para el pixel (x; y) va a ser similar en magnitud al pixel (x; y) si

$$|\nabla f(x,y) - \nabla f(x',y')| \leq T,$$

donde T es un cierto umbral en amplitud.

La dirección del vector gradiente vendrá dado por la ecuación (2.7). Un píxel de borde con coordenadas (x', y') en el vecindario predefinido para el píxel (x; y) va a ser similar en ángulo al pixel (x; y) si

$$|\alpha(x,y) - \alpha(x',y')| \leq A,$$

Ecuación 2. 13

donde A es un cierto umbral angular. Hay que tener en cuenta que la dirección del borde en el punto (x; y) es perpendicular a la dirección del gradiente en ese punto. Sin embargo, de cara a comparar las direcciones, la ecuación (2.13) es equivalente.

Se enlazará un punto, en el vecindario predefinido para (x; y), con el píxel (x; y) si se cumplen simultáneamente las condiciones de magnitud y de ángulo. Este procedimiento se repetirá para todos los pixels de borde de la imagen.

Tras encontrar un pixel de borde que enlaza con el (x; y) se procede a probar con este nuevo pixel de borde, hasta que no se puedan enlazar más pixels de borde. Todos estos pixels enlazados se etiquetan con cierto valor de gris y se procederá a enlazar los restantes pixels de borde con otra etiqueta. Al Final habrá tantas etiquetas como conjuntos de pixels enlazados, y que corresponderán a los contornos o fronteras correspondientes a todos los objetos detectados de la imagen.

2.3.3 Técnicas de Umbrales

Fundamentos

Supongamos que el histograma de los niveles de gris de una imagen I(x; y) es el que se muestra en la Figura 2.15(a). La imagen I(x; y) está compuesta de objetos claros sobre fondo oscuro de tal forma que los niveles de gris está agrupados en dos modos predominantes. Una forma de separar los objetos del fondo consiste en seleccionar un umbral T que separe esos modos. Entonces, cualquier punto (x, y) para el que se cumpla que

$$I(x, y) > T,$$

Ecuación 2. 14

se lo etiqueta como objeto; en otro caso, como fondo. La Figura 2.15 (b) muestra el histograma de otra imagen en un caso más general. En este caso el histograma de la imagen está caracterizado por tres modos dominantes. Esto ocurrirá cuando tengamos dos tipos de objetos claros sobre fondo oscuro, por ejemplo. Se puede utilizar el mismo principio para clasificar cada punto (x; y). Si

$$T_1 < I(x, y) \leq T_2$$

entonces se lo etiqueta como primer objeto, si

$$I(x, y) > T_2$$

como segundo objeto y si

$$I(x, y) \leq T_1$$

como fondo. En general, este tipo de clasificación con varios umbrales es menos viable, ya que es más difícil determinar esos umbrales que aíslen de forma efectiva las regiones de interés, especialmente cuando el número de modos del histograma aumenta.

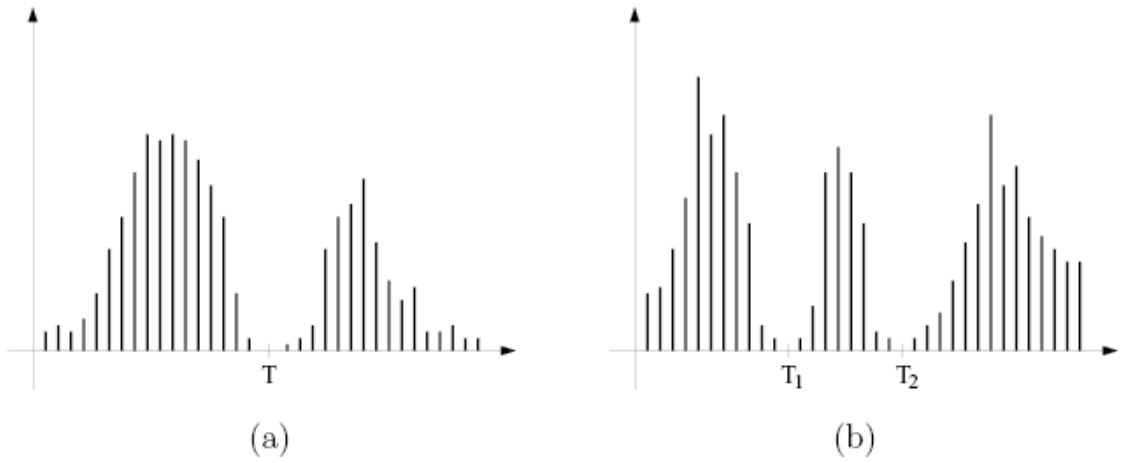


Figura 2. 15 Histogramas de nivel de gris que se pueden segmentar con (a) un único umbral y (b) con múltiples umbrales.

En este caso es mejor emplear umbrales variables. En general, un método de umbral se puede ver como una operación en la que se hace un test de cada pixel con respecto a una función T de la forma

$$T = T(x, y, p(x, y), I(x, y)],$$

Ecuación 2. 15

donde I(x; y) es el nivel de gris del punto (x; y) y p(x; y) denota cualquier propiedad local de ese punto (como por ejemplo el nivel de gris medio en un vecindario centrado en (x, y). El método de umbral dará lugar a otra imagen B(x, y) definida por

$$B(x, y) = \begin{cases} 1 & \text{si } I(x, y) > T \\ 0 & \text{si } I(x, y) \leq T. \end{cases}$$

Ecuación 2. 16

En este caso un pixel con etiqueta 1 de la imagen B corresponderá a objetos, mientras que un pixel con etiqueta 0 corresponderá al fondo. Cuando T dependa sólo del nivel de

gris I(x; y) se denomina umbral global (en la Figura 2.15(a) se puede ver un ejemplo en este caso). Si T depende tanto del nivel de gris I(x; y) como de la propiedad local p(x; y), el umbral se denomina local. Si, además, depende de las coordenadas espaciales x e y, el umbral se denomina dinámico.

En la Figura 2.16(a) se puede ver el histograma de la imagen de la Figura 2.16(a) para el que se ha fijado un umbral T. En la Figura 2.16(b) se muestra el resultado de segmentar dicha imagen con ese umbral T. Como se puede apreciar el histograma de esta imagen es bimodal.

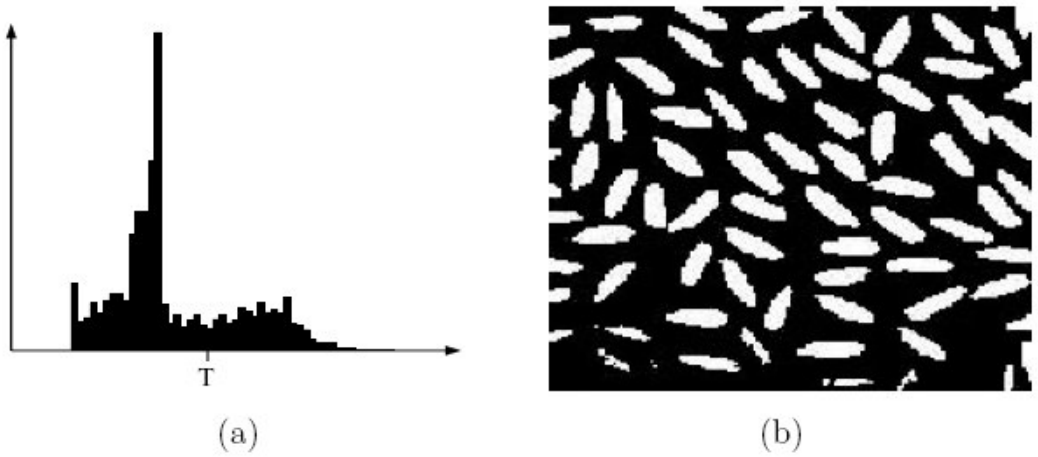


Figura 2. 16 (a) Histograma y umbral T para la imagen de la Figura 2.11(a). (b) Imagen segmentada usando el umbral T.

Sin embargo, el valle no está claramente marcado debido fundamentalmente a que la iluminación de la imagen no es constante. Esto influye enormemente en el histograma como veremos a continuación. Sin embargo, en este caso la segmentación es bastante adecuada, excepto en la zona inferior donde la iluminación de la imagen es peor.

2.3.4 Segmentación Orientada a Regiones

Preliminares

Sea R la región correspondiente a la imagen a segmentar. Vamos a ver el proceso de segmentación como un proceso en el cual dividimos la región *R* en n subregiones *R1,R2,…,Rn*, tal que:

$$\bigcup_{i=1}^{n} R_i = R$$

R_i es una región conectada, $i = 1, 2, \ldots, n$

$R_i \cap R_j = \emptyset$ para todo i y j con $j \neq i$

$P(R_i) = \text{CIERTO}$ para $i = 1, 2, \ldots, n$

$P(R_i \cup R_j) = \text{FALSO}$ para todo i y j adyacentes con $j \neq i$,

Ecuación 2. 17

El primer enunciado indica que la segmentación debe ser completa, es decir, que todo pixel debe estar en una región. El segundo requiere que todos los pixels pertenecientes a una región estén conectados. El tercer enunciado lleva consigo regiones disjuntas, por eso mismo, los conjuntos de pixels *R1,R2,…,Rn* son una partición de R. El cuarto enunciado se refiere a la propiedad que deben cumplir los pixels dentro de cada región segmentada (por ejemplo P(Ri) =CIERTO si todos los pixels de Ri tienen el mismo nivel de gris dentro de una cierta tolerancia). Finalmente, el último enunciado indica que las regiones adyacentes Ri y Rj deben ser distintas con respecto a la propiedad P.

Crecimiento de Regiones

Como el nombre indica, el crecimiento de regiones es un procedimiento mediante el cual se agrupan pixels o subregiones en regiones mayores. El procedimiento más sencillo se denomina agregación de pixels, que comienza a partir de un conjunto de pixels semilla, de forma que a partir de cada semilla se crecen regiones añadiendo pixels a dicha semilla de entre aquellos pixels vecinos que tienen propiedades similares. El resultado de la segmentación dará lugar como mucho a tantas regiones como semillas haya. Sin embargo, puede darse el caso de que dos de esas semillas correspondan a pixels de la misma región.

En este caso el crecimiento desde una de las semillas absorberá a la otra, que en este caso deberá ser descartada. Un ejemplo sencillo, pero muy usado en la práctica de similitud es la diferencia absoluta en el nivel de gris. Fijado un umbral T se va calculando la diferencia en valor absoluto del nivel de gris del pixel en cuestión (en el vecindario de la región crecida hasta el momento) con respecto al nivel de gris de la semilla y si no se supera ese umbral T se añade a la región.

Dos problemas fundamentales en el crecimiento de regiones son: por un lado, la selección de las semillas o puntos de partida que representen adecuadamente a las regiones de interés; y por otro, la elección de las propiedades adecuadas que permitan ir añadiendo pixels durante el proceso de crecimiento.

La selección de los puntos de partida en muchos casos depende de la naturaleza de la imagen a segmentar. Por ejemplo, en aplicaciones militares con imágenes de infrarrojos, los blancos de interés normalmente desprenden calor, por lo que corresponden a pixels claros frente a fondo oscuro. En este caso, los pixels claros son una elección natural para las semillas. En ausencia de conocimiento a priori sobre el problema, un procedimiento que suele dar lugar a buenos resultados consiste en calcular para cada pixel de la imagen las mismas propiedades que luego se emplearán durante el proceso de crecimiento.

Si el resultado del cálculo de dichas propiedades da lugar a agrupaciones de los valores para dichas propiedades, se pueden emplear como semillas los pixels cercanos a los centroides de dichas agrupaciones. Por ejemplo, los pixels correspondientes a los valores máximos de cada modo del histograma de nivel de gris podrán servir como semillas cuando se emplee como propiedad la similitud en el nivel de gris.

La selección del criterio de similitud depende no sólo del problema considerado, sino también del tipo de imagen disponible. En imágenes en color o multiespectrales (fotografías desde satélite, por ejemplo) se pueden emplear propiedades de similitud en los diferentes canales disponibles. Sin embargo, en general, lo habitual es trabajar con imágenes de intensidad con un único canal disponible. En este caso las propiedades se basan en descriptores locales a partir de una única imagen como son la intensidad y propiedades espaciales (basadas momentos y textura). En cualquier caso siempre hay

que tener en cuenta la conectividad durante el proceso de crecimiento para que el resultado tenga significado dentro de su contexto. Otro problema añadido en el crecimiento de regiones es la regla de parada. Básicamente, el crecimiento termina cuando no existen más pixels en el vecindario de la región ya crecida que cumplan el criterio de similitud. Los criterios de similitud mencionados son locales y no tienen en cuenta la “historia" del crecimiento de la región. Se pueden emplear criterios adicionales que incrementen la potencia del algoritmo de crecimiento de regiones, utilizando el concepto de similitud entre el pixel candidato y los pixels de la región crecida hasta ese momento, o usando el tamaño y la forma de la región crecida. La utilización de estos tipos de descriptores se basa en la suposición de que un modelo de los resultados esperados ya está parcialmente disponible.

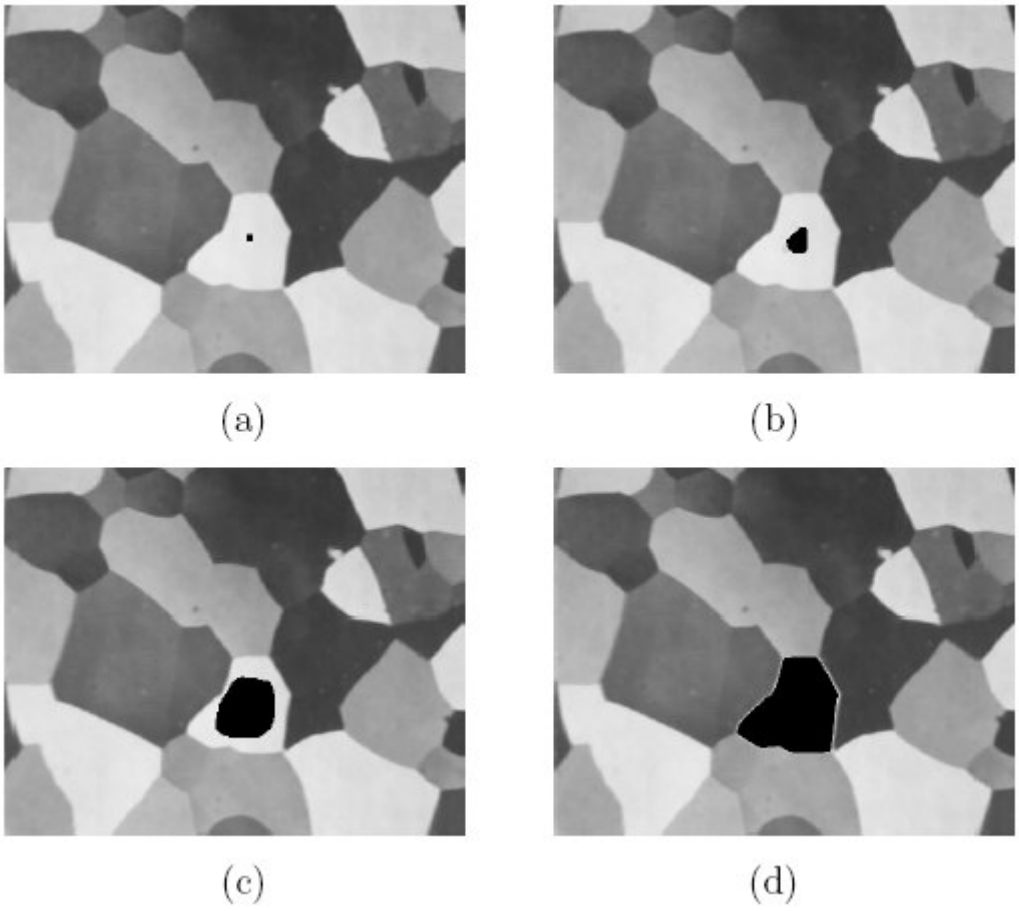


Figura 2. 17 (a) Imagen en la que se ha marcado un punto interior a la región a segmentar. (b)-(d) Proceso de crecimiento de la región marcada.

En la Figura 2.17(a) se muestra una imagen con varias regiones en la que se muestra una única semilla en el interior de una de las regiones a segmentar. El criterio empleado para el crecimiento es que el valor absoluto de la diferencia entre los niveles de gris del pixel candidato y la semilla no exceda el diez por ciento de la diferencia entre el mayor nivel de gris de la imagen completa y el menor. Además los pixels agregados a la región deben tener conectividad tipo ocho con respecto a los pixels ya incluidos en la región. En la Figura 2.17(b) se muestra el proceso de crecimiento tras unas pocas iteraciones, en la Fgura 2.17(c) se muestra el proceso para la mitad de las iteraciones y en la Figura 2.17(d) el resultado del crecimiento de la región para cuando ninguno de los pixels vecinos de la región cumple el criterio de similitud.

División y Fusión de Regiones

Un proceso alternativo al crecimiento de regiones es subdividir inicialmente la imagen en un conjunto arbitrario de regiones disjuntas y posteriormente fusionar y/o dividir estas regiones con el objetivo de que finalmente se cumplan las condiciones dadas por los enunciados de las ecuaciones 2.17.

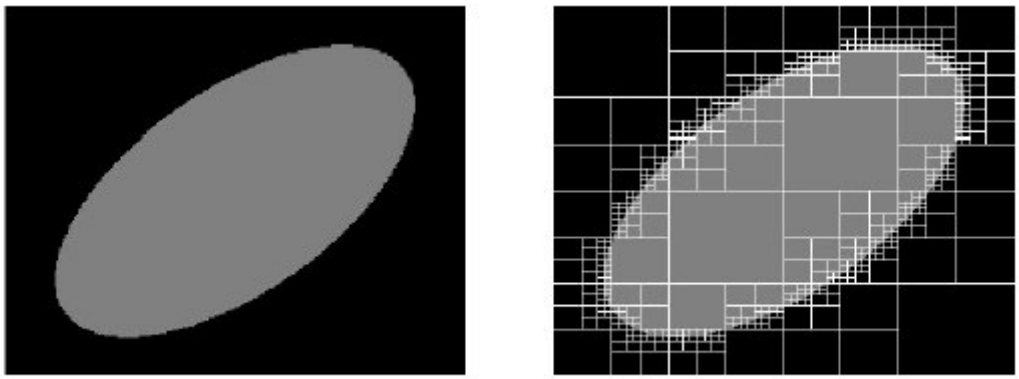


Figura 2. 18(a) Imagen sintética. (b) División de la imagen correspondiente a la representación *quadtree*.

Sea R la región correspondiente a la imagen completa y sea P el predicado de similitud elegido. Un procedimiento para segmentar R consiste en subdividir recursivamente cada

región en cuadrantes más y más pequeños hasta que se cumpla para cada región Ri que P(Ri) = CIERTO. Es decir si P(R) = FALSO, se divide la imagen en cuatro regiones (cuadrantes). Si para alguno de los cuadrantes P es FALSO, se vuelve a dividir dicho cuadrante en cuatro subcuadrantes y así sucesivamente. Esta técnica de división de una imagen da lugar a un tipo de representación bastante conveniente denominada *quadtree*, es decir, un árbol de regiones en el que cada región tiene exactamente cuatro descendientes. La raíz del árbol corresponde siempre a la imagen completa R. Cada nodo corresponde a una subdivisión. En la Figura 2.18(a) podemos ver una imagen sintética generada por ordenador de una elipse clara sobre un fondo oscuro. En este caso el criterio de similitud es que la diferencia en valor absoluto en los niveles de gris sea menor que cierto umbral T. En la Figura 2.18(b) podemos ver el resultado de la división de la imagen en cuadrantes correspondientes a los nodos finales del *quadtree*. Las zonas más uniformes dan lugar a regiones mayores, mientras que las zonas de transición dan lugar a regiones pequeñas, de forma que siempre se cumpla el predicado P en cada región.

Tras la etapa de división, la partición final contiene regiones adyacentes con propiedades idénticas, por lo que no se cumple la propiedad correspondiente a la ecuación (44). Esto se puede solucionar permitiendo la fusión de regiones adyacentes cuando la unión de ambas cumplan el predicado P.

Podríamos resumir el procedimiento de división y fusión de regiones como sigue:

1. Dividir una región Ri es sus cuatro cuadrantes disjuntos siempre que *P(Ri) = FALSO.*
2. Fusionar dos regiones adyacentes *Ri* y *Rj* siempre que *P(Ri , Rj) = CIERTO*.
3. Parar el algoritmo cuando no sea posible realizar más divisiones y fusiones.

2.4 MÉTODOS DE INTERPOLACIÓN DE IMÁGENES

En la rama matemática del análisis numérico, se denomina interpolación a la construcción de nuevos puntos dados partiendo del conocimiento de un conjunto de puntos dados discretos.

En ingeniería y ciencias es frecuente disponer de un cierto número de puntos obtenidos por muestreo o a partir de un muestreo o experimento y pretender construir una función que los ajuste.

Otro problema estrechamente ligado con el de la interpolación es la aproximación de una función complicada por una más simple. Si tenemos una función cuyo cálculo resulta costoso, podemos partir de un cierto número de sus valores e interpolar dichos datos construyendo una función más simple. En general, por supuesto, no obtendremos los mismos valores evaluando la función obtenida que si evaluásemos la función original, si bien dependiendo de las características del problema y del método de interpolación usado la ganancia en eficiencia puede compensar el error cometido.

En el mundo de la imagen digital, la interpolación aplica este mismo patrón para conseguir un tamaño mayor de la imagen inicial, rellenando la información que falta con datos «inventados» a partir de un algoritmo específico.

La aplicación de la interpolación de imágenes en el área de la Reconstrucción 3D a partir de tomografías e imágenes médicas en general se encuentra en que frecuentemente es necesario tener una mayor resolución de la imagen o poseer un mayor número de tomografías, ya que en el estudio no se tomaron las suficientes. En este caso es necesario generar nuevas imágenes de tomografía a partir de las ya existentes.

Existen varios algoritmos, los más conocidos son:

- **Interpolación por aproximación.** Es uno de los métodos más antiguos que se basa en obtener el promedio de valores de los 2 pixeles más próximos. La interpolación bilineal es una mejora de la anterior, promediando en este caso 4 pixeles adyacentes. La interpolación Bilineal explora los cuatro puntos vecinos de

un punto (x,y), y asume que la función de luminosidad (brillo) tambien es bilineal en estos vecinos.

- **Interpolación bicúbica.** Es el método de interpolación considerado estándar (promedia 16 pixeles adyacentes).Existen varias técnicas para usar modificar el tamaño de una imagen. Éstos generalmente tienen un ajuste entre la velocidad y el grado a que ellos reducen los artefactos visuales. El método más simple para agrandar una imagen por un factor determinado, es reproducir cada pixel 4 veces. Claro, esto conducirá a producir bordes dentados más pronunciados que en la imagen original. Lo mismo se aplica para reducir una imagen por un divisor del entero de la anchura de cada pixel. El caso más general de cambiar el tamaño de una imagen por una cantidad arbitraria requiere interpolación del colores entre los pixeles.
- **Interpolación en escalera** (*Stair Interpolation*): Se basa en la interpolación bicúbica con la diferencia que se va interpolando en incrementos de un 10% en cada paso con respecto al anterior.
- **Interpolación S-SPline:** Este método de interpolación determina el color de un pixel «desconocido» basándose en la totalidad de colores de la imagen, a diferencia que los métodos anteriores.

2.5 MÉTODOS DE RECONSTRUCCIÓN DE SUPERFICIES

Los gráficos tridimensionales son una herramienta útil en cualquier ámbito de la ciencia donde se necesite visualización de datos. En medicina es de mucha utilidad poder visualizar estructuras anatómicas en un espacio tridimensional que simule al objeto real que en este caso vendría a ser la anatomía humana de cada paciente. Debido a que el médico necesita visualizar la estructura interna de éste, es necesario buscar técnicas que permitan reconstruir de manera fiables órganos del paciente en estudio a partir de alguna técnica, esto es lo que se conoce como Reconstrucción volumétrica de estructuras anatómicas.

Existen varias técnicas y métodos para realizar Reconstrucción tridimensional, pero el más eficiente y aplicado en el área de medicina es el algoritmo Marching Cubes y Marching tetrahedra [18], que transforma modelos basados en de vóxeles en modelos basados en superficies; aunque en general esas transformaciones no pueden reproducir todos los detalles.

En esta sección se presenta una perspectiva general de los métodos de Marching Cubes y Tetrahedra para la reconstruccióon de superficies, y en el siguiente capítulo se hará una profundización de estos métodos. Estos métodos, si bien su objetivo principal es la reconstrucción tridimensional, funcionan también como métodos de segmentación donde el criterio de segmentación es la intensidad de los voxels, que generan las isosuperficies buscadas.

En 1987 Lorensen y Cline introdujeron la descripción del algoritmo “Marching Cubes” en la convención anual del Siggraph[15]. Este método se basa en el estudio del modelo tras su división en celdas, como el caso límite de los voxels; Estas celdas están dispuestas de modo que seccionan el objeto en las tres dimensiones. El algoritmo analizará las celdas en subgrupos de ocho contiguas que forman un cubo mayor siendo éstas sus esquinas. Un primer análisis intuitivo nos hacer a ver la idea de que si alguna de estas equinas (que son voxels de la imagen) falta se puede aproximar a un polígono cerrando las esquinas vecinas y dirigiendo su vector normal hacia el ausente. Formalmente el algoritmo se apoya en una tabla pregenerada de arrays de bits, 8, que proporciona una cardinalidad de 256 configuraciones distintas de ausencia o existencia de las esquinas de nuestro “*cubo de análisis”* y que enlaza con una tabla de polígonos equivalentes correspondientes a su forma más cercana. Estas 256 configuraciones nacen tras el desarrollo de 16 casos básicos mediante rotaciones simétricas y reflexiones.

El modelo sería *rastreado* completamente hasta finalmente unificar con operaciones regularizadas todos los polígonos y obtener así finalmente un modelo de enmallado válido [19].

El concepto de marching tetrahedra es el mismo que se aplica en Marching Cubes. La principal diferencia es que marching tetrahedra utiliza como figura elemental de corte un

tetraedro y no un cubo. Esa consiste en dividir los cubos en una serie de tetraedros. Un tetraedro solo tiene 16 triangulaciones posibles, las que se reducen a 3 por simetría.

Para una explicación más en detalle de los métodos existentes de Reconstrucción de tomografías y su fundamento matemático ver el Apéndice. Los métodos antes mencionados se explicarán y se desarrollarán con detalle en el siguente capítulo.

2.6 MÉTODOS VISUALIZACIÓN DE VOLÚMENES

Existen diferentes métodos para la representación tridimensional de imágenes en una pantalla o monitor. El que es a nivel mundial utilizado por la mayoría de sistemas gráficos actuales es el llamado Raytracing o trazado de rayos, ya que proporciona las herramientas necesarias para representar objetos virtuales con características propias de entornos reales, tales como sombras, iluminación y texturas.

El Raytracing o trazado de rayos es un algoritmo para síntesis de imágenes tridimensionales. Propuesto inicialmente por Turner Whitted en 1980 [20], está basado en el algoritmo de determinación de superficies visibles de Arthur Appel denominado Ray Casting (1968).

En el algoritmo Ray Casting se determinan las superficies visibles en la escena que se quiere sintetizar trazando rayos desde el observador (cámara) hasta la escena a través del plano de la imagen. Se calculan las intersecciones del rayo con los diferentes objetos de la escena y aquella intersección que esté más cerca del observador determina cuál es el objeto visible.

El algoritmo de trazado de rayos extiende la idea de trazar los rayos para determinar las superficies visibles con un proceso de sombreado (cálculo de la intensidad del pixel) que tiene en cuenta efectos globales de iluminación como pueden ser reflexiones, refracciones o sombras arrojadas.

Para simular los efectos de reflexión y refracción se trazan rayos recursivamente desde el punto de intersección que se está sombreando dependiendo de las características del material del objeto intersectado.

Para simular las sombras arrojadas se lanzan rayos desde el punto de intersección hasta las fuentes de luz. Estos rayos se conocen con el nombre de rayos de sombra (shadow rays).

El algoritmo básico de trazado de rayos fue mejorado por Robert Cook (1985) [21] para simular otros efectos en las imágenes mediante el muestreo estocástico usando un método de Monte Carlo; entre estos efectos podemos citar el desenfoque por movimiento (blur motion), la profundidad de campo o el submuestreo para eliminar efectos de aliasing en la imagen resultante.

En la actualidad, el algoritmo de trazado de rayos es la base de otros algoritmos más complejos para síntesis de imágenes (Mapeado de fotones, Metropolis, entre otros) que son capaces de simular efectos de iluminación global complejos como la mezcla de colores (color blending) o las cáusticas.

A continuación se presenta un esquema general a manera de algoritmo representativo del método de Ray-Tracing.

2.6.1 Algoritmo Ray - Tracing

```
Para cada pixel de la imagen{
 Crear un rayo desde el punto de visión a través del pixelActual
 Inicializar NearestT al INFINITO y NearestObject a NULL
 Para cada objeto de la escena {
   Si el rayo intersecta el objetoActual{
     Si t de la intersección es menor que NearestT {
         Poner NearestT = t de la intersección
         Poner NearestObject a objetoActual
     }
   }
 }
 Si NearestObject = NULL{
    Rellenamos pixelActual con el color de fondo
 }
 Sino{
    Lanzar un rayo a cada foco de luz para comprobar las sombras
    Si la superficie es reflectiva, generar un rayo reflectivo
(recursivo)
```

```
    Si la superficie es transparente, generar un rayo refractante
(recursivo)
    Usar NearestObject y NearestT para computar la función de sombreado
    Rellenar este pixel con el color resultante de la función de
sombreado
 }
}
```

3. ANÁLSIS E IMPLEMENTACIÓN DE ALGORITMOS

3.1. SEGMENTACIÓN

En esta sección se detallan los métodos de segmentación basados en regiones clasificados según su funcionamiento.

3.1.1 Método basado en Pixels

Este método de segmentación toma en cuenta solo el valor de gris de un pixel, para decidir si el mismo pertenece o no al objeto de interés. Para ello, se debe encontrar el rango de valores de gris que caracterizan dicho objeto, lo que requiere entonces la búsqueda y el análisis del histograma de la imagen. El objetivo de este análisis, es el de encontrar de una manera óptima los valores característicos de la imagen que establecen la separación del objeto de interés, con respecto a las regiones que no pertenecen al mismo; debido a esta característica y si los valores de gris del objeto y del resto de la imagen difieren claramente, entonces el histograma mostrará una distribución bimodal, con dos máximos distintos, lo que debiera generar, la existencia de una zona del histograma ubicada entre los dos máximos, que no presente los valores característicos, y que idealmente fuera igual a cero, con lo cual se logrará una separación perfecta entre el objeto y la región de la imagen que lo circunda, al establecer un valor umbral ubicado en esta región del histograma. Por lo tanto cada pixel de la imagen, es asignado a una de dos categorías, dependiendo si el valor umbral es excedido o no. Si el valor del histograma ubicado entre los dos máximos, es distinto de cero, las funciones de probabilidad de los valores de gris del objeto y de la región restante, se solaparán, de tal manera que algunos pixels del objeto deberán ser tomados como pertenecientes a la región circundante y viceversa. Conocida la distribución de la función de probabilidad de los pixels del objeto y de la región circundante, es posible aplicar análisis estadístico en el proceso de buscar un umbral óptimo, con el número mínimo de correspondencias

erróneas. Estas distribuciones pueden ser estimadas por histogramas locales, los cuales solamente incluyen las regiones correspondientes de la imagen.

3.1.2 Método basado en Contornos

En el método basado en pixels, el tamaño del objeto de interés depende del nivel de umbral escogido. La variación del tamaño es una característica dada por el hecho de que los valores de gris en el contorno de un objeto cambian gradualmente desde la región circundante hacia el mismo. El método basado en contornos puede ser usado para evitar la variación del tamaño del objeto. Este método se basa en realizar la búsqueda del valor máximo del gradiente, sobre cada línea que forma la imagen. Cuando un máximo es encontrado, un algoritmo de trazado trata de seguir el máximo del gradiente alrededor del objeto, hasta encontrar de nuevo el punto inicial, para luego buscar el próximo máximo en el gradiente.

3.1.3 Métodos Basados en Regiones

En el método de segmentación basado en pixels, la idea fundamental es clasificar un punto como del objeto, solamente tomando en cuenta el valor de gris que el mismo tiene asociado, lo que conlleva a que puntos aislados o pequeñas áreas puedan ser clasificadas como pertenecientes a la región de interés, es decir, allí no se toma en cuenta la conectividad como característica importante del objeto.

Los métodos de segmentación basados en regiones, toman en cuenta un conjunto de puntos de la imagen, a los cuales se les analiza características como, la posición en el espacio de intensidades, las relaciones topológicas (conectividad) y las característica de las fronteras entre dos conjuntos. Dependiendo de como sea analizada la posición en el espacio y las relaciones espaciales existentes entre los pixels, se pueden encontrar métodos de Clasificación [9] y métodos por Crecimiento de Regiones [2].

Los métodos de Clasificación determinan primero una partición del espacio de intensidades y utilizan luego las relaciones de conectividad, para determinar una región. Los métodos de Crecimiento de Regiones utilizan de manera simultánea los dos tipos de información.

3.1.4 Segmentación de Regiones por Clasificación

Inicialmente se utilizan los niveles de gris presentes en la imagen para obtener una partición del espacio. Se asocia a cada pixel la clase de nivel de gris a la cual pertenece. Las regiones son definidas por los conjuntos de pixels conexos pertenecientes a una misma clase. Este método utiliza el cálculo del histograma para realizar la clasificación de las intensidades de la imagen, realizando la búsqueda de los distintos modos del histograma y sus valles correspondientes; las clases son determinadas por los valores entre los valles, de esta forma los puntos de la imagen son etiquetados con la clase correspondiente. Este método de segmentación es eficaz, si la clasificación de las intensidades permite definir, las diferentes regiones homogéneas de la imagen.

Estos métodos son los más básicos y se basan solo en características locales de la imagen. El las secciones siguientes se presentan técnicas más complejas, que realizan la segmentación teniendo en cuenta más características de la imagen.

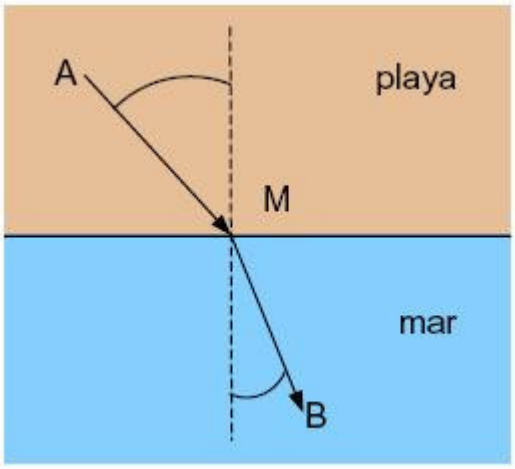


Figura 3. 1 Camino de mínimo costo

3.1.5 Level Sets

En esta primera sección se detallan los conceptos básicos e introductorios para entender el concepto de Level Sets, introducido inicialmente por J. A. Sethian en [1]. Level Sets utiliza como fundamento básico el principio de Camino Mínimo para la extracción de

curvas en las imágenes.

Teoría de Camino Mínimo (Minimal Path)

Para explicar mejor el principio de Camino Mínimo, imaginemos un pequeño ejemplo en el cual la línea costera separa el mar de la playa. Un guardavidas sentado en la playa (en el punto A) ve una joven dentro del agua ahogándose (en el punto B) algunos metros dentro del mar. Asumiendo que el guardavidas puede correr tres veces mas rápido de lo que puede nadar, el menor camino en cuestión de tiempo que le tomara llega a la joven no es el que describe la línea recta que pasa por los puntos A y B, sino que deberá recorrer la línea que va de A a M (por la playa) y al entrar en el agua, la línea que une M a B. En LA Figura 3.1 puede verse este ejemplo.

El principio de Fermat dice que a luz a una determinada frecuencia viaja la distancia que separa dos puntos a la menor velocidad posible. El ejemplo mas claro de esto es la luz viajando en un medio homogéneo, en el cual la velocidad no varía con la posición. En este caso el menor tiempo es equivalente a la menor distancia, lo cual es una línea recta. Cuando el medio no es homogéneo, hay un ángulo de refracción entre los dos medios homogéneos.

El principio de mínimo costo puede ser utilizado para extracción de curvas en imágenes. El objetivo es encontrar el contorno que mejor ajusta el borde de un objeto o línea de interés. Este contorno ubicado entre dos puntos determinados será aquel sobre el cual, la integral del potencial P[8] tendrá valor mínimo. Entonces es necesario contar con algoritmos que encuentren el camino que hace que la integral sobre P sea mínima. Las técnicas clásicas de extracción de caminos se basan en Snakes. Éstos se inician desde un camino cercano a la solución y convergen a un mínimo local de la energía. Este método es detallado en secciones posteriores.
A continuación se presentan las ideas básicas del método presentado por Cohen-Kimmel para encontrar un camino entre dos puntos con el costo mínimo (en términos de potencial P).

[8] P: índice de refracción, o Potencial. Tiene la característica de tomar valores bajos cerca de los bordes.

Método de Cohen-Kimmel en 2D

La energía a minimizar es similar a la de los modelos deformables clásicos donde se combinan las energías de suavizado y de la imagen (apéndice B). El concepto introducido en [4] modifica estas formulaciones para reducir la interacción del usuario, y limita solamente a asignar 2 puntos del contorno de *C*.

Planteo del problema. La mayoría de los contornos deformables no tienen restricciones sobre la parametrización de *s*, permitiendo que diferentes parametrizaciones de *C* lleven a diferentes resultados. En [4], *s* representa el parámetro con la longitud del arco, lo que significa que $\|C'(s)\| = 1$, de la cual obtenemos una nueva formula de energía. En la sección B.2.2 se presenta la nueva formula de energía obtenida a partir de esta modificación.

Dado un potencial $P > 0$ que toma valores mas pequeños cerca de los detalles de interés de la imagen, se buscan caminos a lo largo de los cuales la integral de $\tilde{P} = P + w$ sea mínima (la ecuación de la superficie de mínima acción está detallada en la sección B.2.3). De esta manera la inicialización se reduce solo a la selección de las dos extremidades del camino.

Implementación Numérica

Para encontrar la superficie de mínima acción, se utiliza la búsqueda de grafos y programación dinámica, tomando los pixels de la imagen como nodos del grafo y a la imagen como un grafo dirigido.

Grafo dirigido. Una imagen digital es un arreglo de pixels. En la solución de camino óptimo, el arreglo de pixels es considerado un grafo dirigido. Un costo local es asociado a cada nodo del grafo, y un costo a cada arco. Los costos (ambos, tanto del nodo como del arco) son determinados por la energía que hay que minimizar, llamada función de costo. De esta manera el problema de encontrar el vecino de menor costo entre dos pixels se transforma en encontrar el camino óptimo entre dos nodos del árbol. Se dice que el camino entre dos nodos es óptimo si la suma de los costos de todos los nodos intermedios es (llamado costo acumulado) menor que el costo acumulado de cualquier

otro camino entre los mismos dos puntos. Existe una variedad de algoritmos, basados en programación dinámica que tratan este problema. El algoritmo de Dijkstra es utilizado en este caso.

Algoritmo de Dijkstra. El algoritmo es inicializado al seleccionar el punto de comienzo *s* (también llamado semilla) en el grafo. Una vez obtenido el punto *s*, el costo acumulado de un punto *p* es la suma de los valores de la función de costo del camino óptimo de *s* a *p*. El tamaño de la lista activa es el rango total de posibles costos acumulados, y en la posición i-ésima contiene la lista de pixels cuyo costo acumulado vale i. El costo acumulado vale ∞ en todos lados excepto en el punto inicial *s*, donde vale 0. La lista activa es inicializada al insertar en la primer sublista el pixel inicial.

En cada iteración del algoritmo, el pixel *p* con el menor costo es removido de la lista activa y expandido: el costo acumulado de cada uno de sus vecinos no actualizado es calculado, y la lista activa es reorganizada. La actualización de los costos acumulados de los vecinos es de la siguiente manera: el costo acumulado de un vecino *q* es la suma de el costo acumulado de su padre *p* y el costo del arco de *p* a *q.* Si este nuevo costo es menor que el costo anterior de *q* este nuevo costo pasa a ser su costo, la lista activa es actualizada y la estructura de camino guarda un puntero de *q* a *p*. Como en cada iteración algún pixel obtiene su valor final, y la realiza la búsqueda por el nodo de mínimo costo, la complejidad del algoritmo es $O(N\log_2 N)$ donde N es el numero de pixels de la imagen. Los pixels agregados son marcados. El camino óptimo entre el pixel inicial y cualquier otro punto se obtiene siguiendo hacia atrás los punteros del camino desde el punto final hasta el punto inicial. En el algoritmo 1 se puede ver el pseudocódigo del algoritmo de Dijkstra.

Algorithm 1 Algoritmo de Dijkstra

- Entrada:
  - Función de costo $P(P(p,q)$ = costo del arco (p,q) en el grafo)
  - Pixel semilla s
- Salida:
  - Mapa de caminos M
- Estructuras de datos:
  - Arreglo de costos acumulados CC
  - Lista activa ordenada L
  - Marcados de pixels recorridos E
- Comienzo:

1. poner $CC(s) = 0$ y $CC(n) = \infty$ $\forall$n en el conjunto de vértices del grafo
2. poner a s en la lista L
3. While (!L.vacía)
   a) elegir p, el pixel en L con $CC(p)$ mínimo
   b) setear Ep = true
   c) quitar p de L
   d) For (cada vecino q de p tal que Eq = false)
      1) $u = CC(p) + P(p,q)$
      2) If ($u<CC(q)$) then
         a′ $CC(q) = u$
         b′ actualizar la posición de q en L
         c′ agregar la marca de q a p en M

3.1.6 Fast Marching

Para calcular el mapa de potenciales *U,* se utiliza la ecuación descrita en el apéndice B.2.1. El frente crece desde un circulo infinitesimal alrededor de p_0 hasta que cada punto dentro del dominio de la imagen tiene asignado un valor para *U*. El valor *U*(*p*) es el tiempo *t* en le cual el frente pasa por el punto *p*. Sethian detalla este método en [10] y en [7].

El concepto de Fast Marching trata de introducir orden en la selección de los puntos. Este ordenamiento se basa en el principio de que la información se propaga hacia afuera. El algoritmo de Fast Marching elige en cada iteración un punto de prueba con el menor valor

de potencial posible en esa iteración. Esta técnica de tomar en cada paso solamente el grupo de puntos necesarios fue introducida originalmente para calcular el camino de mínimo costo en un grafo entre dos puntos dados, y solo necesita una pasada sobre la imagen. Para realizar esta tarea de manera eficiente y en tiempo mínimo, los puntos son almacenados en una estructura de min-heap. Como la complejidad de cambiar el valor de un elemento del árbol está limitada en el peor caso en el hecho de recorrer el árbol de abajo hacia arriba, y por lo tanto la operación tiene un costo de $O(log_2 N)$. Entonces, el trabajo total tiene un costo de $O(N log_2 N)$ para el Fast Marching, sobre una grilla de N puntos.

Encontrar el menor camino entre cualquier punto p y el punto inicial p_0 se reduce a realizar back-propagation en el mapa de potenciales mínimos calculado con anterioridad.

Este algoritmo otorga como resultado una imagen que contiene en cada punto el costo de llegar de la semilla al punto indicado. Esta información debe ser post procesada para obtener una segmentación. Por ejemplo, una vez obtenidos los costos de una segmentación que busca el cerebro, se deben buscar los valores que limitan esa región. Al aplicar un filtro de thresholding (ver apéndice), se pueden filtrar los pixels que se encuentren por debajo de un tiempo t (esto indica en que momento el frente de avance paso por allí). Esto es suficiente para obtener la segmentación.

3.1.7 Contornos Activos (Snakes)

Snakes presenta un modelo para el seguimiento de las características relevantes mediante la definición de campos de potencial sobre objetos deformables. Básicamente, se debe realizar el seguimiento de una zona determinada. A esto se añade la aparición de regiones con las mismas características que la zona a seguir, pero que no se encuentran conectadas inicialmente a ella. Evolucionando en el espacio del objeto las nuevas regiones se fusionan entre sí. En este momento se deben recombinar todos los datos obtenidos para poder continuar con el seguimiento de las características deseadas. Este método de representación 3D se basa, por una parte en un modelo de Contorno Activo (Snakes) guiado por un proceso de minimización de energía, y por otra parte se emplean técnicas de etiquetado de regiones para saber cuando inicializar los snakes.

Contornos Activos como Base del Seguimiento

Un contorno activo o snake es un modelo de contorno paramétrico propuesto por Kass, Witkin y Terzopoulos[5] y compuesto por un conjunto de puntos característicos (snaxels v_i) cuya dinámica está regida por la influencia de una energía de deformación o potencial que se pretende minimizar para alcanzar un punto de estabilidad. El snake es un conjunto de snaxels (ver apéndice) que van a ser atraídos por puntos en la imagen. Un punto de la imagen va a atraer más o menos a los snaxels dependiendo de la energía asociada a ese punto, la cual viene definida por las funciones que se exponen en el apéndice. Dicho potencial está compuesto por dos tipos de energía de signo contrario: energía interna y externa. La definición clásica de los términos ponderados que componen la energía global es la siguiente:

Energía Interna. Define tanto la relación espacial entre los snaxels como las restricciones de forma del contorno.

- **Energía de Contorno:** Esta función fuerza a los snaxels v_i a expandirse uniformemente. Ver apéndice.
- **Energía de Curvatura:** Esta energía estima la curvatura en un punto. Su objetivo es mantener el contorno suave localmente. Ver apéndice.
- **Energía de Longitud:** Esta energía interna mantener la longitud o tamaño de la curva cerca del valor esperado para esta. Ver apéndice.

