Universidad Autónoma de Querétaro
Facultad de Informática
Maestría en Ciencias Computacionales

"MÉTODO INTERACTIVO MULTIESCALA DE SEGMENTACIÓN DE IMÁGENES"

TESIS

Que como parte de los requisitos para obtener el grado de Maestro en Ciencias Computacionales

Presenta:
J Bertoldo Ramos Montoya

Dirigido por:
Dr. Iván R. Terol Villalobos

SINODALES

Dr. Iván R. Terol Villalobos
Presidente — Firma

Dra. Norma Maricela Ramos Salinas
Secretario — Firma

M.C. Rosa María Romero González
Vocal — Firma

M.S.I. Lilia López Vallejo
Suplente — Firma

MC Alberto Lamadrid Alvarez
Suplente — Firma

M.C. Alejandro Santoyo Rodríguez
Director de la Facultad

Dr. Luis Gerardo Hernández Sandoval
Director de Investigación y Posgrado

Centro Universitario
Querétaro, Qro.
Mayo de 2007
México

RESUMEN

En esta tesis se presenta un algoritmo o método para hacer la segmentación de una imagen y además nos permita pasar de una segmentación más fina a una menos fina o viceversa. Se explican los algoritmos principales que se crearon para segmentar y fusionar regiones. La imagen de entrada se procesa para poder obtener segmentación más adecuada. Después se crea la imagen de vertientes de la imagen de entrada, la cual contiene la partición más fina de la imagen, al mismo tiempo se genera un grafo *Minimum Spanning Tree* (MST), que se crea en el proceso de inundación durante la creación de las vertientes de la imagen procesada. A cada partición de la imagen de vertientes se le asigna un nodo del MST, cada arista de MST esta valorada, este valor se asigno durante la creación del MST durante el proceso de inundación de la imagen procesada, el valor que se asignó a las aristas es el cálculo del área de las vertientes o zonas que forman el dique o separación de aguas durante el proceso de inundación de vertientes, se podrán utilizar otros valores como altura o volumen de las vertientes. Una vez que ya esta creado el MST, lo llamaremos MST asociado a la imagen, el cual está formado por nodo y aristas valoradas, cada nodo representa un mínimo en la imagen de vertientes (Cada partición contiene su mínimo), Todas las operaciones serán efectuadas sobre el grafo asociado MST, a partir de este grafo MST y la imagen de vertientes, se genera la nueva imagen segmentada. Si hablamos de subárbol nos referimos al MST, y las regiones son a la imagen, ya que cada región en la imagen tiene asociado un subárbol en el MST asociado a la imagen. Las operaciones sobre el grafo son dividir cierto subárbol en n subárboles, a los subárboles generadas se podrán volver a dividir. Otra operación es la de fusionar subárboles. Se hacen pruebas finales sobre diversas imágenes; de rostros, de cerebro, etc., se ve un capítulo sobre fundamentos de morfología matemática y otro sobre estructura de datos y algorítmica

(**Palabras Clave:** Vertientes, *Minimum Spanning Tree*, Partición)

SUMMARY

In the present work, an interactive method for segmenting images is proposed. This method is based on a multi-scale approach that permits going from finer scales to coarser ones (also from coarser scales to finer ones). The main goal of the proposal consists in proposing a simple method that allows carrying out a segmentation of the original image in a simple way. Thus, user with a minimum of knowledge in image processing can compute a segmentation of the image. The method is based on a minimum spanning tree structure (MST) which is built during immersion process of the image to compute a catchment basins. Each catchment basin is associated with a given node in the MST, where the edges of the MST graph are weighted according to some measures (high, volume, area) that form the criteria to extract the different segmentations of the image. A pre-processing step is carried out on the image in order to simplify image segmentation process. After the MST is completed, a set of transformations can be applied on the MST structure. Working with the MST structure has an important advantage because of the small number of nodes to be processed in the MST structure with regard to the information of the image. The two principal transformations working on the graph are the split and merge operations. The changes on the MST by the splitting and merging nodes will imply the same changes on the image by merging and splitting the associated regions. The first operation called split allows us to compute a set of sub-graphs according to the regions required by the user. Each region of the segmented image will be associated to a sub-graph. In a recursive way the user can split again each sub-graph in another set of sub-graph. On the other hand, the second operation permits the merging two of sub-graphs. In this case, the regions of image to be merged are selected by the user. This operation is only made if there is an edge connecting the sub-graphs on the MST. Both basic operations permit building a set of more complex operations on the graph. The method proposed in the present work was validated with a set of images with different structural characteristics. This validation shows the performance of the proposal.