Energía de Imagen. Esta energía, de signo contrario a la anterior y también denominada energía externa, se define en función de las características hacia las que se pretende atraer el snake (contornos, puntos de intensidad elevada, transformada de distancia). Los tres términos de energía deben estar en el mismo rango, en este caso en el intervalo [0,1]. Los parámetros $\alpha,\ \beta,\ \sigma\ \text{y}\ \gamma$sirven para dar más prioridad a una u otra energía. Mediante esta formulación se intenta determinar en que medida otra configuración del contorno hace variar las propiedades del Snake. Minimizando esta energía total se asocia los nuevos puntos con los que tenían unas propiedades parecidas en secciones anteriores. El problema del seguimiento con snakes se formula como una continuada minimización de esta energía a través de los cambios sucesivos del potencial a lo largo de

la secuencia.

Proceso de Reconstrucción 3D

Para el caso de generación de regiones 3D, debe considerarse una nube de puntos[9] a partir de la cual iniciar el crecimiento. Para esto puede utilizarse una esfera o un cilindro. De esta manera se dispone de un conjunto de puntos completamente incluido dentro de la región buscada. Con el conjunto de puntos se comienza la iteración hasta lograr que los valores de energía se estabilicen.

Existen una serie de aspectos que deben ser evaluados para permitir la adecuación de Snakes a estas imágenes con complejidad semántica (ejemplo, resonancias magnéticas de cabeza, con distinción de materia gris y materia blanca). Estos aspectos son cubiertos en detalle en [8], destacándose los siguientes:

1. **Mínimos Locales:** Dado que en las secciones aparecen distintos tipos de materia, se origina un excesivo número de contornos que potencialmente pueden atraer a los snakes situados sobre las zonas de interés. Además el límite de algunas zonas no está bien definido y el contorno resultante es abrupto. Cuando se calcula el término de continuidad de ese contorno aparecen mínimos locales entre los puntos a los que se puede expandir los snaxels a estudio. La presencia de mínimos locales puede hacer que el snake detenga su expansión en dichos puntos de forma incorrecta. Para resolver esta cuestión se puede realizar un filtrado de los puntos vecinos con mínimos locales.
2. **Discontinuidad Espacial:** Cuando realizamos el seguimiento de un contorno y este cambia bruscamente de forma (el contorno que sigue el snake pasa a tener un perímetro mayor que la vecindad a la que este puede expandirse) se provoca una discontinuidad espacial en el eje Z (altura de sección) que dificulta una buena adaptación a los cambios de curvatura. Los efectos negativos derivados de los cambios drásticos en la forma del contorno no se solucionan aumentando el ámbito de la vecindad sobre el que realizamos la búsqueda del contorno, porque al

[9] Lo ideal es utilizar una figura 3D conocida como el origen de los puntos.

relajar dicha búsqueda se facilita la atracción de los snaxels hacia contornos no deseados. Este problema podría solucionarse extrayendo previamente un modelo morfológico que guiase el proceso de seguimiento.

3. **Aparición de Regiones Nuevas:** En determinados tipos de sección es normal la aparición de nuevas regiones impuestas por la forma del objeto a estudio. Por lo tanto se debe conocer las coordenadas Z donde aparecen dichas regiones para efectuar el lanzamiento de los snakes.
4. **Factores de Ponderación:** Los factores de ponderación de términos α, β, σ y γ deben sintonizarse experimentalmente para obtener resultados adecuados.

Fusión de Regiones. El comportamiento del snake es bueno para cada región aislada, es decir cuando no existe interacción entre las distintas regiones de interés. Sin embargo la fusión de las regiones plantea serios problemas. Este tipo de situaciones son cubiertas en [8]. Estos casos plantean un gran aumento de la complejidad del algoritmo.

3.2 RECONSTRUCCIÓN DE SUPERFICIES

En esta sección se presentan los métodos de Marching Cubes y Tetrahedra para la reconstrucción de superficies. Si bien su objetivo principal es la reconstrucción tridimensional, funcionan también como métodos de segmentación donde el criterio de segmentación es la intensidad de los voxels, que generan las isosuperficies buscadas.

3.2.1. Algoritmos basados en Marching

Los algoritmos de Marching son métodos simples y bastante populares para extraer isosuperficies de funciones implícitas o información tridimensional discreta. Estos se basan en subdividir una superficie o volumen en formas más elementales y detectar uno o más contornos, ubicando el color, valor, o intensidad en los vértices de la figura elegida. De esta manera, por medio de interpolación es posible detectar cuales aristas de la figura son intersecadas por la superficie y reconstruir (a partir de los puntos de intersección), la superficie con figuras mas elementales, como ser triángulos. A continuación se presentan dos algoritmos de Marching que utilizan esta técnica.

El la Figura 3.2 se presenta un ejemplo de una imagen, en esta se resaltan dos isosuperficies.

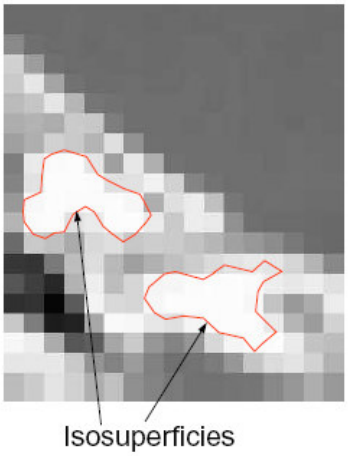


Figura 3. 2 Vista 2D de una isosuperfice.

3.2.2 Marching Cubes

Marching Cubes es un algoritmo para extraer Isosuperficies a partir de datos volumétricos. La idea fundamental es que se puede definir un cubo (voxel) que contenga en sus 8 vértices la información de la intensidad que corresponde a ese punto. Si uno o más vértices tienen valores mayores que el de la superficie buscada y uno o mas tienen valores menores, entonces se sabe que el cubo es cortado por la isosuperfice que se está buscando. Al determinar los lugares del cubo por donde pasa la superficie se pueden generar triángulos que unan los puntos de intersección. Al unir todos los triángulos generados se obtiene la superficie buscada.

Este algoritmo tiene dos componentes principales. La primera es decidir como definir la sección o secciones de la superficie que cortan un cubo. Si se clasifican los vértices como dentro o fuera de la superficie, existen 256 posibles combinaciones. Dos de estos casos son triviales: todos los puntos dentro y todos los puntos fuera. Para el resto de las configuraciones hace falta determinar, para cada arista del cubo, si corta la superficie.

Luego, en una segunda pasada, por medio de interpolación se calcula el punto exacto por donde la superficie corta las aristas. Una vez que se tienen los puntos exactos por donde la superficie corta al cubo, solo hace falta unir estos puntos para formar los triángulos. Descartando casos simétricos solo quedan 14 de los 254 casos restantes. Cuando un solo vértice es menor que el valor buscado, se forma un solo triángulo que corta a este cubo. Existen 8 casos simétricos a este. Lo mismo sucede cambiando la normal del triángulo, haciendo el caso opuesto; en este caso un vértice es mayor y los otros 7 menores que el color buscado. El resto de los casos del algoritmo se detallan en [6]. En la Figura 3.3 se muestran los principales casos de Marching Cubes.

Todos los demás casos no triviales pueden ser resueltos agregando entre 1 y 4 triángulos a la Isosuperfice. Los vértices de cada triángulo pueden ser calculados por medio de interpolación lineal entre los vértices de los voxels, o simplemente tomado el punto medio de las aristas como vértice. Si se decide utilizar el primer caso las superficie obtenida tendrá será mas suave y regular.

3.2.3. Marching tetrahedra

El concepto de marching tetrahedra es el mismo que se aplica en Marching Cubes. La principal diferencia es que marching tetrahedra utiliza como figura elemental de corte un tetraedro y no un cubo. Esa consiste en dividir los cubos en una serie de tetraedros. Un tetraedro solo tiene 16 triangulaciones posibles, las que se reducen a 3 por simetría. El la Figura 3.4 se pueden ver los 2 casos básicos de triangulación.

Originalmente se construyo la malla de tetraedros dividiendo el cubo (voxel) en 5 tetraedros. Desafortunadamente esta división introduce ambigüedades debido a que la simetría las subdivisiones de los cubos debe alternarse entre cubos para alinear las caras de los tetraedros. Entonces existen dos posibles divisiones la malla de cubos. Esta ambigüedad solo pude ser resuelta usando interpolación cúbica en lugar de lineal. Sin embargo este enfoque es mucho más complejo y resulta en una Lookup Table de 59 casos, muchos más que en el método original de Marching Cubes.

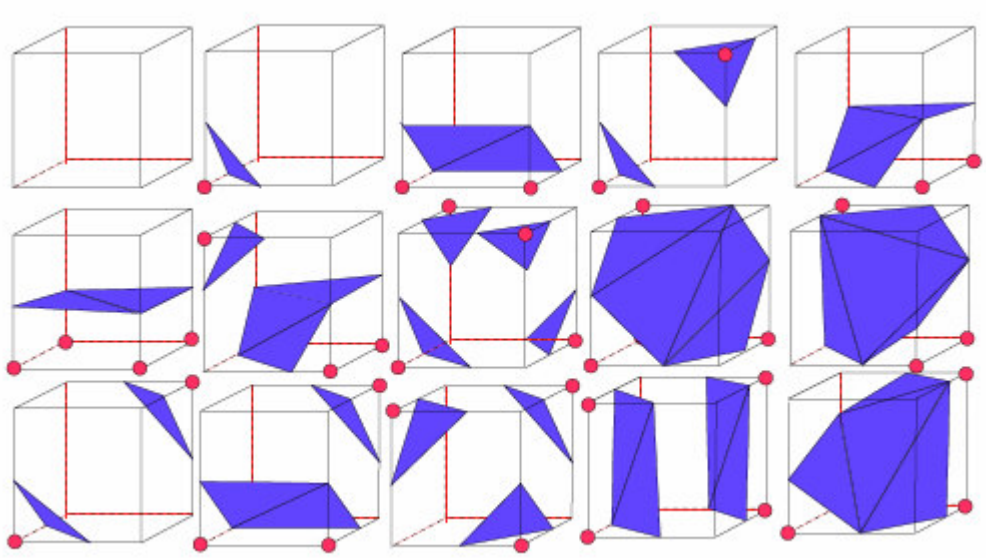


Figura 3. 3 Casos de triangulación de Marching Cubes

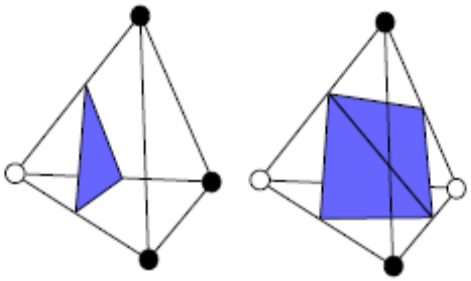


Figura 3. 4 Casos de triangulación de Marching Tetrahedra

En [3] se describe una nueva división de la malla. Ésta se basa en dividir los cubos en tetraedros centrados. Los tetraedros de esta nueva división se encuentran centrados respecto del cubo. Al contrario que el método descrito con anterioridad en el que se subdivide el cubo, en este caso los tetraedros siempre tienen la misma forma, lo cual elimina las ambigüedades de la malla de triángulos resultante.

La principal desventaja de las técnicas basadas en tetraedros es que crean una cantidad de triángulos aun mayor que Marching Cubes para unos datos de entrada determinados. Esto agrava el segundo problema de Marching Cubes, es decir la regularidad de la malla de la superficie resultante.

Existe una gran cantidad de trabajo para tratar de mejorar la triangulación que resulta de MC y MT. El objetivo de este post-procesamiento es mejorar los tiempos de renderizado y almacenamiento. Sin embargo triángulos de baja calidad[10], llevan a renderizaciones de mala calidad.

Una buena técnica para la optimización del resultado de MT fue introducida por Treece, Prager y Gee en [12]. En este paper se introduce el llamado Regularised Marching Tetrahedra. RMT utiliza una serie de técnicas (basadas en pre y post processing) para lograr una triangulación mas regular y simplificada.

3.3 POST-PROCESADO DE SUPERFICIES

3.3.1. Suavizado de superficies - Algoritmo de Taubin

Una vez realizada la segmentación y aplicado un filtro de reconstrucción de superficie, como por ejemplo Flat Contour (sección 4.2.2), se obtiene una representación inicial de la superficie que limita con la región buscada. En este caso la superficie sigue el contorno de los voxels incluidos por la segmentación. Los triángulos de la malla se encuentran dispuestos de manera ortogonal entre sí. En una malla de estas características es difícil detectar las formas de las diferentes estructuras o tejidos reconstruidos. Una de las alternativas para mejorar la calidad de la superficie es aplicar un filtro de suavizado. Este filtro debe trabajar sobre los puntos de la malla, para relajarla y permitir que desaparezcan las irregularidades. Estas irregularidades pueden verse como ruido (señales de alta frecuencia).

Sin embargo es muy importante que la malla conserve la topología, ya que si esta se deforma se perderán detalles. Teniendo en cuenta el origen de esta malla (una imagen médica), estos detalles representan partes de la anatomía y no pueden perderse.

[10] Malos por su tamaño y/o forma.

Aproximación por procesamiento de señal

El análisis de Fourier es una herramienta natural cuando se trata de resolver el problema de suavizado de señales. Las diferentes señales[11] son descompuestas en subespacios ortogonales asociados con diferentes frecuencias, con el contenido de baja frecuencia de la se˜nal mirado como información subyacente, y el contenido de alta frecuencia como ruido. El efecto es relajar la malla, mejorando la forma de las celdas y distribuyendo los vértices de forma mas homogénea. Este filtro trabaja sobre líneas y polígonos.

El algoritmo trabaja de la siguiente manera. Para cada vértice *v*, se realiza un análisis geométrico y topológico para determinar que vértices están conectados con *v*, y que celdas están conectadas con *v*. Después se crea un arreglo de conectividad para cada vértice. [12]Luego comienza una iteración sobre todos los vértices. Para cada vértice *v*, las coordinadas de *v* son modificadas usando una función de interpolación llamada Windowed Sinc. El algoritmo y los fundamentos matemáticos son descritos en [11]. Básicamente, hay 2 variables de configuración del filtro: iteraciones y pass- band. Se debe tener cuidado particular para no suavizar con muy pocas iteraciones. La segunda variable es la especificación de PassBand para el filtro Windowed Sinc. Por cuestiones de diseño, el PassBand es especificado por un valor de punto flotante entre 0 y 2. Valores de PassBand más bajos resultan en mayor suavizado. Un buen valor por defecto para el PassBand es 0.1.

[11] Funciones definidas en un dominio determinado

[12] El arreglo de conectividad es una lista de listas de vértices que se agrega directamente a cada vértice.

4. MÉTODOS DESARROLLADOS

En este capítulo se describen las implementaciones y distintas alternativas originales para la realización del trabajo. Se discuten el desarrollo de técnicas para segmentación, reconstrucción tridimensional, post-procesamiento de superficies y para visualización de imágenes médicas.

4.1. SEGMENTACIÓN

4.1.1. Voxel Grow

En el método de Voxel Grow el objetivo es buscar las regiones uniformes directamente sobre la imagen basándose en características locales. Este método puede actuar a partir de una o varias semillas, las cuales son elegidas por el usuario. El concepto que utiliza este método es que una misma región de la imagen es homogénea en las intensidades de color de los voxels que contiene. Así, a partir de una (o varias) semilla(s) se hace crecer la región que se tomará como interior. Este método minimiza la interacción con el usuario, el que solo debe señalar semillas dentro de las regiones que desea segmentar y determinar un delta.

Crecimiento de regiones

El crecimiento de las regiones se lleva a cabo a partir de las semillas que fueron elegidas por el usuario. Las semillas contienen la siguiente información:

- **Posición:** son las coordenadas x, y, z del punto que el usuario a seleccionado. Sabemos que este punto se encuentra dentro de la región que nos interesa reconstruir.
- **Color:** el color o intensidad de este voxel es importante para determinar el rango de colores que estarán incluidos en la región. A partir de este color se llevará a cabo la exploración del resto del volumen.

Estos datos deben ser interpretados correctamente para sacar mejor provecho de ellos. El usuario tiene acceso a la imagen, desde los diferentes planos. En estos puede detectar las regiones que le son de interés y seleccionar las semillas que desee dentro de estas.

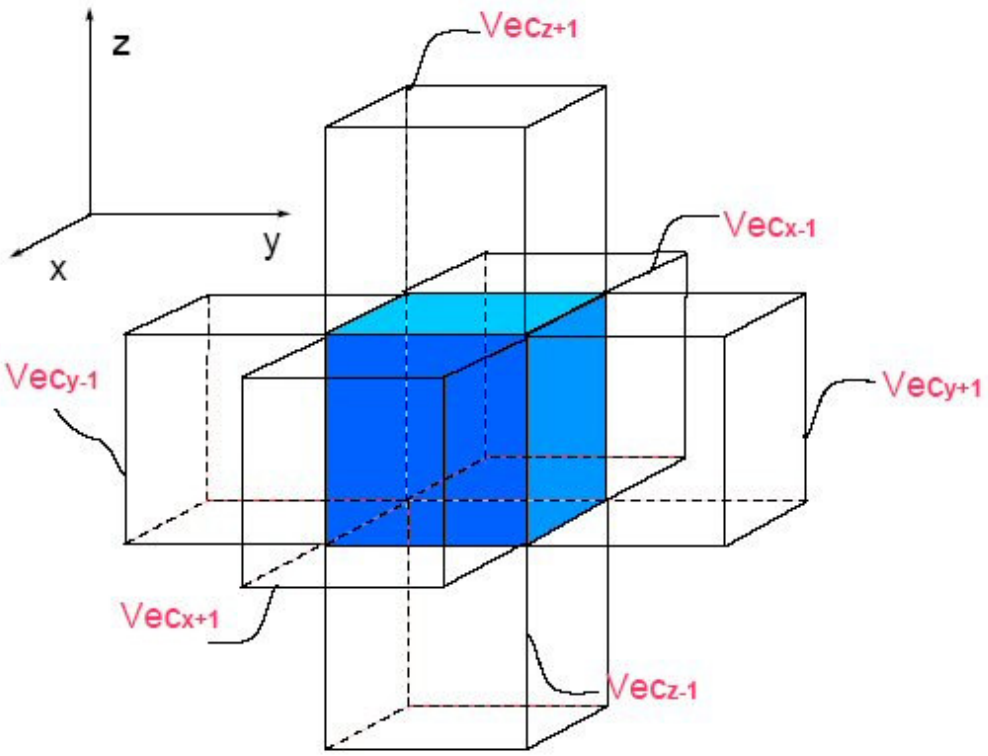


Figura 4. 1 Conectividad 6 de un cubo.

Al igual que en el caso de Marching Cubes, cada uno de estos valores esta representado por un voxel. Un voxel tiene 6 vecinos (>x, <x, >y, <y, >z, <z , ver Figura 4.1), para cada uno de estos se debe verificar si pertenece o no a la región.

Para determinar si un voxel está dentro o fuera de la región se utiliza una técnica muy simple. Ésta consiste en comparar la variación de color entre el voxel y un delta ingresado por el usuario.

Algoritmo de crecimiento

Para calcular la región se utiliza una serie de estructuras auxiliares. En principio se dispone del volumen de datos, la imagen organizada de manera de matriz cúbica. La segunda estructura es una matriz de iguales dimensiones que la imagen. Esta inicialmente esta llena en 0 en todas las posiciones. Aquí se marcará la semilla para luego hacer crecer la región. Por ultimo de utilizará una estructura de fila (FIFO - First In First Out), en ésta se ubica inicialmente la semilla y luego los vecinos que pertenecen a la región para ser visitados. En el algoritmo 2 se puede ver detallado el pseudocódigo del algoritmo de crecimiento por voxel. El algoritmo sucesivamente retira elementos de la cola. Estos elementos son los voxels del volumen que ya han sido visitados. Para cada uno de ellos debemos visitar sus vecinos y se debe decidir si este pertenece o no a la región buscada. Para comparar vecinos se utiliza conectividad de a 6. En la figura se detalla la conectividad 6, es esta se recorren los vecinos en X, en Y y en Z.

Uno de los aspectos más destacables de esta técnica es que al crecer siempre por vecinos, mantiene conectividad entre los elementos que incluye dentro de la región segmentada.

Algorithm 2 Algoritmo de crecimiento por voxel (Voxel Grow)

1. Inicializar el volumen auxiliar AuxV en 0
2. Marcar la semilla en AuxV
3. Agregar la semilla a la cola C
4. While (! C.vacía())
   - *a*) Punto p = C.primer_elemento()
   - *b*) For (cada vecino v de p)
     - 1) If (pertenece(v))
       - *a'* marcar v en AuxV
       - *b'* agregar v a C
     - 2) else
       - *a'* marcar que p no pertenece y fue visitado

Los vecinos del voxel que no han sido visitados son marcados según pertenezcan o no al volumen. Aquellos que pertenecen al volumen además de ser marcados son encolados para visitar vecinos suyos en una siguiente pasada. De esta manera, cuando la cola se vacía, ya no hay más elementos que considerar.

Tipos de crecimiento

Existen tres criterios diferentes para considerar que un voxel pertenece o no a la región. Uno de ellos es considerar la variación de color de un voxel respecto del color de la semilla, el otro es considerar localmente la variación de color respecto del vecino que estoy visitando actualmente, y el último tiene en cuenta el gradiente tridimensional de la imagen. Estos criterios se consideran a continuación.

Respecto de la semilla

En este caso se toma siempre como referencia el color de la semilla. Cada nuevo voxel que se agrega a la región es incluido si la diferencia de color que existe entre el y el color de la semilla se mantiene dentro de un umbral determinado con anterioridad por el usuario. Este umbral es el Δ. Este se compara directamente con la diferencia de intensidad.

Esta técnica da como resultado regiones que contienen voxels cuyos valores se encuentran dentro de un rango determinado. La función pertenece(Voxel *v*) verifica que

$$|intensidad(v) - intensidad(semilla)| < \Delta$$

Respecto del vecino

A diferencia del caso anterior, aquí se considera que un punto pertenece a la región si su diferencia con su vecino permanece debajo del umbral Δ. El color del voxel que esta siendo considerado en este momento es comparado con el de su vecino (el voxel que sacamos de la cola en esta iteración). En esta técnica, quedan excluidos aquellos voxels que tienen grandes variaciones de color con vecinos. La comparación realizada es

$$|intensidad(v) - intensidad(vecino)| < \Delta$$

Esta técnica es buena para detectar regiones bien delimitadas, donde los bordes se encuentran bien definidos.

Respecto del gradiente

De las tres técnicas esta es tal vez la más precisa, ya que utiliza el gradiente de la imagen para determinar los límites de una región. El gradiente de una imagen determina que tan rápido se produce la variación de color en una imagen. Este gradiente esta calculado para cada voxel con sus 6 vecinos, o sea que el gradiente de color en un punto esta dado por la variación de colores de un voxel con sus 6 vecinos. En este caso se calcula el gradiente para toda la imagen (esto da como resultado una Imagen Gradiente, esta contiene en cada punto la magnitud del gradiente en ese punto en la imagen original) y luego se realiza la propagación a partir de la semilla, pero dentro de la imagen gradiente. En este caso el umbral introducido por el usuario determina hasta que valor de gradiente se debe avanzar. Por ejemplo: si el usuario introduce un gradiente de 3.5, el crecimiento se realizará hasta encontrar vecinos que tengan intensidades que superen este valor. El gradiente de la imagen es mayor cuando la variación de color es mayor, es decir, las zonas de alto gradiente determinan límites entre regiones. En este caso se utiliza el gradiente para “frenar” el avance de la región. La comparación que se realiza en este caso es

$$intensidad(imagenGradiente(v)) < \Delta$$

Fusión de regiones

La fusión de regiones es un gran problema en todos los métodos de segmentación que se basan en el crecimiento a partir de una semilla. En el caso de tener que rodear algún objeto o si se tienen dos semillas, las cuales deban luego componerse en una sola al crecer y encontrarse, pueden generarse inconsistencias fácilmente. Cuando dos frentes se encuentran, deben combinarse. Los puntos que se encuentran en ambos frentes deben combinarse y modificar la topología de la región. En el caso de Voxel Grow, se trabaja sobre los valores discretos de la imagen, y no existe superficie aun. Solamente

existe el volumen y dentro de el un conjunto de puntos que están seleccionados. En este punto solo se conocen los voxels que pertenecen a la región. Los problemas anteriormente mencionados no se presentan ya que la región crece naturalmente, rodeando obstáculos sin tener que combinar puntos.

Evolución

En la segmentación de imágenes se trata de reconocer regiones homogéneas dentro de imágenes de manera automática. Se intenta dar más semántica a las imágenes por medio de su interpretación. En una imagen plana la unidad mas pequeña es el pixel, este no nos proporciona mucha información de forma y tamaño del tejido u órgano que representa. Sin embargo al agrupar pixels de acuerdo a ciertas características (tanto grupales como individuales), se pueden determinar regiones dentro de la imagen. Estas regiones pueden representar cosas (un auto, una persona, un tejido, etc.). Voxel Grow intenta explotar estas similitudes y características comunes entre pixels que pertenecen a una misma región para delimitarla.

Los tipos de crecimiento funcionan bastante bien en general, pero fallan con algunos casos especiales. Por ejemplo: cuando existe una zona que gradualmente varía de color, llegando a intensidades altas (o bajas) de manera muy gradual. En este caso el gradiente es bajo entre vecinos, permitiendo que el frente avance y se “filtre” hacia zonas que no pertenecen a la región. También se producen casos dentro de las imágenes médicas en donde pequeños orificios pueden permitir que la región se filtre hacia lugares que no pertenecen a la región buscada. Para resolver este tipo de inconvenientes se plantaron algunas alternativas al algoritmo de crecimiento que se explican a continuación.

Crecimiento con Esfera

Frecuentemente sucede en las segmentaciones que pequeños orificios permiten que el frente de avance se “escape” de la zona buscada y se incorporen partes de la imagen que en realidad no son de interés. Esto resulta en segmentaciones de mala calidad que no representan elementos u objetos reales de la imagen. Por la manera en que se realiza el crecimiento es difícil detectar situaciones en que el frente se escapa por zonas que no se

encuentran bien limitadas. El crecimiento se realiza preguntando por los vecinos inmediatos de un voxel. Es decir que la frente de propagación tiene un voxel de ancho. En otras palabras, si existe algún "camino" por el cual pase un voxel y este camino cumpla con la condición de crecimiento con radio 1[13], el frente avanza. Este camino puede encontrarse a través de algún tejido intermedio, segmentando porciones de la imagen que no pertenecen a la zona de interés.

Para solucionar este inconveniente se planteó la siguiente alternativa. En vez de crecer por medio de un voxel se crece por medio de una esfera. La esfera puede tener un radio variable, de manera que se pueda tomar una esfera mayor para ser más restrictivo y ajustarla de acuerdo a la región que se desee segmentar. En algunos casos puede desearse segmentar una zona que por sus características es muy delgada, por ejemplo un hueso. En este caso se desearía utilizar un radio menor, para permitir que el frente se desplace por lugares delgados o que poseen regiones muy delgadas. Sin embargo para regiones más grandes como el cerebro, se obtienen mejores resultados utilizando radios mayores. Esto permite que el cerebro sea reconocido enteramente evitando que la región se propague hacia lugares que no corresponden a este. La esfera se detiene al no poder avanzar completa dentro de la nueva zona.

[13] Esto significa que la condición se cumple a distancia 1 (uno) del vecino. En este caso es muy fuerte la localidad, no se tiene en cuenta lo que sucede a mas de 1 voxel de distancia.

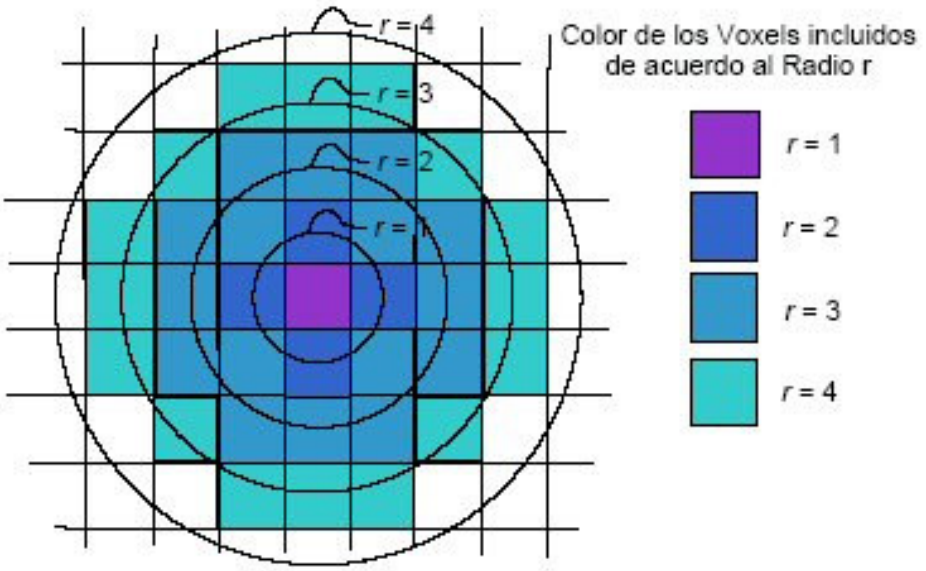


Figura 4. 2 **Crecimiento con esfera**

Implementación con VTK[14]

Para implementar esta alternativa se utilizó una clase de VTK que permite determinar que voxels se encuentran dentro de una esfera de radio y centro determinados. Este función implícita (vtkImplicitFunction) determina que voxels se encuentran dentro de la región buscada, determinada por la función utilizada. En este caso se utiliza la función de la esfera. Los voxels que serán incluidos dentro de la esfera, son aquellos que se encuentren enteramente dentro de la esfera de radio *r*.

En la figura 4.2 se puede ver un ejemplo de los voxels que quedan incluidos de acuerdo a la variación del radio de la esfera. El la imagen se presenta un ejemplo en 2D para simplificar su legibilidad.

[14] VTK (Visualization Tool Kit) Herramienta de apoyo a la programación para visualización científica. Ver http://www.vtk.org

Crecimiento por diferencia

El algoritmo de crecimiento por medio de la esfera varía levemente respecto del original. Su funcionamiento se muestra en la Figura 4.3. En este caso se considera que se agregarán nuevos voxels a la región siempre y cuando todos los voxels dentro de un radio determinado también pertenezcan a la región. De esta manera se tiene la seguridad de que no se agregarán elementos que no conserven una cierta homogeneidad con la región que los rodea. Al iniciar esto desde un punto determinado, se puede determinar que todos los voxels que serán incluidos cumplen con la restricción aplicada al la semilla. Al introducir el concepto de radio, se puede establecer por que lugares el frente puede o no avanzar.

Este crecimiento se llama “por diferencia” porque en cada iteración se incluyen al resultado todos los nuevos voxels que no estaban incluidos en la vuelta anterior. Cada vez que se mueve el centro de la esfera, nuevos voxels son consultados para saber si pertenecen o no a la región buscada. Estos nuevos voxels serán los agregados en el resultado si es que cumplen con la condición de inclusión.

En la Figura 4.3 puede verse cuáles son los voxels que son incorporados a la región.

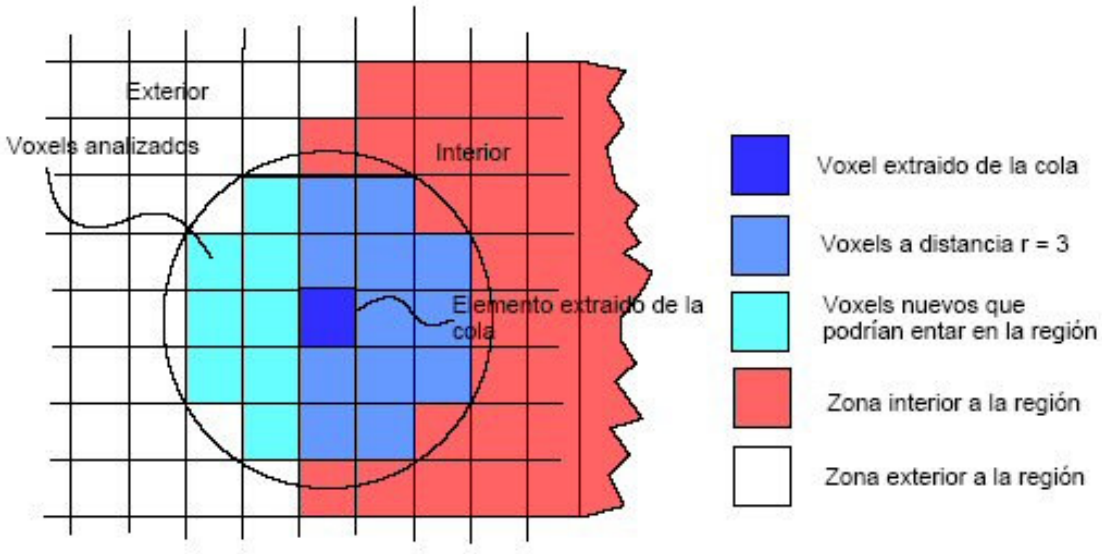


Figura 4. 3 Detalle crecimiento por diferencia.

Crecimiento por vecino

Este es un enfoque más restrictivo del crecimiento. Es este enfoque la consultas son iguales que en el ejemplo expuesto anteriormente, pero existe una sutil diferencia respecto del criterio que existe para incorporar nuevos voxels al resultado. Cada vez que retira un voxel de la cola (en este punto el algoritmo debe verificar si agrega los vecinos de ese voxel al resultado), verifica que todos los voxels que estén a radio *r* cumplan con la condición de inclusión.

Algorithm 3 Voxel Grow con esfera por vecino

1. Elegir la(s) semilla(s).
2. Incorporar a la región la(s) semilla(s), y encolar en C.
3. While (!C.vacía())
   - *a*) Recuperar el primer elemento de la cola.
   - *b*) For (vecinos v de conectividad 6)
     1) Calcular los vecinos v_i de v con radio r
     2) If (todos los v_i pertenecen a la región)
        - a' agregar v al resultado
        - b' agregar v a la C

En este caso la incorporación de nuevos voxels a la región es mas restrictiva, se incorpora un voxel si todos los voxels que se encuentran a distancia *r* de él pertenecen a la región. Esta técnica ofrece la ventaja de detenerse a una distancia *r* del límite que se esta buscando. Esto permite crecer los últimos pasos utilizando otra técnica sabiendo que el límite aún no se ha alcanzado.

En el Algoritmo 3 se puede apreciar que los elementos de la cola son retirados y los vecinos (con conectividad 6) de estos son consultados. El elemento que se encuentra en la cola, ya ha sido incluido en la región con anterioridad, pero el vecino puede estar aun en el exterior. Para que este vecino quede incluido, todos los voxels a distancia *r* deben cumplir con la condición de pertenencia.

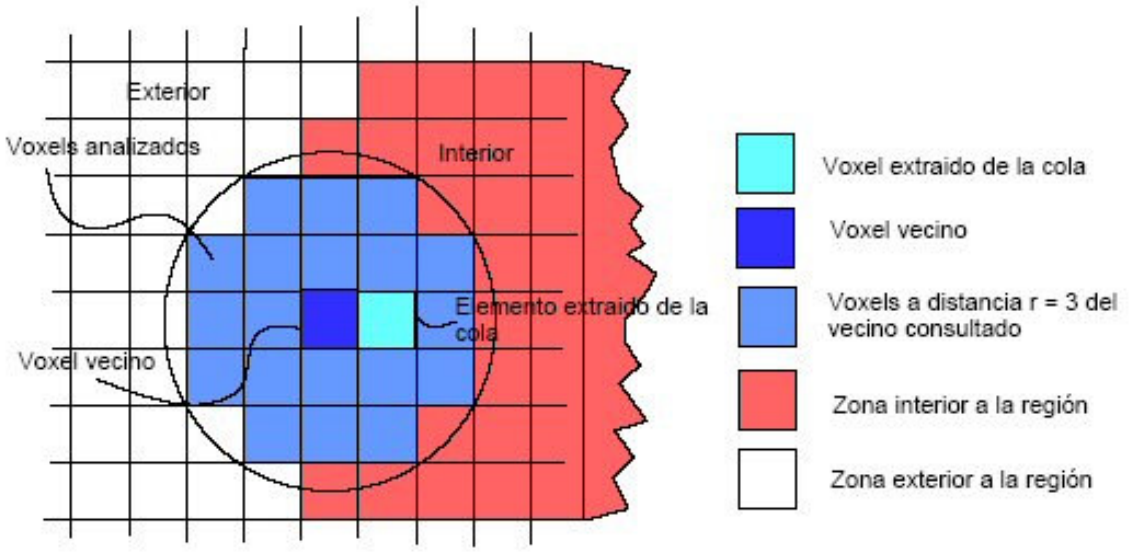


Figura 4. 4 Crecimiento por vecino

De esta manera nos aseguramos que estamos al menos a distancia *r* del límite de la región.

Los algoritmos que utilizan una esfera pare recorrer el volumen tiene un inconveniente, es temporalmente muy costoso calcular qué voxels están dentro de la esfera de radio *r.* Por eso se planteó una alternativa menos costosa en tiempo y más simple. Para resolver este problema se implemento el algoritmo utilizando un cubo de tamaño variable.

Adaptación con un Cubo

Como se menciono anteriormente, calcular que voxels se encuentran dentro de un radio *r* determinado tiene costos muy altos en tiempo. Por esta razón se decidió plantear una alternativa para este algoritmo. La alternativa es utilizar un cubo en lugar de una esfera. La idea es mover las “paredes” del cubo y preguntar sucesivamente por los elementos que conforman cada una de sus (seis) caras.

Esta modificación del algoritmo anterior pregunta por los vecinos de una de las caras del cubo. Si todos los elementos vecinos de una de las paredes cumplen con la condición de

crecimiento, entonces todos esos voxels son agregados al resultado. En esta punto pueden plantearse dos alternativas, al igual que en el caso de la esfera. Puede incorporarse la cara completa a la cola (para que luego se pregunte por estos voxel en una iteración siguiente) o simplemente el voxel vecino al central (crecimiento mas restrictivo, similar al caso de crecimiento por vecino planteado para la esfera).

En la figura 4.5 se muestra los voxels incluidos dentro del cubo de radio $r = 1$[15]. Para realizar el crecimiento, se prueban los voxels vecinos a una cara completa. Estos se encuentran resaltados en la imagen.

Radio de estiramiento

El control que estos métodos otorgan sobre el crecimiento permite detener el frente de avance a una distancia arbitraria del borde de la región. Teniendo control de esto, puede utilizarse otra técnica para avanzar los últimos pasos. Por ejemplo, si se avanzó una primera pasada con radio 3 y creciendo por vecino, al terminar la pasada la región seleccionada se encuentra a 3 voxels de distancia del límite buscado. En este punto se puede aplicar una pasada que (con el mismo criterio de inclusión), crezca de a un voxel, pero no mas de 3 voxels de distancia. Así el frente no podrá "escaparse" por lugares peque˜nos, pero si podrá llegar hasta los límites de la región. Ver imagen.

[15] En el crecimiento por Cubo, se interpreta de manera diferente el radio. En este caso se toma *r* a partir del voxel centro. Por ejemplo, para un radio de 2, el frente del cubo tendrá 5x5 voxels de tamaño (el voxel central, mas 2 voxels de cada lado).

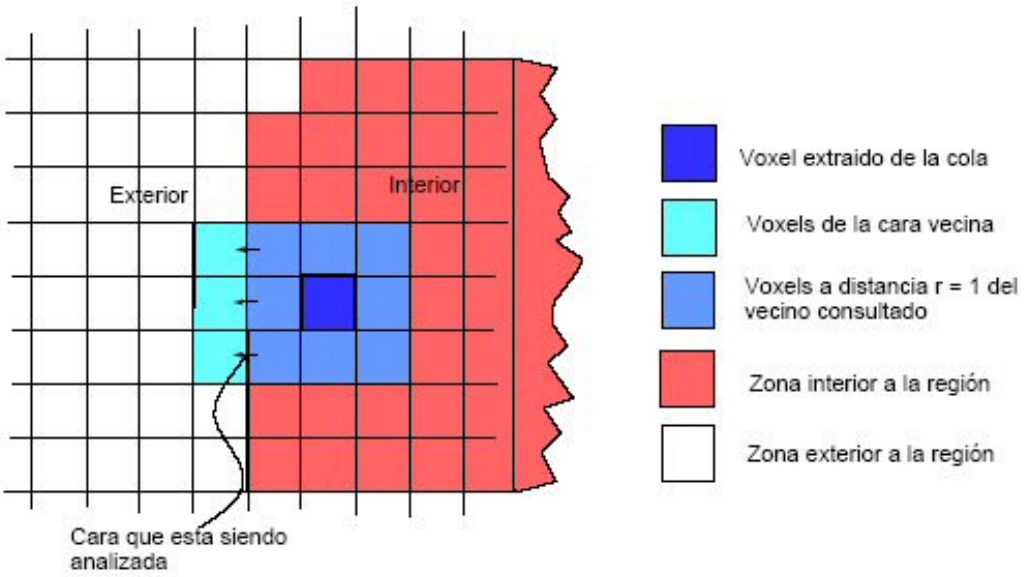


Figura 4. 5 Crecimiento por Cubo.

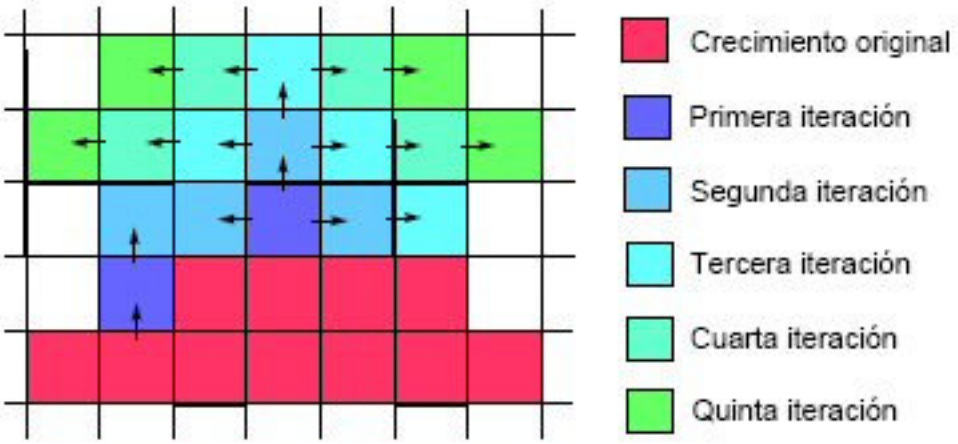


Figura 4. 6 Crecimiento por inflado de Voxel Grow.

Ventajas

Realizar crecimientos dentro de las regiones utilizando figuras básicas como pueden ser una esfera o un cubo tiene una serie de ventajas sobre los crecimientos básicos basados en un simple voxel. El crecimiento en estas condiciones es más homogéneo, ya que se tiene en cuenta valores a mayor distancia y no solo los vecinos inmediatos. Los cambios

de región en las imágenes pueden ser graduales, no siempre se encuentran tan bien limitadas. Para evitar pasar por alto límites de región que no estén muy marcados y evitar que el frente de avance escape, se toman en cuenta valores que se encuentran a mayor distancia, exigiendo que todos los elementos cumplan con la restricción de crecimiento.

Estos enfoques además permiten poner cota a los lugares por los cuales nuestro frente de avance pueda escapar. Al configurar la propagación con radios mayores se restringe a que el frente no avance por orificios de peque˜no diámetro. Esto mejora ampliamente los resultados obtenidos, ya que en imágenes médicas con mucha frecuencia se presentan situaciones en que diferentes regiones se encuentran conectadas por zonas pequeñas o difusas.

Otro aspecto importante es permitir al usuario seleccionar más de un punto de inicio para el crecimiento. Los crecimientos anteriormente descritos permiten crecer desde mas de un punto, solo se deben agregar las diferentes semillas a la cola para que estas serán evaluadas. Otro punto interesante de destacar es que pueden designarse distintos criterios de inclusión para las diferentes semillas. El usuario podría desear que diferentes semillas crezcan de diferente manera según la localización de las mismas, esto puede ser por razones arbitrarias o porque las características de la imagen varían en distintos puntos.

4.2. RECONSTRUCCIÓN DE SUPERFICIES

Una vez realizada la segmentación se debe reconstruir la superficie que la envuelve. Para realizar esta reconstrucción se pueden utilizar diferentes técnicas como por ejemplo Marching Cubes (3.2.2).

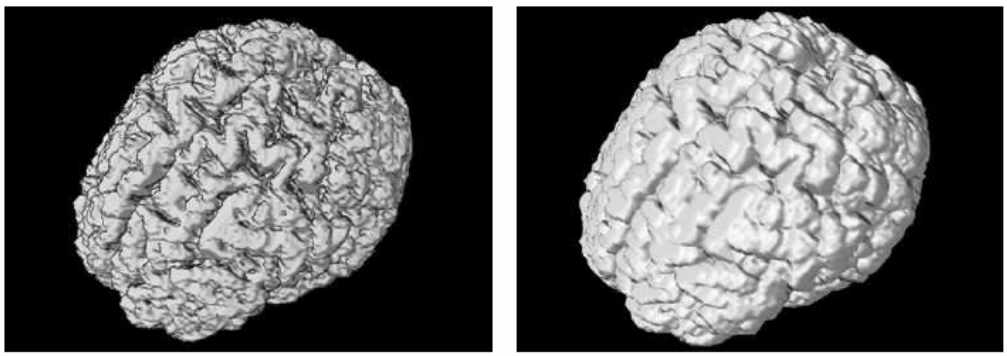
4.2.1 Utilización de Marching Cubes

Las técnicas de segmentación presentadas permiten recuperar de una imagen las diferentes regiones (en el caso de imágenes médicas representan tejidos o estructuras del cuerpo) de acuerdo a su homogeneidad u otras características. Pero toda esta información se encuentra en forma de imagen. Una vez que se tiene la imagen segmentada (representada en forma binaria con valores dentro/fuera) se debe extraer de esta imagen la representación poligonal de la región para luego ser renderizada. El

objetivo de este trabajo no solo es segmentar, sino también proveer de una reconstrucción tridimensional fiel y de buena calidad que permita reconocer estructuras complejas del organismo como órganos internos, diferentes tejidos y lesiones.
Una vez que se obtiene una segmentación a partir de una imagen (la segmentación es una imagen binaria de tres dimensiones), a partir de esta se genera una superficie tridimensional que la representa. Una de las técnicas mas conocidas para generar superficies a partir de imágenes tridimensionales es Marching Cubes. MC construye una Isosuperfice a partir de imágenes tridimensionales. Utilizando interpolación lineal detecta en que puntos una superficie corta un campo escalar, dando como resultado una malla de triángulos. Esta representa la superficie que limita dentro del volumen de datos original con la región buscada. El funcionamiento de este método se encuentra detallado en la sección 3.2.2.

Problemas

Marching cubes es una de las técnicas de reconstrucción de superficies más conocidas. Al ser uno de los algoritmos más usado, esta probado tanto en su desempeño como en la calidad de los resultados. Como se mencionó anteriormente, produce mallas de muy buena definición y con muy buenos tiempos de ejecución. Sin embargo la triangulación que produce no es de muy buena calidad. MC produce mallas con topologías muy complejas, a costo de tener mucha cantidad de triángulos. Este es un precio muy alto a pagar ya que la cantidad de triángulos repercute directamente en los tiempos de renderización.



(a) Reconstruccion por MC sin suavizado previo (b) Reconstruccion por MC con un suavizado gausiano previo

Figura 4. 7 **Ejemplo de reconstrucción por Marching Cubes**

En renderización de superficie es importante que la malla este formada por triángulos regulares. Cuando los triángulos que la forman son demasiado des- parejos en su superficie, forma, tamaño de sus ángulos, etc., puede resultar en inconsistencias entre la topología original de la superficie y la percepción que el usuario tiene de esta. MC suele producir triangulaciones que tienen gran cantidad de triángulos muy peque˜nos o con ángulos muy agudos. En la imagen renderizada, los triángulos con ángulos muy agudos y poca superficie pueden aparecer como bordes agudos en la superficie. Triángulos cuya superficie es muy pequeña aparecen como puntas agudas. Al aplicar MC directamente sobre el resultado de la segmentación, se obtienen resultados de baja calidad debido a que este algoritmo busca (por medio de interpolación) un valor exacto. Como la imagen tiene solo dos valores de intensidad, el límite entre una región y la otra es muy abrupto. Esto provoca que la superficie resultante quede con un enmallado irregular y angulado, resultando en imágenes de baja calidad. Para resolver este problema se decidió aplicar un filtro Gaussiano al resultado de la segmentación. Al aplicar de este filtro se suavizan los bordes de la región segmentada. Si se utiliza MC una vez aplicado el filtro sobre la imagen, los resultados mejoran considerablemente. La superficie es mucho mas suave y se ajusta mucho mejor a la región real (Figura 4.7).

La alternativa a MC para reconstruir la superficie es reconstruir las caras de los voxels para luego procesarla y obtener una superficie más suave.

3.2.2. Flat Contour

Flat Contour es una técnica de reconstrucción de superficie desarrollada especialmente para ser aplicada sobre una segmentación. Esta se basa en encontrar los límites entre la zona segmentada (o interior) y la zona exterior. Este algoritmo recorre el volumen de datos (resultado de la segmentación) buscando los limites de esta. Cada vez que encuentra un límite genera las caras de los voxels que la limitan.

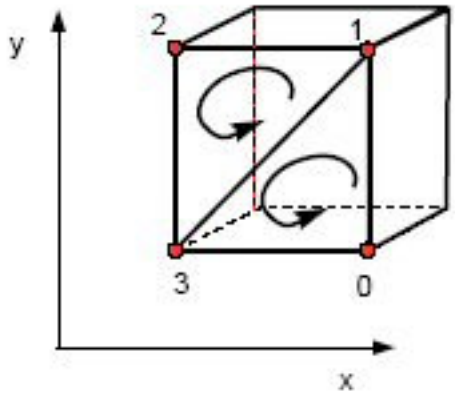


Figura 4. 8 Ejemplo de generación de una cara para un voxel.

Los límites pueden ser de dos tipos:

- Interior a exterior
- Exterior a interior

La única diferencia entre estos dos casos es la dirección que debe tener la cara generada. Las caras de los triángulos generados siempre deben encontrarse en dirección hacia la zona exterior. Como el recorrido de la superficie es secuencial, el tiempo de ejecución es constante, dependiendo directamente del tamaño de la imagen y no de la cantidad de triángulos. Este algoritmo genera dos triángulos por cada cara de voxel. En la Figura 4.8 se muestra la numeración de los vértices de la cara del cubo y la forma en que son unidos para formar cada uno de los triángulos. El orden en que son agregados los puntos no es arbitrario, ya que de esta orden depende de la dirección en que apunta la cara del triángulo, es decir la dirección que tendrá la normal de la superficie del triángulo.

Es fácil notar que en esta superficie se encontraran muchas veces puntos que están repetidos, ya que varios triángulos comparten los vértices de un mismo voxel. VTK tiene clases de utilidad que permiten crear conjuntos de puntos que no tengan elementos repetidos. Esta clase se llama vtkMergePoints. Esta almacena un conjunto de puntos, proporcionando funcionalidad para insertar elementos nuevos sin repetir. Al insertar un punto este retorna su Id, con esta información ya puede hacerse el link (enlace) del

triángulo.

Al recorrer todo el volumen, este algoritmo va reconstruyendo la superficie, resultando en una mala que envuelve la región segmentada. Como las caras de los voxels son generados al encontrar los limites de la región, la malla siempre se resulta cerrada. La malla es el limite ente la región interior y la exterior.

La principal ventaja de este método es que los triángulos generados son muy homogéneos, todos tienen la misma escala y relación entre sus lados. Esto da una gran ventaja sobre otros métodos como MC. MC parte de una malla con triángulos de mala calidad, esta es muy difícil de mejorar.

Sin embargo existen casos en la segmentación que pueden generar inconsistencias en la malla resultante.

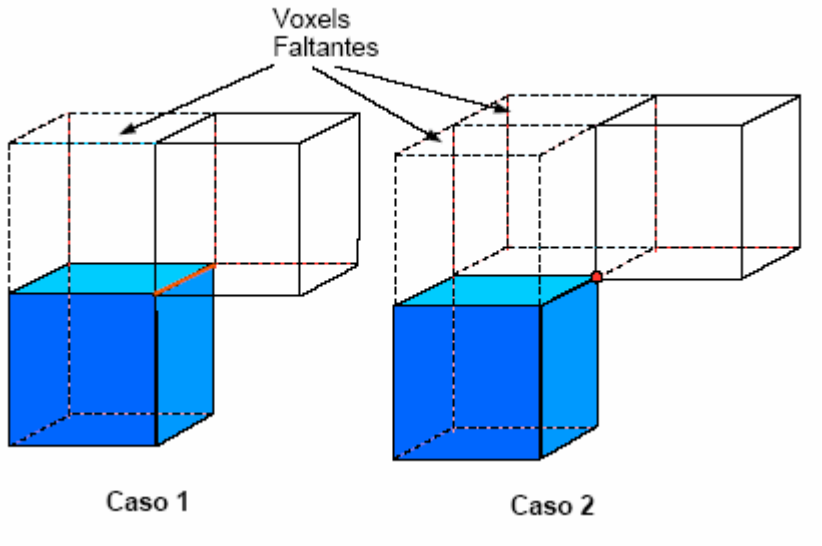


Figura 4. 9 **Casos patológicos**

Casos patológicos

Existen algunos casos puntuales de la segmentación que generan inconsistencias en la malla resultante si no se tienen en cuenta especialmente. El caso más claro de esta situación es cuando el dos voxels se encuentran unidos solo por una arista y ninguno de

los dos vecinos que se encuentran entre ellos esta presente. Este tipo de situación se presenta cuando dos frentes de avance se acercan lo suficiente como para casi tocarse pero desde direcciones diferentes. Como para el crecimiento de la región se utiliza conectividad 6 (vecinos en X, en Y y en Z), es imposible que se generen estas situaciones entre vecinos directos. Es decir, cuando se encuentra uno de estos casos, este fue generado por frentes de avance distintos que se encontraron luego de rodear algún objeto o zona.

En la Figura 4.9, pueden verse los dos casos problemáticos. Los voxels faltantes se indican en líneas punteadas. El otro caso es cuando dos voxels se encuentran unidos por un vértice. Es fácil notar en estos casos que la región en realidad debe estas desconectada en esa zona, pero debido a la forma de la propagación se producen este tipo de anomalías.

Cabe destacar que estos casos particulares no producen problemas al generar la malla, pero si son problemáticos en las posteriores etapas del procesamiento. Generalmente la superficie es post-procesada para suavizarla. La topología de la superficie generada con segmentaciones que presentan este tipo de anomalías no se comporta de la manera esperada al aplicar algoritmos de suavizado, ay que estos algoritmos desplazan los puntos de la malla. En estos casos el algoritmo de suavizado mueve los puntos de una manera incorrecta introduciendo errores en la malla.

Una de las maneras mas simples de solucionar este tipo de problemas es identificarlos y eliminarlos, es decir, antes de generar la superficie, se debe preprocesar la imagen para detectar estos casos y eliminarlos. La manera más simple de salvar estos casos es eliminar el voxel que produce la anomalía. Si un voxel produce una situación de este tipo, este voxel es eliminado de la región marcándolo como voxel externo. Otra alternativa es agregar voxels a la región para llenar los lugares vacíos alrededor de la anomalía. En este trabajo se opto por eliminar el voxel conflictivo, ya que la intención final del trabajo es también proveer la funcionalidad de inflado de la superficie. Si la superficie a ser inflada (es decir va a crecer), al eliminar voxels se le da mas margen la crecimiento.

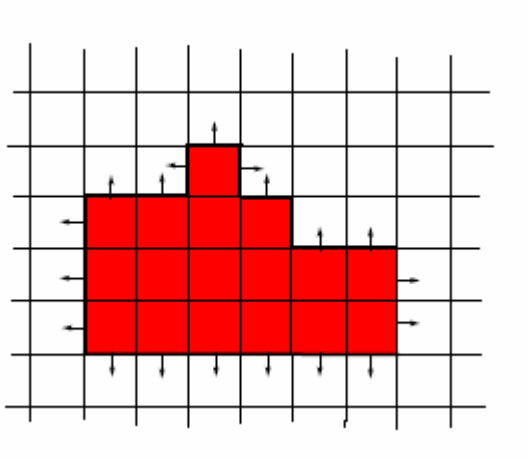


Figura 4. 10 **Ejemplo en 2D de la superficie que limita una segmentación**

Una vez generadas todas las caras de los voxels que limitan con la región exterior, se tiene la superficie que delimita la región segmentada. En la figura 3.10 se presenta un ejemplo en 2D para una segmentación. Las líneas resaltadas representan las caras que fueron generadas de los cuadrados que limitan con la zona exterior. Además se indican las direcciones de las normales para cada una de las caras. Es importante destacar este detalle, ya que las caras de los triángulos deben estar orientadas hacia el exterior. Para lograr esto, hay que identificar de qué lado se encuentra el voxel que esta siendo visitado. Si el voxel se encuentra en el interior de la región, las caras del voxel deben apuntar en dirección del vecino, de otra manera deben apuntar hacia el voxel visitado. Para cambiar la dirección de la cara (sentido positivo o negativo de la normal), solo deben tomarse los puntos del triángulo en orden inverso. Por ejemplo, si los puntos son $<p_0, p_1, p_2>$ (en ese orden) para que la cara quede en dirección del vecino, solo debe declararse $<p_0, p_2, p_1>$ para que la cara "mire" en la dirección contraria.

4.3. POST-PROCESADO DE SUPERFICIES

4.3.1 Crecimiento por gradiente

Luego de reconstruir una superficie utilizando el método de Flat Contour, se sabe que los puntos de la malla que la componen se encuentran dentro de la región buscada. Entonces, hace falta un paso de post-procesado de la superficie que desplace esos puntos a los límites de la región.

Para resolver este problema, se planteó la alternativa de “inflar” la superficie teniendo en cuenta el mapa de gradiente de la imagen original. Sabiendo que las magnitudes de gradiente altas indican límites de una región, la idea básica consiste en desplazar los puntos de manera inversamente proporcional a la magnitud en dirección a la normal calculada en dichos puntos. En otras palabras, cuando un punto se encuentra en una zona de gradiente bajo, este es desplazado en la dirección y el sentido de la normal en dicho punto. Por el contrario, si el gradiente es alto, el punto se mueve una distancia menor (inversamente proporcional al gradiente) y también en dirección y sentido de la normal. De esta manera se asegura que la superficie siempre crezca, pero no demasiado.

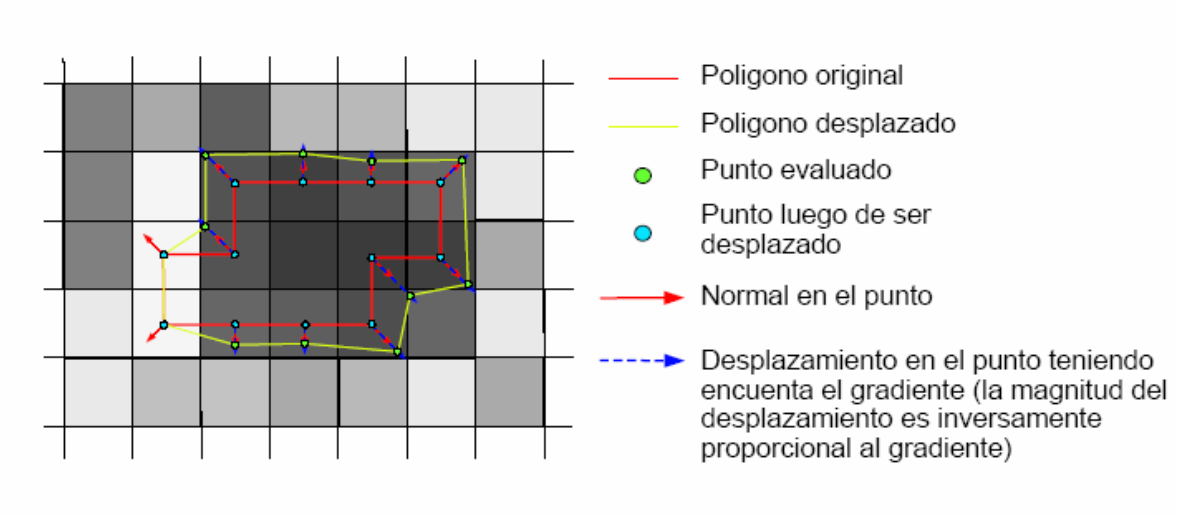


Figura 4. 11 Ejemplo crecimiento por gradiente.

En la figura 4.11 se ve un ejemplo de como funciona el algoritmo. Dependiendo de la magnitud del gradiente en el punto se calcula el desplazamiento a ser aplicado. El desplazamiento es en sentido de la normal. Algunos puntos no son desplazados ya que la magnitud del gradiente es muy grande. En el ejemplo los pixels contienen diferentes intensidades de gris para los diferentes gradientes (mayor el gradiente, mayor la intensidad de gris).

Los datos de entrada del algoritmo son:

1. **Malla:** la información poligonal de la malla generada (probablemente usando FlatContour) a partir de la segmentación.

2. **Imagen gradiente:** A partir de la imagen original se genera una imagen que contiene la magnitud del gradiente en cada punto.
3. **Umbral:** El umbral determina hasta que punto de moverán los puntos (mientras la magnitud del gradiente sea menor que le umbral, el punto puede moverse).
4. **Cantidad de iteraciones:** cantidad de pasadas que se realizarán sobre los puntos de la malla. Este parámetro se utiliza porque en cada iteración cada punto se mueve una distancia muy pequeña (se multiplica el desplazamiento por un factor de 0,5). Al aumentar la cantidad de iteraciones, el resultado se estabiliza.

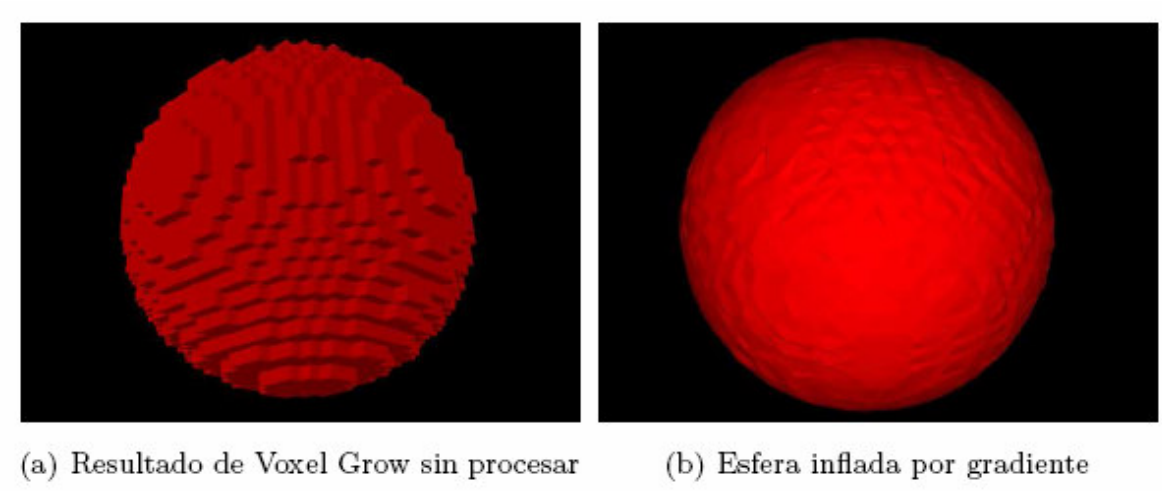
En el algoritmo 4 se puede ver el pseudocódigo del crecimiento.

Algorithm 4 Pseudocódigo de crecimiento por gradiente

1. Para todo punto de la malla P
   a) If (magnitud de gradiente de P < umbral) encolarlo en cola C.
2. While (iteración < máximas_iteraciones && !C.vacía)
   a) punto P = C.front()
   b) M = calcular_magnitud_gradiente (P)
   c) ctedesplazamiento = $\frac{(umbral-M)}{umbral}$
   d) N = calcular_normal_en_el_punto(P)
   e) P' = P + N * ctedesplazamiento * 0.5
   f) M' = calcular_magnitud_gradiente (P')
   g) If (M' < umbral)
      1) mover punto P a la posición P' en la malla
      2) encolar P' en cola C'
   h) If (C.vacía)
      1) C = C'
      2) iteración ++

Resultados

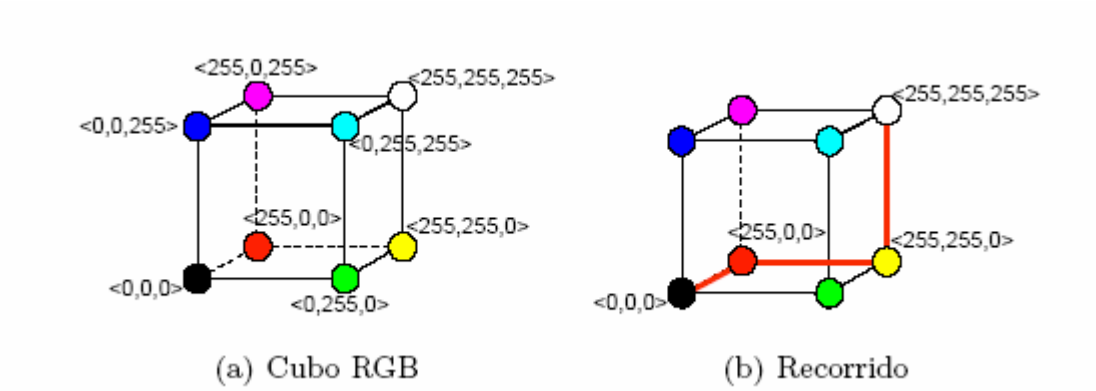
Como caso de prueba esencial y a modo de validación, se probó el algoritmo sobre una esfera. En la Figura 4.12 se ve la comparación antes de realizar el inflado, y luego del inflado.



(a) Resultado de Voxel Grow sin procesar (b) Esfera inflada por gradiente

Figura 4. 12 **Validación del inflado por gradiente desde una esfera.**

El algoritmo funciona bastante bien, pero se producen inconsistencias en la malla al desplazar los puntos. En algunos casos los vértices de los triángulos de cruzan, esto provoca que la malla tenga inconsistencias haciendo este algoritmo poco apropiado.



(a) Cubo RGB (b) Recorrido

Figura 4. 13 **Cubo RGB.**

4.4. MEJORAS EN LA VISUALIZACIÓN

4.4.1. Gamas de colores especiales

Los instrumentos de diagnóstico por imágenes como CT o MRI producen conjuntos de datos (campos escalares) que pueden tener diferentes profundidades (valores de intensidad). Las imágenes pueden tener valores en diferentes rangos como por ejemplo 0..255 (para 8 bits), 0..1024 (para 10 bits), $0..2^{32}$ (para 32 bits), etc. dependiendo del tipo de estudio. Para mostrar estas imágenes generalmente se utiliza una escala de grises, pero esta puede tener algunas limitaciones.

Un aspecto importante que hay que tener en cuenta es que las imágenes poseen valores de intensidad, no de color. Para poder visualizar estos valores de manera mas cómoda, se realiza una conversión entre los diferentes valores y colores del espectro visible, donde cada color representa una intensidad. Para hacer esta conversión (valor de intensidad a color) se debe utilizar alguna estructura o método auxiliar. En este trabajo se utilizó una Lookup Table (LUT). Esta posee para cada valor de intensidad el color correspondiente. Los colores son representados como una tripla en formato RGB (<R,G,B>). En la figura 3.13 se muestra la disposición de colores en la Cubo RGB. La imagen muestra los colores primarios (tanto RGB como CMY). En el momento en que debe realizarse la renderización, se consulta esta tabla para saber con que color se deben pintar los puntos de la superficie de acuerdo al valor de intensidad que cada punto tiene.

Escala de grises

La manera mas común de visualizar imágenes médicas es utilizando escalas de grises. Las imágenes extraídas tanto de un tomógrafo como de una MRI son generalmente impresas con una impresora láser en placas de acetato. Para ver estas placas se usa contraluz, esto resalta los pequeños detalles de la imagen. Como la impresora tiene muy alta definición, esta técnica permite detectar detalles muy pequeños. Los profesionales médicos esta acostumbrados a ver imágenes en gamas de grises. En imágenes en gamas de gris fácilmente se pueden identificar las diferentes estructuras y los limites entre las diferentes regiones. Este tipo de gamas facilita la tarea del profesional para identificar tejidos o lesiones.

Sin embargo, para representar estas imágenes en un monitor (de computa- dora), el cual trabaja normalmente con profundidades de color de 24 bits, solo se dispone de 256 intensidades de gris. En algunos casos son suficientes, pero existen muchas imágenes que poseen una profundidad de imagen mucho mayor. Al utilizar estas gamas de color en imágenes de mas de 256 valores de profundidad, se pierde gran cantidad de detalles. Para esto se diseñaron tablas de color especiales.

Gamas de color alternativas

Para mejorar la visualización de imágenes médicas con las de 256 valores de intensidad, se planteó la alternativa de realizar recorridos en el Cubo RGB que proporcionen rangos más amplios de color. Para lograr una gama de grises se realiza un recorrido en línea recta desde el blanco (RGB = <256,256,256>) hasta el negro (RGB = <0,0,0>). Este recorrido solo posee 256 valores distintos, ya que para obtener una intensidad de gris, las intensidades de las tres componentes (R, G y B) deben ser iguales. Los valores serían: <0,0,0>, <1,1,1>, <2,2,2>,<3,3,3>, etc. hasta <256,256,256>. En la figura 3.13 se muestra el recorrido que esta gama realiza sobre el Cubo RGB. Para lograr más cantidad de colores, se pueden realizar recorridos más largos. Un buen ejemplo es comenzar en <0,0,0> y aumentar la intensidad de la componente roja (<1,0,0>,<2,0,0>,<3,0,0>...<255,0,0>) hasta llegar a 255. En esta escala se tiene 256 colores que van del negro al rojo. A partir de aquí se puede recorrer el camino que va del rojo al amarillo disminuyendo gradualmente la intensidad del rojo y aumentando el verde (<254,1,0>,<253,2,0>,<252,3,0>...<0,255,0>). Hasta este momento la gama ya tiene 512 colores. Para terminar se recorre la distancia que hay hasta el blanco (<1,255,1>, <2,255,2>, <3,255,3>...<255,255,255>), esto resulta en una tabla de 768 colores.

Este ejemplo resulta en una tabla de 768 colores. Fácilmente se pueden implementar tablas de 1024, 2048 o mas colores, sin embargo no es tan simple implementar tablas que tengan dos o mas órdenes más de cantidad de valores (tablas de 1.000.000 de valores por ejemplo). Estos casos deben ser tenidos en cuenta especialmente, ya sea para pasar por alto el nivel de detalle o para visualizar la información de otra manera.

3.4.1.3. Implementación en TCL [16]

Esta tabla fue implementada como un filtro TCL, esta se debe cargar explícitamente. Una vez cargada la imagen se le asigna la tabla de colores con la que se desea visualizarla. Las tablas de color tienen parámetros de configuración como:

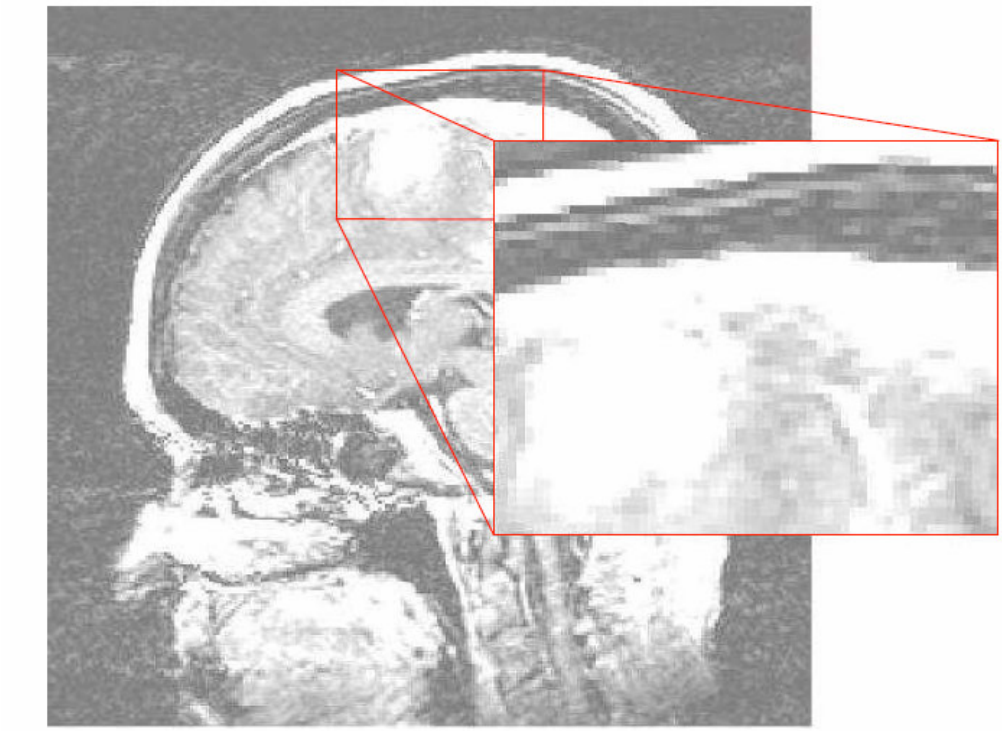
- **Gamma:** este parámetro define cual es la gama que se utilizará (que recorrido se realiza en el Cubo RGB).
- **numOfCols:** este parámetro determina la cantidad de entradas que tendrá la tabla. Si una tabla realiza un recorrido que tiene 256 valores (ej: escala de grises), pero el parámetro numOfCols es configurado en 1024, la tabla tendrá colores repetidos. Si, en cambio, el recorrido tiene 768 valores, pero este parámetro tiene el valor 256, en la tabla no se utilizarán todos los colores. Esto puede ser aprovechado por el usuario para variar la gama de colores con la que representa una imagen 3D.
- **rangeMin:** con este valor se indica en que valor comienza la tabla. Si de antemano el usuario sabe que las intensidades del la imagen que tienen valores menores que 1200 representan espacios vacíos, puede no asignar color a esos valores.

Estos parámetros permiten obtener una variedad de diferentes tablas de color. El criterio y entendimiento del usuario es fundamental para lograr una buena visualización.

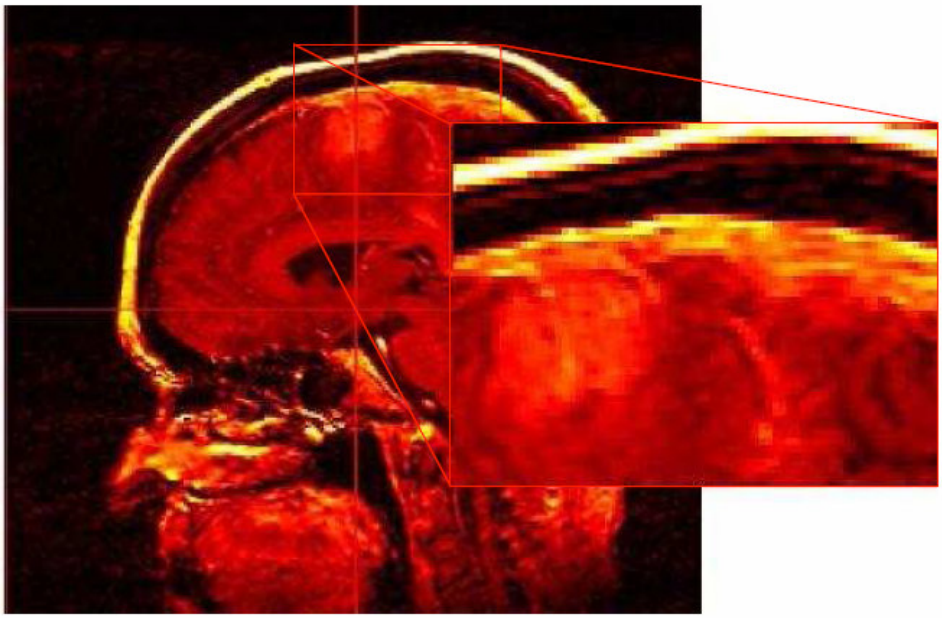
Comparaciones

Para notar las diferencias entre las distintas tablas se presenta a continuación una comparación entre dos diferentes tablas de color para una misma imagen. En la figura 4.14 se puede ver la misma imagen con dos gamas de color diferentes.

[16] TCL (del inglés Tool Command Language) es un lenguaje de script. Se usa principalmente en programas rápidos, aplicaciones en secuencia, entornos gráficos y pruebas.



(a) Escala de grices



(b) Gama de color

Figura 4. 14 Gamas de color

5. DISEÑO DE LA APLICACIÓN

5.1. Generalidades

Se diseñó e implementó una aplicación con el objetivo principal de evaluar los métodos de segmentación y de reconstrucción de superficies explicados en el capítulo 3. Adicionalmente, la aplicación se puede utilizar con otros fines, mas allá de la segmentación, ya sea como procesador de imágenes, o como sistema de visualización tridimensional.

El sistema se compone de dos módulos principales: segmentación y visualización. La Visualización se ofrece en dos posibilidades: a partir de la segmentación y a partir de las imágenes originales de tomografía, siendo dos tipos de visualización diferentes; de superficies y de volúmenes respectivamente.

5.1.1 ALGORITMO DE SEGMENTACIÓN DESARROLLADO

Como el nombre indica, el crecimiento de regiones es un procedimiento mediante el cual se agrupan pixels o subregiones en regiones mayores. El procedimiento más sencillo se denomina agregación de pixels, que comienza a partir de un conjunto de pixels semilla, de forma que a partir de cada semilla se crecen regiones añadiendo pixels a dicha semilla de entre aquellos pixels vecinos que tienen propiedades similares. El resultado de la segmentación dará lugar como mucho a tantas regiones como semillas haya. Sin embargo, puede darse el caso de que dos de esas semillas correspondan a pixels de la misma región. En este caso el crecimiento desde una de las semillas absorberá a la otra, que en este caso deberá ser descartada. Un ejemplo sencillo, pero muy usado en la practica de similitud es la diferencia absoluta en el nivel de gris. Fijado un umbral ***t*** se va calculando la diferencia en valor absoluto del nivel de gris del pixel en cuestión (en el vecindario de la región crecida hasta el momento) con respecto al nivel de gris de la semilla y si no se supera ese umbral ***t*** se añade a la región.

Dos problemas fundamentales en el crecimiento de regiones son: por un lado, la selección de las semillas o puntos de partida que representen adecuadamente a las regiones de interés; y por otro, la elección de las propiedades adecuadas que permitan ir añadiendo

pixels durante el proceso de crecimiento. Hay diferentes formas de evaluar la pertenencia o no de un voxel a una región. El algoritmo de crecimiento desarrollado usa los siguientes criterios:

• **Crecimiento según semilla.** En este caso, si la diferencia entre el valor de la semilla *s* y el del voxel analizado se mantiene dentro de un límite de tolerancia *t* dado, se acepta su incorporación. Las regiones resultantes se componen de voxels que satisfacen el criterio enunciado en la ecuación 1:

$$valor(v) - valor(s) <= t \qquad (1)$$

• **Crecimiento según el vecino.** Esta opción considera que un voxel *v* pertenece a la región si la diferencia entre su intensidad y la del vecino *v'* que lo precede se mantiene por debajo de *t,* como muestra la ecuación 2.

$$valor(v) - valor(v') <= t \qquad (2)$$

• **Crecimiento por gradiente.** La magnitud del gradiente en un punto es una medida de la variación de intensidades alrededor de ese punto [4, 7]. Esta alternativa de crecimiento utiliza este valor para establecer los límites de la región, ya que el proceso de llenado se detiene si el gradiente en un voxel *v* supera el umbral dado. Entonces, la condición de aceptación de un voxel en este caso es el enunciado en la ecuación 3:

$$valor(gradiente(v)) <= t \qquad (3)$$

Una región inicialmente queda constituida por el conjunto de semillas que han sido especificadas. Como se mencionó antes, el algoritmo de crecimiento se basa en extender la región por medio de la incorporación progresiva de los voxels vecinos a los que ya están integrados, mientras satisfagan el criterio de aceptación establecido. Para ello, los voxels visitados que cumplen la condición son insertados en una lista hasta el momento de ser procesados e integrados a la componente de interés.

En cada ciclo del algoritmo, se debe extraer uno de los elementos de la lista para incluirlo a la región y luego analizar la posible extensión sobre sus vecinos. Con el fin de aumentar las capacidades del algoritmo de crecimiento, se ha analizado una estrategia novedosa de

elección de los voxels candidatos en cada paso. La idea es proveer al proceso de conocimiento a través de información de algunas propiedades de la región en cuestión. Las posibilidades en este punto son variadas, aunque en el caso de aplicaciones médicas no resulta sencillo debido a la complejidad de las estructuras consideradas. No obstante, de acuerdo al tipo de componente que se desee segmentar, se pueden considerar algunos aspectos, como medidas de volumen, modelos de formas, etc., que permitan guiar el proceso de crecimiento de una forma más inteligente que en el caso convencional. Se ha considerado una primera estrategia basada en la posibilidad de establecer un volumen estimado para la componente de interés, y así determinar la región que mejor se adapte a ese volumen.

Uno de los aspectos interesantes de esta técnica es que las regiones crecen a través de la incorporación de los vecinos de los voxels ya incluidos, por lo que se asegura la conectividad entre los elementos de la región segmentada. Dependiendo del tipo de material considerado, el esquema anterior suele provocar desbordes de la región segmentada. Esto puede suceder cuando el llenado se dirige, a través de vías estrechas, hacia otras zonas de la imagen que en realidad corresponden a un material diferente, pero que poco difieren en intensidad. Este problema se hace más evidente cuanto mayor es la tolerancia admitida. Con el fin de evitar este tipo de situaciones, se ha considerado una opción más elaborada que, en lugar de tener en cuenta la similitud entre elementos individuales, realiza un crecimiento por entorno de voxels. En cada punto de la imagen, su entorno queda definido como el conjunto de voxels que lo rodean a distancia *r* en cada dirección (Figura 5.1, vista sobre un plano).

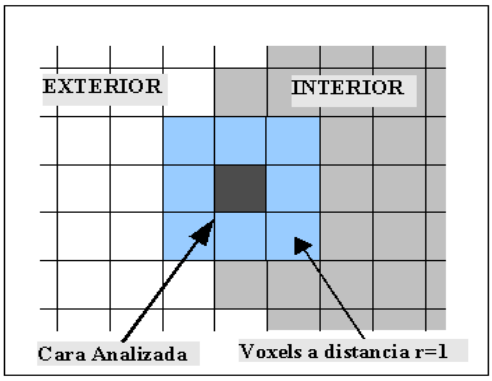


Figura 5. 1 Esquema de crecimiento por entorno

Cuanto mayor es el valor de *r*, más restrictiva resulta la condición de incorporación, ya que se exige que todos los elementos cumplan el criterio de aceptación para ser incorporados, razón por la cual pueden originarse huecos dentro de la región segmentada.. Esta estrategia brinda mayor flexibilidad que el caso anterior, a la vez que sigue reduciendo las oportunidades de escape del proceso de llenado por conductos delgados.

Para presentar una ilustración del proceso de segmentación con el algoritmo implementado, en la Figura 5.2 se muestra una imagen correspondiente a un conjunto de cortes tomográficos. Se escoge la zona de interés, que en este caso, para afectos de ilustración se tomó la zona de los ojos. Se aplica un filtro de ajuste de intensidad sobre la imagen y se colocan las semillas necesarias (en este caso una semilla en cada zona ocular). Al colocar en funcionamiento el algoritmo, estas semillas se van expandiendo, teniendo en cuenta los parámetros de inclusión de voxels y tolerancia para evitar que la semilla se expanda a zonas no deseadas.

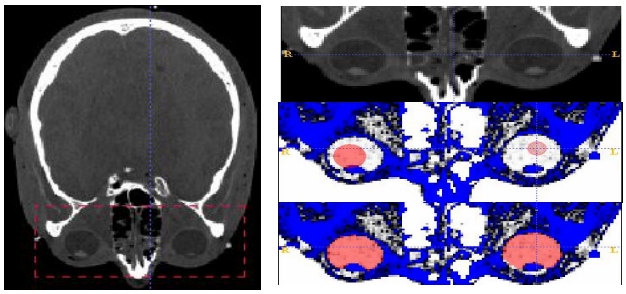


Figura 5. 2 Proceso de segmentación con algoritmo de crecimiento dinámico de regiones.

La Figura 5.3 muestra el resultado del proceso de segmentación de la zona ocular presentado anteriormente como ejemplo. Una vez segmentada esta región, se procede a reconstruir la zona y visualizarla en un espacio 3D, aislada de las demás estructuras, que para este caso no eran de interés, como lo muestra la Figura 5.3.

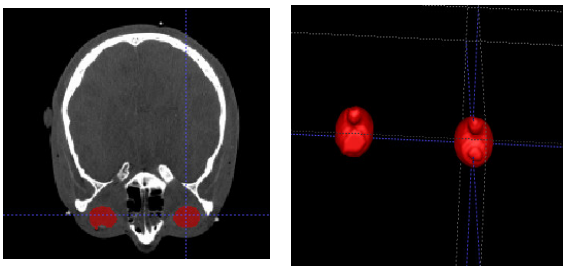


Figura 5. 3 Imagen de tomografía perteneciente a un volumen segmentado y su visualización en un espacio 3D.

5.1.2 VISUALIZACIÓN 3D

La visualización 3D es el proceso mediante el cual se despliega una imagen con aspecto tridimensional en una pantalla de computador. Esto se logra a partir de un conjunto de operaciones y transformaciones matemáticas aplicadas a los modelos de imágenes, a través de la computación gráfica, dando efectos de profundidad, rotación, traslación y escalado.

Una vez se ha extraído la región de interés de las imágenes médicas, el siguiente paso es reconstruirlo y visualizarlo en un espacio tridimensional, con el fin de que el usuario pueda interactuar con el modelo. La herramienta que se realizó como apoyo a la visualización tridimensional consta de dos módulos: volúmenes y superficies.

Visualización de volúmenes

El módulo de visualización de volúmenes permite desplegar un modelo tridimensional sin necesidad de haber hecho una segmentación previa. Aquí se podrán visualizar algunas estructuras de interés a partir de la modificación de parámetros de transparencia y funciones de transferencia de opacidad, teniendo en cuenta siempre el nivel de intensidad de las imágenes. La Figura 5.4 muestra una imagen de éste módulo, donde se puede ver la imagen 3D y un conjunto de controles de interfaz gráfica que permiten cambiar los parámetros de visualización del volumen. Las técnicas utilizadas para la visualización de volúmenes se basan en un conjunto de algoritmos de R*ay Casting* [15], en el cual se determinan las superficies visibles en la escena que se quiere sintetizar trazando rayos desde el observador (cámara) hasta la escena a través del plano de la imagen. Se

calculan las intersecciones del rayo con los diferentes objetos de la escena y aquella intersección que esté más cerca del observador determina cuál es el objeto visible.

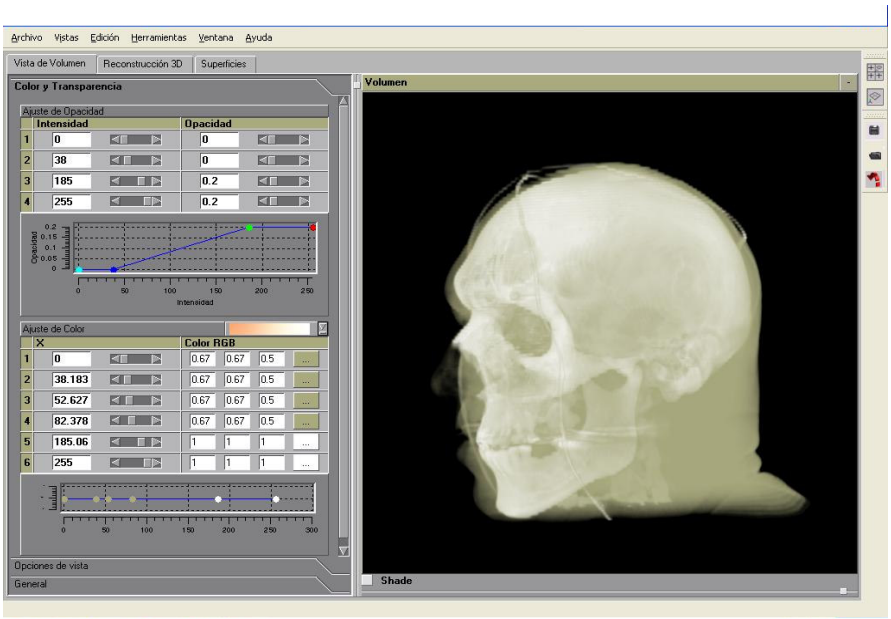


Figura 5. 4 Visualización de volúmenes usando técnicas de Ray Casting

Debido a que la visualización de volúmenes es un proceso que se realiza en tiempo real, a menudo es de mucho interés visualizar en un mismo espacio las imágenes originales en diferentes vistas junto con la visualización del volumen, por esta razón se desarrolló un modelo de múltiples vistas como se muestra en la Figura 5.5.

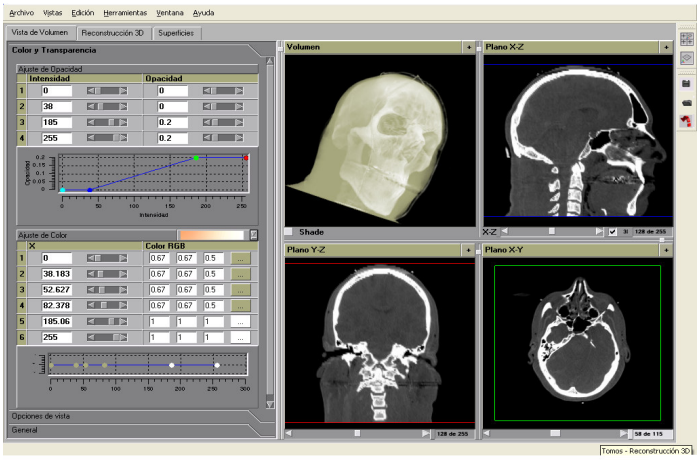


Figura 5. 5 MÚLTIPLES VISTAS EN EL MÓDULO DE VISUALIZACIÓN DE VOLÚMENES

En realidad las imágenes originales pertenecen a una única vista, por lo tanto se realiza una interpolación de pixels en cada imagen para generar las otras vistas, lo cual es de mucha utilidad al momento de analizar las estructuras anatómicas y compararlas con el modelo 3D para facilitar el diagnóstico.

La herramienta desarrollada permite apreciar los resultados del proceso de segmentación mediante visualizaciones de los diferentes cortes sobre los ejes cartesianos (como los de la Figura 5.6) o a través de proyecciones arbitrarias del objeto mediante *volume rendering* [15]. Además de estas facilidades, se ha incluido la posibilidad de generar los datos necesarios para la visualización de las superficies de las regiones segmentadas dentro del volumen, mediante opciones de *surface rendering* [17].

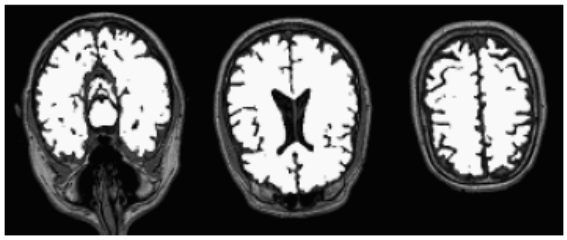


Figura 5. 6 Diferentes cortes de imágenes tomográficas segmentadas.