(Key Words: Multi-scale, Minimum Spanning Tree, Segmentation, Partition)

A mi director de tesis, Dr. Iván Terol Villalobos, cuya guía y enseñanzas fueron indispensables para la realizar la presente tesis. A mi esposa Laura, a mis hijos Ricardo y Ximena, a mis padres, hermanos, amigos que siempre me alentaban.

AGRADECIMIENTOS

A Dios, a quien le debemos todo el bien que hacemos. Quiero agradecer al Dr. Iván Terol Villalobos, por su valioso tiempo dedicado a este proyecto, su gran paciencia y compromiso.

Y a mis sinodales: Dra. Norma Maricela Ramos Salinas, M.C. Rosa María Romero González, M.S.I. Lilia López Vallejo, MC Alberto Lamadrid Alvarez por sus acertadas observaciones en la revisión de este documento.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE FÓRMULAS

ÍNDICE DE DEFINICIONES

1. INTRODUCCIÓN

Objetivo: Minimizar el esfuerzo requerido del usuario para obtener la salida deseada. Imágenes reales tales como imágenes en medicina o de materiales, difícilmente se puede proveer al especialista de un método automático puesto que para su realización se requiere del conocimiento sobre las imágenes por parte del especialista, quién frecuentemente toma sus decisiones al analizar la (o las)

Varias propiedades importantes son tomadas en cuenta para la creación de una técnica multi-escala: a) Invariancia a las rotaciones, b) Invariancia a las traslaciones, c) Invariancia bajo cambios de iluminación. Otros requerimientos sobre el efecto de la transformación deben ser tomados en cuenta. En particular, d) la transformación debe permitir realmente una simplificación de las imágenes, e) no debe crear nuevas estructuras en las escalas menos finas, f) (causalidad) las escalas menos finas deben ser originadas por los que pasa en las más finas.

La técnica tradicional de segmentación de imágenes en morfología matemática (MM) por medio de la transformación por vertientes (*watershed*) constituye el paradigma de la segmentación en MM. El problema de esta técnica es la sobre-segmentación de las imágenes. La solución a este problema consiste en la detección de marcadores que señalen cada una de las regiones de interés en la imagen. El conjunto formado por la transformación por vertientes y los marcadores se presenta como una herramienta interesante para crear algoritmos de segmentación interactivos.

La interacción con el usuario: Generalmente se aplica un proceso de adaptación de imagen, este es para eliminar una posible sobre segmentación, ajustar lo máximo posible las regiones a la imagen original, etc. Luego se define el nivel de resolución inicial, después se ejecuta el proceso *Minimum Spanning Tree*; este proceso segmenta la imagen en el nivel deseado.

A partir de la imagen segmentada se puede fusionar regiones adyacentes o re segmentar regiones hasta obtener la segmentación deseada o aislamiento de una región (ver ***Figura 1.1.***). Para esto se tienen las siguientes herramientas:

Selección de un nivel de resolución: Dado un número específico de regiones (N) el algoritmo proporciona las N regiones más significativas. Esta acción es global, es decir se aplica a toda la imagen, sin embargo el usuario puede estar interesado en algunas particiones de la imagen, que desee re segmentar (más finas) o fusionar (más ásperas), para estos casos se cuenta con 2 herramientas.

- *Partición más fina*: Selecciona con un clic la región a re segmentar e introduce el número de particiones deseadas.

- *Partición menos fina*(o más áspera): Sólamente con el ratón arrastra una primera partición hasta una segunda partición vecina.

Figura 1.1. a) Se marcan con el ratón las regiones correspondientes a la mejilla y frente.

Figura 1.1. b) Se obtiene la fusión de las regiones marcadas en el rostro (aplicando el procedimiento fusiona)

Figura 1.1. c) Se re segmenta en dos una región,

Figura 1.1. Fusiona y re-segmenta.