VISUALIZACIÓN DE SUPERFICIES

Una de las representaciones comúnmente utilizadas para la descripción de superficies consiste en la determinación de mallas de polígonos (generalmente triángulos). Usualmente, para generar esta descripción se utilizan técnicas tradicionales como *Marching Cubes* [16] o *Marching Tetrahedra* [17]. Estas alternativas son veloces y se comportan bien en la mayoría de los casos, permitiendo la generación de isosuperficies a partir de un valor de umbral especificado. Sin embargo, suelen conducir a situaciones ambiguas, las que pueden generar huecos en la superficie resultante y además poseen las desventajas inherentes a los métodos basados en *thresholding*.

Se presenta un enfoque eficiente para la generación del modelo de superficie, el cual se aplica sobre los resultados de una segmentación previa. El algoritmo se basa en encontrar el límite entre la zona segmentada (interior) y el exterior de la misma con el fin

de obtener un conjunto de triángulos que lo describan. Esta alternativa consiste de dos pasos, como se describe a continuación.

Triangulación inicial. Cada región segmentada provee un marco de referencia para la generación de la descripción poligonal de su superficie. En un primer paso, se recorre el volumen segmentado en búsqueda de voxels adyacentes correspondientes a interior y exterior (o a dos regiones diferentes), considerando las caras de los voxels que corresponden al contorno de la región de interés. A partir de estas caras frontera se puede generar una triangulación inicial, dividiendo cada una de ellas en dos triángulos, a partir del trazado de una diagonal. Cada esquina de un voxel frontera constituye un vértice numerado de la malla regular en que se convierte la imagen y entonces cada triángulo se puede describir por la tripla de vértices que lo determinan. Para obtener superficies cerradas y coherentes, se debe tener cuidado al especificar la dirección del vector normal a cada triángulo y por lo tanto el sentido de los vértices.

Suavizado de superficies. Si bien el dominio generado a partir del conjunto de triángulos que describe la superficie de cada región segmentada corresponde al objeto en estudio, la imagen resultante puede ofrecer un aspecto escalonado que entorpece la visualización. Esto se debe a que los triángulos se generan a partir de las caras de los voxels y por lo tanto muchos de ellos resultan ortogonales entre sí (Figura 5.7-a). Una manera de mejorar la calidad de la superficie es aplicar un postprocesamiento de suavizado, de manera que disminuya las irregularidades y lograr una apariencia más semejante a las formas originales de los materiales (Figura 5.7-b).

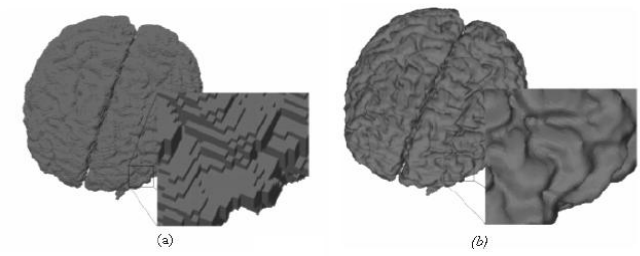


Figura 5. 7 Descripción poligonal de superficies: (a) triangulación inicial – (b) superficie suavizada

El suavizado se realiza mediante la aplicación de un filtro promedio. Para ello, se consideran –para cada vértice de la malla generada que se desea suavizar- todos los triángulos que poseen un vértice coincidente con él y se calcula su nueva posición como el promedio de las posiciones de los demás vértices de los triángulos involucrados. El proceso se puede aplicar en forma iterativa, hasta obtener los resultados deseados. Al tratar con superficies de forma convexa, este procesamiento puede provocar una reducción del volumen del cuerpo, ya que los puntos tenderán a moverse hacia el interior de la región, siendo mayor el efecto según la cantidad de iteraciones efectuadas. Por lo tanto, debe ser aplicado moderadamente y sólo a efectos de visualización.

Otra posibilidad es realizar un análisis de frecuencias sobre los vértices que componen la malla de polígonos, según se describe en [18]. A partir de él, es posible clasificar cada uno de los vértices como de baja o alta frecuencia y determinar si es necesario procesarlos. El propósito de este algoritmo es suavizar la superficie, distribuyendo los vértices de forma más homogénea.

El módulo de visualización de superficies permite guardar, cargar, importar y exportar modelos generados en otras herramientas. La Figura 5.8 muestra un modelo cargado, al cual se le han agregado características como textura e iluminación, para una mejor visualización.

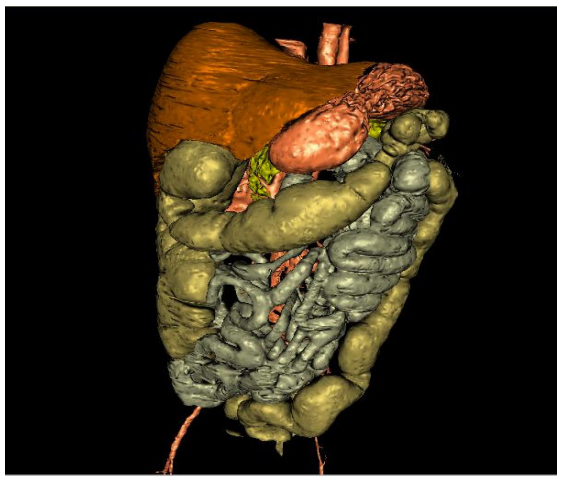


Figura 5. 8 Visualización con textura e iluminación

También es posible visualizar todas las vistas en una misma ventana, con el fin de que tanto las tres vistas de las imágenes e incluso el modelo de superficie coincidan en un mismo modelo (Figura 5.9).

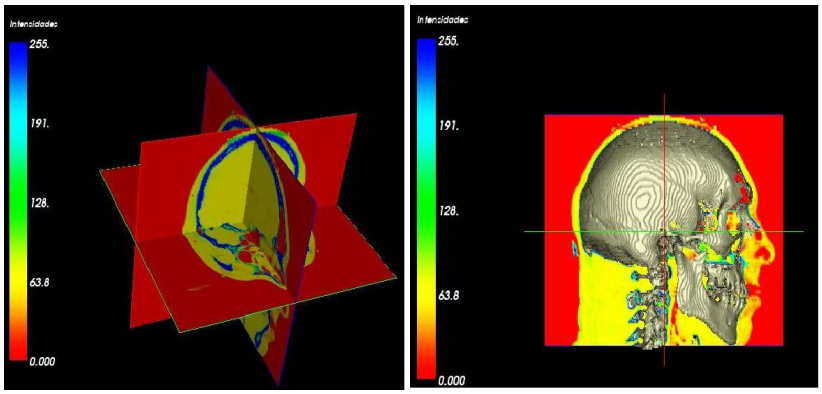


Figura 5. 9 Visualización de modelos en vista múltiple

5.1.3 Requerimientos

Los requerimientos tenidos en cuenta en la etapa de diseño establecieron que la aplicación debe:

Proveer una interfaz uniforme para segmentar imágenes tridimensionales sin importar el método de segmentación. De esta manera se puede establecer una comparación en los mismos términos de los resultados de los métodos, y/o seleccionar el mas adecuado de acuerdo a la imagen utilizada.

Proveer de un sistema de visualización interactivo 3D. Inherentemente al tratar con imágenes tridimensionales, y por el hecho de realizar la reconstrucción tridimensional de superficies, es necesario poder ver e interactuar con la información también en 3 dimensiones de una manera natural.

Proveer una interfaz de usuario amigable. Una interfaz amigable ayuda a la usabilidad, fundamental en un sistema 3D.

Tener un tiempo de respuesta aceptable para la ejecución de los algoritmos y la visualización, ya que se desea proveer entorno de alta interactividad.

Portabilidad a diferentes plataformas. Si bien este requerimiento no es fundamental,

permite evaluar y aprovechar ciertas características disponibles en plataformas no-Windows (por ejemplo, capacidad de procesamiento y graficación tridimensional ampliada en workstations Sillicon Graphics).

Para ver los requerimientos funcionales específicos que satisface la aplicación referirse a la sección 5.3.4.
Para satisfacer estos requerimientos, se realizó un análisis sobre las posibles soluciones a cada uno de ellos. A continuación se enumera cada uno y se discute la solución adoptada para satisfacerlo.

Tiempo de respuesta aceptable

El procesamiento de imágenes es costoso en tiempo de ejecución y en memoria, y más aún cuando se trata de imágenes tridimensionales. Por lo tanto se debe tener en cuenta la utilización eficiente de los recursos de memoria y procesador.

Para satisfacer estas condiciones, se optó como lenguaje de implementación a C++. No se tuvo en cuenta a Java ya que si bien presenta ciertas características deseables (portabilidad, orientación a objetos, garbage collection, rico en clases de utilidad, cantidad de herramientas de desarrollo, etc.) su performance no es óptima para procesar grandes volúmenes de datos.

La elección de C++ introduce ciertas dificultades en la implementación; por ejemplo, se debe tener especial cuidado con el manejo de la memoria, y lidiar con la ausencia de clases de utilidad para desarrollar la aplicación. Sin embargo, se utilizó como técnica de manejo de memoria los punteros inteligentes[17], y como clases de utilidad se utilizó a la librería STL[18] que viene acompañada por cualquier compilador estándar de ANSI-C++.

[17] Conocidos habitualmente como SmartPointers en inglés. Un puntero inteligente tiene la propiedad de liberar la memoria del objeto al que apunta cuando nadie mas los referencia. Esto se logra gracias a un esquema de *reference counting*. De esta manera, el llamado a los destructores es realizado de manera implícita cuando nadie más hace referencia al objeto.

[18] Standard Template Library. Consiste de un conjunto de clases de utilidad (vectores, tablas de hash, algoritmos) que adoptan como paradigma de desarrollo a la programación genérica.

Interfaz uniforme para realizar la segmentación

Cada método de segmentación está controlado por diversas variables de ajuste y parámetros de entrada. Por lo tanto, para lograr esta uniformidad fue necesario realizar una abstracción del proceso. En la sección 5.4 se ve plasmada la abstracción en el diagrama de clases correspondiente.

Visualización e interactividad 3D

Este requerimiento plantea diversos obstáculos, ya que se debe mantener un entorno de visualización 3D además de permitir interactuar con el mismo. La solución natural al problema de visualizar objetos en 3 dimensiones es utilizar al lenguaje gráfico OpenGL, que es un estándar adoptado por los fabricantes de la gran mayoría de placas de vídeo aceleradoras y disponibles en muchas plataformas. El inconveniente con la utilización directa de OpenGL es que no provee la noción de objeto tridimensional, sino que trabaja a una granularidad más fina, es decir, a nivel de triángulo, polígono, etc. por lo que se debía hacer una abstracción para definir la noción de "objeto 3D" para trabajar de manera mas "cómoda".

Por otra parte, para la interacción, OpenGL no provee ninguna facilidad, y se debía recurrir a una librería de utilidad[19] que permita interactuar con los cenarios tridimensionales generados por OpenGL.

Lamentablemente, el uso de esta librería no cumple con el requerimiento de interfaz amigable con el usuario, ya que los controles de interfaz que provee son bastante limitados.
Sin embargo, luego de una búsqueda intensiva y evaluación de las alternativas, se optó por VTK[20], un framework para desarrollar aplicaciones de visualización tridimensional que posee las características deseadas.

[19] Específicamente, GLUT. http://www.vtk.org

[20] Visualization Toolkit. http://www.vtk.org

Interfaz amigable con el usuario

Existen en la actualidad una infinidad de frameworks para desarrollar Interfaces gráficas de usuario con C++ y por lo tanto se realizó una evaluación de ellos y seleccionar el más adecuado. Básicamente los criterios de búsqueda apuntaban a que se tratara de un framework en desarrolado en lenguaje C++, OpenSource, y multiplataforma. Adicionalmente, debía soportar OpenGL y obviamente poder vincularlo a VTK. De acuerdo a estos criterios, las opciones resultantes fueron:

- **QT.** Este es un framework completo, y posee todas las características deseadas. Multiplataforma y con buen soporte técnico.
- **wxWindows.** Reune las características deseadas, pero en etapas de prueba quedó en evidencia que el tiempo que toma en aprender a utilizar el framework es muy grande.
- **FLTK.** Resultó seleccionado. Un framework sencillo y completo. Como ventaja adicional provee widgets para incorporar visualizaciones generadas con OpenGL por lo que facilitaba aún más el proceso de desarrollo de la aplicación.

Existen gran cantidad de frameworks, cada uno con sus ventajas y desventajas. De las opciones estudiadas se concluyó que FLTK es adecuado, pero QT posee algunas prestaciones, como tipos de controles de interfaz gráfica que eran necesarios en algunos módulos. Por esta razón se decidió utilizar FLTK y QT en aquellos módulos donde fuera necesario.

Portabilidad

La portabilidad a otras plataformas no es esencial pero permite utilizar ciertas características disponibles en cierto tipo de hardware (como son mayor disponibilidad de recursos de memoria, graficación, procesador).

Todos los componentes utilizados (ANSI-C++, STL, VTK, FLTK, QT) pueden ser portados a otras plataformas con solo recompilar y enlazar el código fuente, logrando la portabilidad de la aplicación.

5.2. Arquitectura general

Antes de avanzar sobre los detalles de la arquitectura seleccionada para construir la aplicación, es conveniente aclarar que la decisión se vio fuertemente influenciada por la propia estructura del proceso de visualización. Dicho proceso se puede ver en la Figura 5.10, y se puede definir como una sucesión de *transformaciones* sobre los datos, que luego son *mapeados* a un dispositivo (en este caso, el monitor) para su *representación*.

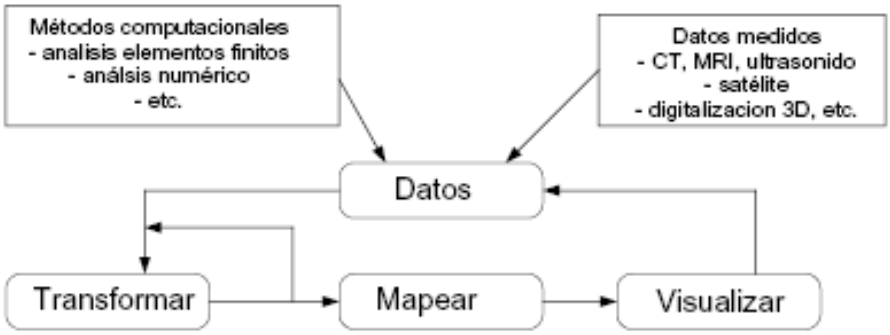


Figura 5. 10 El proceso de visualización

Debido a las características del proceso de escogió un tipo de estructura *data-flow* como estilo arquitectural el modelo, específicamente el modelo de *Pipes & Filters*, donde los componentes que intervienen son:

- **Pipes** que son los encargados de mover los datos (ver mas adelante los *Data Objects*) de la salida de un filtro a la entrada de otro. Los pipes son los que determinan la topología del pipeline.
- **Filters** que son los componentes que transforman los datos de entrada en datos de salida (ver mas adelante los *Process Objects*). Adicionalmente, los filtros usan poca información contextual para llevar a cabo la transformación, de manera que tratan de no mantener ningún estado común entre las diferentes instanciaciones de un mismo filtro, para dar independencia y reutilización a los componentes.
- **Pipeline** que es el resultado de la interconexión de los filtros. La ejecución del

pipeline da como resultado la aplicación de todas las transformaciones y transferencia de datos entre los filtros.

Para implementar la aplicación, y debido a que VTK tiene también un estilo arquitectónico *Pipes & Filters,* se construyó un pipeline general, a través del cual fluyen los datos (imágenes y superficies) y se realizan las operaciones de segmentar, reconstruir y visualizar. Los detalles del pipeline, se dan en la sección 5.3.

5.3. Pipeline

El pipeline consiste de objetos para representar los datos (*data objects*) y objetos que operan sobre los datos (*process objec*ts). Se le llama *red* a una configuración particular de pipeline (aquel grafo que definen los Pipes y los Filtros).

5.3.1. Componentes

Data Objects

Los *Data Objects* representan la información, además de proveer métodos para crear, acceder y borrar a dicha información. La modificación directa de los datos representados por los *Data Objects* no está permitida.

Esta capacidad está reservada para los *Process Objects*. Existen métodos que permiten obtener características de los datos representados. Por ejemplo, determinar el valor máximo o mínimo de los datos, o determinar el tamaño y la cantidad de valores en el objeto.

Los *Data Objects* difieren en lo referente a su representación interna, ya que esta determina la eficiencia en el almacenamiento o en el acceso a los datos, y por lo tanto en la eficiencia de los *Process Objects* que acceden a esta información. Por lo tanto, es posible utilizar diferentes tipos de *Data Objects* para representar los mismos datos, dependiendo de las necesidades de eficiencia ó generalidad en su procesamiento.

Process Objects

Los *Process Objects* operan en los datos de entrada para generar los datos de salida. Un

process object puede derivar nuevos datos desde su entrada, o bien hacer una transformación a los mismos. La entrada a un *process object* está compuesta de uno o más *Data Objects* como también de parámetros locales que definen para controlarlo.
Los *Process Objects* pueden ser subcategorizados como *source objects* (fuente de datos), *filter objects* (filtros), o *mapper objects* (mapeadores). Esta categorización depende de que un objeto inicie, mantenga o termine un flujo de datos dentro del pipeline.

Los *source objects* son la interfaz a las fuentes de datos externas (por ejemplo, archivos), o generan datos de acuerdo a parámetros locales. En VTK los *source objects* heredan de la clase abstracta Source.

Los *filter objects* requieren como entrada 1 o más *data objects* y generan como salida 1 o más *data objects* y como se dijo anteriormente son controlados mediante los parámetros locales. En ITK y VTK, los *process objects* heredan de la clase ProcessObject.

Los *mapper objects* corresponden a los sumideros en el pipeline. Estos requieren uno o más *data objects* de entrada, y concluyen el flujo de datos en el pipeline. Generalmente los *mapper objects* son los encargados de convertir los datos en primitivas de graficación (por ejemplo, comandos OpenGL en VTK a través de la clase vtkRenderer), pero pueden llevar a cabo otro tipo de procesamiento (como ser exportar datos a un archivo, o comunicarse con un sistema externo).

5.3.2. Conexiones en el pipeline

Es importante determinar como se realiza la conexión entre los *data objects* y los *process objects* en el pipeline. Debido a que se utiliza C++ como lenguaje de base, la conexión de los filtros está determinada por las reglas de chequeo de tipos del lenguaje, por lo tanto la consistencia del pipeline está determinada en tiempo de compilación.

Las conexiones se realizan a través de los métodos SetInput() y GetOutput() que asignan la entrada de un filtro, y obtienen la salida del mismo. Por ejemplo,

```
filter2->SetInput(filter1->GetOutput());
```

indica que la salida del filtro 1 debe ser conectada a la entrada del filtro 2. En tiempo de compilación, el compilador C++ hace una verificación de tipos, dando consistencia al armado del pipeline.

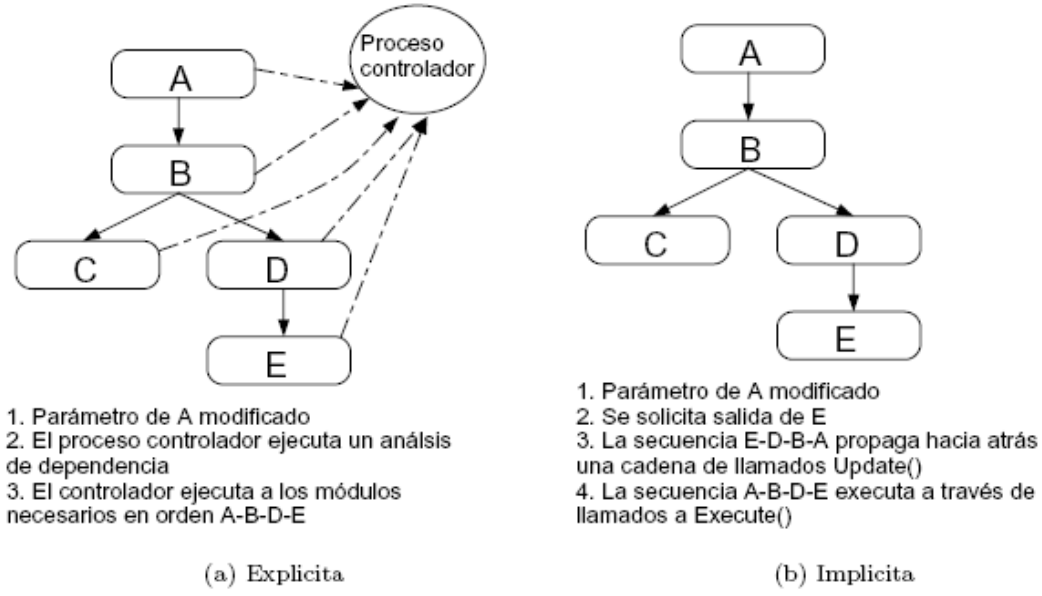


Figura 5. 11 Ejecución explícita e implícita del pipeline.

5.3.3. Ejecución del pipeline

Para ser útil, el pipeline debe procesar los datos para generar un resultado. El procedimiento de dar ejecución a cada *process object* es llamado *ejecución del pipeline*.

Frecuentemente el pipeline es ejecutado más de una vez ya sea por el cambio de algún parámetro o de alguna entrada en algún *process object*. Cuando alguno de estos cambios ocurre, se debe ejecutar nuevamente el pipeline para generar resultados que estén actualizados de acuerdo a los nuevos parámetros y/o entradas. Para ganar en performance, los *process objects* debe ejecutarse *solamente* si un cambio ocurre en su entrada.

La ejecución del pipeline requiere de una adecuada sincronización en cuanto a la ejecución de los *process objects*. Es deseable ejecutar los filtros solamente cuando sus

entradas se encuentran actualizadas. Por lo tanto hay 2 alternativas para sincronizar la ejecución del pipeline: ejecución explícita, y ejecución implícita. (Figura 5.11).

Ejecución explícita

En la ejecución explícita, hay un componente centralizado encargado de llevar un registro de los cambios en cada *process object*. Cuando se requiere la salida actualizada este componente hace un análisis de dependencia sobre el grafo de procesamiento y ejecuta aquellos *process objects* que son necesarios. Además, el uso de este componente centralizado hace posible llevar más control sobre los mecanismos de sincronización y facilita la construcción de pipelines de procesamiento distribuidos.

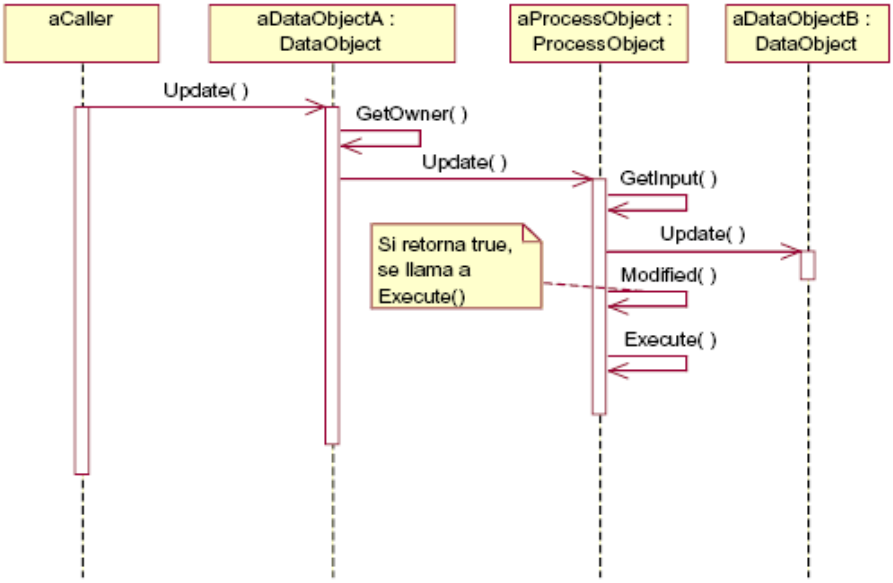


Figura 5. 12 Diagrama de secuencia de los llamados Update() y Execute() para la ejecución implícita. Este modelo es el implementado en la aplicación.

Ejecución implícita

En cambio, en la ejecución implícita, un *process object* se ejecuta solamente si su entrada o alguno de sus parámetros cambió. El control implícito es implementado mediante un proceso de 2 pasadas (ver diagrama de secuencia en la figura 5.12).

Primero, cuando se solicita actualizar la salida para un objeto particular, este objeto solicita actualizar sus entradas. De la misma manera, este proceso es repetido recursivamente hasta que se encuentran los *source objects*, los cuales ejecutan si cambiaron. Luego, a la vuelta de la recursión cada *process object* examina sus entradas para ver si han cambiado y determina de esta manera si ejecuta o no. Este proceso continúa hasta que el objeto que inició el pedido de actualizar, ejecuta y termina así con el proceso. En particular, la clase abstracta ProcessObject tiene a los métodos abstractos Update() y Execute() para llevar a cabo el proceso descripto anteriormente. El modelo seleccionado para la arquitectura de la aplicación, es el de ejecución implícita.

5.3.4. Diagrama del Pipeline

Se diseñó un pipeline (ver figura 5.13) que tiene en cuenta los siguientes requerimientos funcionales:

- Cargar imágenes (volúmenes) que representan las tomografías/resonancias.
- Seleccionar una imagen.
- Alterar la imagen seleccionada mediante la aplicación de filtros Imagen- Imagen (marca (1) en la figura). Un ejemplo de esto, es pasar un filtrado anisotrópico[21] seguido de un suavizado gausiano[22].
- Visualizar los planos XY, XZ, YZ de la imagen seleccionada.
- Modificar la gama de colores con que se visualiza la imagen seleccionada (marca (2) en la figura).
- Visualizar información adicional de la imagen seleccionada (marca (3) en la figura).
- Seleccionar un método de segmentación y configurar sus parámetros.
- Segmentar la imagen seleccionada con el método de segmentación seleccionado.

[21] Este filtro es un suavizado que promedia el valor de un voxel con el de sus vecinos si el gradiente entre ellos es menor que un delta determinado. Este permite definir mejor los bordes en una imagen homogeneizando zonas comunes.

[22] Suavizado de la imagen utilizando una función de Gauss

- Guardar una imagen procesada.
- Cargar superficies.
- Seleccionar un método de reconstrucción de superficies, con algún filtro Imagen-PolyData, como ser Marching Cubes.
- Reconstruir una superficie desde una imagen de acuerdo al método de reconstrucción seleccionado.
- Seleccionar una superficie.
- Alterar la superficie seleccionada mediante la aplicación de filtros PolyData-PolyData (marca (4) en la figura). Un ejemplo de esto, es suavizar la imagen y luego reducir la malla mediante una técnica de decimación.
- Guardar una superficie como información poligonal.
- Visualizar información adicional de la superficie seleccionada (marca (5) en la figura).

Este pipeline general, es estático, ya que no permite alterar las conexiones entre los filtros, ni agregar/eliminar filtros. Pero para darle versatilidad a la aplicación, se diseñó un esquema de *mini-pipelines* (elipses en la figura 5.13) en los que se puede agregar dinámicamente una serie de filtros programados en lenguaje TCL. Esto se trata con más detalle en la sección 5.6.

Como se puede ver en el ejemplo de la figura 5.14, los puntos de entrada y de salida a estos *mini-pipelines* se mantienen constantes, por lo que es posible incorporarlos dentro del pipeline estático, pero internamente, se puede agregar y sacar filtros (siempre respetando la consistencia entre los tipos de entrada/salida para conectar a los filtro) que en combinación con el lenguaje interpretado TCL hace a la flexibilidad de la aplicación.

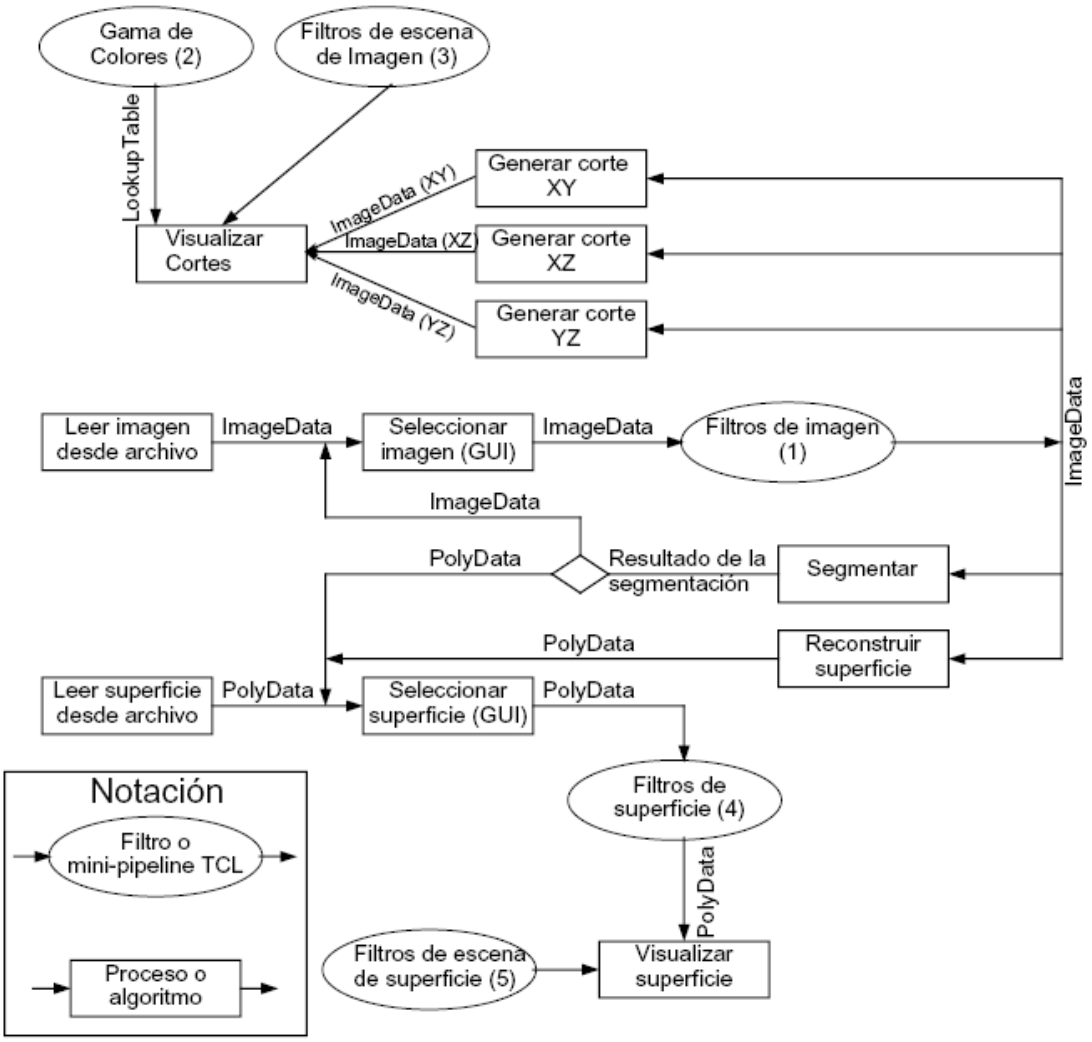


Figura 5. 13 Pipeline implementado en la aplicación

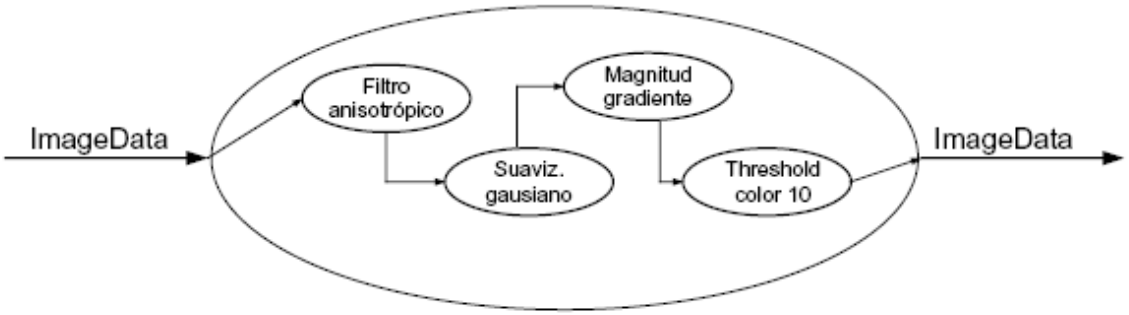


Figura 5. 14 Ejemplo de un mini pipeline para el mejoramiento de la imagen

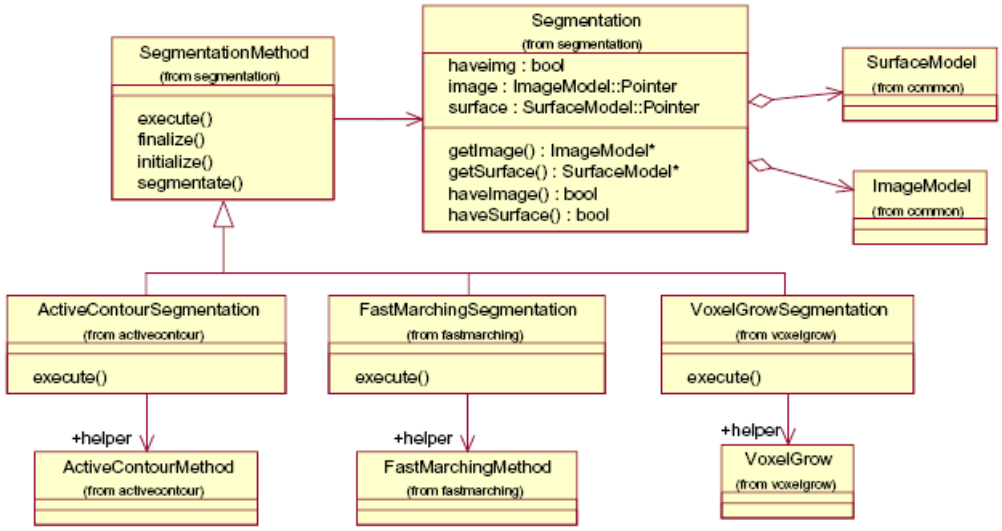


Figura 5. 15 Diagrama de clases de la abstracción de la segmentación.

5.4. Abstracción de la Segmentación

Con el objetivo de unificar a los métodos de segmentación, se realizó una abstracción del proceso, de manera que la aplicación y el usuario ven de manera uniforme a todos los métodos, además de permitir agregar en el futuro otros métodos de segmentación.

El diagrama de clases de la Figura 5.15 se ve dicha abstracción. La clase principal es SegmentationMethod y el método execute() dentro de la misma. El punto de entrada para

el algoritmo de segmentación es el método segmentate() el cual hace un llamado a los métodos initialize(), execute() y finalize() siguiendo este orden. Como entrada, se provee una instancia de la clase ImageModel que representa la imagen que se desea segmentar. El resultado de la segmentación se almacena en una instancia de SegmentationModel que provee métodos para indicar si la segmentación produjo una imagen como resultado, una superficie (por ejemplo el método ActiveContourSegmentation) o ambas.

5.4.1. Segmentación por Voxel Grow

Para el caso especial de la segmentación por el método VoxelGrow (ver sección 5.2) se diseñó una jerarquía de clases que permita armar configuraciones flexibles para el algoritmo. El punto de configuración viene dado en la parte de aceptación de un voxel vecino, y para ello, se generalizó la condición de decisión por medio de la clase abstracta VoxelProcessor utilizando el patrón *Strategy*. En la Figura 5.16 se puede ver el diagrama de clases.

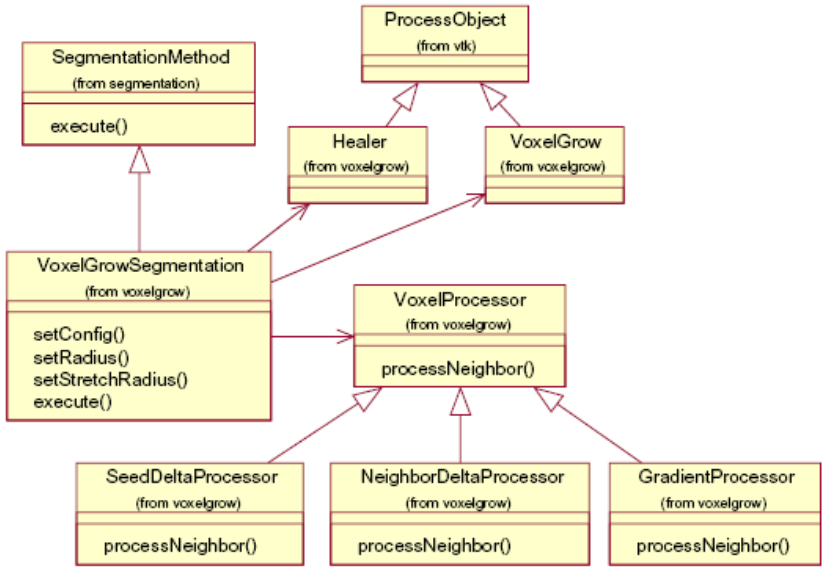


Figura 5. 16 Diagrama de clases del método de segmentación VoxelGrow.

5.5. Diagrama general de clases

En la figura 5.17 se puede ver el diagrama general de clases. Se omitieron las clases relacionadas a la interfaz del usuario (salvo DesktopGUI que es la única que se

muestra y por el solo hecho de ser la que contiene la referencia al pipeline general), y las clases que forman parte de abstracción de segmentación de imágenes, ya que hay un diagrama especial con su explicación en la sección 5.4.

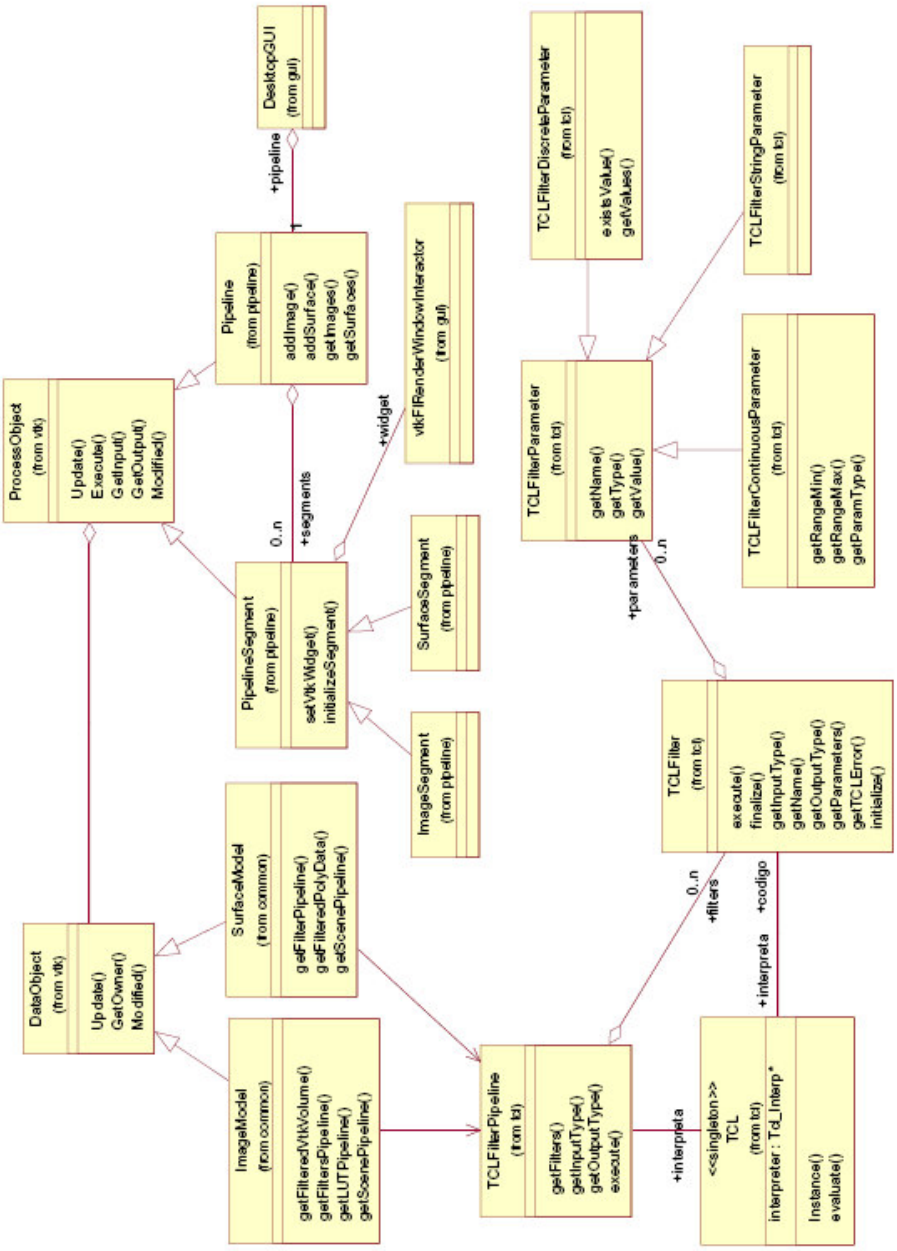


Figura 5. 17 Diagrama general de clases

6. COMPARACIONES Y RESULTADOS

En este capítulo se tiene por objetivo tomar casos de prueba para evaluar el desempeño de los métodos de segmentación y reconstrucción de superficies implementados.

Los datos de prueba están clasificados de acuerdo al origen. Para la etapa de pruebas se tomaron imágenes de tomografía obtenidos de la Librería Nacional de Medicina de Estados unidos (NLM)[23], en cuyo dominio poseen una base de datos de tomografías, imágenes de Resonancia Magnética y crioseccionamiento de un ser humano, proyecto que lleva por nombre VHP (*Visible Human Project*)[24].

Las imágenes obtenidas a partir de MRI destacan mejor los tejidos blandos (cerebro, tejido conectivo, etc.) mientras que en las TC se pueden observar mejor los tejidos como el hueso, haciéndola mas apropiada para extraer tanto el hueso como la piel.

6.1. Algoritmos

En el proceso de comparación existen tres etapas para llegar de la imagen cruda (original) a la superficie final. Segmentación. En las pruebas se utilizarán dos diferentes algoritmos de segmentación:

Tipo de estudio	Caso	Dimensiones (X x Y x Z)	Intensidades	Tejido a segmentar
Esfera	1	50 x 50 x 50	1500	-
MRI	2	256 x 256 x 124	833	Cerebro
MRI	3	256 x 256 x 124	833	Tumor
CT	4	256 x 256 x 125	256	Hueso
CT	5	256 x 256 x 125	256	Piel

Tabla 6. 1 Datos de prueba

Estos son los métodos de segmentación, reconstrucción de superficie y mejorado de superficie. En algunos casos también se utilizaron filtros especiales sobre las imágenes para mejorar su calidad (Suavizado de Gauss, Suavizado Anisotrópico, Thresholding, etc.).

[23] NLM http://www.nlm.nih.gov/
[24] VHP http://www.nlm.nih.gov/research/visible/visible_human.html

6.2. Datos

Para realizar las pruebas, se tomaron como datos de origen las imágenes referidas en la Tabla 6.1.

6.3. Métricas

Para realizar la evaluación se tiene en cuenta las siguientes métricas:

- **Tiempo:** se toma en cuenta el tiempo de la segmentación y de la reconstrucción tridimensional, incluyendo los tiempos de ejecución de filtros de pre y post procesado si es que se aplicó alguno.
- **Calidad de la superficie:** se tiene en cuenta la calidad de la triangulación. Esto incluye la cantidad de triángulos y un indicador mínimo y medio de la métrica NAR[25](ver sección B.3). La razón por la cual se utiliza esta métrica es que por la forma en que las computadoras realizan la graficación, con triángulos que se aproximan a equiláteros se logran mejores resultados de renderización.

6.4. Configuración de Hardware

Las pruebas se realizaron sobre un equipo con las siguientes características de hardware:

- AMD Athlon 1.8 Mhz
- 512 Mb RAM
- Windows XP
- Video GeForce MX 5200 128Mb RAM

6.5. Casos de prueba

6.5.1. Caso 1

Para el caso 1, la imagen que se toma como origen es una esfera (Figura 6.1). En el caso particular del caso 1c (cuadro 6.4 e imagen (c) de la Figura 6.2) el inflado por gradiente da

[25] Normalised Aspect Ratio.

como resultado una imagen de no mucha calidad debido a que el inflado produce que se inviertan los triángulos de la imagen.

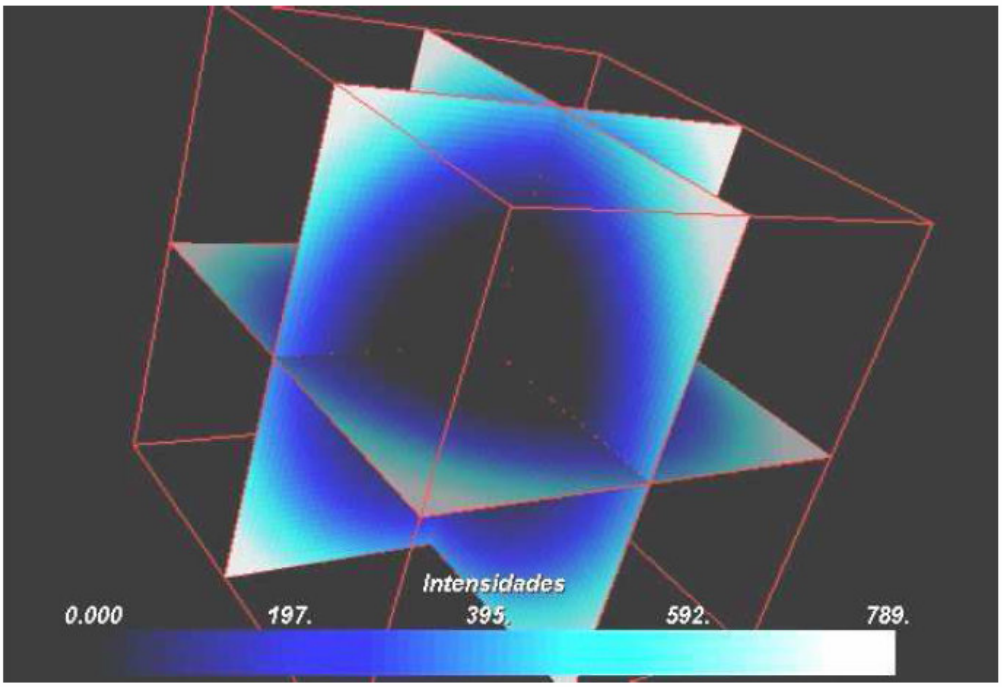


Figura 6. 1 Imagen de origen del caso 1.