1. I. Organización de la tesis.

El algoritmo para lograr la segmentación multiescala de una imagen utiliza un grafo especial llamado MST (*Minimum Spanning Tree*), en esta tesis se estudia inicialmente lo que se necesita saber respecto a Imágenes, posteriormente se estudian grafos y árboles, recorridos de MST, etc. También se explica paso a paso como se va creando el algoritmo, y finalmente se muestran los resultados.

1. II. Material y Equipo empleado.

La interfase gráfica con el usuario o la interfase que interactúa con el usuario final para realizar una segmentación más adecuada de la imagen de interés a los requerimientos del usuario, está basada en un programa en Visual C++ 5.0 que proporciono el Dr. Iván R. Terol Villalobos, al cual se le agrego un menú con los siguientes elementos *minumum spanning tree*, divide *mst* y fusiona *mst*, se utilizaron una gran variedad de funciones y procedimientos ya elaborados del programa para terminar estas opciones del menú agregado. Todas las imágenes de prueba fueron proporcionadas por el Dr. Ivan R. Terol Villalobos

El procesamiento de las imágenes se llevó a cabo en una computadora laptop portátil con las siguientes características: microprocesador AMD k6 533 GHz, 256 Mb de RAM y HD de 5 Gb.

2. REVISION DE LITERATURA.

2. I. Filtrado Morfológico.

La conclusión sobre el estudio de estos problemas nos lleva a buscar proponer un método de segmentación de imágenes tomando en cuenta tanto las técnicas multi-escala como la interactividad del usuario con las imágenes. El método que se buscará proponer estará basado en la metodología de procesamiento de imágenes conocida como morfología matemática (MM). En esta técnica, el filtrado morfológico juega el papel primordial en la segmentación de las imágenes. Por esta razón, parte del trabajo a desarrollar, estará enfocada al filtrado de las imágenes y de manera particular al filtrado morfológico conexo.

Es bien conocido que la idea intuitiva de la noción de conectividad está ligada a la segmentación de imágenes binarias. Es decir, el objetivo es el de separar las componentes conexas, presentes en la imagen, en un conjunto de formas elementales que van a ser procesadas de manera individual. Las transformaciones conexas, toman decisiones de manera individual sobre cada forma elemental. Idealmente, estas formas elementales corresponden a nuestra percepción visual de las principales partes de un objeto.

En el caso de imágenes binarias compuesta de formas elementales asimilables a partículas convexas que se tocan formando componentes conexas más complejas (componentes no convexas) existe una solución simple utilizando la metodología de procesamiento y análisis de imágenes conocida como morfología matemática (MM). En este caso, esta solución consiste en determinar la función distancia sobre la imagen binaria y calcular la transformación conocida como vertientes (*watershed*) sobre la inversa de la función distancia (frecuentemente la función distancia filtrada). Esto permite en general separar correctamente las formas elementales.

Cuando se tienen formas elementales más complejas, el uso de transformaciones tales como el esqueleto morfológico o la bisectriz condicional son utilizadas. En el caso de imágenes en niveles de gris, el problema permite tratar cualquier tipo de formas, sin embargo es más complejo del punto de vista de tratamiento de imágenes.

Es fundamentalmente en imágenes en niveles de gris, que el filtrado morfológico juega un papel fundamental utilizando la MM. Frecuentemente, el filtrado morfológico es usado para detectar los objetos o regiones de interés de una imagen. En el caso de imágenes en niveles de gris, la técnica de segmentación conocida como vertientes y marcadores (watershed-plus-marker approach) es utilizada en MM [Meyer, F., Beucher S., et al 1990].

La idea principal en esta técnica es el de encontrar un conjunto de marcadores que señalan las zonas de interés de la imagen. Es en esta etapa que el filtrado morfológico juega el papel primordial, en este caso, en la detección de marcadores. En particular, dentro de las diferentes familias de filtros morfológicos, el filtrado por reconstrucción, que forma una clase de filtros morfológicos conexos, se presenta como una de las herramientas fundamentales para la realización de esta tarea. Después de haber detectado un conjunto de marcadores, este conjunto es utilizado para modificar los mínimos del gradiente de la imagen (generalmente).

La transformación por vertientes (*watershed*) es aplicada posteriormente sobre el gradiente modificado para extraer los contornos de las regiones marcadas. Esta técnica presenta ciertas limitaciones, entre ellas una de las más importantes es un problema de resolución. En efecto, principalmente sobre las regiones de tamaño pequeño o regiones delgadas y alargadas, frecuentemente es imposible o muy difícil de imponer marcadores.

Debido principalmente a este inconveniente, la técnica de segmentación por zonas planas fue propuesta [Crespo, J., et al 1993].

En esta técnica, de manera similar, un conjunto de marcadores debe ser detectado sobre la imagen original. Posteriormente, un proceso de fusión de zonas planas a partir del conjunto de marcadores es realizado sobre la imagen original hasta obtener una partición de la imagen. La etapa de modificación de la homotopía (imposición de mínimos sobre el gradiente de la imagen) es eliminada. Generalmente un pre-procesamiento de la imagen original es realizado utilizando filtros por reconstrucción bajo la forma de filtros alternados secuenciales o filtros alternados bajo el operadores supremo o ínfimo [Crespo, J., Schafer et al 1994]; [Crespo, J., Serra, J., Schafer et al 1995]; [Salembier, P., Serra, J. et al 1995]

Este tipo de trabajo ha motivado el uso del filtrado morfológico, no únicamente como una herramienta para la detección de marcadores, sino propiamente como una herramienta de segmentación que permite aislar regiones para ser procesadas de manera individual. Sin embargo, aunque las transformaciones llamadas por reconstrucción han permitido resolver diferentes problemas de procesamiento de imágenes, estas presentan un problema principal llamado encadenamiento (leakage), en el cual dos o más formas elementales pueden ser tratadas como una sola. Puesto que en las imágenes reales en niveles de gris pueden existir conexiones delgadas entre las formas elementales, algunas regiones que deberían ser separadas para ser analizadas de manera individual cada una de ellas o eliminadas de la imagen, son conservadas como parte de otra forma elemental.

Esto implica que, el concepto clásico de arco-conectividad ligado a las transformaciones por reconstrucción, presenta limitaciones [Salembier, Ph. and Oliveras, A. et al 1996]. Otra de las limitaciones del concepto clásico de arco-conectividad es el hecho de no poder tratar un conjunto de componentes conexas separadas a cierta distancia entre ellas como una sola forma elemental.

Este tipo de problemas se presenta frecuentemente en física de materiales, imágenes médicas (por ejemplo de células), etc., donde un conjunto de

componentes conexas forman una sola identidad y deben ser tratadas como tal para entender el fenómeno que dio lugar a este tipo de formas.

Una solución elegante a este tipo de problemática es la generalización de la noción de conectividad. Esta noción fue establecida por [Serra, J. et al 1988] a partir del concepto de clases conexas. Sin embargo, dicho concepto empezó a ser realmente explotado en los últimos cinco años.

Diferentes trabajos se han realizado sobre el tema de conectividad, entre estos, los estudios más relevantes fueron realizados por Heijmans [Heijmans, et al 1997], [Ronse, C. et al 1998], [Serra, J. et al 1988] y [Salembier, Ph. and Oliveras, A. et al 1996]; [Salembier, Ph., Garrido, L., et al 2000]. Heijmans introduce la noción de operadores sobre granos, mientras que Ronse propone otra familia de axiomas para caracterizar la conectividad. Ambos trabajos son estudios puramente binarios lo que resulta paradoxal puesto que el filtrado morfológico conexo (en particular las transformaciones por reconstrucción) han probado su eficiencia en segmentación de imágenes, codificación y predicción de movimiento. De esta problemática parte el trabajo de Serra quién caracteriza la conectividad desde un punto de vista general: conectividad en retículas.

Finalmente, Salembier se enfoca fundamentalmente al problema práctico y algorítmico. Otros trabajos interesantes en la generalización de la conectividad es el concepto de nivelaciones (levelings) que son filtros morfológicos conexos propuestos en 1998 [Meyer, F., et al 1998a], [Meyer, F., et al 1998b] ; [Meyer, F., Maragos, P., 2000]. Meyer se aleja de la axiomática propuesta por Serra para definir las clases conexas enfocándose a estudiar la conectividad a partir de las transiciones en las imágenes de entrada y salida. La consecuencia directa de este estudio es la proposición de una clase de transformaciones conexas a las cuales llama nivelaciones.