Caso	1a
Tejido	Ninguno. Se trata de una imagen con fines de prueba.
Filtros pre-procesado imagen	GradientMagnitude
Método segmentación	Marching Cubes. Color corte: 7
Configuración método segmentación	-
Filtros post-procesado imagen	-
Método reconstrucción	Marching Cubes.
Filtros post-procesado superficie	-

Tabla 6. 2 Caso 1a

Caso	1b			
Tejido	Ninguno. Se trata de una imagen con fines de prueba.			
Filtros pre-procesado imagen	-			
Método segmentación	Voxel Grow			
Configuración método segmentación	Semilla 1	X:25, Y:25, Z:25		
	Radio	2		
	Radio estiramiento	7		
	Método aceptación	Gradiente		
	Umbral	7		
Filtros post-procesado imagen	-			
Método reconstrucción	Flat Contour. Color afuera: 0			
Filtros post-procesado superficie	Suavizado Taubin	PassBand:	0.1	
		Iteraciones	500	

Tabla 6. 3 Caso 1b

Caso	1c			
Tejido	Ninguno. Se trata de una imagen con fines de prueba.			
Filtros pre-procesado imagen	-			
Método segmentación	Voxel Grow			
Configuración método segmentación	Semilla 1	X:25, Y:25, Z:25		
	Radio	2		
	Radio estiramiento	7		
	Método aceptación	Gradiente		
Filtros post-procesado imagen	-			
Método reconstrucción	Flat Contour. Color afuera: 0			
Filtros post-procesado superficie	Inflado por gradiente	Límite gradiente:	20	
		Iteraciones	100	
		Distancia media	0.1	

Tabla 6. 4 Caso 1c

Caso	1d
Tejido	Ninguno. Se trata de una imagen con fines de prueba.
Filtros pre-procesado imagen	-
Método segmentación	Voxel Grow
Configuración método segmentación	Semilla 1: X:25, Y:25, Z:25 Radio: 2 Radio estiramiento: 7 Método aceptación: Gradiente Umbral: 7
Filtros post-procesado imagen	Suavizado Gaussiano: Desvío 1.0, Radio 1.0
Método reconstrucción	Marching Cubes. Color corte 1.2
Filtros post-procesado superficie	-

Tabla 6. 5 Caso 1d

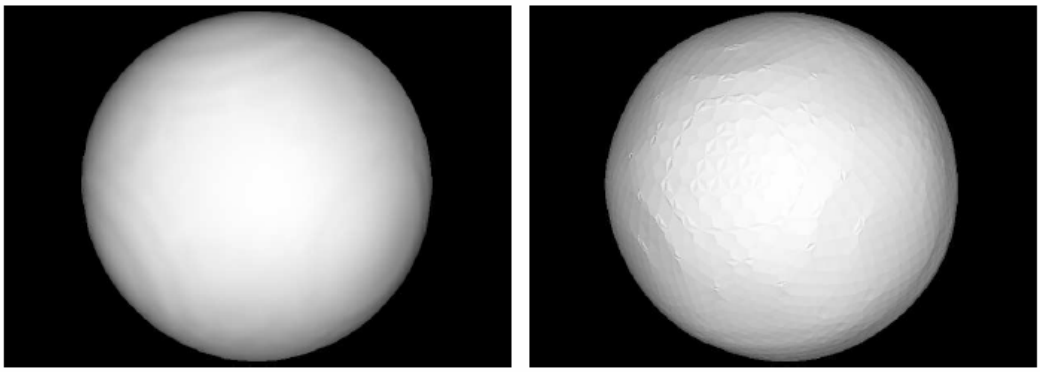
Caso	1e
Tejido	Ninguno. Se trata de una imagen con fines de prueba.
Filtros pre-procesado imagen	-
Método segmentación	Level Sets - Fast Marching
Configuración método segmentación	Semilla 1: X:25, Y:25, Z:25 Sigma: 1.0
Filtros post-procesado imagen	-
Método reconstrucción	Marching Cubes. Color corte: 20
Filtros post-procesado superficie	-

Tabla 6. 6 Caso 1e

El caso que fue suavizado con el método de inflado por gradiente arrojó resultados de mala calidad. La estructura de la malla y la representación gráfica resultaron ser malas, por esta razón este método no es utilizado en los casos posteriores. Este método produce inconsistencias en la malla.

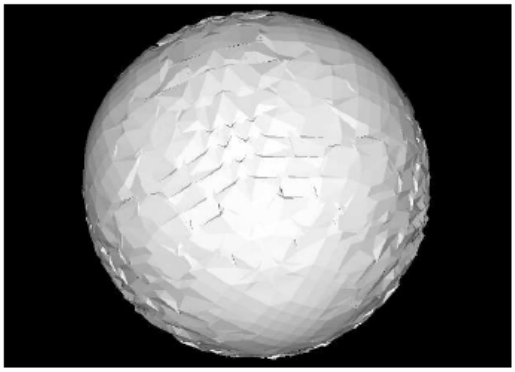
6.5.2. Caso 2

El objetivo de este caso es segmentar el tejido del cerebro. En la figura 5.3 se ve la imagen que es usada (con los cortes XY, XZ e YZ combinados) como origen para realizar la segmentación, donde se pueden observar el cerebro y el tumor.

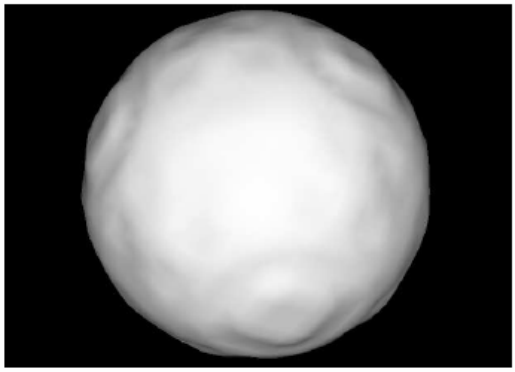


(a) Caso 1a

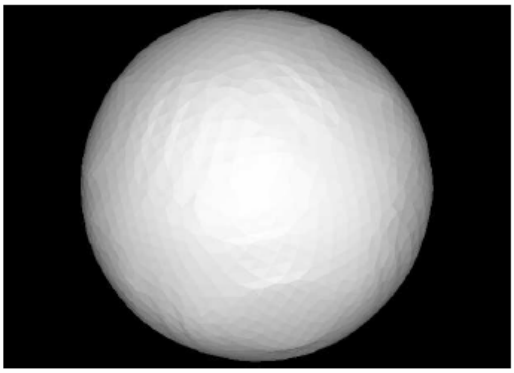
(b) Caso 1b



(c) Caso 1c



(d) Caso 1d



(e) Caso 1e

Figura 6. 2 Imágenes generadas para el caso 1

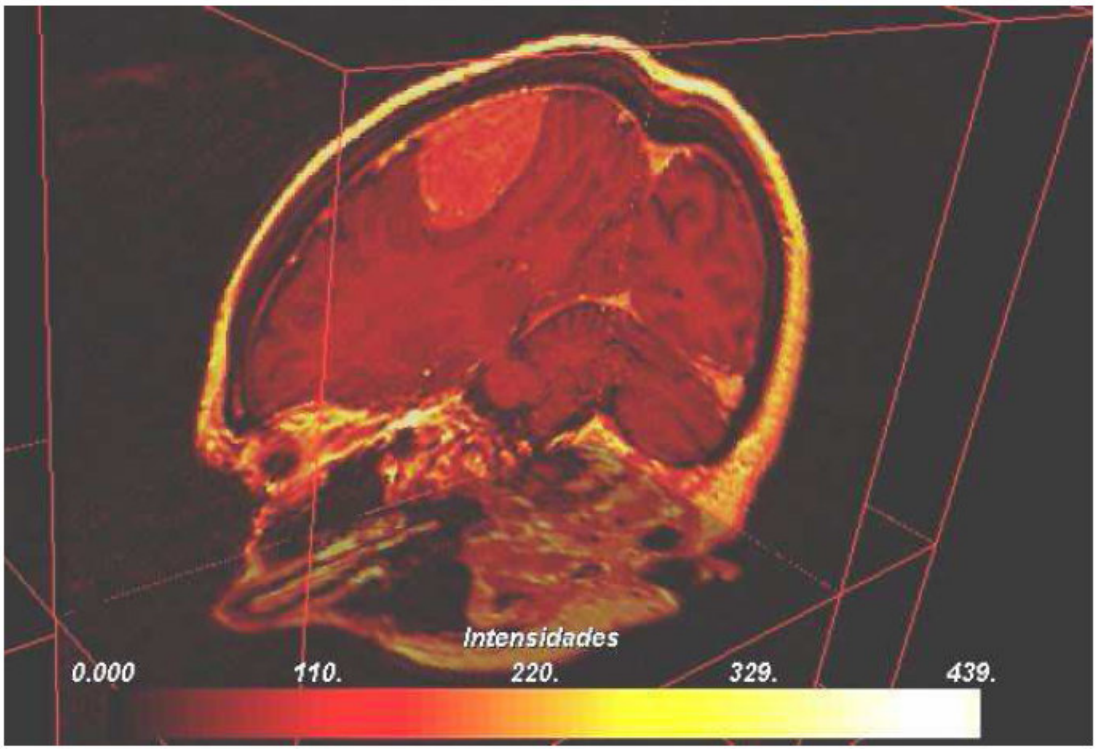


Figura 6. 3 Imagen de origen del caso 2

Caso	2a		
Tejido	Cerebro		
Filtros pre-procesado imagen	Suavizado Anisotrópico	Radio difusión	1.1
		Gradiente límite	10
		Iteraciones	6
Método segmentación	Voxel Grow		
Configuración método segmentación	Semilla 1	X:91, Y: 144, Z: 90	
		Método: Gradiente	
		Umbral: 15	
	Radio	2	
	Radio Estiramiento	2	
Filtros post-procesado imagen	Suavizado Gauss	Desvío	1.1
		Radio	1.1
Método reconstrucción	Marching Cubes. Color corte: 132		
Filtros post-procesado superficie	-		

Tabla 6. 7 Caso 2a

Caso	2b		
Tejido	Cerebro		
Filtros pre-procesado imagen	Suavizado Anisotrópico	Radio difusión	1.1
		Gradiente límite	10
		Iteraciones	6
Método segmentación	Voxel Grow		
Configuración método segmentación	Semilla 1	X:91, Y: 144, Z: 90	
		Método: Gradiente	
		Umbral: 15	
	Radio	2	
	Radio Estiramiento	2	
Filtros post-procesado imagen	-		
Método reconstrucción	Flat Contour. Color afuera: 0		
Filtros post-procesado superficie	Suavizado Taubin	PassBand:	0.1
		Iteraciones	100

Tabla 6. 8 Caso 2b

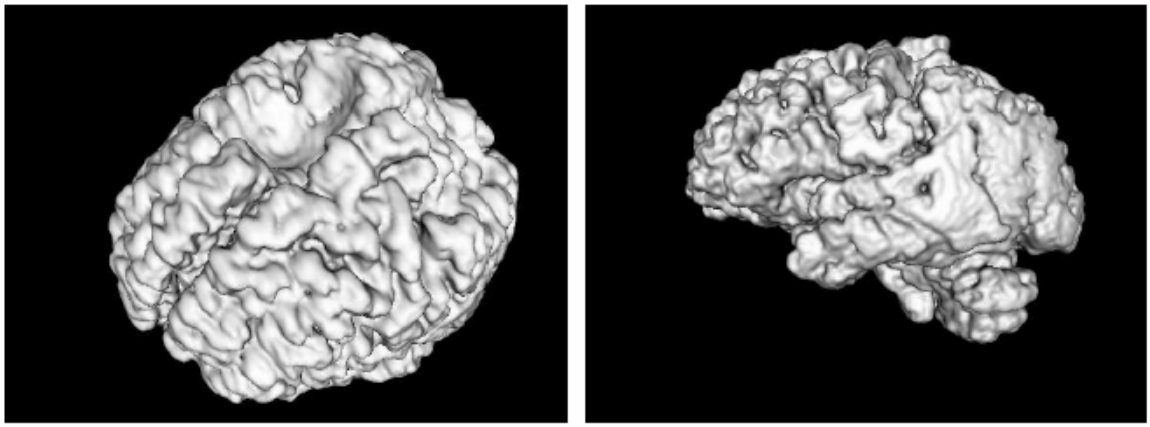
Caso	2c		
Tejido	Cerebro		
Filtros pre-procesado imagen	Suavizado Anisotrópico	Radio difusión	1.1
		Gradiente límite	10
		Iteraciones	6
Método segmentación	Fast Marching		
Configuración método segmentación	Semilla 1	X:91, Y: 144, Z: 90	
	Sigma	1.2	
Filtros post-procesado imagen	Threshold	Color	600
		Modo	ByLower
	Suavizado Gauss	Desvío	1.1
		Radio	1.1
Método reconstrucción	Marching Cubes. Color corte: 29		
Filtros post-procesado superficie	-		

Tabla 6. 9 Caso 2c

Los resultados de la segmentación del cerebro son bastante buenos teniendo en cuenta que es una estructura con una topología muy compleja. Tal vez el mejor resultado sea el de Level Sets, este recuperó la mayor parte del tejido.

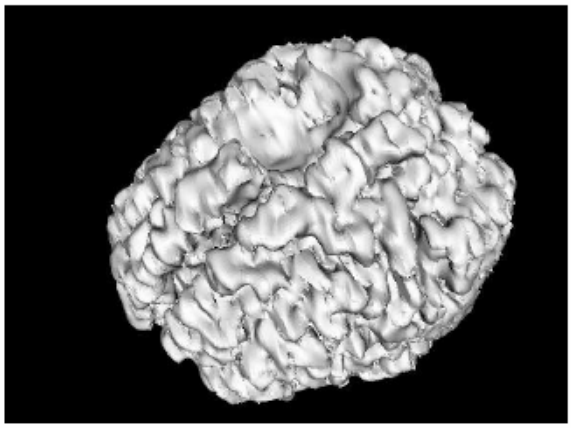
6.5.3. Caso 3

El objetivo de este caso es segmentar el tejido del tumor. En la figura 6.3 se ve la imagen que es usada (con los cortes XY, XZ e YZ combinados) como origen para realizar la segmentación, donde se pueden observar el cerebro y el tumor.

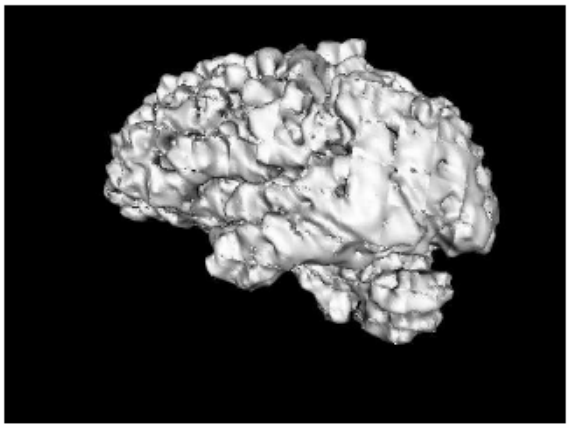


(a) Caso 2a. Vista 1

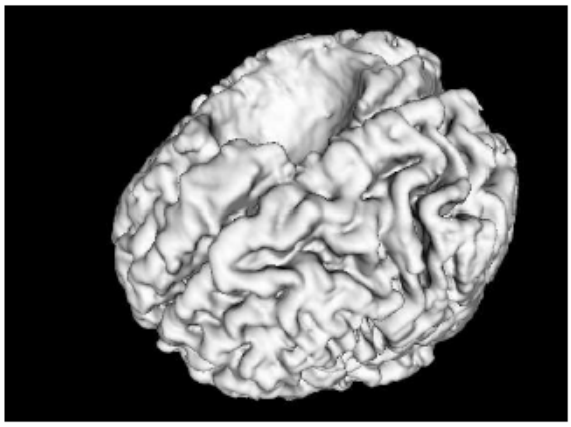
(b) Caso 2a. Vista 2



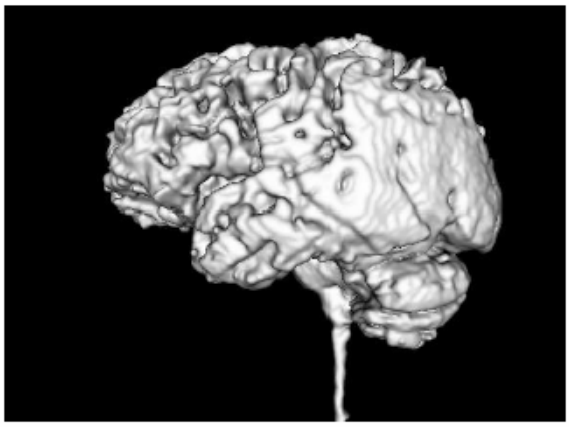
(c) Caso 2b. Vista 1



(d) Caso 2b. Vista 2



(e) Caso 2c. Vista 1



(f) Caso 2c. Vista 2

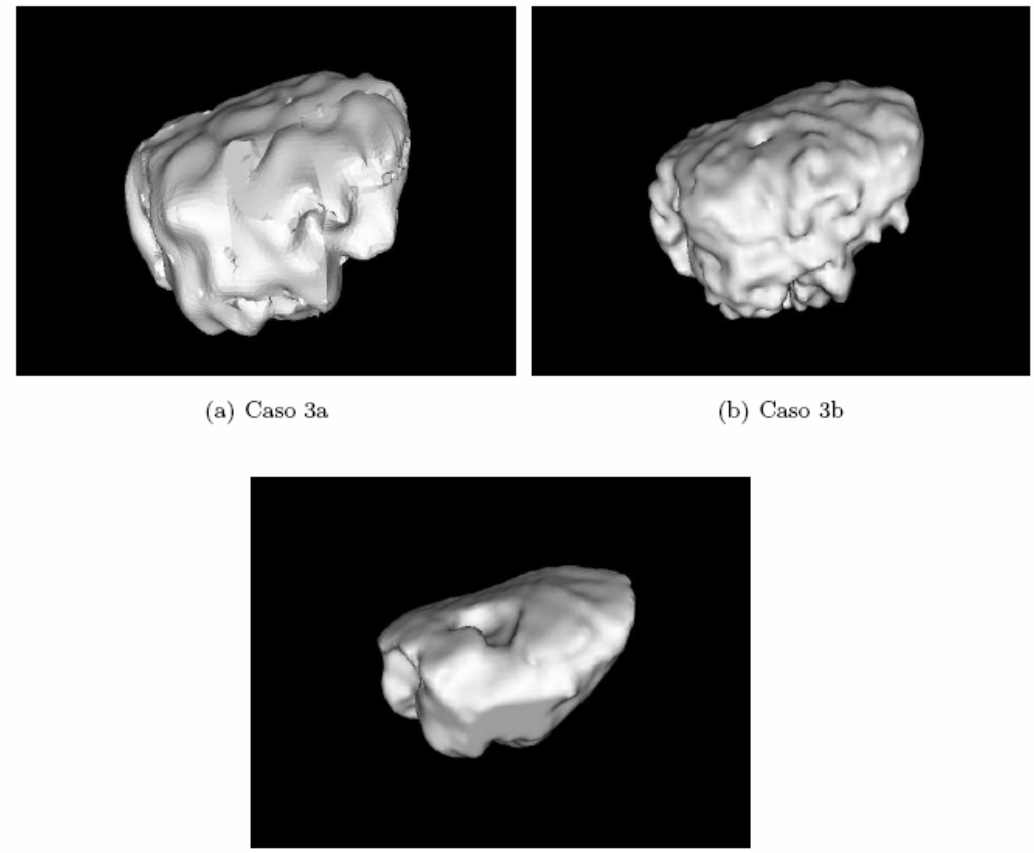
Figura 6. 4 Imágenes generadas para el caso 2

Caso	3a		
Tejido	Tumor		
Filtros pre-procesado imagen	Suavizado Anisotrópico	Radio difusión	1.1
		Gradiente límite	10
		Iteraciones	6
Método segmentación	Voxel Grow		
Configuración método segmentación	Semilla 1	X:123, Y: 176, Z: 50	
		Método: Gradiente	
		Umbral: 20	
	Radio	2	
	Radio Estiramiento	4	
Filtros post-procesado imagen	Suavizado Gauss	Desvío	1.1
		Radio	1.1
Método reconstrucción	Marching Cubes. Color corte: 105		
Filtros post-procesado superficie	-		

Tabla 6. 10 Caso 3a

Caso	3b		
Tejido	Tumor		
Filtros pre-procesado imagen	Suavizado Anisotrópico	Radio difusión	1.1
		Gradiente límite	10
		Iteraciones	6
Método segmentación	Voxel Grow		
Configuración método segmentación	Semilla 1	X:123, Y: 176, Z: 50	
		Método: Gradiente	
		Umbral: 20	
	Radio	2	
	Radio Estiramiento	4	
Filtros post-procesado imagen	-		
Método reconstrucción	Flat Contour. Color afuera: 0		
Filtros post-procesado superficie	Suavizado Taubin	PassBand:	0.1
		Iteraciones	100

Tabla 6. 11 Caso 3b



(a) Caso 3a

(b) Caso 3b

(c) Caso 3c

Figura 6. 5 Imágenes generadas para el caso 3

Caso	3c
Tejido	Tumor
Filtros pre-procesado imagen	Suavizado Anisotrópico: Radio difusión 1.1; Gradiente límite 10; Iteraciones 6
Método segmentación	Fast Marching
Configuración método segmentación	Semilla 1: X:123, Y: 186, Z: 50; Sigma 1.5
Filtros post-procesado imagen	Threshold: Color 600, Modo ByLower; Suavizado Gauss: Desvío 1.1, Radio 1.1
Método reconstrucción	Marching Cubes. Color corte: 161
Filtros post-procesado superficie	-

Tabla 6. 12 Caso 3c

En este caso los tres resultados son buenos, pero existen diferencias entre ellos. El método de Level Sets tuvo menor interferencia del ruido, produciendo un resultado más prolijo.

6.5.4. Caso 4

El origen es una tomografía computada (figura 5.6), y el objetivo es segmentar y visualizar el Hueso.

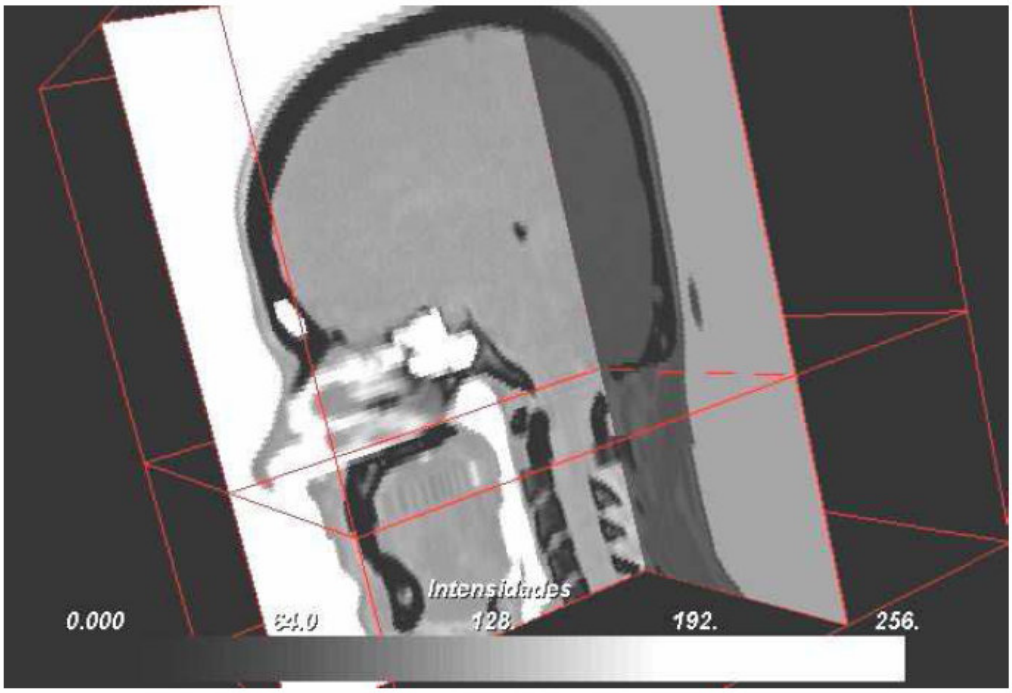


Figura 6. 6 Imagen de origen para el caso 4

Caso	4a		
Tejido	Hueso		
Filtros pre-procesado imagen	-		
Método segmentación	Voxel Grow		
Configuración método segmentación	Semilla 1	X:128, Y: 214, Z: 32	
		Método: semilla	
		Umbral: 4	
	Semilla 2	X:132, Y: 207, Z: 93	
		Método: semilla	
		Umbral: 4	
	Semilla 2	X:113, Y: 14, Z: 98	
		Método: semilla	
		Umbral: 4	
	Radio	0	
	Radio Estiramiento	0	
Filtros post-procesado imagen	-		
Método reconstrucción	Flat Contour. Color afuera: 0		
Filtros post-procesado superficie	Suavizado Taubin	PassBand:	0.1
		Iteraciones	100

Tabla 6. 13 Caso 4a

Caso	4b		
Tejido	Hueso		
Filtros pre-procesado imagen	-		
Método segmentación	Voxel Grow		
Configuración método segmentación	Semilla 1	X:128, Y: 214, Z: 32	
		Método: semilla	
		Umbral: 4	
	Semilla 2	X:132, Y: 207, Z: 93	
		Método: semilla	
		Umbral: 4	
	Semilla 2	X:113, Y: 174, Z: 98	
		Método: semilla	
		Umbral: 4	
	Radio	0	
	Radio Estiramiento	0	
Filtros post-procesado imagen	Suavizado Gauss	Desvío	1.0
		Radio	1.2
Método reconstrucción	Marching Cubes. Color corte: 0.55		
Filtros post-procesado superficie	-		

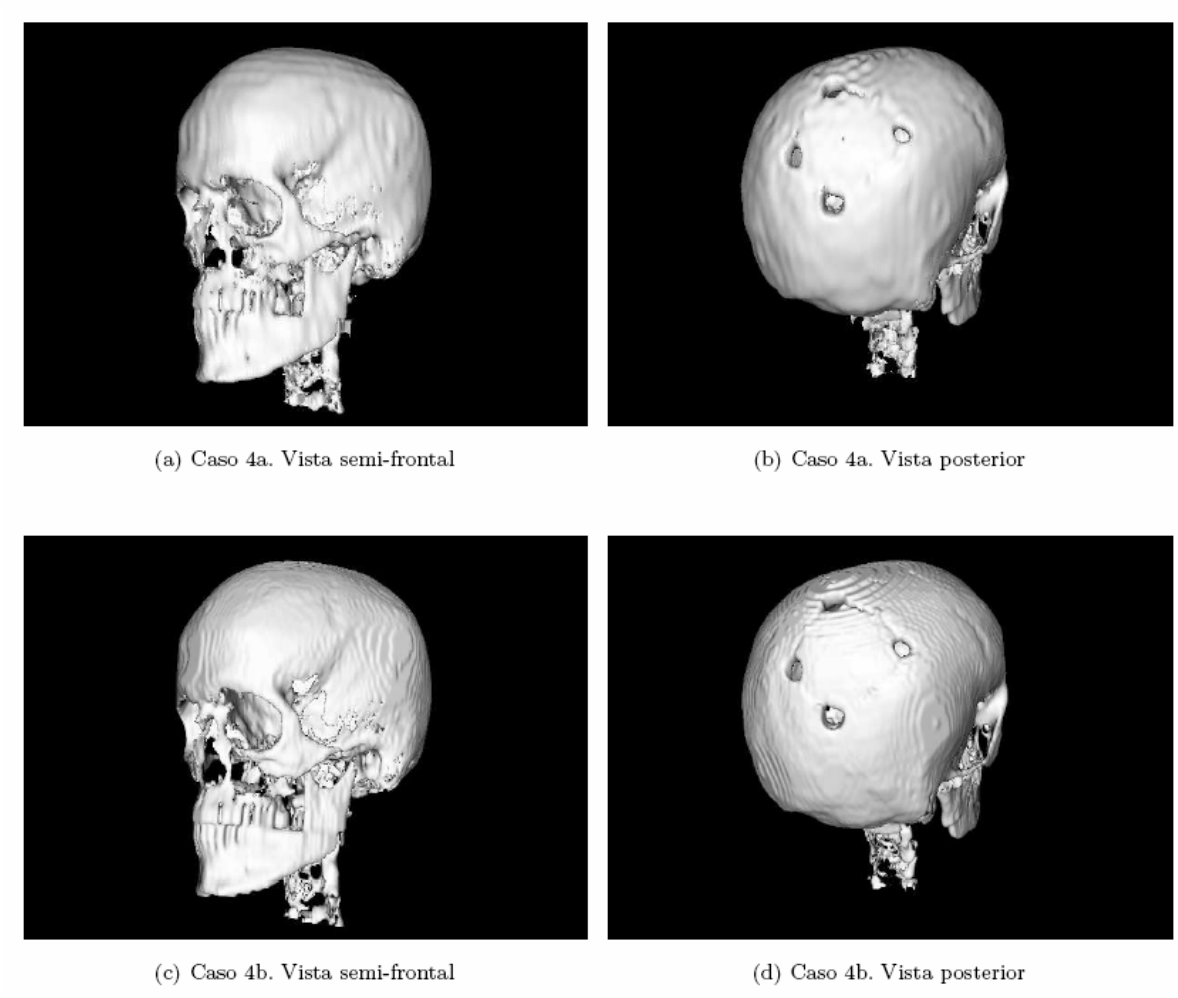
Tabla 6. 14 Caso 4b

Caso	4c
Tejido	Hueso
Filtros pre-procesado imagen	-
Método segmentación	Marching Cubes, Color de corte: 4.0
Configuración método segmentación	-
Filtros post-procesado imagen	-
Método reconstrucción	-
Filtros post-procesado superficie	-

Tabla 6. 15 Caso 4c

Caso	4d		
Tejido	Hueso		
Filtros pre-procesado imagen	-		
Método segmentación	Marching Cubes, Color de corte: 4.0		
Configuración método segmentación	-		
Filtros post-procesado imagen	-		
Método reconstrucción	-		
Filtros post-procesado superficie	Suavizado Taubin	PassBand:	0.1
		Iteraciones	100

Tabla 6. 16 Caso 4d



(a) Caso 4a. Vista semi-frontal

(b) Caso 4a. Vista posterior

(c) Caso 4b. Vista semi-frontal

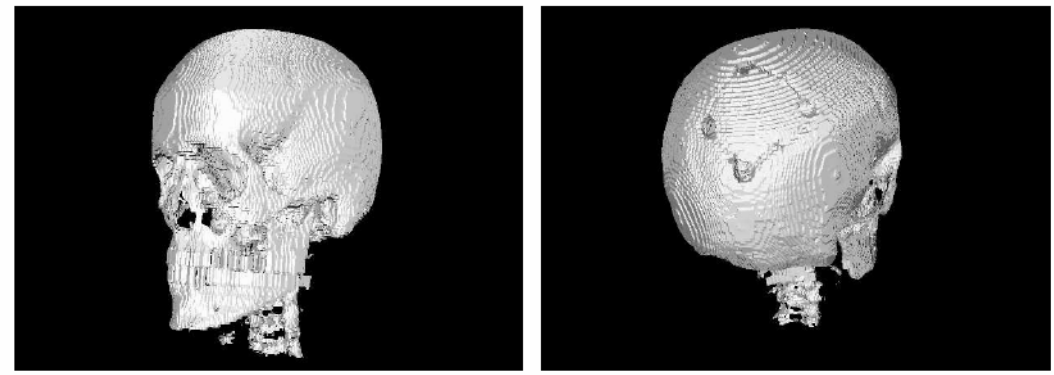
(d) Caso 4b. Vista posterior

Figura 6. 7 Imágenes generadas los casos 4a y 4b

En este caso del hueso, utilizar Level sets no dio buenos resultados. Las segmentaciones obtenidas solo recuperaban una parte del tejido, en ningún caso se pudo obtener la calavera completa. Por medio de Voxel Grow se obtuvo un buen resultado y una reconstrucción completa del cráneo. Otro buen resultado se obtuvo al aplicar Marching Cubes directamente sobre la Imagen original, pero este resultado no necesariamente se corresponde con el tejido original, ya que Marching Cubes no tiene en cuenta peque˜nas variaciones que pueden delimitar las región (Obtiene una isosuperfice).

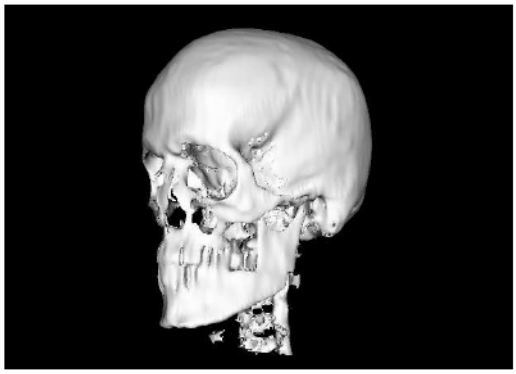
6.5.5. Caso 5

El origen es una tomografía computada (figura 6.6), el objetivo es segmentar y visualizar la piel.

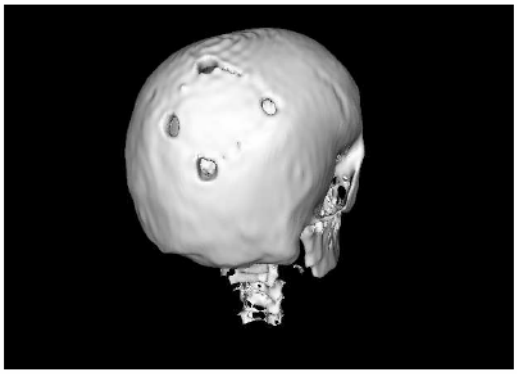


(a) Caso 4c. Vista semi-frontal

(b) Caso 4c. Vista posterior



(c) Caso 4d. Vista semi-frontal



(d) Caso 4d. Vista posterior

Figura 6. 8 Imágenes generadas para los casos 4c y 4d

Caso	5a		
Tejido	Piel		
Filtros pre-procesado imagen	-		
Método segmentación	Voxel Grow		
Configuración método segmentación	Semilla 1^2	X:0, Y:0, Z: 0 Método: semilla Umbral: 5	
	Radio	2	
	Radio Estiramiento	0	
Filtros post-procesado imagen	-		
Método reconstrucción	Flat Contour. Color afuera: 1		
Filtros post-procesado superficie	Suavizado Taubin	PassBand:	0.1
		Iteraciones	100

Tabla 6. 17 Caso 5a

Caso	5b
Tejido	Piel
Filtros pre-procesado imagen	-
Método segmentación	Voxel Grow
Configuración método segmentación[3]	Semilla 1: X:0, Y:0, Z: 0; Método: semilla; Umbral: 5 Radio: 2 Radio Estiramiento: 0
Filtros post-procesado imagen	Suavizado Gauss: Desvío 1.0; Radio 1.2
Método reconstrucción	Marching Cubes. Color corte: 0.68
Filtros post-procesado superficie	-

Tabla 6. 18 Caso 5b

Caso	5c
Tejido	Piel
Filtros pre-procesado imagen	-
Método segmentación	Marching Cubes, Color de corte: 190.0
Configuración método segmentación	-
Filtros post-procesado imagen	-
Método reconstrucción	-
Filtros post-procesado superficie	-

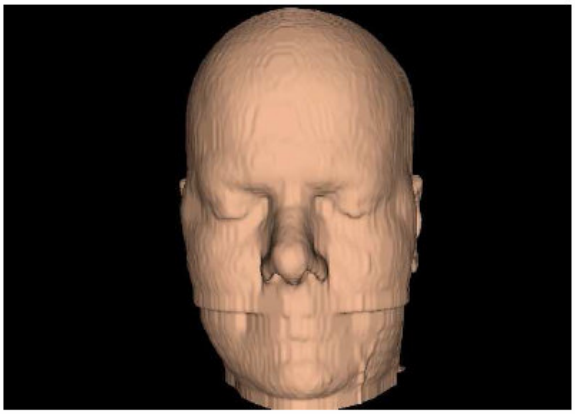
Tabla 6. 19 Caso 5c

Caso	5d
Tejido	Piel
Filtros pre-procesado imagen	-
Método segmentación	Marching Cubes, Color de corte: 190.0
Configuración método segmentación	-
Filtros post-procesado imagen	-
Método reconstrucción	-
Filtros post-procesado superficie	Suavizado Taubin: PassBand: 0.1; Iteraciones 100

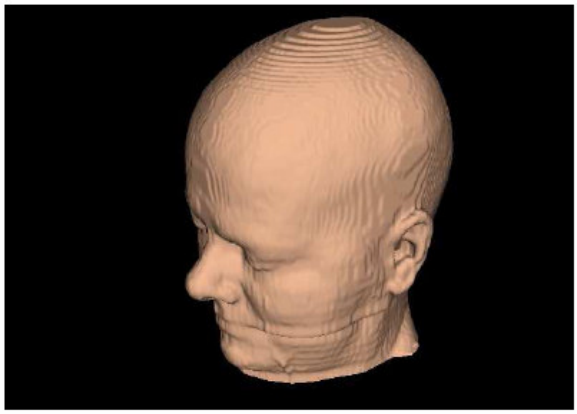
Tabla 6. 20 Caso 5d

(a) Caso 5a. Frente

(b) Caso 5a. Lateral

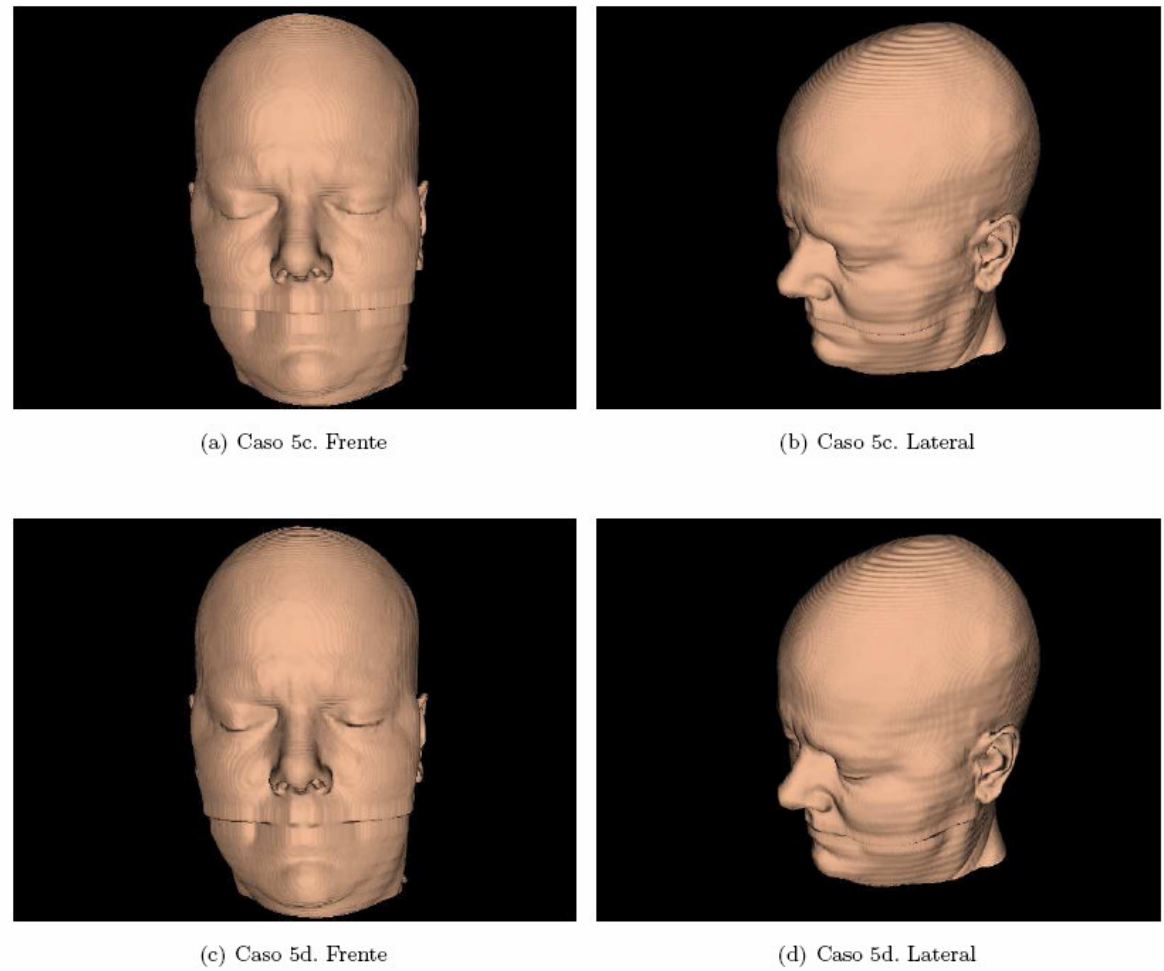


(c) Caso 5b. Frente



(d) Caso 5b. Lateral

Figura 6. 9 Imágenes generadas para los casos 5a y 5b



(a) Caso 5c. Frente (b) Caso 5c. Lateral

(c) Caso 5d. Frente (d) Caso 5d. Lateral

Figura 6. 10 Imágenes generadas para los casos 5c y 5d

La visualización de la piel es fundamental para poder ubicar espacialmente (dentro del individuo) el lugar exacto en donde se encuentra una estructura o tejido. En el caso de la tomografía es fácil encontrar el límite de la piel, ya que se puede segmentar el aire que rodea al paciente. En este tipo de imágenes (CT) el límite entre el tejido y el aire se encuentra marcado muy nítidamente. Para obtenerlo fácil se colocó una semilla en el exterior de la imagen y luego se creció en esa zona.

6.6. Resultados

En las figuras 6.11, 6.12 y 6.13 se pueden ver imágenes combinadas de los diferentes tejidos de las segmentaciones realizadas en los casos de prueba.

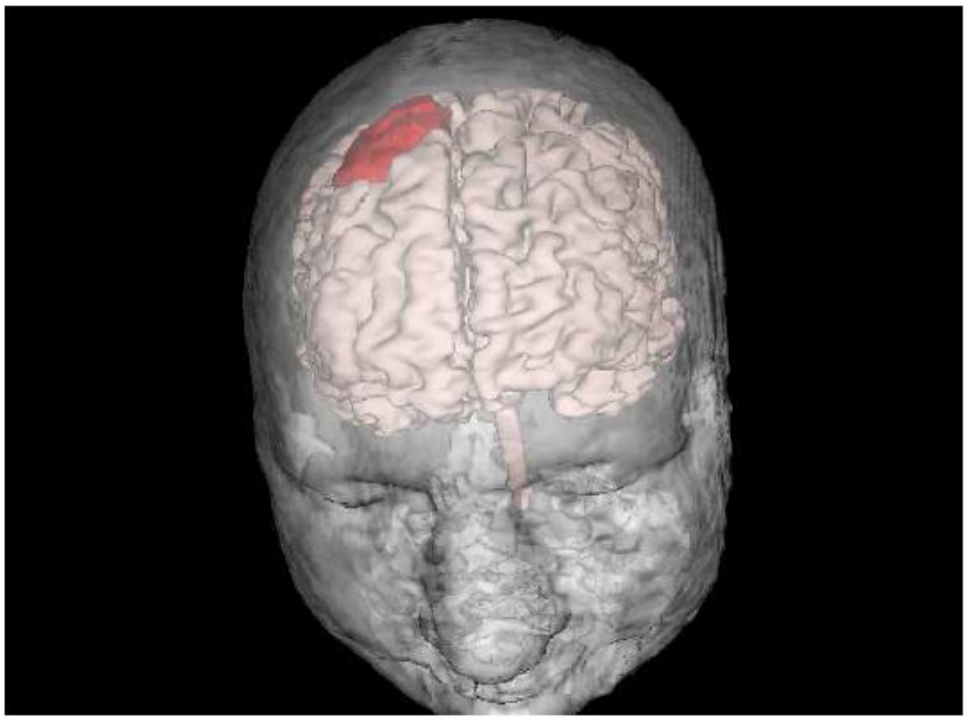


Figura 6. 11 Combinación de tumor, cerebro y piel. Vista de frente

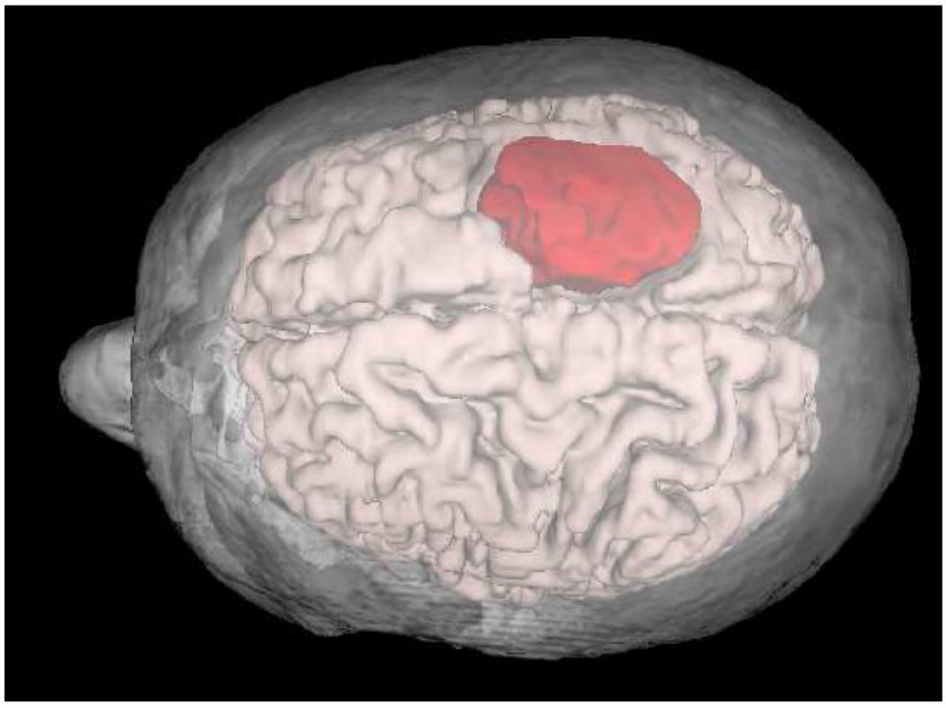


Figura 6. 12 Combinación de tumor, cerebro y piel. Vista de arriba

En estas dos imágenes puede verse con claridad la estructura del cerebro y el tumor dentro del paciente. La superficie de la piel no es de la mejor calidad ya que es obtenida a partir de una tomografía que posee mucho ruido.