Este estudio permite tener un soporte teórico para estudiar la noción de conectividad en el caso de imágenes en niveles de gris, no únicamente en el sentido arco-conexo, si no también extender dicha noción. A pesar de que el trabajo de Meyer no muestra una relación entre las nivelaciones y la noción de clases conexas, [Serra, J., et al 2000] establece dicha relación dando un énfasis principal en noción de marcadores. Puesto que en general el filtrado es frecuentemente utilizado en la detección de un conjunto de marcadores con el objetivo de segmentar una imagen, la generación de nuevos filtros morfológicos usando la noción de marcadores enriquece el concepto de filtro morfológico.

Las consecuencias más importantes de lo comentado anteriormente son:

1) La técnica de vertientes y marcadores muestra problemas de resolución. Una opción a este tipo de problema es la técnica conocida como segmentación por zonas planas.

2) La conectividad clásica conocida como arco-conectividad, presenta como noción, ciertas inconveniencias en el tratamiento de imágenes en niveles de gris.

La conectividad extendida a partir de la noción de nivelaciones (levelings) o más importante aún, la definición de clases conexas por Serra, permite tener una visión más amplia del tratamiento de imágenes. Esto permite pensar en la generación de nuevas transformaciones morfológicas.

2. II. Filtros no crecientes multi-escala.

Parte del interés de este trabajo se dirige al estudio del filtrado morfológico bajo las nociones de conectividad y marcadores para la proposición de un método de segmentación interactivo multi-escala. Uno de los estudios que se realizarán en este proyecto se enfocará a una clase de filtros morfológicos no crecientes que se

describen a continuación. Sin embargo, durante el desarrollo de la tesis y conjunto con mi director de tesis, otras familias de filtros actualmente bajo estudio podrían ser usadas para la propuesta del método de segmentación.

Esta clase de transformaciones no crecientes fueron propuestas en [Terol-Villalobos, I., R., 1995]; [Terol-Villalobos, I. R., 1996a]; [Terol-Villalobos, I., R., 1996b], a las cuales se les llamó filtros morfológicos por pendientes (MSF). Su objetivo era el poder modificar la imagen gradiente sin necesidad de imponer marcadores como en el caso de la técnica tradicional vertientes y marcadores, y poder utilizar la transformación de vertientes directamente sobre la imagen filtrada sin imponer marcadores. Para su construcción, criterios de gradientes morfológicos fueron utilizados.

Su extensión a familias de filtros secuenciales y su estudio desde el punto de vista de mapeos de actividad, permitió enriquecer esta familia de filtros [Terol-Villalobos, I. R. and Cruz-Mandujano, et al 1998a]; [Terol-Villalobos, I. R., et al 1998b]. Dado que estos filtros no son transformaciones conexas, un estudio reciente permitió extender estos filtros sobre particiones usando la noción de zona plana [Terol-Villalobos, I.R., et al 2001] con el fin de generar MSF conexos. Un resultado fundamental en este estudio es que estos filtros no crean nuevos mínimos ni máximos lo cual nos permitió definir una técnica multi-escala de filtrado.

Dado que las transformaciones básicas, la dilatación y la erosión sobre particiones fueron propuestas en este trabajo, todas las transformaciones morfológicas sobre particiones pueden ser definidas. Entre ellas, la apertura y la cerradura sobre particiones fueron utilizadas para la creación de nuevos mapeos de contraste [Mendiola-Santibañez, J. D., Terol-Villalobos, I.R., et al 2001a], [Mendiola-Santibañez, J. D., Terol-Villalobos, I.R., et al 2001b], [Mendiola-Santibañez, J. D., Terol-Villalobos, I.R., et al 2002].

3 METODOLOGIA

PRIMERA PARTE:

LOS FUNDAMENTOS**.**

3. I. Introducción.

Morfología es un método de procesamiento de imágenes que nació después de los años 60 y desde entonces ha ido ganando una gran popularidad e importancia [Serra 1988]. Actualmente es aceptada por una gran variedad de aplicaciones de análisis de imágenes como una útil caja de herramientas. Un solido conocimiento de esta metodología permite desarrollar rápidamente algoritmos y programas para resolver problemas de segmentación y reconocimientos de patrones.

La morfología tiene aplicaciones en áreas como medicina (Imágenes de resonancia magnética, rayos X, etc), biología, radar, sonar, inspección industrial, robótica, ciencia de materiales, huellas digitales, identificaciones, reconocimiento de documentos, etc. Trabaja sobre imágenes binarias y de niveles de gris, así como con imágenes de color y 3-D. A priori cualquier aplicación que involucren imágenes que comparten un conjunto definido de características (Tamaño, y figura de los objetos a ser detectados) es un buen candidato para una solución morfológica.

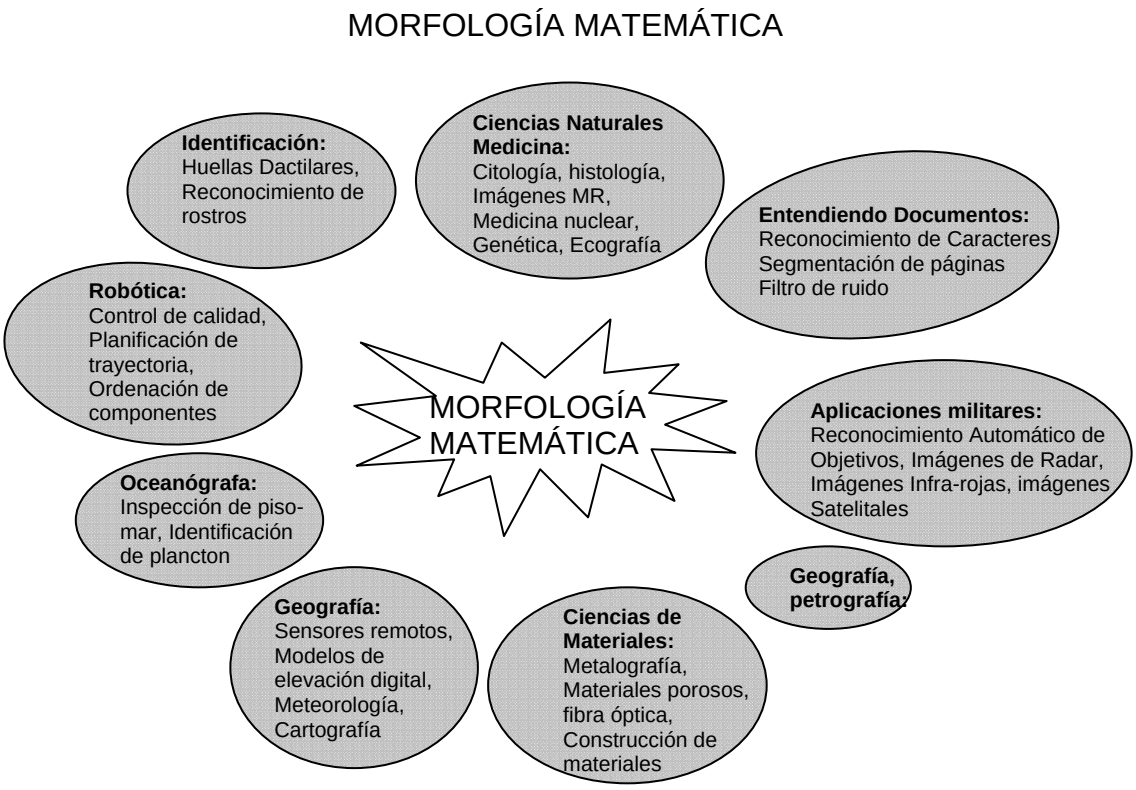


Figura 3.1. Campos de aplicación del análisis morfológico de imágenes. [Luc Vicent at al 1997]

La propuesta de la morfología hacia el análisis de imágenes es natural y atractiva: Esta comienza considerando las imágenes binarias como conjuntos, y las imágenes en niveles de gris como funciones o relieves topográficos. Esos conjuntos y funciones son entonces transformados—en el dominio espacial—por medio de operadores morfológicos, cuyas definiciones están basadas usualmente en elementos estructurantes, figuras particulares que son trasladadas sobre la imagen y usadas como prueba. De estas operaciones básicas, se derivan complejos operadores, que son usados para objetivos más específicos tales como detección de vertientes y salientes, extracción de líneas de valles y crestas en imágenes de niveles de gris (Mínimos y máximos), etc.