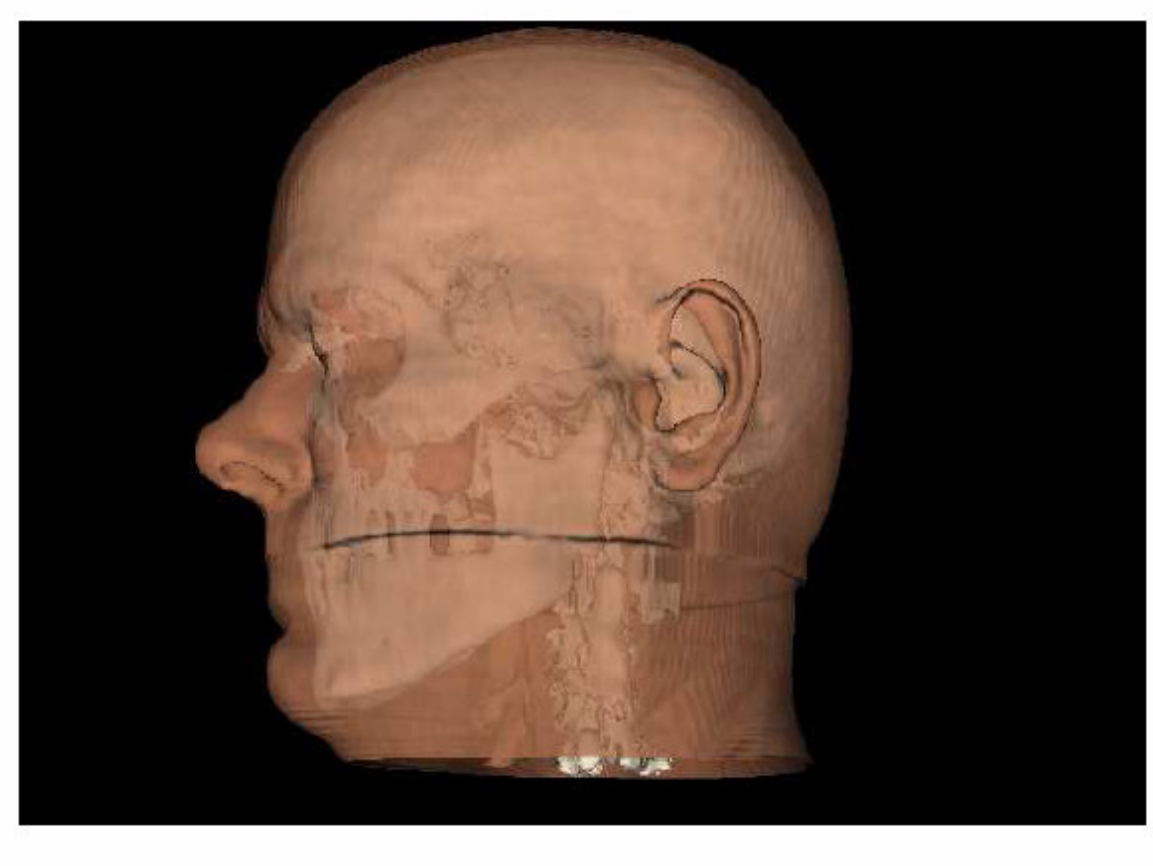


Figura 6. 13 Combinación de piel y hueso

En este caso puede apreciarse claramente el hueso del cráneo dentro de la piel de paciente. A la superficie de la piel se le aplicó una textura que la hace similar a la piel humana.

En la siguiente figura se pueden ver los valores numéricos obtenidos a partir de las pruebas anteriormente realizadas.

Caso			Tiempos (ms)		Calidad triangulación		
Tipo	Id	Descr.[a]	Seg.	Rec.	# Tri.	NAR min	NAR med
Esfera	1a	Mc	961	77	3704	0.57735	0.70257
	1b	Vg+Fc+Tb	961	3275	7020	0.15404	0.65647
	1c	Vg+Fc+Cg	961	501	7020	0.04756	0.57090
	1d	Vg+Mc	961	190	5768	0.2768	0.5874
	1e	Fm+Mc+Tb	5999	60	5864	0.16955	0.57794
MRI	2a	Vg+Mc	104671	17153	279088	0.02732	0.57235
	2b	Vg+Fc+Tb	104671	58815	396228	0.00126	0.58206
	2c	Ls+Mc	683833	14776	314362	0.0001	0.57741
MRI	3a	Vg+Mc	104671	13640	16792	0.04232	0.57094
	3b	Vg+Fc+Tb	147191	20321	22928	0.00242	0.57582
	3c	Ls+Mc	843424	10635	12720	0.03695	0.57049
CT	4a	Vg+Fc+Tb	20880	70852	520216	0.00094	0.58415
	4b	Vg+Mc	20880	11747	428680	0.05175	0.56968
	4c	Mc	0[b]	5107	539050	0.01062	0.56978
	4d	Mc+Tb	0	76029	539050	0.06236	0.55806
CT	5a	Vg+Fc+Tb	350775	48570	283112	0.00362	0.58641
	5b	Vg+Mc	350775	12609	252588	0.06272	0.55356
	5c	Mc	0[c]	4497	348653	0.00717	0.56282
	5d	Mc+Tb	0	51906	348653	0.00170	0.62693

[a]Vg: Voxel Grow; Ls: Level Sets; Sn: Snakes; Fc: Flat Contour; Mc: Marching Cubes; Tb: Suavizado Taubin;
Cg: Suavizado Crecimiento Gradiente

[b]En los casos 4c y 4d no hay tiempo de segmentación, ya que se procede directamente con MC

[c]En los casos 5c y 5d no hay tiempo de segmentación, ya que se procede directamente con MC

7. CONCLUSIONES

Luego de realizar una serie de pruebas, se observó que los resultados no eran de buena calidad debido al ruido que poseía la imagen. Para eliminarlo, luego de varias pruebas, se llego a la conclusión que un pre-filtrado Anisotrópico mejora ampliamente la calidad de las segmentaciones.

- En cuanto a los métodos de segmentación, se destaca Voxel Grow, por su simplicidad y calidad de sus resultados. Los resultados que este obtiene son comparables con métodos más complejos y elaborados como puede ser Level Sets con Fast Marching, pero con tiempos de ejecución mucho menores, en algunos casos del orden de 6 a 1.

- Se estudiaron métodos de interpolación de imágenes, entre los cuales se destacó la interpolación Bilineal por su efectividad al aplicarse a datos en todas las dimensiones deseadas, en los casos en que la resolución o cantidad de imágenes sea insuficiente.

- También se desarrolló un método de reconstrucción de superficies (Flat Contour), con el objetivo de partir de una malla con triángulos de mejor calidad (en este caso todos los triángulos tienen la misma forma), para luego suavizarla. En la práctica se pudo comprobar, luego de suavizar las superficies (con filtros de suavizado), que los triángulos se deforman mucho, y no se conserva la calidad. Estos resultados fueron comparados directamente con los obtenidos al aplicar Marching Cubes sobre la segmentación. Este último resulto ser mucho mas rápido y la triangulación buena en comparación con el caso anterior.

- Se desarrolló un entorno computacional que permite la visualización 3D tanto de superficies como de volúmenes, permitiendo al usuario una total interacción a través de acercamientos, rotación, traslación, manejo de transparencias y vista de árbol donde se asocie los diferentes órganos, de tal forma que se puedan habilitar o deshabilitar las diferentes estructuras. Esta herramienta está diseñada para ser de utilidad en la medicina, siendo una ayuda en el diagnóstico.

Apéndice A

Glosario

Isosuperfice. Superficie que representa un valor constante de una función es- calar.

Snaxels. Elemento de una Snake.

Segmentación. Procesado digital que consiste en reconocer de forma automática los objetos de una escena sin ninguna intervención previa por parte del usuario.

Histograma. Es la representación gráfica de las frecuencias relativas con las que aparecen los distintos colores en una determinada imagen.

Path. Camino que une dos puntos.

Thresholding. Filtro que separa valores que estén por debajo, por enésima o entre un determinado valor umbral.

ProcessObject. Objeto de visualización que es la abstracción de un proceso o algoritmo. Por ejemplo, el algoritmo de creación de isosuperficies Marching Cubes es implementado como un Process Object.

DataObject. Objeto que es la abstracción de datos. Por ejemplo, el archivo de un paciente en un hospital puede ser un Data Object. Los típicos objetos de visualización incluyen Structured Grids y volúmenes.

GenericProgramming. "Programar con conceptos", donde concepto se define como la familia de abstracciones que están relacionadas por un conjunto común de requerimientos.

Seed. Semilla, algunas técnicas de segmentación utilizan semillas para determi nar a partir de que punto de debe comenzar a crecer la región segmentada. Se interpreta que la semilla siempre se encuentra dentro de la región bus- cada.

Visualización. Es el proceso de explorar, transformar y ver los datos como imágenes (u otras formas sensoriales) para ganar conocimiento acerca de estos datos.

Voxel. Es un pixel con volumen. Es un punto tridimensional, tiene cualidades de altura, ancho, profundidad.

LookupTable. Tabla de búsqueda. Generalmente utilizada para realizar transformaciones. Ejemplo: valor escalar a color.

Apéndice B

Fundamentos matemáticos

B.1. Conceptos matemáticos utilizados por el método de Contornos Activos (Snakes)

B.1.1. Función de energía global

Cada punto tiene asociado un valor de energía. La formulación clásica de la función de energía global es la siguiente:

$$E_{Total}(V) = \alpha(v)E_{cont}(v)dv + \beta(v)E_{curv}(v)dv + \sigma(v)E_{long}(v)dv + \gamma(v)E_{imagen}(v)dv$$

Los valores de energía tienen que ver con la continuidad del contorno, su curvatura y su longitud. Estos son los valores de energía internos. El ultimo valor corresponde a la energía de la imagen para “emparejar” el contorno con la imagen.

B.1.2. Función de energía de contorno

El valor $\overline{d}$ indica la distancia media entre snaxels vecinos. Los puntos de la imagen donde el snaxel quedaría separado más de la media de sus vecinos más próximos, atraen menos al snaxel. La energía viene dada por el valor absoluto de la distancia media entre snaxels menos la distancia entre snaxels vecinos.

$$E_{cont} = \left|\overline{d} - \|v_i - v_{i-1}\|\right|$$

B.1.3. Función de energía de curvatura

Tratando de minimizar esta función, se intenta mantener el contorno local- mente suave. De esta manera mantenemos la restricción de forma (curvatura global).

$$E_{curv} = \left\| \frac{u_{i+1}}{\|u_{i+1}\|} - \frac{u_i}{\|u_i\|} \right\|^2$$

$$u_i = v_i - v_{i-1}$$

B.1.4. Función de energía de longitud

Esta indica cuanto difiera la longitud de la curva de la longitud de la curva buscada. Su rol es acercar la longitud del contorno a la longitud esperada al minimizar.

$$E_{long} = \left(\sum_{i=1}^{n-1} \|v_{i+1} - v_i\| - L \right)^2$$

para contornos abiertos

$$E_{long} = \left(\sum_{i=1}^{n-1} \|v_{i+1} - v_i\| \, \|v_1 - v_n\| - L \right)^2$$

para contornos cerrados.

B.2. Conceptos matemáticos utilizados por Level Sets

B.2.1. Función de propagación del frente

La ecuación de propagación del frente es la siguiente:

$$\frac{\partial C}{\partial t} = \frac{1}{\tilde{P}} \overrightarrow{n}$$

B.2.2. Función de energía presentada por Cohen-Kimmel

Considerando un modelo de energía simplificado que no tiene término de segunda derivada obtenemos la expresión:

$$\dot{E(C)} = \int_\Omega w \dot{\|C'\|}^2 + P(C)\partial s$$

Asumiendo que

$$\|C'(s)\|$$

obtenemos que:

$$E(C) = \int_{\Omega} w + P(C(s))\partial s$$

Ahora obtuvimos una expresión en la cual las fuerzas externas están incluidas en el potencial externo.

B.2.3. Superficie de mínima acción

La superficie de mínima acción *U* se define como la energía mínima integrada a lo largo de un camino que va entre el punto p_0 y cualquier punto *p*:

$$U(p) = \inf_{A_{p0,p}} E(C) = \inf_{A_{p0,p}} \left\{ \int_{\Omega} \widetilde{P}(C(s))\partial s \right\}$$

Donde $A_{p0;p}$es el conjunto de todos los caminos entre p_0y *p*. El camino mínimo entre p_0 y cualquier punto $p_1$puede ser fácilmente deducido de este mapa de acción. Asumiendo que el potencial *P* es siempre positivo, el mapa de acción tendrá un solo mínimo local que es el punto de comienzo p_0, y el camino mínimo puede ser calculado por medio de back-propagation en el mapa de energía.

B.3. Normalised Aspect Ratio - NAR

Este parámetro permite determinar la calidad de un triángulo basándose en dos valores:

- **Radio del círculo circunscripto (C_c):** la distancia que existe entre en centro del triángulo y el mas cercano de los vértices.
- **Radio del círculo inscripto (C_i):** la distancia que existe entre en centro del triángulo y el mas lejano de los vértices.

$$NAR = \frac{C_i}{C_c}$$

Este valor tiende a 1 a medida que el triángulo de acerca a la forma del equilátero.

Apéndice C

Matemáticas en la Reconstrucción tridimensional

INTRODUCCIÓN

La Reconstrucción Digital 3D, es una rama de la VISION ARTIFICIAL por computador que consiste en tratar de reproducir la geometría de un objeto real, en un modelo digital con el cual se pueda interactuar por medio de rotaciones, acercamientos y/o modificaciones.

En los últimos años, los enormes avances tecnológicos en el campo de los computadores, fotografía y video digital, etc., han permitido el desarrollo de nuevas disciplinas científicas como es el caso de la visión por computador. El sistema de visión humano es muy eficiente y nos suministra información útil y muy variada sobre nuestro entorno. Somos capaces, por ejemplo, de idéntificar y reconocer fácilmente objetos y formas, también podemos detectar y seguir con facilidad objetos en movimiento en nuestro campo de visión, gracias a la visión binocular podemos estimar la distancia que nos separa de los objetos presentes en nuestro entorno, etc.. Todas estas capacidades de la visión humana y muchas otras que no mencionamos aqui son complejas de modelizar y formalizar desde el punto de vista matemático. La visión por computador es una disciplina científica de reciente desarrollo que estudia la modelización e implementación en computador de procesos propios de la visión. La visión por computador empezó siendo una disciplina esencialmente tecnológica donde se creía que todos los problemas se irían resolviendo fácilmente en base al incremento de potencia de cálculo de los computadores y las mejoras en la calidad de los dispositivos de adquisición como son las cámaras digitales. Pero pronto se descubrió que el tipo de problemas que aparecen son complejos y difíciles de analizar. De tal forma que para muchos de ellos no existe actualmente una solución plenamente satisfactoria. Esta complejidad de los problemas ha llevado a un esfuerzo científico muy importante en la disciplina donde las Matemáticas juegan un papel fundamental. De hecho, actualmente, el perfil de un investigador en la disciplina de Visión por Computador, es un perfil mucho más científico que técnico y con una sólida formación

matemática.

El objetivo de este capítulo es presentar una síntesis sobre la importancia del papel de las Matemáticas en el campo de la Visión por Computador aplicado a la Reconstrucción 3D.

C.1 Tipos de Reconstrucción

Existen diferentes tipos de Reconstrucción. Primero que todo hay que hacer énfasis en que la reconstrucción bidimensional (produce imágenes) es diferente de la Reconstrucción tridimensional (produce visualización de objetos 3D). Debido a que este proyecto se enfoca en la reconstrucción tridimensional a partir de las imágenes reconstruidas bidimensionalmente, se obviará el tema de la reconstrucción 2D, partiendo del hecho de que ese tipo de reconstrucción ya está bastante estudiado y los mismo instrumentos como tomógrafos y escáners ya realizan este tipo de reconstrucción automáticamente. Sin embargo, para efectos de conocimiento se presentará en este capítulo un resumen matemático tanto de reconstrucción 2D como de reconstrucción tridimensional, aunque sólo este último tópico es el abordado en el presente proyecto, como se menciona en los objetivos y alcances.

La reconstrucción tridimensional es un campo de estudio muy extenso ya que existe una gran variedad de técnicas, diferenciandose dependiendo del tipo de datos de entrada que se tengan. Se tienen varias situaciones para realizar la reconstrucción:

1. A partir de un conjunto de vistas a partir de cámaras.
2. A partir de dos imágenes (geometría epipolar)
3. A partir de tres imágenes (tensor trifocal)
4. A partir de muchas imágenes.

En esta investigación se presenta una reconstrucción a partir de muchas imágenes, que son las que se obtienen de un sistema de tomografía axial computarizada. Para analizar las técnicas de reconstrucción tridimensional se presentará una base matemática basada en los siguientes aspectos:

- La Geometría Proyectiva.
- Las Transformadas Integrales.
- El Cálculo Variacional
- Los Modelos Probabilísticos
- Las Ecuaciones en Derivadas Parciales Geométricas
- La Optimización y El Análisis Numérico

En la última sección de este capítulo se hará una explicación teórica y científica del método de reconstrucción para muchas imágenes utilizado y se hará una comparación de esta técnica frente a los métodos tradicionales de reconstrucción a partir de proyecciones.

C.2. La Geometría proyectiva para Reconstrucción de escenas a partir de imágenes obtenidas de cámaras.

El modelo de cámara proyectivo es el más sencillo y el que más se utiliza habitualmente. Viene definido por un plano de proyección que representa el plano imagen y un foco. La proyección de un punto 3D en la imagen viene dada por la intersección entre la recta que une dicho punto y el foco, y el plano de proyección. En la figura 5.1, podemos observar una ilustración de cómo se aplica esta modelo en pintura. El pintor mira a través de un foco una escena 3D y ha intercalado entre él y la escena una cuadrícula que representa el plano de proyección. Apoyándose en la proyección de la escena 3D sobre esta cuadrícula va pintando la escena.

En principio, las coordenadas en las que se representan los puntos son coordenadas euclídeas, esto es 3 coordenadas para un punto 3D y 2 coordenadas para un punto en el plano. Sin embargo, cuando buscamos la expresión matemática que determina como se proyecta un punto 3D en el plano, resulta mucho más conveniente trabajar en los correspondientes espacios proyectivos, de tal manera que un punto 3D se identifica con 4 coordenadas $\mathbf{X} = (x; y; z; v)$; cuando v es distinto de 0; recuperamos las coordenadas euclídeas del punto haciendo $(x/v; y/v; z/v)$: Trabajar en el espacio proyectivo nos permite

manejar corréctamente puntos en el infinito ($v = 0$). Además la aplicación que determina como se proyecta un punto 3*D* en el plano, que en coordenadas euclídeas es no-lineal, en coordenadas proyectivas es lineal y viene dada por una matriz $P \in M_{3x4}$*:* De tal manera que dado un punto 3D **X** en coordenadas proyectivas, su correspondiente proyección **x** en el plano viene dada por la expresión

$$\mathbf{x} = P\mathbf{X}$$

La matriz P depende de la posición y orientación del plano de proyección en el espacio 3*D* y del sistema de referencia elegido en el plano de proyección. Un problema fundamental que surge con mucha frecuencia en Visión por Ordenador es el de calibrar una cámara que consiste básicamente en encontrar la matriz P*.* La primera pregunta que surge es si cualquier matriz P no nula determina una proyección admisible en el sentido de que corresponda físicamente a una configuración real del modelo de proyección. La respuesta a esta pregunta es negativa. Para que P determine una proyección admisible es necesario que sus coeficientes verifiquen una cierta relación algebraica. El estudio de las relaciones algebraicas que aparecen de forma natural al plantear el problema de calibración de cámaras involucra técnicas de álgebra computacional para su resolución.

Existen muchas formas distintas de plantear el problema de calibración en función del tipo de información y número de cámaras que queramos calibrar. La manera más simple es el uso de un calibrador. Un calibrador es un objeto del cual conocemos con exactitud las coordenadas 3*iD* de algunos puntos. Por ejemplo, en este caso, conocemos (porque lo hemos medido físicamente), las coordenadas 3D de los puntos que corresponden a las esquinas de los rectángulos negros. Por otro lado, para cada esquina 3D $\mathbf{X}_i$ calculamos en la imagen las coordenadas 2*D* (en pixels) de su proyección $\mathbf{x}_i$ en el plano imagen. A partir de las correspondencias $\mathbf{X}_i \leftrightarrow \mathbf{x}_i$ es posible calcular P resolviendo un cierto sistema de ecuaciones lineales.

En muchas ocasiones no es posible la utilización de calibradores para calibrar las cámaras. En general el estudio del problema de la calibración de un sistema de varias cámaras que están observando una misma escena 3D requiere de un fuerte aparato matemático basado principalmente en la geometría y el álgebra. Del hecho de que las

cámaras están observando una misma escena se pueden derivar muchos tipos de relaciones, en su mayoría algebraicas entre las matrices de proyección de las diferentes cámaras.

Las transformadas integrales

Las transformadas integrales son una herramienta muy poderosa que posee múltiples utilidades en Visión por Ordenador. En esta sección veremos algunas aplicaciones concretas relacionadas con el análisis multiescala lineal, el reconocimiento de formas planas y la compresión de imágenes.

El análisis multiescala lineal

La transformada integral por excelencia viene dada por la convolución de una función de entrada con un cierto núcleo de convolución. En nuestro caso, la función de entrada será una imagen bidimensional que formalmente definimos como una aplicación $I : R^2 \rightarrow R$; donde *I*(*x; y*) representa el nivel de gris de la imagen en dicho punto. Por simplicidad en la exposición supóndremos que el dominio donde está definida la imagen es todo R^2 y que la imágen es de niveles de gris (y no de color en cuyo caso tendría 3 canales). En una misma imagen coexisten informaciones a diferentes escalas sobre la escena que estamos observando. Por ejemplo en una panorámica de un bosque podemos observar el bosque en su conjunto, los árboles individuales, e incluso si la resolución de la imagen es suficientemente buena, los detalles de las hojas de los árboles. La existencia de una información tan variada y compleja en una imagen hace muy complicado su análisis. Para intentar simplificar la información presente en una imagen y poder analizarla más fácilmente, una herramienta muy utilizada consiste en extraer de la imagen original una secuencia de imágenes, que representan la imagen a escalas diferentes. La manera más sencilla de hacer esto es convolucionar la imagen original con un núcleo de convolución gaussiano, es decir, construimos la secuencia

$$I_\sigma(x,y) = \int_{R^2} G_\sigma(x-u, y-v) I(u,v) du dv$$

donde

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

σ representa la escala del análisis y cuanto mayor sea, mayor simplificación se produce en la imagen original.

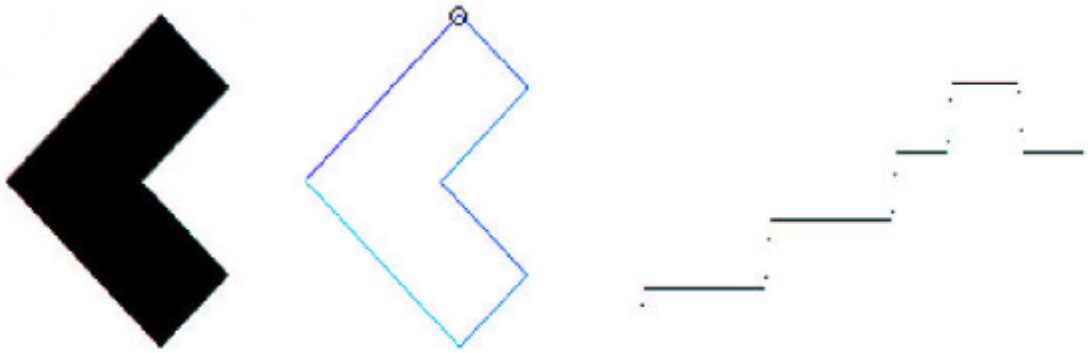


Figura C.1: Ilustración de una forma y la función de orientación de su contorno tomando como punto inicial la esquina marcada.

El reconocimiento de formas planas

La transformada de Fourier es una herramienta muy útil en Matemáticas tanto desde el punto de vista teórico como práctico. La transformada de Fourier de una función *f*((*x*) en el intervalo [*a; b*] viene dada por la expresión

$$\widehat{f}(w) = \int_a^b f(x) e^{-iwx} dx$$

A continuación veremos una sencilla aplicación de la transformada de Fourier a la descripción y reconocimiento de formas planas. Consideraremos formas sencillas definidas por un conjunto en el plano cuya frontera es una única curva cerrada como se muestra en la figura 3. Las formas de este tipo vienen caracterizadas por la geometría de

su contorno. Con objeto de utilizar una representación del contorno adecuada para comparar contornos entre sí, vamos a asociar a la curva cerrada que delimita el contorno la función unidimensional T(*s*) que para cada punto de la curva determina su orientación, es decir, el ángulo que forma la normal a la curva en dicho punto con el eje *x;* en la figura 5.1 puede verse tal representación. Por tanto para comparar dos formas lo que haremos será comparar las funciones de orientación asociadas. Ahora bien, queremos que dicha comparación sea independiente del tamaño, posición y orientación de los objetos y por supuesto, independiente del punto inicial que se ha tomado en la curva para obtener su función de orientación. Dado que estamos comparando funciones periódicas, la transformada de Fourier resulta una herramienta idónea y realizamos la comparación de las funciones de orientación de 2 formas utilizando su representación en el espacio de Fourier. Comparando los coeficientes de Fourier podemos asociar a 2 formas una medida de similitud entre ellas, de tal forma que cuanto menor sea esta medida mayor parecido habrá entre las formas, en la figura 5.2 se ilustra esta técnica.

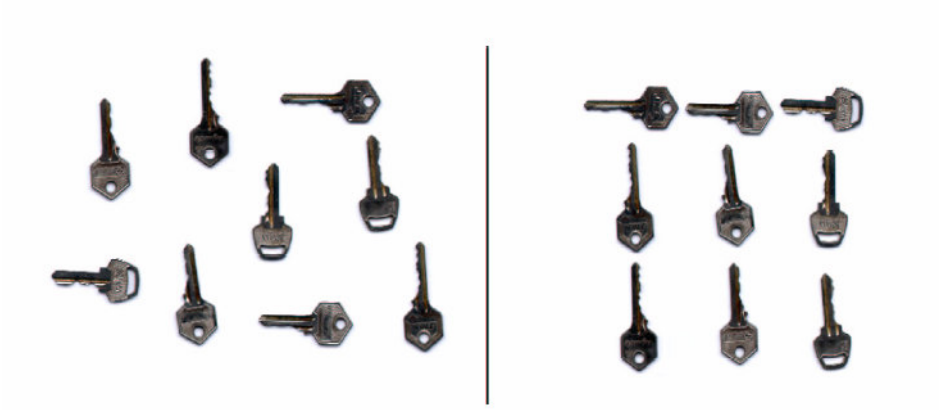


Figura C.2: Reconocimiento de formas. A la izquierda observamos un grupo de llaves, a la derecha su clasificación por criterios de similitud. Las llaves idénticas se encuentran ordenadas por columnas.

El cálculo variacional

La modelización de la solución de un problema como el mínimo de un cierto funcional de

energía es una herramienta de gran utilidad en el campo de la Visión por Ordenador. En esta sección vamos a ver algunas aplicaciones de esta técnica a problemas como la eliminación de ruido en una imagen, el seguimiento de objetos en una secuencia video o la reconstrucción 3*D* de objetos a partir de un par estéreo de imágenes.

La eliminación de ruido en una imagen

Normalmente, existe siempre un cierto ruido en las imágenes producido por los dispositivos de adquisición, manipulaciones posteriores, etc. El ruido en una imagen se presenta habitualmente en forma de irregularidades de la función imagen. Formularemos el problema de la eliminación de ruido de la siguiente forma: Dada una imagen $I(x; y)$, pretendemos encontrar otra imagen $I^0(x; y)$ tal que por un lado sea una función más regular que $I(x; y)$ y por otro lado se parezca a $I(x; y)$ lo más posible. Existen muchas formas de expresar matemáticamente estas condiciones, una forma sencilla consiste en buscar $I^0(x; y)$ minimizando el funcional

$$E(I') = \int_{R^2} (I' - I)^2 + \alpha \int_{R^2} \Phi(\|\nabla I'\|)$$

donde $\Phi(.)$ es una función positiva y **α** representa el balance entre el término que ajusta I a I^0 y el término que suaviza la función I^0 minimizando globalmente su gradiente, módulo la función $\Phi(.)$.

Para calcular los posibles mínimos de esta energía se iguala a cero la derivada del funcional de energía lo que da lugar a la denominada ecuación en derivadas parciales de Euler-Lagrange del funcional. A título de ejemplo, en el caso del funcional anterior, la ecuación de Euler-Lagrange resultante es:

$$(I' - I) - \alpha div\left(\frac{\Phi'(\|\nabla I'\|)}{\|\nabla I'\|}\nabla I'\right) = 0$$

Por tanto, observamos como la minimización del funcional se reduce a resolver una ecuación en derivadas parciales. En la figura 5.6 se ilustra el resultado de aplicar esta

técnica.

C.3.2 Seguimiento de objetos en una secuencia video

Cuando filmamos una secuencia de imágenes, se producen movimientos debidos al propio movimiento de la cámara o al desplazamiento de objetos en la escena. Dadas 2 imágenes consecutivas de la secuencia que denotaremos por $I_1(x; y)$ y $I_2(x; y)$ consideramos el problema de encontrar una función $h(x; y) = (u(x; y); v(x; y))$ denominada flujo, que determina el movimiento de cada punto entre la imagen I_1 e I_2. Utilizaremos como hipótesis de base que el nivel de gris de un punto no varia cuando el punto cambia de posición, es decir $I_1(x; y) = I_2((x; y)+h(x; y))$: Ahora bien, esta hipótesis no es suficiente para definir $h(x; y)$, pues en general, para cada punto en I_1, hay múltiples combinaciones de puntos en I_2 que verifican la hipótesis. Para poder resolver el problema, a˜nadiremos como condición que el flujo $h(x; y)$ sea regular. Ello nos lleva a minimizar el funcional

$$E(h) = \int_{R^2} (I_1(x,y) - I_2((x,y) + h(x,y)))^2 + \alpha \int_{R^2} Traza\left(\nabla h^T D(\nabla I_1)\nabla h\right)$$

el término de regularización es un poco especial porque regulariza el flujo h pero permitiendo discontinuidades en los bordes de los objetos de I_1.

C.3 Reconstrucción 3*D* de objetos a partir de un par estéreo de imágenes

Una vez calibrado un sistema de cámaras, el principal problema a resolver es la puesta en correspondencia de puntos en ámbas imágenes. El problema se puede formular en los mismos términos que en el cálculo del flujo en una secuencia video. La única diferencia es que podemos reducir la complejidad del problema utilizando la información suministrada por la denominada Geometría Epipolar. En un par estéreo, dado un punto en una imagen, su correspondiente en la otra imagen debe estar sobre una recta denominada recta epipolar. Si las cámaras están calibradas, la recta epipolar que corresponde a cada punto es conocida y por tanto ello simplifica considerablemente la complejidad del problema. En la figura 8 se muestra un resultado de aplicar este tipo de técnicas a la reconstrucción 3*D*.

Los Modelos Probabilísticos

Sin duda, los modelos probabilíticos son de una gran utilidad en Visión por Ordenador

y existe toda una comunidad de investigadores en Visión que utilizan como base dichos modelos, una imagen puede modelizarse como la realización de una cierta variable aleatoria y muchos filtros y transformaciones en imágenes pueden modelizarse como procesos estocásticos. En esta sección vamos a ilustrar como se utilizan los modelos probabilísticos para modelizar la formación de escenas naturales. En concreto veremos el denominado ”modelo de hojas muertas”que modeliza la estructura resultante de la superposición de objetos que se acumulan de forma aleatoria, por ejemplo, y de ahí su nombre, la estructura de hojas caidas en un bosque en oto˜no. La estructura del objeto que va cayendo (la hoja) se modeliza a través de los denominados conjuntos aleatorios. El objetivo que se plantea es estudiar las características geométricas de las partes visibles (denominadas celdas) de la estructura resultante después de superponer aleatoriamente las hojas. Existen resul- tados realmente interesantes como que la longitud media de los segmentos resultantes de intersectar una recta con las celdas es exáctamenta la mitad de la longitud media de los segmentos resultantes de interceptar las hojas originales. En la figura 9 se ilustra un modelo de hojas muertas donde la hoja esta compuesta por rectángulos de tamaño y orientación uniformemente distribuidos.

C.4 Reconstrucción a partir de muchas imágenes

Para hacer Reconstrucción y seleccionar el método adecuado, es necesario conocer detalladamente el tipo de imágenes que se requieran analizar. En el caso de imágenes tomográficas, es necesario un proceso complejo que no solamente consiste en aplicar algoritmos de reconstrucción sino también de método de reconocimiento y caracterización de las estructuras que se visualizan en las imágenes en escala de grises.

Esta caracterización es sumamente necesaria debido a que sin ésta no se podría hacer una reconstrucción de cada órgano presente en la imagen, y como resultado de la reconstrucción tendríamos solamente un volumen poco apto para una manipulaqción médica o útil para posteriores estudios diagnósticos. El preceso de caracterización y clasificación de estructuras en las imágenes es a lo que llamamos segmentación y ya se abordó en capítulos anteriores.

Una característica sumamente importante en la reconstrucción 3D en el área de medicina es que el proceso de reconstrucción no se puede hacer directamente sobre las imágenes originales, sino que se debe hacer sobre un conjunto de imágenes ya segmentadas.

Hay diferentes algoritmos para reconstrucción a partir de imágenes segmentadas, el que se escogió para la realización de este proyecto fue el algoritmo Marching Cubes mencionado en el capítulo anterior. Se escogió este algoritmo por su eficiencia y capacidad de obtener estructuras tridimensionales muy fiables. Por este motivo es uno de los algoritmos pioneros hoy en dia en la Reconstrucción 3D aplicado a medicina. Otra de las ventajas de éste algoritmo es que a parte de funcionar sobre imágenes segmentedas, tiene la capacidad de segmentar en casos particulares, donde los niveles de intensidas vecinos a cada voxel son muy similares a los voséeles circundantes.

Para hacer una explicación más teórica y científica de éste método se explicará a continuación y en la siguiente sección se hará una comparación de esta técnica frente a los métodos tradicionales de reconstrucción a partir de proyecciones.

C. 4.1 Reconstrucción 2D de imágenes de tomografía por emisión de positrones

Existe una gran diferencia entre Reconstrucción bidimensional (2D) y Reconstrucción tridimensional (3D). La primera hace referencia a la obtención del conjunto de imágenes originales de tomografía, las cuales son resultado de un gran numero de cálculo de proyecciones debido a que la tromografía se captura a partir de un conjunto de rayos X en diferentes direcciones, asi que para obtener una sola imagen a partir de ese conujunto de datos resultantes de la radiación a la que fue sometida el paciente es necesara una reconstrucción bidimensional (2D).

La reconstrucción tridimensional es un paso más allá de las técnicas clásicas ya que se basa en el resultado de la reconstrucción 2D (cuyo resultado son el conjunto de imágenes de tomografía) para así realizar el proceso que posteriormente nos dará como resultado una visualización tridimensional de los órganos internos y externos del paciente, simulando la realidad.

Las imágenes de tomografía de emisión se generan mediante un algoritmo de reconstrucción, a partir de un conjunto de proyecciones adquiridas del objeto o paciente bajo examen. El procedimiento clásico de reconstrucción de imagen es la retroproyección filtrada (FBP). Este método es rápido y sencillo, pero no utiliza información estadística.

Es un buen método para aplicaciones en las que el número de cuentas es alto (como tomografía de rayos X o CT), pero es peor cuando hay un bajo número de cuentas, como en imágenes de medicina nuclear.

Los métodos iterativos de reconstrucción de imagen se han propuesto como alternativas a FBP. Estas técnicas tienen un coste computacional más alto que FBP pero producen imágenes de mejor contraste y relación señal-ruido. Los métodos iterativos eliminan los artefactos de líneas presentes en las imágenes FBP, reduciendo los falsos positivos y los falsos negativos cuando las lesiones están en la proximidad de
órganos calientes. En esta sección se introducen brevemente las bases matemáticas del método FBP para seguidamente presentar los métodos estadísticos de reconstrucción iterativa, principalmente los basados en la estimación de la máxima verosimilitud. También se comenta la técnica de subconjuntos ordenados para acelerar su cómputo, así como el uso de probabilidades *a prior* bayesianas, lo que permite la incorporación de información *a priori* (tal como restricciones de suavidad o información topológica parcialmente especificada) y así mejorar la calidad de la imagen.

El objetivo de todas las modalidades de imagen médica es visualizar los órganos internos del cuerpo de una manera no invasiva, para obtener información estructural y anatómica, como en la tomografía computerizada (*computed tomography*, CT), o funcional, como en la tomografía por emisión de positrones (*positron emission to mography*, PET) o en la tomografía por emisión de fotón único (*single photon emission tomography*, SPECT).

El principio de la reconstrucción de imagen en todas las modalidades de tomografía es que un objeto se puede reproducir exactamente a partir de un conjunto de sus proyecciones tomadas desde diversos ángulos. En una aplicación práctica, en cualquier modalidad de tomografía computerizada se puede obtener solamente una estimación de

la imagen real del objeto bajo estudio. La fidelidad de la reconstrucción en cada caso dependerá de las respuestas a una serie de preguntas sobre la adquisición y el proceso previo de los datos adquiridos, de la implementación numérica de las fórmulas matemáticas de reconstrucción, y del post-procesado de las imágenes reconstruidas [1].

Denominaremos $f(x, y)$ a la distribución espacial de la densidad de actividad que se desea visualizar. Utilizaremos aquí dos dimensiones (2D), puesto que las imágenes que deben ser reconstruidas representan cortes bi-dimensionales (2D) transversales del cuerpo; aunque actualmente es posible realizar estudios PET en tres dimensiones (3D) en este trabajo no se abordará la problemática de la reconstrucción 3D. Dado que la línea de respuesta (*LOR*) del par de fotones generados en la aniquilación se conoce, el número de cuentas registrado a lo largo de la misma, definida por t (su distancia del origen) y h (su ángulo con respecto al eje x), según la figura 1, es una medida de la concentración total del radiofármaco en ese punto del «sinograma» $P(t, \theta)$. Esta función $P(t, \theta)$ representa para cada punto (t, θ) el valor la integral de línea definida por esos dos parámetros de la función $f(x,y)$, que haciendo el cambio a coordenadas polares queda expresada de la siguiente forma:

$$P(t, \theta) = \int_{(t, \theta)\text{línea}} f(x, y)\, ds = \\ = \int_{\text{línea}} f\big(t\cos(\theta) - s\sin(\theta),\, t\sin(\theta) + s\cos(\theta)\big)\, ds \qquad (1)$$

Así descrita la función $P(t, \theta)$ representa la transformada de Radon [2] de la función $f(x, y)$. En PET, el número de cuentas a lo largo del conjunto de líneas entre los detectores constituye un muestreo de la transformada de Radon de la densidad de la actividad $f(t, \theta)$. Para recuperar una aproximación de la función de distribución espacial se debe aplicar un algoritmo apropiado de inversión o de reconstrucción. La ecuación (1) se cumple solamente si se pueden despreciar las propiedades de atenuación del medio. En PET, se puede hacer una corrección de la atenuación en los datos y plantear el problema nuevamente dentro del marco de la transformada Radon [1].

El problema de la reconstrucción es ahora el siguiente:

dado $P(t, \theta)$ hay que calcular $f(x, y)$. La solución a dicho problema fue publicada por Johann Radon en 1917 [3], con una fórmula de inversión que expresa f en términos de P, aunque la implementación de esta fórmula en problemas prácticos, como el problema de la reconstrucción de imagen en CT, se ha efectuado mucho más tarde.

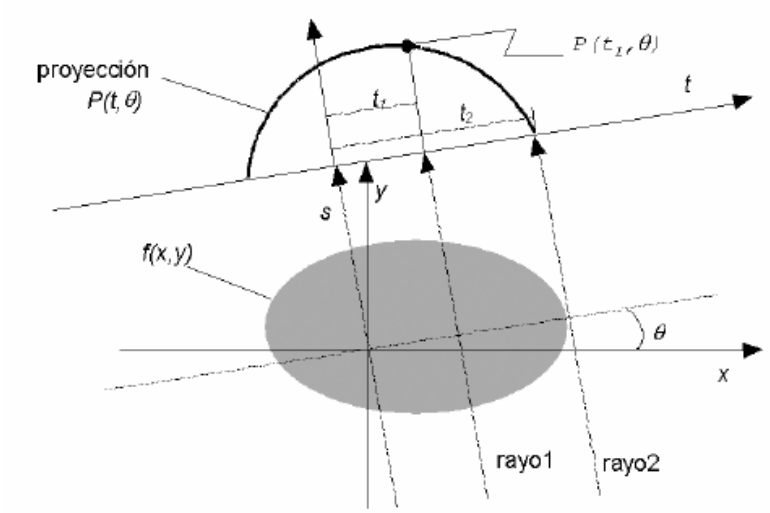


Figura C.3. Un objeto $f(x, y)$ y su proyección $P(t, \theta)$ para un ángulo h. Cada rayo se define por su distancia perpendicular t desde el origen y su orientación h.

Aunque la fórmula de inversión se puede expresar en matemáticas abstractas, éste es solamente el principio para un problema aplicado [4].

En casi todos los casos las medidas físicas no pueden definir exactamente el conjunto completo de integrales de la ecuación (1). De hecho, según demostraron Smith *et al.* [5], un objeto queda determinado únicamente por el conjunto infinito, pero por ningún conjunto finito, de sus proyecciones. La carencia del conjunto completo de integrales de línea conduce a inexactitudes y distorsiones en la reconstrucción, debidas a efectos no lineales, al ruido y a la insuficiencia de datos [6]. Los efectos no lineales se pueden originar a partir de procesos no lineales en los detectores, mientras que el ruido puede ser la incertidumbre estadística general de una medida o de un componente adicional, como la dispersión. Los datos pueden ser insuficientes debido a un muestreo inadecuado o a la

falta de datos de una región. La distribución desconocida de la atenuación puede distorsionar las medidas de la distribución original, dando por resultado errores en la reconstrucción.

En las últimas dos décadas se han desarrollado varios algoritmos que se pueden considerar como métodos para aproximar la transformada de Radon inversa. Se pueden implementar para la reconstrucción en varias modalidades tomográficas y no solamente para la PET. Es importante mencionar aquí también que, debido a muchas razones (implementación en ordenador, aproximaciones
numéricas, etc.), estos métodos no son todos equivalentes [7].

En [1] se encuentra una clasificación y descripción de los algoritmos existentes de reconstrucción de imagen, según el esquema siguiente:

1. Métodos directos de Fourier.
2. Convolución en el espacio de la señal y filtrado en el espacio de la frecuencia:
• Convolución en el espacio de la señal y retroproyección.
• Convolución en el espacio de la frecuencia y retroproyección.
• Retroproyección y convolución en el espacio de la señal.
• Retroproyección y convolución en el espacio de la frecuencia filtrado.
3. Métodos iterativos.
4. Métodos de series y funciones ortogonales.

Estas técnicas de reconstrucción de imagen proporcionan soluciones al problema de aproximar la transformada inversa de Radon. Sin embargo, el conjunto de variables consideradas como proyecciones es generalmente un conjunto de variables aleatorias, con densidad de probabilidad dependiente de integrales de línea. Una asunción fundamental de estos métodos es, por lo tanto, que el conjunto de parámetros medidos podría representar con alta exactitud (como consecuencia de la ley de grandes
números) los valores medios de este proceso (valores exactos de las proyecciones).

En el caso de la tomografía por rayos X, por ejemplo, en que se procesan en principio una gran cantidad de cuentas por ángulo de proyección, la asunción antedicha es válida y

estos métodos producen resultados satisfactorios. Sin embargo, en tomografía de emisión (PET, SPECT) el número total de cuentas es bajo, y consecuentemente hay mucho ruido estadístico, debido al retardo de los detectores, al tiempo muerto (puesto que los circuitos de coincidencia no pueden procesar más de un evento durante la ventana de la coincidencia temporal) y al «pileup» de los pulsos (cuando diversos pulsos del amplificador coinciden y producen un solo pulso que distorsiona la información de energía y contribuye a la pérdida de eventos válidos [8]). Esto limita el número total de cuentas de emisión detectadas a unas 106-107 para un solo anillo de detectores PET, cuando en tomografía de transmisión de rayos X el número de cuentas puede alcanzar las 1015-1016 [9].