Dos aspectos muy importantes de la morfología matemática son la segmentación y la granulometría.

Segmentación es la tarea de particionar una imagen en regiones significativas. Esto es equivalente a detectar y extraer de la imagen los objetos o zonas de interés. Estos objetos o zonas pueden ser contados, medidos, clasificados, etc.

Granulometría se refiere al tamaño de la información de las imágenes sin antes de segmentar,

La segmentación multiescala consiste de una familia de particiones que representan a la imagen en diferentes niveles de resolución, El nivel más áspero considera la imagen como un todo (una sola región) y las particiones finas están siempre incluidas en algunas ásperas. Esto significa que un nivel más fino es obtenido de la re segmentación de las regiones de un nivel específico. En nuestro método interactivo multiescala, se define primero el número de particiones deseadas inicialmente, es decir el nivel de granularidad, posteriormente se puede re segmentar regiones para obtener otras con niveles más finos, también se puede fusionar regiones adyacentes para obtener regiones más ásperas o menos finas.

3. II. Fundamentos de morfología matemática.

Del punto de vista práctico, la morfología matemática (MM) es una teoría del tipo conjuntista, en donde la imagen puede ser vista como un conjunto con el cual se va hacer interactuar otro conjunto de prueba definido a priori en base a un conocimiento tanto de las imágenes a tratar, así como de las transformaciones morfológicas. Este conjunto de prueba llamado elemento estructural o de estructura interactúa con la imagen para extraer información de forma de la misma. En MM existe toda una gama de transformaciones, cada una de ellas con ciertas características que la caracterizan, que permiten resolver diferentes problemas de procesamiento de imágenes. Sin embargo, buena parte de estos filtros son construidos a partir de las transformaciones de base conocidas como la erosión y la dilatación.

3. II.1. Erosión y dilatación morfológicas.

Las transformaciones básicas en la morfología matemática son la erosión y dilatación morfológicas de un conjunto X por un conjunto μB llamado elemento estructural o estructurante μB, donde B puede tener cualquier forma (cuadrado, disco, recta, etc.) mientras que μ es un factor de escala. Por ejemplo, consideremos el elemento estructural elemental con la forma de un cuadrado (3x3 píxeles) conteniendo el origen, $\hat{B}$ su transpuesto ($\hat{B} = \{-x : x \in B\}$). De esta forma, el elemento estructurante de tamaño μB estará formado por un conjunto de tamaño $(2\mu+1) \times (2\mu+1)$ píxeles.

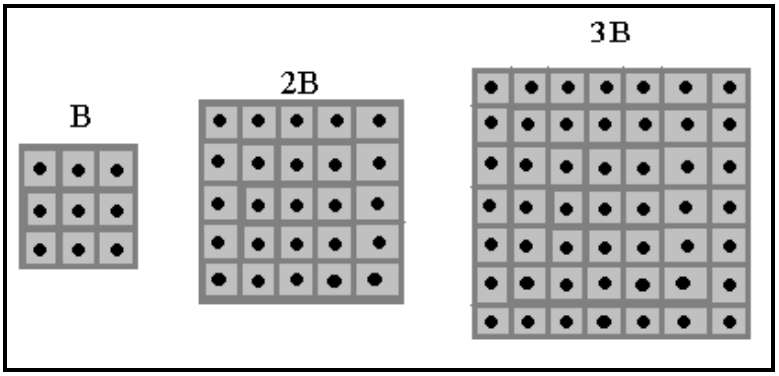


Figura 3.2. Elementos estructurales.

En el caso de imágenes binarias o de conjuntos tenemos;

Definición 1. (Erosión morfológica). Sean dos conjuntos X y λB. La erosión morfológica del conjunto X por el conjunto λB (elemento estructural o estructurante) está dada por;

$$\varepsilon_{\lambda B}(X) = X \Theta \lambda \overset{\vee}{B} = \bigcap_{b \in \lambda \overset{\vee}{B}} X_b = \{x : \lambda B_x \subset X\} \qquad \textbf{(1)}$$

Del punto de vista geométrico, podemos decir que la erosión de un conjunto X por un conjunto λB, es el lugar de centros del elemento estructural cuando este se encuentra totalmente incluido en X. Ver ***Figura 3.3*** (a).