Existen otros casos, especialmente cuando se requieren tiempos cortos de observación, en los que el número de cuentas medidas no se puede considerar bastante grande. Por lo tanto, se espera que un método de reconstrucción reconstrucción de imagen en tomografía de emisión que incorpore la naturaleza estocástica del proceso de emisión produzca imágenes mejores que las técnicas convencionales. Las secciones siguientes describen detalladamente los algoritmos de reconstrucción de imagen en PET, comenzando por el clásico de retroproyección filtrada y siguiendo por los métodos iterativos donde el criterio de optimización para aproximarnos a la solución óptima del problema de la reconstrucción de imagen se basa en la estimación de máxima verosimilitud.

Reconstrucción de imagen en pet con métodos de retroproyección filtrada

Los métodos de retroproyección filtrada se basan en el teorema de «cortes de Fourier» o teorema de la proyección, que afirma que (figura 5.4): «La transformada unidimensional de Fourier de la proyección de una imagen *f* (*x, y*), obtenida a partir de rayos paralelos entre sí y formando un ángulo h con el eje *x,* es el corte o muestreo de la transformada bidimensional de Fourier de la imagen *F*(*u,* v) a lo largo de una línea que forma un ángulo h con el eje *u*.»

Según este teorema, si disponemos de las proyecciones de una imagen es posible determinar cuál es esa imagen calculando una transformada bidimensional inversa de Fourier.

El resultado anterior indica que tomando P proyecciones de un objeto en los ángulos h1, h2, ..., h*P* y obteniendo la transformada continua de Fourier de cada una de ellas, podemos determinar los valores de *F*(*u,* v) —transformada bidimensional del objeto— en líneas que pasan por el origen formando los ángulos h1, h2, ..., h*P* (figura 2).

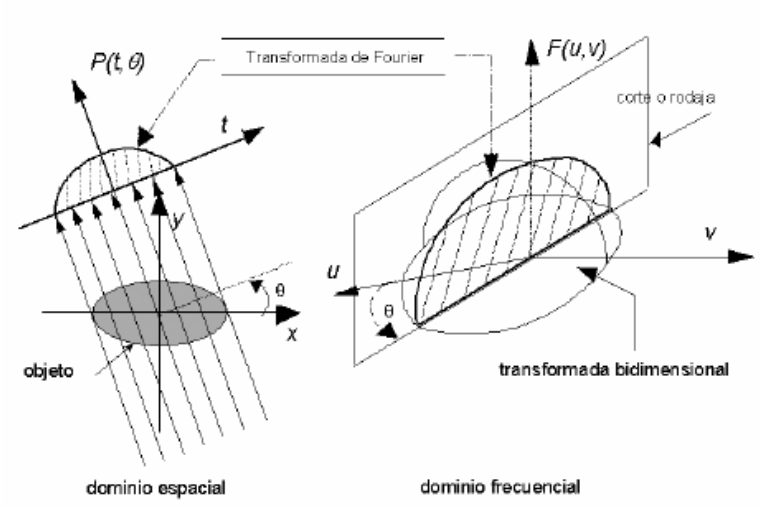


Figura C.4. El teorema de cortes de Fourier da una relación entre la transformada unidimensional de Fourier de una proyección y la transformada bidimensional de Fourier del objeto.

Tomando un número infinito de proyecciones podemos determinar *F*(*u,* v) en cualquier punto del plano
(*u,* v) y no sólo en los puntos de los ejes radiales. La funciónimagen *f(x,y)* se obtiene a partir de *F*(*u,* v) usando latransformada inversa,

$$f(x,\ y) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} F(u,\ v)\, e^{j\,2\pi(ux\,+\,vy)}\, du\, dv \qquad (2)$$

Algoritmo de retroproyección filtrada

Aunque el teorema de cortes de Fourier sugiere un algoritmo sencillo de reconstrucción, es conveniente reescribir las ecuaciones y presentarlas como un nuevo algoritmo.

Para ello, se sustituyen las coordenadas rectangulares (*u,* v) por las polares (o, θ) *u* = o cos θ v = o sin h (3) *du dv* = o *d*o *d* θ de forma que la transformada inversa de la imagen se convierte en *f* (*x, y*) = 2n

$$\begin{aligned} u &= \rho \cos\theta \\ v &= \rho \sin\theta \\ du\, dv &= \rho\, d\rho\, d\theta \end{aligned} \qquad (3)$$

De forma que la trasformada inversa de esta imagen se convierte en:

$$f(x, y) = \int_0^{2\pi} \int_0^{\infty} F(\rho, \theta) e^{j2\pi\rho(x\cos\theta + y\,\mathrm{sen}\,\theta)}\, \rho\, d\rho\, d\theta \qquad (4)$$

Esta ecuación, considerando por separado las variaciones de θ entre (0..180] y (180..360] y la propiedad

$$F(\rho, \theta + 180) = F(-\rho, \theta) \qquad (5)$$

se puede escribir como

$$f(x, y) = \int_0^{\pi} \left[\int_{-\infty}^{+\infty} F(\rho, \theta) |\rho|\, e^{j2\pi\rho t}\, d\rho \right] d\theta \qquad (6)$$

donde hemos simplificado la expresión tomando $t = x \cos\theta + y \,\mathrm{sen}\,\theta$ (7) Si sustituimos la transformada bidimensional *F*(o, θ) por la transformada unidimensional de la proyección *S*h (o), obtenemos,

$$f(x, y) = \int_0^{\pi} \left[\int_{-\infty}^{+\infty} S_\theta(\rho) |\rho|\, e^{j2\pi\rho t} d\rho \right] d\theta \qquad (8)$$

Esta integral puede expresarse también como

$$f(x, y) = \int_0^{\pi} [P_\theta(t) * h(t)]\, d\theta \qquad (9)$$

lo que se presenta como una forma sencilla de estimar la imagen *f* (*x, y*). Esta ecuación representa una operación de filtrado (i.e. convolución), donde la respuesta en frecuencia del filtro es *H*(o) = YoY. El resultado de esta convolución se conoce por ello como *proyección filtrada*. La suma de las diferentes proyecciones filtradas (una por cada h) permiten estimar así la imagen *f* (*x, y*). El algoritmo de retroproyección filtrada no considera la existencia de errores. En el proceso de reconstrucción, sin embargo, aparecen errores de varios tipos, especialmente especialmente los debidos a disponer de unos datos insuficientes y aquéllos producidos por la presencia de un ruido aleatorio en las proyecciones.

En primer lugar, un muestreo por debajo del necesario (submuestreo) o no disponer de un número adecuado de proyecciones, introduce errores en la reconstrucción.

Ambos existen por la discretización del algoritmo continuo de retroproyección filtrada: la imagen se reconstruye a partir de un número finito de muestras y a partir de un número finito de proyecciones. Un problema fundamental de las imágenes tomográficas en general es que los objetos, y por tanto sus proyecciones, no están limitados en banda. En otras palabras, el ancho de banda de las proyecciones es mayor que la más alta frecuencia capaz de ser captada con una frecuencia de muestreo dada. Para reconstruir adecuadamente por retroproyección filtrada un objeto, el número de muestras por proyección debe cumplir Nyquist y adaptarse al tamaño del más pequeño objeto que se desea ver (resolución). Si ese número es *K,* existe un número óptimo de proyecciones que es, grosso modo, el mismo *K*. Una matriz de imagen adecuada es en tal caso de $K \times K$.

Por otra parte, la utilización del filtro *H*(o) = YoY aumenta el ruido de altas frecuencias existente en las proyecciones. Es posible evitar en parte este realce del ruido eligiendo ventanas adecuadas, aunque ello implica distorsionar ligeramente la imagen.

Reconstrucción iterativa de imagen en pet basada en la estimación de máxima verosimilitud

En los últimos años, se han desarrollado modelos matemáticos para la reconstrucción de imagen en PET que consideran las características estadísticas de Poisson de la emisión de positrones dentro de la fuente. Desde esta perspectiva, el problema de la reconstrucción de imagen en PET se puede considerar como un problema estadístico estándar de estimación con datos incompletos. Los datos adquiridos en PET se consideran incompletos por el hecho de que, aunque se conoce el par de detectores donde el evento fue registrado, no se conoce el origen del evento de aniquilación (y por lo tanto el punto donde se produjo el positrón después del decaimiento radiactivo del radiofármaco del trazador).

En estadística hay un método iterativo general conocido como maximización de la esperanza (expectation – maximization, EM), presentado por Dempster et al. [10] en su forma más general. El nombre de la técnica proviene del hecho que en cada iteración existe un paso de cálculo de expectativa (*expectation step*) que usa las estimaciones actuales de los parámetros para realizar una reconstrucción del proceso Poisson inobservable, seguida por un paso de máxima verosimilitud (*maximum like- lihood*) que utiliza esta reconstrucción para revisar dichas estimaciones [11].

El principio de máxima verosimilitud en la reconstrucción de imagen para tomografía de emisión fue introducido inicialmente por Rockmore y Macovski [12]. Una implementación práctica del método EM para el problema de la reconstrucción de imagen en PET fue publicada por Shepp y Vardi [13] y extendida a la tomografía por transmisión por Lange y Carson [14].

El algoritmo EM se aplica en la tomografía por emisión como técnica iterativa para calcular las estimaciones de máxima verosimilitud (maximum likelihood estimation, MLE) a partir de los parámetros de la densidad de actividad. Así, los datos adquiridos se consideran muestras de un conjunto de variables aleatorias con funciones de densidad de

probabilidad relacionadas con la distribución del objeto según un modelo matemático del proceso de adquisición de los datos. Usando el modelo matemático, es posible calcular, para cualquier densidad inicial de la distribución real, la probabilidad de que haya producido los datos observados. En el conjunto de todas las imágenes posibles, que representan una distribución potencial del objeto, la imagen que tiene más alta dicha probabilidad es la estimación de máxima verosimilitud del objeto original.

Descripción del algoritmo EM para PET

En una cámara PET hay un conjunto de detectores colocados típicamente en forma de anillo. El par de fotones producidos en un evento de aniquilación se detecta por la coincidencia en un par de detectores que definen un volumen cilíndrico (un «tubo de detectores» —*detector tube*— [15] o una línea de respuesta —*line of response, LOR*—). El conjunto de datos registrados en un estudio PET se representa por el vector [$y(1)$, $y(2)$, ..., $y(J)$], donde $y(j)$ es el número total de coincidencias registradas en el tubo j y J es el número total de LORs. Si N es el número total detectores en el anillo, entonces el número de los LORs es: $J = N(N - 1)/2$. Los eventos de coincidencia registrados incluyen eventos de dispersión y coincidencias accidentales (*accidental coincidences* o *randoms*) [8]. Además, no todos los eventos producidos dentro de la fuente se detectan, debido a la atenuación de los tejidos, los fotones que atraviesan el anillo de detectores sin interaccionar con el mismo y pasan desapercibidos, u otras ineficacias debidas a diversas razones. La densidad de actividad dentro de la fuente, en el caso de dos dimensiones (2D) representada por la función $f(x, y)$, tiene que ser estimada usando el vector **y** de datos medidos. Para su implementación en el ordenador y su visualización, la densidad d está digitalizada en los píxeles $i = 1, 2, ..., I$. A cada píxel i corresponde un número $n(i)$ de eventos desconocido, que representa el número número total de emisiones que ocurrieron en el área de la fuente cubierta por el píxel i, cuya media es $x(i) = = E[n(i)]$, $i = 1, 2..., I$. El problema ahora es estimar $x(i)$, o estimar el número verdadero $n(i)$ en cada píxel, usando los datos observados $y(j)$, $j = 1, 2..., J$ [13].

Una emisión en el píxel i se detecta en el LOR j con probabilidad conocida:

$$a(i, j) = P\,(\text{evento detectado en el LOR } j \mid \text{evento emitido en el píxel } i) \qquad (10)$$

donde $a(i, j) \geq 0$. La matriz de probabilidad $a(i, j)$ se determina a partir de la geometría del tomógrafo y otras características del sistema.

Las variables $y(j)$ son independientes y siguen la estadística de Poisson, con esperanza matemática:

$$\tilde{y}(j) = E[y(j)] = \sum_{i=1}^{I} x(i)a(i, j) \qquad (11)$$

Puesto que $x(i)$ son variables Poisson independientes, una combinación lineal de estas variables como la de la ecuación anterior tiene también una distribución según el modelo de Poisson. La verosimilitud de los datos observados está por tanto dada por la expresión:

$$L(x) = P(y \mid x) = \prod_{j=1}^{J} e^{-\tilde{y}(j)} \frac{\tilde{y}(j)^{y(j)}}{y(j)!} \qquad (12)$$

La función de verosimilitud $L(\mathbf{x})$ expresa la probabilidad, bajo el modelo de Poisson de emisión de fotones, de observar el número de eventos adquirido si la densidad verdadera es $x(i)$. La maximización de esta función usando el método de máxima verosimilitud, proporciona el esquema iterativo de corrección del vector de imagen, que se puede aplicar directamente al problema de la reconstrucción de imagen en PET. Si $x(k)$ denota la estimación de x en la iteración k, se puede definir una nueva estimación
$x(k + 1)$ como:

$$x^{(k+1)} = x^{(k)}(i) \sum_{j=1}^{J} a(i, j) \frac{y(j)}{\tilde{y}^{(k)}(j)}, \quad i = 1, 2, ..., I \qquad (13)$$

donde $\sim y\,(k)$ es la proyección al espacio de datos del vector de imagen $\mathbf{x}(k)$ estimado en la iteración k. Es decir, contiene los valores previstos de los datos registrados si $\mathbf{x}(k)$ fuese el vector de imagen verdadero (y desconocido) de la distribución de actividad en la fuente.

La suma en la expresión anterior es la retroproyección de la relación: $\frac{y(j)}{\tilde{y}^{(k)}(j)}$ en el espacio

de imagen con peso $a(i, j)$. Por lo tanto, la forma general del proceso de actualización de la imagen en la iteración $(k + 1)$ es:

$$x^{(k+1)}(i) = x^{(k)}(i)C^{(k)}(i), \quad i = 1, 2, ..., I \qquad (14)$$

donde $C(k)$ son coeficientes multiplicativos que actualizan el valor del píxel i en la iteración $(k + 1)$. Estos coeficientes se calculan usando el vector y de los datos del sinograma y la proyección de la imagen estimada $x(k)$ al espacio de los datos, usando un operador de proyección **P** y viceversa (del espacio de los datos al espacio de la imagen) con el operador transpuesto **PT**.

Es importante notar aquí:

1. Dada una imagen inicial **x**(0) no negativa y los valores $a(i, j)$ e $y(j)$ no negativos (como es el caso en PET), entonces todas las imágenes producidas por el algoritmo EM son no negativas. Ésta es una ventaja
importante del método en comparación con el método de retroproyección filtrada o las técnicas
algebraicas, que pueden producir valores negativos, sin sentido, para la densidad de emisión en ciertos
píxeles. Esta no negatividad se obtiene sin imponer restricciones adicionales, que compliquen el trabajo
de cómputo, y es inherente a la formulación del método.

2. Para cada vector de imagen **x**(*k*) producido por el algoritmo, la suma de eventos en todos los píxeles es igual a la suma de los eventos registrados en los sinogramas. Esta característica resulta inmediatamente
de la fórmula iterativa (13), a partir de la cual se puede demostrar que:

$$\sum_{i=1}^{I} x^{(k)}(i) = \sum_{j=1}^{J} y(j) \qquad (15)$$

Esto significa que el algoritmo EM está normalizado en sí mismo y la redistribución de la actividad en los píxeles que ocurre después de cada iteración, no implica aumento o

disminución de la actividad total. Según lo notado por Lewitt y Muehllehner [16], desde el punto de vista práctico, las características de no negatividad y la normalización automática son más importantes que el hecho de que el algoritmo EM progrese en la dirección del máximo global de la función de la verosimilitud.

Otra característica importante del algoritmo EM de reconstrucción iterativa de imagen es que está basado en el modelo estocástico exacto de las medidas de la proyección, a diferencia de la convolución – retroproyección o de las técnicas de Fourier, que son deterministas y no tienen en cuenta la naturaleza estocástica de los datos [12,14].

El algoritmo EM para la tomografía de emisión puede proporcionar un modelo físicamente exacto de la reconstrucción, puesto que permite la incorporación directa de muchos factores físicos, que, si no son considerados, pueden introducir errores en la reconstrucción final. Estos factores se pueden incluir en la matriz de transición $a(i, j)$ [14] y pueden ser:

a) Información de corrección de atenuación, que se puede incorporar en $a(i, j)$ en vez de corregir los
datos del sinograma antes de la reconstrucción [17].

b) Dispersión y correcciones de coincidencias accidentales.

c) Efectos del rango del positrón.

d) Información sobre el tiempo-de-vuelo *(time-offlight)*[18, 19].

e) Información sobre la ventana de resolución de coincidenciatemporal para cada par de detectores y también información sobre la naturaleza del radiofármaco específico que se está usando (con decaimiento radiactivo conocido).

f) Normalización para el muestreo redundante de las proyecciones y la eficacia de cada par de detectores

(puesto que la eficacia de detección no es igual para todos los pares de detectores y puede en la práctica
estar en el rango de un 80% para la mayoría de los radioisótopos [8]).

g) Variación en la resolución espacial (aunque es más significativa en SPECT que en PET). Una desventaja importante de los algoritmos EM de reconstrucción, como sucede con la mayoría de las técnicas algebraicas e iterativas, es su ritmo lento de convergencia a una imagen aceptable y el alto coste de cómputo para una implementación práctica. Sin embargo, se ha encontrado que los algoritmos EM producen resultados superiores en comparación con los métodos de convolución-retroproyección [20,21], reduciendo los artefactos (tiene mejor relación señal-ruido) y reduciendo el error en la valoración del metabolismo del radioisótopo, especialmente en zonas de baja actividad [22].

Estudios recientes [22,23] han verificado la superioridad de las técnicas EM en términos de características de ruido y detectabilidad de lesiones, especialmente después de estudios detallados y sistemáticos de los espectros de frecuencia de la imagen, de sesgo y de varianza [24], respecto a la retroproyección filtrada. Además, la reconstrucción iterativa EM no requiere datos de proyección igualmente espaciados y permite utilizar un conjunto incompleto de datos [25]. Especialmente en el caso de SPECT, se ha encontrado que el algoritmo EM proporciona una cuantificación mejor de la imagen [26] y una definición mejor de los bordes del objeto que las técnicas de retroproyección filtrada.
Por otra parte, hay también estudios [27] indicando que las técnicas de máxima verosimilitud no presentan una mejora significativa en la calidad de imagen comparado a la retroproyección filtrada. Otros [28] concluyen que con el post-procesado adecuado de las imágenes de la retroproyección filtrada, los artefactos asociados a esta técnica de reconstrucción se podrían reducir y mejorar la calidad de la imagen a niveles muy cerca de la calidad ofrecida por el algoritmo EM, usando medios de cómputo más simples.

Las observaciones anteriores se deben a que las técnicas EM son altamente dependientes de la estructura de la imagen y los niveles de actividad, y también es común pasar varias decenas de iteraciones hasta comenzar a conseguir mejoras significativas. Además, las imágenes producidas en la tomografía de emisión con el algoritmo EM pueden llegar a ser más ruidosas y tener distorsiones importantes cerca de los bordes,

durante las iteraciones antes de que las imágenes converjan hacia la estimación de la máxima verosimilitud [29, 30]. Un alto valor de la verosimilitud no significa necesariamente una calidad mejor de la imagen. Se necesita imponer restricciones adicionales en la ecuación (13) para evitar el ruido inherente de Poisson.

Desarrollos recientes y modıficaciones del algoritmo EM

Las conclusiones de las investigaciones en curso sobre métodos EM para reconstrucción de imagen en PET, son [31]: (i) el algoritmo EM converge lentamente, con un número típico de 30 a 60 iteraciones necesarias para conseguir una buena calidad de imagen, y (ii) si se sigue iterando para obtener valores de verosimilitud más grandes, después de un cierto punto óptimo, las imágenes vuelven a ser ruidosas: aparece un efecto de «tablero de ajedrez» *(checkerboard effect)* [32] que llega a ser visible en la imagen, junto con un artefacto de intensificación de los bordes, que hacen la interpretación de la imagen prácticamente imposible. Sin embargo, hay un cierto debate [33] sobre si las imágenes deterioradas después de muchas iteraciones EM pueden todavía proporcionar información clínica útil.

El problema de la convergencia lenta de la técnica EM ha sido tratado por varios autores. Se han presentado versiones aceleradas del algoritmo EM para superar esta desventaja. Aceleración aquí significa que se intenta alcanzar el máximo de la función de verosimilitud de la ecuación (12) más rápido que con el algoritmo EM convencional de la ecuación (13). Ciertamente, el objetivo de las técnicas de aceleración es reducir el coste de cómputo y no cambiar las características de convergencia del método [31]. No obstante, no hay garantía de que con técnicas aceleradas se puedan obtener imágenes mejores o que una calidad mejor o igual de imagen se pueda alcanzar más rápidamente. Las técnicas que aplican esquemas de actualización secuenciales muestran un funcionamiento acelerado, puesto que la imagen se actualiza en pasos que implican solamente parte de los píxeles o de los sinogramas.

El algoritmo *«space-alternating generalized expectation maximization»* (SAGE) [34] utiliza la retroproyección de la estimación de imagen expresada por la ecuación (13) para calcular los factores de corrección para la actualización de uno o un subgrupo de píxeles

durante una iteración. De esa manera, se alcanza una aceleración sobre el método EM usando consideraciones estadísticas y por lo tanto el aumento continuo en la función objetivo de la ecuación (12) está garantizado. El «*image space reconstruction algorithm*» (ISRA) [35] se ha propuesto para reconstruir datos de un tomógrafo con una función de dispersión puntual (PSF) variable. Se ha determinado que ISRA converge a una solución no negativa del estimador de mínimos cuadrados de la densidad de actividad, a condición de que se utilice una imagen inicial positiva [36]. El algoritmo «*weighted least squares*» (WLS) se ha presentado recientemente para mejorar el ritmo de convergencia del algoritmo EM [37] Hudson y Larkin [38] presentaron un método para acelerar el algoritmo EM usando el método de subconjuntos ordenados (*ordered subsets* EM, OSEM) de los datos del sinograma. En esa técnica, el vector de los datos adquiridos se divide en subconjuntos. En cada iteración de actualización EM, la retroproyección de la ecuación (13), que es el proceso que más tiempo consume, se calcula solamente para los tubos de detectores del subconjunto seleccionado. A estos datos se les aplica una función retroproyección para actualizar el valor de los píxeles de la imagen.

Este método crea una nueva estimación de la imagen en una fracción del tiempo requerido por el procedimiento convencional EM, donde todos los píxeles se actualizan con todos los datos del sinograma. Hudson y Larkin en [38] demuestran que el método OSEM puede producir una aceleración del orden del número de los subconjuntos aplicados. Sin embargo, este método no converge a una solución de máxima verosimilitud, a excepción del caso de datos sin ruido.

Los métodos de aceleración para las técnicas MLE (*maximum likelihood estimation*) de reconstrucción de imagen en tomografía de emisión pueden proporcionar una convergencia más rápida hacia estimaciones de máxima verosimilitud. No obstante, no garantizan una calidad mejor de imagen que el algoritmo estándar EM. Se han propuesto varias modificaciones del método MLE que incorporan información *a priori* para caracterizar la distribución de actividad y ruido en los datos adquiridos.

Hay también un cierto debate [39] sobre el aumento del ritmo de convergencia de reconstrucción con el uso probabilidades *a priori* bayesianas (*Bayesian priors*). La idea fundamental de los métodos bayesianos de reconstrucción de imagen es la siguiente [40]:

En la derivación del algoritmo EM se intenta maximizar la probabilidad: *P(datos de coincidencia* Y v*ector de imagen)* (o *P*(*y* Y *x*)) (16) para observar los datos adquiridos dada una estimación actual de la distribución de actividad en la fuente. Según el teorema de Bayes, se puede escribir:

$$P(\text{imagen} \mid \text{datos}) =$$
$$= [P(\text{datos} \mid \text{imagen}) - P(\text{imagen})] / P(\text{datos}) \quad (17)$$

En esta ecuación se expresa la probabilidad condicional de que la imagen sea correcta dado: (i) el conjunto de datos medidos en términos de la probabilidad *P(datos* Y *imagen)* que se calcula normalmente con el algoritmo EM; (ii) una constante *P(datos)* de normalización, y (iii) la probabilidad *a priori P(imagen).*

Se puede obtener ahora la imagen más probable maximizando el valor en la parte derecha de la ecuación (17), llamada distribución *de probabilidad a posteriori* del vector de imagen [41]. Este método hace uso de una cierta información *a priori* «razonable» para evitar el deterioro de la imagen que ocurre al maximizar solamente *P*(datos Y imagen) sin restricciones, como en el algoritmo EM. Esta información *a priori* se incorpora en el término *P*(imagen) de la expresión anterior y representa una va*loración a priori* de como se espera que sea la imagen reconstruida [42, 43]. Los métodos bayesianos de reconstrucción permiten la incorporación de la información anterior (tal como restricciones de suavidad o información
parcial topológica especificada) y por lo tanto reducen la sensibilidad al ruido.

Un método bayesiano recientemente presentado [44], basado en el algoritmo OSL («*One-Step Late*») propuesto por Green [45] y modificado por Lange [39], hace uso de la «Median Root Prior» (MRP), para la función *a priori P*(imagen). Según esta técnica, se introduce un factor multiplicativo adicional en el esquema de actualización de los píxeles:

$$x^{(k+1)}(i) = M^{(k)}(i) \cdot C^{(k)}(i) \cdot x^{(k)}(i) \qquad (18)$$

donde:

$$M(i) = \frac{1}{1 + \beta \dfrac{x(i) - \text{med}(x, i)}{\text{med}(x, i)}} \qquad (19)$$

med(*x*, *i*) es la mediana de los píxeles vecinos del pixel *i*. El coeficiente bayesiano b representa un factor de peso *a priori* con valores entre 0 a 1.0. El método MRP no preserva el número total de cuentas medidas en el vector de imagen, al contrario que el algoritmo EM y no converge a una estimación de máxima verosimilitud. No obstante es muy eficiente quitando el ruido sin enmascarar las estructuras
localmente monótonas Como resumen, las técnicas EM, SAGE, WLS y ISRA emplean diversos esquemas que implican el operador *P* de proyección y retroproyección para calcular los coeficientes
de corrección, según las ecuaciones siguientes:

$$\text{EM:} \quad C^{(k)} = P^T[y \,/\, P(x^{(k)})] \qquad (20)$$

$$\text{SAGE:} \quad C^{(k)} = P^T[y \,/\, P(x^{(k)})] \qquad (21)$$

donde $P(x^{(k)})$ se actualiza continuamente

$$\text{WLS:} \quad C^{(k)} = P^T\,(y \,/\, P[x^{(k)}])^2 \qquad (22)$$

$$\text{ISRA:} \quad C^{(k)} = P^T[y] \,/\, P^T\,P[x^{(k)}] \qquad (23)$$

En todas estas técnicas, es difícil (si no imposible) probar su convergencia; no obstante se aceptan puesto que se ha demostrado que producen imágenes exactas en experimentos de simulación y durante reconstrucciones sobre datos clínicos. En cualquier caso, cada método introduce su conjunto de parámetros (tales como el coeficiente b en la técnica MRP, etc.), que parecen ser de importancia crucial en el resultado final y deben ser seleccionados con cuidado. Esta selección es frecuentemente un proceso
empírico y depende altamente de la tarea específica en cada caso, y por esa razón se puede cuestionar la aplicabilidad universal de estos métodos en el contexto de reconstrucción de imagen para tomografía de emisión. Sin embargo, todas estas técnicas y otras que existen actualmente, prueban la flexibilidad del método EM para ajustarse a un problema médico específico como la tomografía por emisión de positrones y se deben evaluar siempre sobre una base dependiente de la tarea a efectuar (taskdependent) [33].

El ritmo lento de convergencia es una de las desventajas principales de los algoritmos de reconstrucción iterativa de imagen. Eso significa que hacen falta de varias iteraciones para alcanzar el máximo, o por lo menos un valor cercano al máximo, de la función objetivo (verosimilitud).

El método EM con subconjuntos ordenados (OSEM) presenta una solución a este problema, junto con otras alternativas recientemente presentadas [46]. Otro problema asociado a la observación anterior es la carencia de una regla robusta de parada (*stopping rule*), que pueda indicar el fin del proceso iterativo, aunque en el pasado se han hecho algunos estudios en esa dirección [47]. El problema es debido principalmente al hecho de que la implementación directa del algoritmo EMu otros similares es intrínsecamente inestable y se agrega ruido estadístico a las imágenes reconstruidas mientras siguen las iteraciones. Para remediar esta situación, se han presentado los métodos bayesianos, tales como el métodoMRP, como alternativas al posible post-procesado (filtrado, etc.) de las imágenes, o parando temprano el proceso iterativo.

C.4.2.Técnicas de reconstrucción tridimensional (3D)

Después de lo estudiado en los temas anteriores estamos en disposición de poder abordar la reconstrucción de la escena. La calidad de la reconstrucción que se obtendrá dependerá de la información que tengamos sobre el sistema de captura de las imágenes. Identificaremos tres casos, a) cuando se conocen tanto los parámetros intrínsecos como extrínsecos de la cámara, b) cuando solo se conocen los parámetros intrínsecos de la cámara, c) cuando no se conocen ningún tipo de parámetros, es decir, solo conocemos el conjunto de puntos en correspondencia.

En el primer caso vamos a ver que será posible reconstruir la escena de forma única usando triangulación sobre los puntos en correspondencia. En el segundo caso veremos que podremos obtener una reconstrucción y estimar los parámetros extrínsecos, pero esto será salvo un factor de escala indeterminado. En el tercer caso, obtendremos también una reconstrucción pero esta será correcta salvo una transformación proyectiva global desconocida.

Reconstrucción por triangulación

Este es el caso más simple. Al conocer todos los parámetros del sistema la reconstrucción es directa. Como se muestra en la figura que muestra la geometría epipolar, el punto P es proyectado en un par de puntos en correspondencia p_i y p_d se sitúa en la intersección de los dos rayos definidos por los centros de proyección O_i y O_d y su respectivo punto de proyección p_i y p_d. Las ecuaciones de dichos rayos pueden ser calculadas, y por tanto su intersección, pero el efecto del ruido sobre las coordenadas de los puntos de proyección conducirá a que los rayos no se corten en el espacio; por tanto su intersección deberá ser estimada como el punto del espacio de menor distancia a ambos rayos. La estimación de este punto es el núcleo del algoritmo de triangulación.

Sea ap_i (aЄR) el rayo i a través de O_i y p_i. Sea $T+bR^T p_d$ (bЄR) el rayo d a través de O_d y p_d expresado en el sistema de referencia izquierdo. Sea w un vector ortogonal a i y d. El problema es determinar el punto medio P', del segmento paralelo a w que une i y d.

La solución a este problema es muy simple ya que los puntos extremos del segmento, digamos $a_0 p_i$ y $T+b_0 R p_d$ pueden calcularse resolviendo el sistema lineal de ecuaciones

$$ap_i + c(p_i \times R^T p_d) = T + bR^T p_d$$

para a,b y c.

Es importante señalar que el determinante del sistema anterior es siempre distinto cero salvo cuando los rayos son paralelos. También es posible calcular la reconstrucción directamente sobre las imágenes rectificadas en lugar de tener que usar las imágenes originales.

¿Cuanto es de válida la hipótesis de que se conocen los parámetros intrínsecos y extrínsecos?. Si la geometría del sistema no cambia con el tiempo, los distintos parámetros pueden ser estimados de acuerdo a los algoritmos ya estudiados de la calibración de la cámara.

Ejercicio: Si T_i y R_i, y T_d y R_d son los parámetros estimados de la cámara izquierda y derecha respectivamente en el sistema de referencia del mundo, probar que los

parámetros extrínsecos del sistema estéreo, T y R están dados por $R = R_d R_i^T$, $T = T_i - R^T T_d$.

Reconstrucción salvo un factor de escala

Ahora supondremos que solo conocemos los parámetros intrínsecos del sistema y a partir de ellos y del conjunto de puntos en correspondencias calcularemos los parámetros extrínsecos y la reconstrucción 3D de la escena. El método que expondremos hace uso de la matriz esencial del sistema por tanto necesitaremos al menos 8 puntos en correspondencias para poder estimarla.

El problema: Bajo la hipótesis del conocimiento de los parámetros intrínsecos y n≥8 puntos en correspondencia calcular la reconstrucción 3D de los puntos a partir de sus proyecciones.

A diferencia del método de triangulación ahora desconocemos la distancia base que existe entre las cámaras por tanto la reconstrucción que se obtendrá será verdadera salvo una constante de escala. Este valor puede ser fijado del conocimiento de la distancia real entre dos puntos de la escena.

El primer paso requiere la estimación de la matriz esencial, E, que solo puede ser conocida salvo una constante de proporcionalidad; por

tanto buscaremos una normalización conveniente de E. De la definición de la matriz esencial tenemos $E^T E = S^T R^T R S = S^T S$ o

$$\mathbf{E}^T\mathbf{E} = \begin{bmatrix} T_y^2 + T_z^2 & -T_x T_y & -T_x T_z \\ -T_y T_x & T_z^2 + T_x^2 & -T_y T_z \\ -T_z T_x & -T_z T_y & T_x^2 + T_y^2 \end{bmatrix}$$

Puede observarse de esta matriz que su traza es el doble de la norma al cuadrado del vector T, $Tr(E^T E) = 2\|T\|^2$. Por tanto dividir los elementos de la matriz esencial por $N = \sqrt{Tr(\mathbf{E}^T\mathbf{E})/2}$ es equivalente a normalizar la longitud del vector traslación a la unidad.

Usando la normalización la matriz anterior se puede rescribir como

$$\hat{\mathbf{E}}^T\hat{\mathbf{E}} = \begin{bmatrix} 1-\hat{T}_x^2 & -\hat{T}_x\hat{T}_y & -\hat{T}_x\hat{T}_z \\ -\hat{T}_y\hat{T}_x & 1-\hat{T}_y^2 & -\hat{T}_y\hat{T}_z \\ -\hat{T}_z\hat{T}_x & -\hat{T}_z\hat{T}_y & 1-\hat{T}_z^2 \end{bmatrix}$$

donde el símbolo Λ indica normalización tanto en la matriz como en los elementos. Recuperar los valores de $\hat{\mathbf{T}}$ a partir de cualquier fila o columna de la matriz es una tarea fácil. Sin embargo, dado que los elementos de las filas o columnas de la matriz son cuadráticas en los elementos de T existe una indefinición de un signo global en los valores estimados.

Supongamos de momento que $\hat{\mathbf{T}}$ ha sido calculado con el valor del signo verdadero. Vamos a calcular la rotación. Definimos $\mathbf{w}_i = \hat{\mathbf{E}}_i \times \hat{\mathbf{T}}$, es decir el producto vectorial de la fila i-esima de $\hat{\mathbf{E}}$ por $\hat{\mathbf{T}}$. Si R_i denota la fila i-esima de la matriz de rotación, R, se puede demostrar que $R_i = w_i + w_j \times w_k$ donde (i,j,k) desarrolla todas las permutaciones cíclicas de 1,2,3.

En resumen, dada una estimación de la matriz esencial normalizada obtenemos cuatro estimaciones de la pareja (,R). Las cuatro estimaciones son la consecuencia de la ambigüedad en el signo de y E. La reconstrucción 3D de los puntos observados resuelve la ambigüedad y encuentra la única solución correcta. Para cada una de las cuatro parejas (,R) calculamos la tercera componente de los puntos en el sistema de referencia de la cámara de la izquierda.

Teniendo en cuenta la ecuación vectorial que relaciona los vectores coordenados, de un mismo punto de la escena en los sistemas de referencia de las cámaras $P_d = R(P_i - T)$, $Z_d = \mathbf{R}_3^T(\mathbf{P}_i - \hat{\mathbf{T}})$, y

$$\mathbf{p}_i = \frac{f_i}{Z_i}\mathbf{P}_i \ , \ \mathbf{p}_d = \frac{f_d}{Z_d}\mathbf{P}_d$$

obtenemos

$$\mathbf{p}_d = \frac{\mathrm{f}_d \mathbf{R}(\mathbf{P}_i - \hat{\mathbf{T}})}{\mathbf{R}_3^T(\mathbf{P}_i - \hat{\mathbf{T}})}$$

Por tanto para la primera componente de p_d tenemos

$$\mathrm{x}_d = \frac{\mathrm{f}_d \mathbf{R}_1^T(\mathbf{P}_i - \hat{\mathbf{T}})}{\mathbf{R}_3^T(\mathbf{P}_i - \hat{\mathbf{T}})}$$

Finalmente, substituyendo P_i por su expresión en p_i y despejando Z_i obtenemos

$$Z_i = f_i \frac{(f_d \mathbf{R}_1 - x_d \mathbf{R}_3)^T \hat{\mathbf{T}}}{(f_i \mathbf{R}_1 - x_d \mathbf{R}_3)^T \mathbf{p}_i}$$

Las otras coordenadas de P_i se calculan de las ecuaciones de la perspectiva, y las coordenadas de P_d de la ecuación anterior que relaciona ambos vectores a los parámetros extrínsecos.

Recordemos que solo una de las cuatro estimaciones de (,R) da estimaciones consistentes de las coordenadas ZTˆ$_i$ y Z_d (es decir ambas positivas) de todos los puntos. Los detalles se exponen en el siguiente algoritmo

---

ALGORITMO RECONSTRUCCION_EUCLIDEA

La entrada esta formada por un conjunto de puntos en correspondencia en coordenadas cámara, y una estimación de la matriz esencial normalizada.

1. Calcular el vector de traslación normalizado a partir de la matriz cuadrado de la matriz esencial. Tˆ
2. Calcular los vectores w y calcular las filas de la matriz de rotación.

3. Reconstruir los valores de las coordenadas Z_i y Z_d de cada uno de los puntos a partir de las ecuaciones anteriores.
4. Si el signo de Z_i y Z_d de los puntos reconstruidos son
    a. Ambos negativos para algún punto, cambiar el signo de e ir al paso 3 Tˆ
    b. Uno negativo y uno positivo, para algún punto, cambiar el signo de cada entrada de E e ir al paso 2 ˆ
    c. Ambos positivos para todos los puntos, terminar.

La salida es el conjunto de puntos reconstruidos salvo un factor de escala.

Nota: La implementación del algoritmo se debe de hacer de manera que se garantice que el algoritmo no itera más de cuatro veces de los pasos 2-4. Además se debe de tener en cuenta que en el caso de desplazamientos muy pequeños, los errores de disparidad pueden ser suficientes para hacer que la reconstrucción 3D inconsistente; cuando esto sucede el algoritmo se queda iterando entre los pasos 2-4.

Reconstrucción no calibrada

En esta sección se muestra que es posible calcular una reconstrucción 3D incluso en la ausencia de cualquier información sobre los parámetros intrínsecos y extrínsecos. El precio a pagar es que la reconstrucción que se obtiene es única salvo una transformación proyectiva global del mundo. Las siguientes figuras muestran reconstrucciones equivalentes desde el punto de vista proyectivo (la primera), desde el punto de vista afín (la segunda) y desde el punto de vista Euclídeo (la tercera)respectivamente En esta sección se muestra que es posible calcular una reconstrucción 3D incluso en la ausencia de cualquier información sobre los parámetros intrínsecos y extrínsecos. El precio a pagar es que la reconstrucción que se obtiene es única salvo una transformación proyectiva global del mundo. Las siguientes figuras muestran reconstrucciones equivalentes desde el punto de vista proyectivo (la primera), desde el punto de vista afín (la segunda) y desde el punto de vista Euclídeo (la tercera)respectivamente

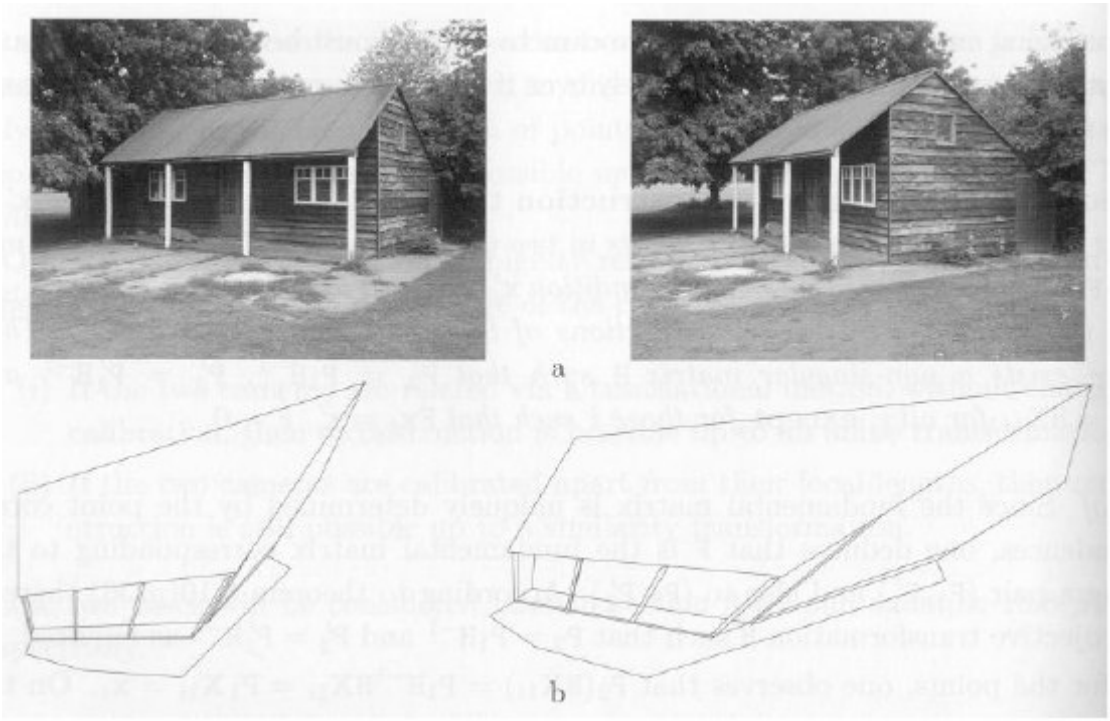


Figura C.5 Trasformación proyectiva

Proyectiva: Obsérvese como se conservan las relaciones de incidencia y colinealidad

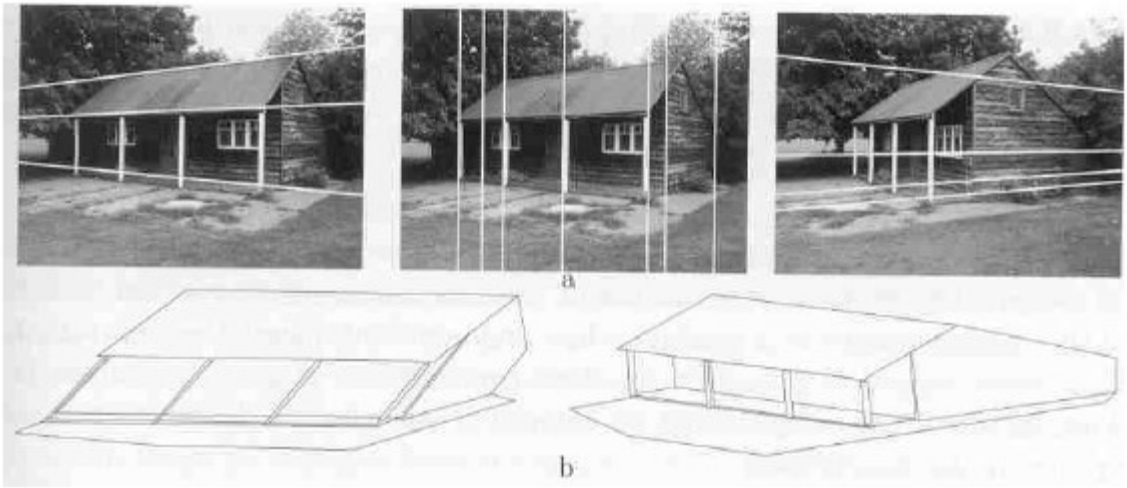


Figura C.5 Trasformación Afín

Afín: Obsérvese como se conservan el paralelismo entre las líneas

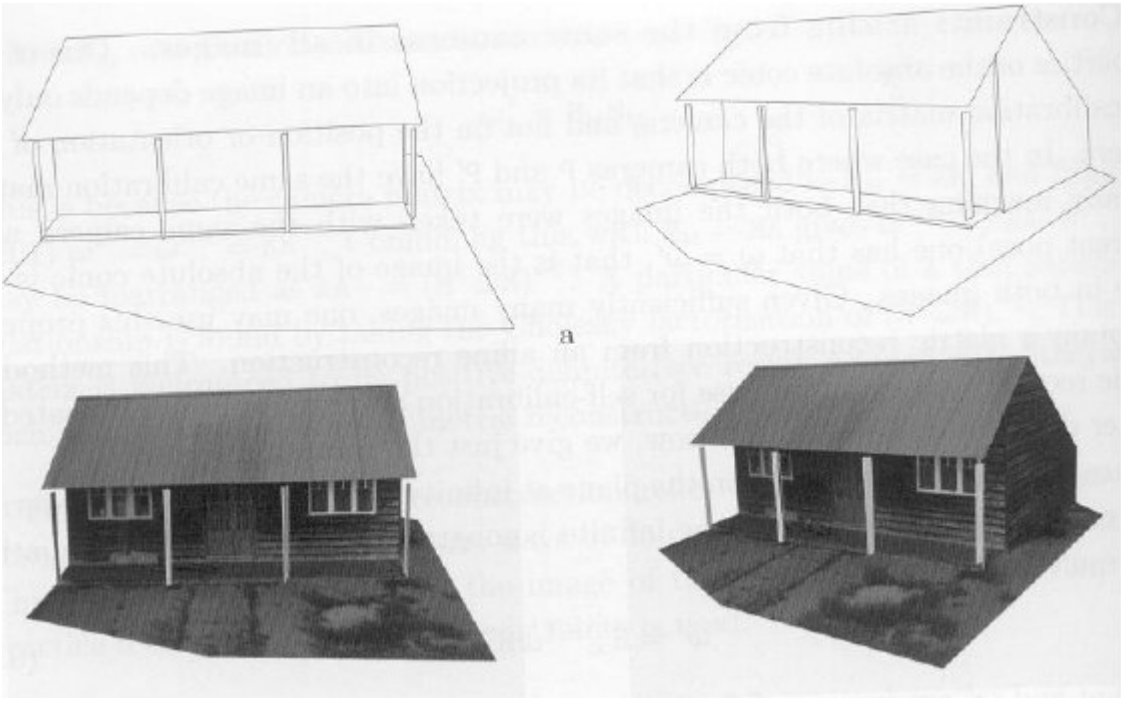


Figura C.5 Trasformación Euclíea

Euclíea: Obsérvese como se conservan los ángulos y la ortogonalidad.

El Problema: Supongamos que solamente tenemos n≥ 8 correspondencias de puntos, el problema es calcular las localizaciones de los puntos 3D a partir de sus coordenadas en las imágenes.

El algoritmo que vamos a estudiar en este caso es solo el dado por la aproximación lineal. Ahora consideramos las dos ecuaciones de proyección x=PX y x'=P'X, que proyectan un mismo punto X del mundo en los dos puntos imagen x y x' a través de las matrices de proyección P y P. Estas dos ecuaciones homogéneas, de forma similar a lo que nos encontramos en el cálculo de la matriz de una homografía 2D, nos permiten obtener cada una de ellas dos ecuaciones lineales independientes en los valores de X si las expresamos como x×PX=0 y x'×P'X=0. Por tanto por cada punto obtenemos un sistema homogéneo de cuatro ecuaciones con cuatro incógnitas que puede expresarse como

$$\mathbf{A} = \begin{pmatrix} x\mathbf{p}^{3T} - \mathbf{p}^{1T} \\ y\mathbf{p}^{3T} - \mathbf{p}^{21T} \\ x'\mathbf{p}'^{3T} - \mathbf{p}'^{1T} \\ y'\mathbf{p}'^{3T} - \mathbf{p}'^{2T} \end{pmatrix}$$

Donde (x,y) son las coordenadas imagen por la matriz P de X, y p^{iT} indica la fila i-ésima de la matriz P. El vector X puede calcularse usando la descomposición SVD de la matriz A

Este problema de resolver el sistema homogéneo de ecuaciones puede resolverse como ya hemos dicho a partir de una solución del sistema homogéneo o podemos transformar el sistema de ecuaciones en otro no-homogéneo y buscar una solución de mínimos cuadrados. Sin embargo el uso del método no-homogéneo supone que el punto no esta en el plano del infinito, lo que no es posible saber a priori. Además ninguno de los dos métodos es invariante proyectivo, es decir que la solución por ambos métodos se afecta si transformamos las coordenadas de los puntos con una transformación lineal. En el caso de transformaciones afines la situación es diferente, ya que el método no-homogéneo es invariante afín.

Función de coste de error-geométrico

Tal y como se ha estudiado ya en otra ocasiones(matriz fundamental) la estimación de máxima verosimilitud permite estimar como variables subsidiarias las coordenadas de los puntos en correspondencia que verifican la geometría epipolar de forma exacta. Por tanto, una vez tengamos correspondencias que verifiquen

exactamente la geometría epipolar cualquier método de triangulación nos permitirá calcular las coordenadas del punto 3D.

Como siempre el uso del error de Sampson nos permite aproximar bastante bien el mínimo de la función de coste con un menor esfuerzo computacional.

Aproximación de Sampson

En este caso, como ya sabemos, lo que nos interesa calcular es el jacobiano de la función de coste $\varepsilon = x'^{T} Fx$. El jacobiano es

$$\mathbf{J} = \partial\varepsilon/\partial\mathbf{x} = [(\mathbf{F}^{T}\mathbf{x}')_1 , (\mathbf{F}^{T}\mathbf{x}')_2 , (\mathbf{F}\mathbf{x})_1 , (\mathbf{F}\mathbf{x})_2]$$

Donde p.e $[(\mathbf{F}^{\mathrm{T}}\mathbf{x}')_1 = f_{11}x' + f_{21}y' + f_{31}$

Entonces la aproximación de primer orden al punto correcto es

$$\begin{pmatrix} \hat{x} \\ \hat{y} \\ \hat{x}' \\ \hat{y}' \end{pmatrix} = \begin{pmatrix} x \\ y \\ x' \\ y' \end{pmatrix} - \frac{x'^T \mathbf{F} x}{(\mathbf{F}x)_1^2 + (\mathbf{F}x)_2^2 + (\mathbf{F}^T x')_1^2 + (\mathbf{F}^T x')_2^2} \begin{pmatrix} (\mathbf{F}^T x')_1 \\ (\mathbf{F}^T x')_2 \\ (\mathbf{F}x)_1 \\ (\mathbf{F}x)_2 \end{pmatrix}$$

La aproximación será precisa si la corrección en cada imagen es pequeña (menor de un píxel), y no es costosa de calcular. Notemos que los puntos finales no verificarán la restricción epipolar de forma exacta.

Solución óptima

El cálculo de una solución optima que nos va a llevar a un algoritmo que no necesita de cálculo iterativo para su solución pasa por la reformulación del problema de mínimo que queremos calcular. En lugar de minimizar una función de coste basada en la suma de distancias entre los puntos estimados y medidos reformularemos el problema en términos de una función de coste de distancias de los puntos medidos a las líneas epipolares de los puntos estimados. Esta reformulación nos permitirá parametrizar la función de coste en términos de un único parámetro lo que nos conduce a poder estimar el valor de dicho parámetro que alcanza el mínimo de la función de coste.

Pasamos de minimizar $d(x,\hat{x})^2 + d(x',\hat{x}')^2$ a minimizar $d(x,l)^2 + d(x',l')^2$ donde l y l' toman valores sobre todas las posibles lineas epipolares en correspondencia. El punto es entonces el punto sobre l más cercano a x y el puntose define de forma similar.

La estrategia de minimización es como sigue

i. Parametrizar el haz de líneas epipolares en la primera imagen por un parámetro t. Por tanto las línea epipolar en la primera imagen se escribe como l(t).
ii. Usando la matriz fundamental F, calcular la correspondiente lineal epipolar l'(t) en la segunda imagen.

iii. Expresar la función distancia $d(x,l(t))^2 + d(x',l'(t))^2$ explícitamente como una función de t.

iv. Encontrar el valor de t que minimiza esta función.

De esta forma, el problema se reduce de encontrar un minimo en un espacio de cuatro dimensiones a un espacio de una sola dimensión.

Con una parametrización adecuada del haz de líneas epipolares la función distancia es una función polinómica racional de t, que puede transformarse en un polinomio de grado 6.

La parte negativa de este algoritmo es que no es generalizable al caso de 3 o más vistas. En cambio los otros algoritmos si lo son.

Notas sobre la implementación

Supondremos que ninguno de los puntos coincide con el epipolo de su imagen ya que en ese caso no es posible determinar las coordenadas del punto X. Entonces bajo esta hipótesis transformaremos las coordenadas de los puntos con una transformación rígida que no afecta a la minimización de la función de coste pero que facilitara la misma. Estas transformaciones llevan el origen de la imagen al punto correspondiente y giran la imagen para que el epipolo caiga sobre el eje x en los puntos (1,0, f) y (1,0, f ') respectivamente.

En este caso la matriz fundamental toma la forma especial especificada en el algoritmo.

Consideremos una línea epipolar en la primera imagen que pasa por el punto $(0,t,1)^T$. Como debe de pasar por el epipolo estará definida por el producto

(0,t,1)x(1,0, f)=(t f, 1 , -t), por tanto la distancia al cuadrado del origen a la línea es

$$d\left(x,l\left(t\right)\right)^2 = \frac{t}{1+\left(tf\right)^2}$$

Usando la matriz fundamental se puede encontrar la línea epipolar asociada a dicho punto en la otra imagen, y se puede calcular la distancia de la línea al origen. La función de coste a minimizar, s(t), es la suma de ambas funciones distancia (ver el algoritmo). La minimización de dicha función puede ser reducida de forma sencilla a la búsqueda de las raices de un polinomio de grado 6 (ver el algoritmo).

Algoritmo

Dada una correspondencia de punto medida x↔x', y una matriz fundamental F, calcular las correspondencias corregidas que minimizan el error geométrico sujeto a la restricción epipolar.

Pasos

1.- Definir matrices de transformación que trasladan los puntos al origen

$$T = \begin{bmatrix} 1 & & -x \\ & 1 & -y \\ & & 1 \end{bmatrix} \text{ y } T' = \begin{bmatrix} 1 & & -x' \\ & 1 & -y' \\ & & 1 \end{bmatrix}$$

2.- Reemplazar F con $T'^{-1} F T^{-1}$. Esta nueva F corresponde a las coordenadas transformadas.

3.- Calcular el epipolo derecho e izquierdo de F. Normalizar el epipolo e para que $e_1^2 + e_2 2=1$, igual para e'.

4.- Formar las matrices de rotación

$$R = \begin{bmatrix} e_1 & e_2 & \\ -e_2 & e_1 & \\ & & 1 \end{bmatrix} \text{ y } R' = \begin{bmatrix} e'_1 & e'_2 & \\ -e'_2 & e'_1 & \\ & & 1 \end{bmatrix}$$

y observar que $Re=(1,0,e_3)^T$ y $R' e'=(1,0,e'_3)^T$.

5.- Reemplazar F por $R' F R^T$ lo que debe de conducir a una matriz del tipo

$$\mathbf{F} = \begin{pmatrix} ff'd & -f'c & -f'd \\ -fb & a & b \\ -fd & c & d \end{pmatrix}$$

6.- Poner $f = e_3$, $f' = e'_3$, $a=F_{22}$, $b=F_{23}$, $c=F_{32}$, $d=F_{33}$.

7.- Formar el polinomio

$g(t)= t((at + b)^2 + f'^2 (ct + d)^2)^2 - (ad - bc)(1+f^2 t^2)^2 (at + b)(ct + d)=0$

y buscar sus 6 raices.

8.- Evaluar la función de coste

$$s(t) = \frac{t^2}{1 + f^2 t^2} - \frac{(ct + d)^2}{(at + b)^2 + f'^2 (ct + d)^2}$$

tanto en cada una de las partes reales de la raices y el valor asintótico para t=∞. Seleccionar el valor t_{min} de t que da el valor más pequeño de la función de coste.

9.- Evaluar las dos líneas l=(tf,1,-t) y l'=F(0,t,1)^T en t_{min} y encontrar los puntos sobre estas rectas más cercanos al origen.

10.- Deshacer las transformaciones de traslación

11.- Las coordenadas 3D se pueden ahora obtener por triangulación.

Reconstrucción de Líneas

Supongamos que una línea en el espacio se proyecta en dos vistas como las líneas l y l'. La línea en el espacio 3D puede ser reconstruida por retroproyección de cada línea a un plano del espacio 3D e intersecando los planos. Los planos definidos por las líneas son $\pi = P^T l$ y $\pi' = P'^T l'$. Es muy conveniente a veces representar la línea 3D como la intersección de los dos planos, es decir representar la línea como una matriz 2x4

$$\mathbf{L} = \begin{bmatrix} \mathbf{l}^T \mathbf{P} \\ \mathbf{l}'^T \mathbf{P}' \end{bmatrix}$$

tal que para cualquier punto X de la recta LX=0.

Notemos que en el caso de puntos en correspondencia un punto del espacio esta sobredeterminado a través de sus proyecciones en dos vistas ya que tenemos cuatro medidas y solo 3 incógnitas. Sin embargo, en el caso de líneas la línea 3D esta especificad de forma exacta por sus proyecciones ya que observamos dos medidas en cada vista y la línea 3D tiene 4 grados de libertad. (estamos discutiendo líneas infinitas no segmentos).

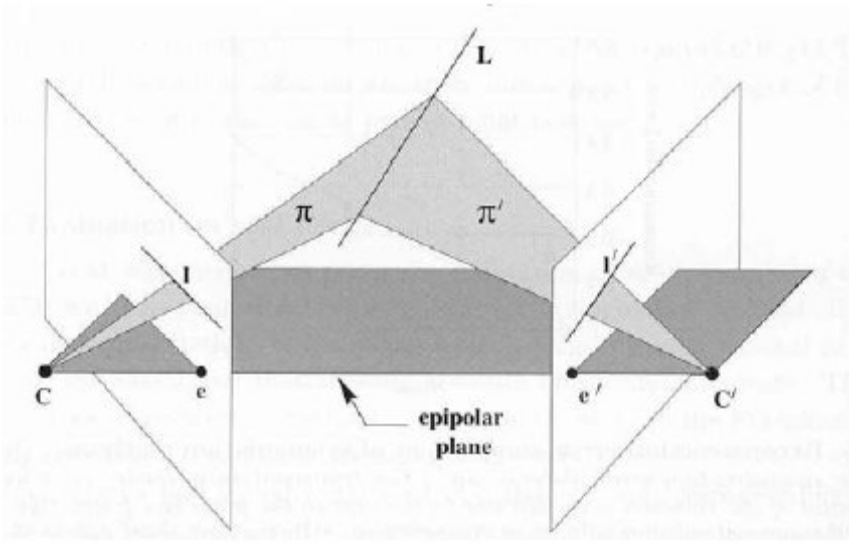


Figura 5.6 plano epipolar

Si la línea 3D cae cerca o dentro del plano epipolar la configuración es degenerada y no podrá obtenerse ninguna reconstrucción ya dicha línea corta a la línea de base que une los centros de las cámaras y sus proyecciones serían líneas epipolares. El caso de degeneraciones para líneas es mucho más importante que el caso de puntos ya que existe toda una familia tri-paramétrica de líneas que da configuraciones degeneradas: un parámetro para indicar el punto de la línea-de-base y dos parámetros para la estrella de líneas que pasa por cada punto de dicha línea.

Cálculo de los puntos de anulación

Ahora vamos a describir como se pueden calcular los puntos de anulación de una imagen a partir de segmentos de línea, aunque el método es válido también para otros problemas de intersección de líneas.

El objetivo ahora es calcular el punto de intersección de líneas definidas a partir de segmentos medidos sobre las imágenes. Ahora supondremos que los segmentos son proyección de líneas paralelas en el mundo.

Debido al ruido de las medidas, es obvio que las líneas que representan dichos segmentos no se cortarán en un único punto. Para buscar dicho punto haremos uso del método de máxima verosimilitud que nos permitirá estimar a la vez las líneas y el punto de cruce coherente con ellas. Para ello ajustaremos a los segmentos estimados un haz de

líneas que se cortan en un punto y trataremos de minimizar la distancia entre las líneas del haz y los segmentos medidos.

Los segmentos medidos se pueden caracterizar adecuadamente por dos puntos por lo cual la distancia a minimizar será una suma de distancias de punto-a- recta. Para ello haremos uso del algoritmo de L-M.

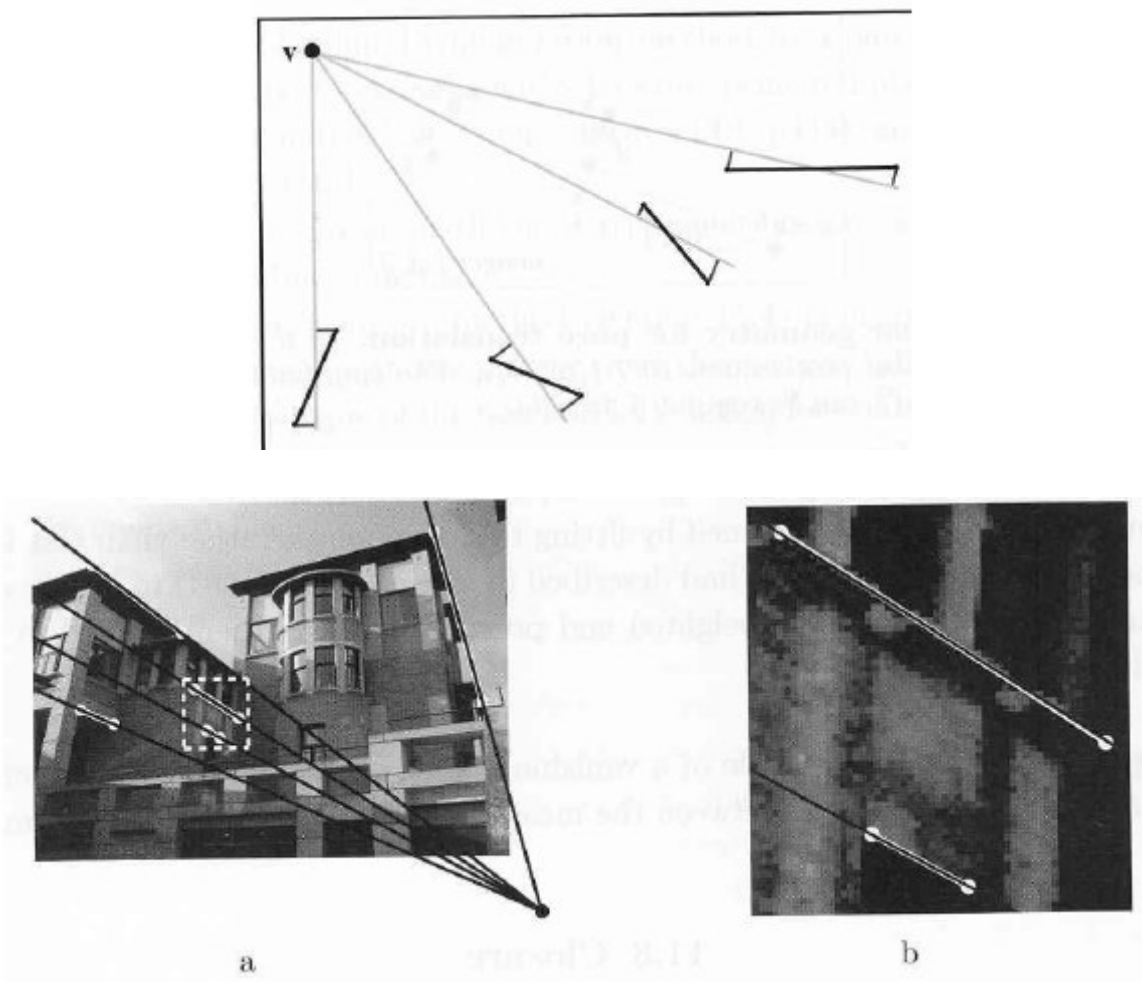


Figura C.7 reconstrucción a partir de líneas y varios puntos

Líneas especificadas por varios puntos

sea descrita por varios puntos, estos pueden reducirse la caso de 2 puntos adecuadamente elegidos. Consideremos un conjunto de puntos x_i en una imagen, con la tercera componente normalizada a1. Sea $l=(l_1, l_2, l_3)^T$ una línea que se supone normalizada

para que $l_1^2 + l_2^2 = 1$. En este caso al distancia de un punto x_i a la recta l es $x_i^T l$, la distancia al cuadrado es $l^T x_i x_i^T l$ y la suma de cuadrados es

$$\sum_i \mathbf{l}^T \mathbf{x}_i \mathbf{x}_i^T \mathbf{l} = \mathbf{l}^T (\sum_i \mathbf{x}_i \mathbf{x}_i^T) \mathbf{l} = \mathbf{l}^T \mathbf{E}\, \mathbf{l}$$

en donde la matriz $E=(x\Sigma i_i x_i^T)$ es por construcción definida positiva y simétrica.

Lemma: La matriz $(E - \varepsilon_0 J)$ es semidefinida positiva y simétrica, donde J es la matriz diag(1,1,0) y ε_0 es la solución más pequeña a la ecuación $\det(E - \varepsilon_0 J)=0$.

Dado que $E - \varepsilon_0 J$ es una matriz simétrica puede escribirse de la forma $E - \varepsilon_0 J = V\,\mathrm{diag}(r,s,0)V^T$ donde V es una matriz ortogonal y r y s son positivos. Se sigue que

$E - \varepsilon_0 J = r\, v_1 v_1^T + s\, v_2 v_2^T$

Donde v_i es la i-esima columna de V. Por tanto $E = \varepsilon_0 J + r\, v_1 v_1^T + s\, v_2 v_2^T$.

$l^T E\, l = \varepsilon_0 + r\,(v_1^T l)^2 + s\,(v_2^T l)^2$

Por tanto, hemos reemplazado la suma de cuadrados de varios puntos por un valor constante ε_0, que no es capaz de ser minimizado, mas la suma de cuadrados ponderada de la distancia a dos puntos v_1 y v_2.

Resumiendo, cuando una línea se representa por un conjunto de puntos, esta representación es equivalente a una representación de dos puntos especiales ponderados por pesos $\sqrt{r}$ y $\sqrt{s}$.

C.5 Reconstrucción 3D a partir de voxels

Uno de los métodos de representación de objetos orgánicos en 3D son los llamados blobs. Un blob es una isosuperficie definida por una expresión de potencial de campo finito. Contando con dicha expresión hemos de obtener la representación del cuerpo de la forma más adecuada. Para la representación de cuerpos en 3D suelen usarse técnicas de poligonalización, que consisten en una obtención del modelo de fronteras de la forma. Existen numerosos algoritmos que resuelven dicho problema para el caso de los blobs, casi todos basados en el padre de todos ellos: *Marching Cubes*. Nuestro objetivo es el estudio de dicho algoritmo para la obtención de un postproceso que nos proporcione unos resultados más correctos, sin un excesivo incremento de su complejidad computacional. Finalmente generalizaremos el postproceso para que pueda aplicarse a una malla poligonal obtenida a partir de cualquier algoritmo de poligonalización de blobs.

.F. Blinn creó un método algebraico general de modelado llamado *"Modelo Blobby"* en el cual podemos expresar un objeto 3D en términos de isosuperficie (superficie de densidad constante) en base a un conjunto de primitivas generadoras de campo [1]. La expresión general para calcular el potencial de campo de un blob en un punto *(x,y,z)* cualquiera del espacio es la que se observa en la siguiente ecuación.

$$V(x,y,z)=\left(\sum_{i=1}^{N} b_i e^{-a_i f_i(x,y,z)}\right)-T$$

En la expresión del potencial de campo (ecuación anterior), a_i es la caída de campo de la primitiva i, b_i es la fuerza de campo de la primitiva i, T es el umbral del campo y N es el número de primitivas. Como podemos observar dicha ecuación es muy difícil de parametrizar puesto que no es lineal. Por tanto convertirla a una ecuación paramétrica que nos determinase un conjunto de puntos donde el valor del campo fuese nulo (condició n de pertenencia a la superficie) sería muy complicado. Por tanto ú nicamente contamos con una expresión matemática que, dado un punto del espacio y un conjunto de primitivas nos dice si el punto queda dentro del campo (valor de $V>0$), fuera

del campo (valor de $V<0$) o en su superficie (valor de $V=0$).

Como puede observarse, el exponente de dicha fórmula es una función *f(x,y,z)*. Dicha función viene definida por el modelo que estemos utilizando. Pueden ser: campos de supercuádricas [2], Metaballs [3] y Soft Objects [4].

Nuestro objetivo es la representación de la frontera del blob para su posterior utilización como herramienta 3D. Podríamos representarlo directamente con una técnica de trazado de rayos rastreando todo el volumen del paralelepípedo que encierra al blob (que puede calcularse con la posición geométrica de las primitivas que lo componen) y buscando los puntos que forman su frontera mediante la función de densidad. Dichos puntos se iluminarían calculando la normal en los mismos mediante la expresión del gradiente (pues es la normal del plano tangente a la superficie en el punto). Pese a ser una técnica sencilla y precisa presenta un alto coste computacional. Para solucionar este problema se suele poligonalizar la frontera del sólido obteniendo un metapolígono que encierre convenientemente el volumen del mismo. Un metapolígono se puede tratar de forma más rápida puesto que existen numerosas técnicas avanzadas de representación de los mismos que incluso se implementan de forma estándar en el propio hardware. Nuestro objetivo es desarrollar una técnica general para la obtención de un modelo de fronteras lo suficientemente preciso de un blob.

Para poligonalizar un blob existen numerosos algoritmos, la mayoría basados en *Marching Cubes*, el cual fue diseñado por *Lorensen y Cline*. Dicho algoritmo consiste en cortar el volumen del paralelepípedo que encierra al blob en un número de cubos de igual tamaño. Posteriormente se hacen intersectar los cubos con la superficie del blob. Dependiendo de la posición de los vértices exteriores / interiores y de los cortes, se poligonaliza de una u otra forma. Existen 14 posibilidades de corte de un cubo con la superficie eliminando rotaciones, traslaciones y simetrías.

Marching Cubes y la mayoría de los algoritmos derivados del mismo tienen el problema común de que en todo momento se analizan cubos de igual tamaño sea cual sea la regió n que se esté estudiando. Esto produce que el algoritmo no obtenga un conjunto de

triángulos que determine de manera óptima la forma. Existe una variante interesante del algoritmo basada en descomponer los cubos que cortan a la superficie en cinco tetraedros [8], que a su vez vuelven a cortar a la superficie, con lo que se obtiene un metapolígono más suavizado que con la técnica original. El problema de dicha técnica es que es equivalente a disminuir el tamaño de cubo, aumentando de forma global el conjunto de triángulos y por tanto generando concentraciones excesivas en zonas donde no es necesario. Lo que perseguimos es una técnica con una medida de precisión local que estudie cada región del blob con diferente nivel de detalle, determinando el modelo de fronteras más aproximado a dicha forma.

Existen diferentes formas de medir la adaptación de una malla poligonal a un sólido basadas en distintos criterios, como por ejemplo el que plantean H.L. de Gougny y M. S. Shephard [10] basado en diferencias de volúmenes con el sólido original. Dichas técnicas de similitud sirven precisamente para medir la adaptabilidad de los distintos algoritmos que estamos mencionando.

Existen otras propuestas de algoritmos no derivadas de *Marching Cubes* basadas en recorrer directamente la frontera del blob graduando la exploración de dicha frontera dependiendo de medidas de precisión locales [6]. Pese a que los resultados son excelentes en cuanto al estudio local de precisión, se producen otros problemas derivados que no han sido resueltos como el de la detección de huecos en el sólido o el de detección de formas separadas.

Igualmente existen propuestas de algoritmos genéricos de generación de mallas

para distintos tipos de superficies [9] basadas en triangulaciones como la de Delaunay. El problema de dichas técnicas es que no se basan en información específica de la propia superficie de estudio por lo que en la mayoría de ocasiones incurren en costes bastante altos. Otras técnicas específicas de generación adaptativa de mallas para blobs como la "*Poligonalización de blobs con rectificación de precisión*" [11] pese a obtener buenos resultados presenta el inconveniente de la excesiva complejidad que supone su implementación, lo que reduce su aplicación práctica y su generalización. El algoritmo que proponemos se plantea como un postproceso de ajuste, con lo que será aplicable a cualquier

poligonalización de blobs de forma sencilla. En los siguientes apartados explicaremos el procedimiento seguido en *Marching Cubes* y plantearemos el postproceso de ajuste sobre dicho algoritmo.

Algoritmo de Marching Cubes

Se trata de un algoritmo para la poligonalización de cualquier tipo de isosuperficies que consiste en lo siguiente: En primer lugar se determina el paralelepípedo que encierra al blob haciendo uso de la información de la posición y tamaño de las primitivas que lo componen. Seguidamente dividimos dicho paralelepípedo en cubos de tamaño constante. Dicho tamaño será definido por el usuario dependiendo de la precisión que desee para la figura resultante.

A continuación se analiza cada cubo por separado. Inicialmente hemos de formar un *cluster* de 8 bits que nos defina la colocación de cada cubo con respecto al blob. Dicho *cluster* se formará con la información de la pertenencia de los distintos vértices del cubo al blob de forma que los vértices interiores caracterizados por tener un potencial >=0 los marcaremos con un 1 y los exteriores que está n caracterizados por tener un potencial <0 será n marcados con un 0.

Una vez tenemos la información sobre la colocación del cubo con respecto al blob procederemos a estudiar qué forma tendrá la porción de superficie asociada a dicho cubo. Para ello *Lorensen y Clyne* [5] estudiaron de cuántas maneras puede un cubo intersectar con una superficie. De las 256 posibles combinaciones (por la definición del cluster tenemos 2^8=256 combinaciones de los valores binarios), eliminaron los casos de colocación físicamente imposibles. Si además tenemos en cuenta simetrías y rotaciones entre los distintos casos el número disminuye a 14.

Por último, para cada una de las 14 combinaciones citadas se estudió la mejor forma de poligonalizarlas. La forma de dichas poligonalizaciones así como los posibles casos de intersección de un cubo con la superficie se pueden observar en la figura 5.8.

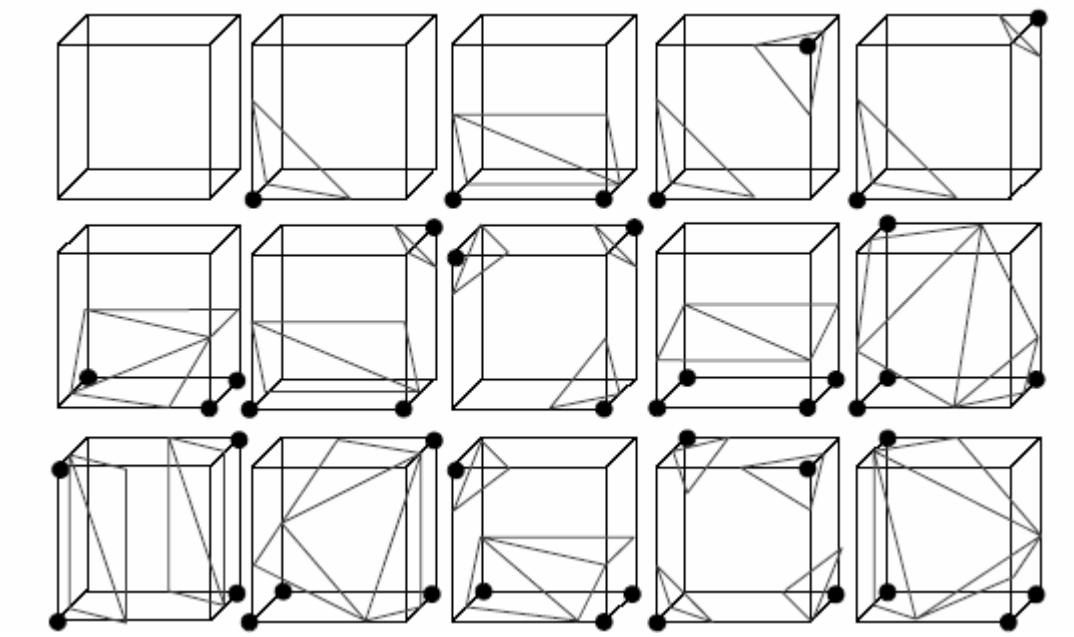


Figura C.8 Casos de intersección de un cubo con un blob.

Ventajas e inconvenientes del algoritmo

El algoritmo presenta grandes ventajas como puede ser su sencillez de implementación con respecto a otros. Por otro lado resulta computacionalmente sencillo (bajo coste) y la malla resultante es igualmente sencilla.

Los inconvenientes del algoritmo son muchos. El principal es que por propia naturaleza del mismo el rastreo del volumen se realiza al mismo nivel de detalle en todas las zonas, por lo que no se asegura la calidad final de la forma. Este problema es derivado de la escasa información que se toma de la superficie y la localidad de la misma. Por otro lado no se asegura un ángulo mínimo entre las normales de polígonos contiguos, con lo que la iluminación del metapolígono deja mucho que desear en la mayoría de los casos (efecto vóxel). Tampoco se asegura la coherencia entre el volumen real y el poligonalizado por lo que en algunos casos la conectividad del metapolígono no es la misma que la del sólido original.

Finalmente no se especifica ninguna forma de determinar el tamaño de cubo óptimo

para una determinada forma.

Estos problemas surgieron en la versión inicial del algoritmo, la cual ha sido modificada por muchos autores que persiguen su mejora desde distintas perspectivas. Por ejemplo en la versión inicial se encontraban huecos en la malla poligonal en distintos casos. Dicho error ha sido solucionado por varios autores añadiendo varios casos más de posibles intersecciones del cubo con el blob. Por otro lado el problema de la coherencia de los volúmenes, ha sido solucionado por varios autores, como es el caso del algoritmo "*Volume preserving MC*". Igualmente el problema del tamaño de cubo puede ser solucionado de forma sencilla calculándolo en base a la información geométrica de las primitivas, tomando dicho tamaño en función de la primitiva más pequeña que tenga la escena. De esta forma se asegura que ninguna primitiva se queda totalmente incluida por un cubo con lo que no obtendríamos intersección y perderíamos la porción de volumen.

El problema de la precisión y el de asegurar las normales entre polígonos contiguos es un problema más complicado. Podríamos introducir cualquier técnica de suavizado de aristas para solucionarlo pero el problema de estas técnicas es que son muy generales y normalmente al no contar con información del sólido original tienden a hacer el metapolígono aún más distinto a la forma real. Perseguimos una técnica específica que se ajuste a las características de los objetos orgánicos.

C.6 Comparación de métodos tradicionales de reconstrucción a partir de proyecciones y métodos basados en voxels

Los métodos de Reconstrucción tridimensional tienen la particularidad de ser específicos para lo que fueron diseñados. Como se mencionó en la primera sección de este capítulo, los métodos se pueden clasificar según el tipo de datos de entrada que se requieran analizar. Reconstruir a partir de un conjunto de imágenes es una tarea compleja, que requiere de algoritmos con grandes cantidades de cálculos, pero éstas operaciones matemáticas sólo tienen sentido si se aplican convenientemente.

Cuando se tiene un conjunto de imágenes en diferentes vistas es conveniente utilizar los

algoritmos de reconstrucción a partir de proyecciones ya que por medio de este se pueden identificar algunos vértices claves en la posterior reconstrucción, se puede realizar un proceso de alineación y correspondencia entre las diferentes vistas, hasta el punto de hallar el sólido buscado.

En el caso de la Reconstrucción a partir de tomografías se debe tener mucho cuidado a la hora de seleccionar el método, ya que a pesar de ser muchas imágenes, la información que se tiene es en una sola vista, que es la que llamamos vista axial. Debido a que tenemos una sola vista de las imágenes no es posible utilizar reconstrucción proyectiva, además antes de hacer el proceso de reconstrucción es necesario hacer un proceso de reconocimiento y segmentación, de tal forma que se puedan identificar y diferenciar cada estructura a reconstruir.

Otro aspecto a tener en cuenta es que los algoritmos de Reconstrucción a partir de proyecciones trabajan con elementos geométicos como líneas, puntos (vértices) y aristas, basándose en el parámetro píxel, ya que se analizan imágenes bidimensionales, mientras que los algoritmos de Reconstrucción a partir de imágenes de tomografía son basados en voxeles, ya que no se tienen imágenes bidimensionales, sino un conjunto de imágenes que se comportan como un volumen, por lo tanto se deben aplicar algoritmos basados en voséeles. En este caso se utilizó el Marching Cubes por su eficacia y fiabilidad.

BIBLIOGRAFÍA

[1] Duncan J., Ayache N., *Medical Image Analysis: Progress over Two Decades and Challenges Ahead*, 2000, IEEE Transactions on Pattern Recognition and Machine Intelligence, Vol.22, No.1

[2] Herman G., *Image reconstruction from projections. The fundamentals of computarized tomography*, 1980, Academic Press

[3] Brodlie K., Word J., *Recent Advances in Volume Visualization*, 2001, Computer Graphics Forum, Vo.20, No.2, pp. 125-148

[4] Castleman K., *Digital Image Processing*, 1996, Prentice Hall

[5] Olabarriaga S., Smeulders A., *Interaction in the segmentation of medical images: A survey*, 2001, Medical Image Analysis, Vol.5, 127-142

[6] Cai, W., Walter, S., Karangelis, G., Sakas, G., *Collaborative Virtual Simulation Environment for Radiotherapy Treatment Planning*, 2000, Eurographics 2000, Vol.19, No.3

[7] Jain A., *Fundamentals of Digital Image Processing*, 1989, Prentice Hall [8] Kass M., Witkin A., Terzopoulos D., *Snakes: Active contour models*, 1988¸ Int.J. Comput. Vis.1(4), 321- 331

[8] Kass M., Witkin A., Terzopoulos D., *Snakes: Active contour models*, 1988¸ Int.J. Comput. Vis.1(4), 321- 331

[9] Ji L., Yan H., *Loop-free snakes for highly irregular object shapes*, 2002, Pattern Recognition Letters 23, pp 579-591

[10] Jain A., Flynn P., *Image Segmentation Using Clustering, in Advances in Image Understanding*, 1996, edited by Kevin Bowyer and Narendra Ahuja, IEEE Computer Society Press, pp 65-83

[11] Jain A., Duin R., Mao J., *Statistical Pattern Recognition: A Review*, 2000, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol.22, No.1

[12] Pardo X., Cabello D., *Biomedical active segmentation guided by edge saliency*, 2000, Pattern Recognition Letters, Vol.21, 559-572

[13] Fan J., Yau D., Elmagarmid A., Aref W., *Automatic image segmentation by integrating color-edge extraction and seeded region growing*, 2001, IEEE Trans. on Image Processing, Vol.10, No.10

[14] Aho A., Hopcroft J., Ullman J. *Data estructures and algorithms*, 1987, Addison-Wesley. Reading

[15] Foley, J., Van Dam, A., Feiner, S., Hughes, J., *Introduction to Computer Graphics*, 1996, Addison- Wesley

[16] Lorensen W., Cline H., *Marching Cubes a high resolution 3D surface reconstruction algorithm*, 1987, Computer Graphics, Vol 21, No.4, pp 163-169

[17] Carneiro B., Silva C., Kaufman A., *Tetra-Cubes: An algorithm to generate 3D isosurfaces based upon tetrahedra*, 1996, IX SIBGRAPI, 205-210

[18] Taubin G., *A signal processing approach to fair surface design*, 1995, SIGGRAPH'95 Conference Proceedings, pp.351-358

[19]**.** Politte, D. G. (1990) Image improvements in Positron Emission Tomography due to measuring differential timeofflight and using maximum – likelihood estimation. *IEEE Trans. Nucl. Sci.*, **37**(2):737-742.

[20]**.** Politte, D. G. (1983) Reconstruction algorithms for timeof flight assisted positron emission tomographs. *M. S.*

Thesis, Ed.: Sever Inst. Technol., Washington Univ., St. Louis, MO.

[21]**.** Shepp, L. A., Vardi, Y., Ra, J. B., Hilal, S. K. & Cho, Z. H. (1984) Maximum likelihood PET with real data. *IEEE Trans. Nucl. Sci.*, **31**(2):910-913.

[22]**.** Llacer, J., Veklerov, E., Baxter, L. R., Grafton, S. T., Griffeth, L. K., Hawkins, R. A., Hoh, C. K., Mazziotta, J. C., Hoffman, E. J. & Metz, C. E. (1993) Results of a clinical receiver operating characteristics study comparing filtered backprojection and maximum likelihood estimator images in FDG PET studies. *J Nuc. Med.*, **34**(7):1198-1203.

[23]**.** Wilson, D. W. & Tsui B. M. W. (1993) Noise properties of filtered-backprojection and ML-EM reconstructed emission tomographic images. *IEEE Trans. Nucl. Sci.*, **40**(4): 1198-1203.

[24]**.** Llacer J. & Bajamonde J. (1990) Characteristics of feasible images obtained from real PET data by MLE, Bayesian and sieve methods. En: *SPIE Proc., Digital Image Synthesis and Inverse Opt.*, **1351**:300-312.

[25]**.** Chen, C-T., Ouyang, X., Wong, W. H., Hu, X., Johnson, V. E., Ordonez, C. & Metz, C. E. (1991) Sensor Fusion in image reconstruction. *IEEE Trans. Nucl. Sci.*, **38**(2):687- 692.

[26]**.** Chornoboy, E. S., Chen, C. J., Miller, M. I. & Snyder, D. L. (1990) An evaluation of maximum likelihood reconstruction for SPECT. *IEEE Trans. Med. Imag.*, **9**(1): 99-110.

[27]**.** Liow, J-S. & Strother, S. C. (1991) Practical tradeoffs between noise, quantitation, and number of iterations for maximum likelihood – based reconstructions. *IEEE Trans. Med. Imag.*, **10**(4):563-571.

[28]**.** O' Sullivan, F., Pawitan, Y. & Haynor, D. (1993) Reducing negativity artifacts in emission tomography: Post-Prepro-cessing filtered backprojection solutions. *IEEE Trans. Med. Imag.*, **12**(4): 653-663.

[29]**.** Politte, D. G. & Snyder, D. L. (1988) The use of constraints to eliminate artifacts in maximum likelihood image estimation for Emission Tomography. *IEEE Trans. Nucl. Sci.*, **35**(1):608-610.

[30]**.** Snyder, D. L., Miller, M. I., Thomas, Jr., L. J. & Politte, D. G. (1987) Noise and edge artifacts in maximum likelihood reconstructions for emission tomography. *IEEE Trans.Med. Imag.*, **6**(3):228-238.

[31]**.** Nuyts, J., Suetens, P. & Mortelmans, L. (1993) Acceleration of maximum likelihood reconstruction, using frequency amplification and attenuation compensation. *IEEETrans. Med. Imag.*, **12**(4): 643-652.**32.** Levitan, E. & Herman, G. T. (1987) A maximum *a posteriori*probability expectation maximization algorithm for image reconstruction in Positron Emission Tomography. *IEEE Trans. Med. Imag.*, **6**(3):185-192.

[33]**.** Herman, G. T. & Odhner, D. (1991) Performance evaluation of an iterative image reconstruction algorithm forPET. *IEEE Trans. Med. Imag.*, **10**(3):336-364. **34.** Fessler, J. A. & Hero, A. (1994) Space alternating generalized expectation maximization algorithm. *IEEE Trans. Sig. Proc.*, **42**: 2667-2677.

[35]**.**Daube-Witherspoon, M. E. & Muehllehner, G. (1986) An iterative image reconstruction algorithm suitable for volume ECT. *IEEE Trans. Med. Imag.* **5**(1): 16-22.

[36]**.** De Pierro, A. R. (1993) On the relation between the ISRA and the EM algorithm for PET. *IEEE Trans. Med. Imag.* **12**(2): 328-333.

[37]**.** Anderson, J. M. M., Mair, B. A., Rao, M. & Wu, C. -H. (1997) Weighted least-squares reconstruction methods for positron emission tomography. *IEEE Trans. Med. Imag.* **2**(16): 159-165.

[38]**.** Hudson, H. M. & Larkin, R. S. (1994) Accelerated image reconstruction using ordered subsets of projection data. *IEEE Trans. Med. Imag.*, **13**(4): 601-609.

[39]**.** Lange, K. (1990) Convergence of the EM image reconstruction algorithms with Gibbs smoothing. *IEEE Trans. Med. Imag.*, **9**(4)439-446.

[40]**.** Nunez, J. & Llacer, J. (1990) A fast Bayesian reconstruction algorithm for Emission Tomography with entropy
prior converging to feasible images. *IEEE Trans. on Med.Imag.* , **9**(2): 159 -171.
[41]**.** Phillips, P. R. (1989) Bayesian statistics, factor analysis, and PET images – Part I: Mathematical background. *IEEE Trans. Med. Imag*., **8**(2):125-132.
[42]**.** Liang, Z., Jaszczak, R. & Hart, H. (1988) Study and performance evaluation of statistical methods in image processing. *Comput. Biol. Med*., **18**(6):395-408.
[43]**.** Liang, Z. (1988) Statistical models of *a priori* information for image processing. En: *Proc. SPIE, Medical Imaging II: Image Formation, Detection, Processing, and Interpretation*,
Vol. **914**, pp. 677-683.
[44]**.** Alenius, S. & Ruotsalainen, U. (1997) Bayesian image reconstruction for emission tomography based on median root prior. *Eur. J. Nuc. Med*., **24**(3): 258-265.
[45]**.** Green, P. (1990) Bayesian reconstructions from Emission Tomography data using a modified EM algorithm. *IEEE Trans. Med. Imag*., **9**(1):84-93.
[46]**.** Browne, J. & de Pierro, A. B. (1996) A row-action alternative to the EM algorithm for maximizing likelihood
in emission tomography. *IEEE Trans. Med. Imag*., **15**(5): 687-699.
[47]**.** Kontaxakis, G. & Tzanakos G. (1996) A stopping criterion for the iterative EM-MLE image reconstruction for PET. En: *SPIE Proc., Medical Imaging: Image Proc.* Ed.: Loew M. H. & Hanson K. M., **2710**: 133-144.
[48]**.** Strauss, L. G. (1996) Fluorine-18 deoxyglucose and falsepositive results: a major problem in the diagnostics of oncological patients. *Eur. J. Nucl. Med*., **10**(23): 1409-1415.
[49]**.** Hoffman, E. J., Cutler, P. D., Digby, W. M. & Mazziotta, J. C. (1990) 3-D phantom to simulate cerebral blood flow and metabolic images for PET. *IEEE Trans. Nucl. Sci*., **37**(2): 616-620.
[50]**.** Fessler, J. A. & Hero, A. (1995) Penalized maximum–likelihood image reconstruction using space-alternating generalized EM algorithms. *IEEE Trans. Imag. Proc*., **4**:1417-1429.
[51]**.** Kontaxakis, G., Strauss, L. G., Thireou, T., Ledesma-Carbayo, M. J., Santos, A., Pavlopoulos, S. & imitrakopoulou-Strauss, A. (2002) Iterative Image Reconstruction for Clinical PET Using Ordered Subsets, Median Root Prior and a Web-based Interface. *Mol.*

Imag. Biol., **4**(3): 219-231. G. Kontaxakis et al. *Re*v.*R.Acad.Cienc.Exact.Fis.Nat. (Esp), 2002; 96* 57