Definición 2. (Dilatación morfológica). Sean dos conjuntos X y λB. La dilatación morfológica del conjunto X por el conjunto λB (elemento estructural o estructurante) está dada por;

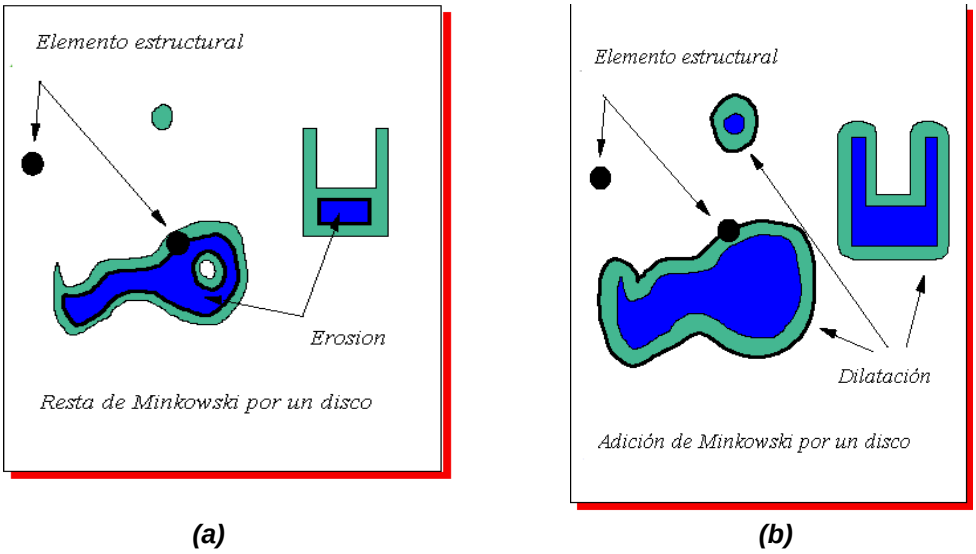
$$\delta_{\lambda B}(X) = X \oplus \lambda \overset{\vee}{B} = \bigcup_{b \in \lambda \overset{\vee}{B}} X_b = \{x : \lambda B_x \cap X \neq \varnothing\} \qquad \textbf{(2)}$$

Observación 3. II.1. Del punto de vista geométrico, la dilatación de un conjunto X por un conjunto λB es el lugar de centros del elemento estructural cuando este conjunto toca al conjunto X. Ver Figura 4 (b).

En estos momentos vale la pena el siguiente comentario. La idea original de estas transformaciones viene del concepto de resta y suma de Minkowski [Luc Vicent at al 1997]. Estas transformaciones están dadas, respectivamente, por las siguientes ecuaciones.

$$X \Theta Y = \left\{ x : \overset{\vee}{Y}_x \subset X \right\} \qquad X \oplus Y = \left\{ x : \overset{\vee}{Y}_x \cap X \neq \varnothing \right\} \quad \textbf{(3)}$$

Cuando el elemento estructural y su transpuesto son iguales (discos, cuadrados por ejemplo), se obtiene el mismo resultado. Sin embargo, con otro tipo de elemento estructural se obtienen resultados diferentes, y principalmente, se puede perder una propiedad fundamental de los filtros que los considera invariantes a la traslación.

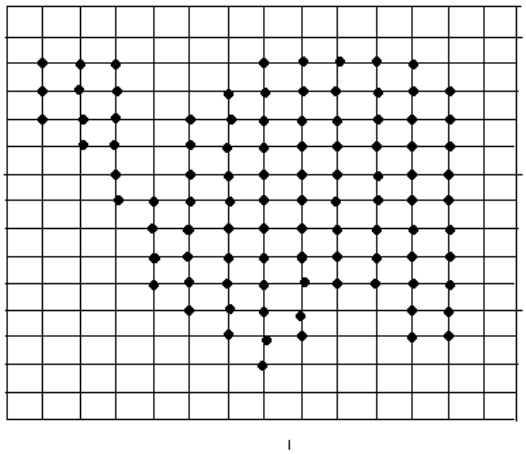


(a) ***(b)***

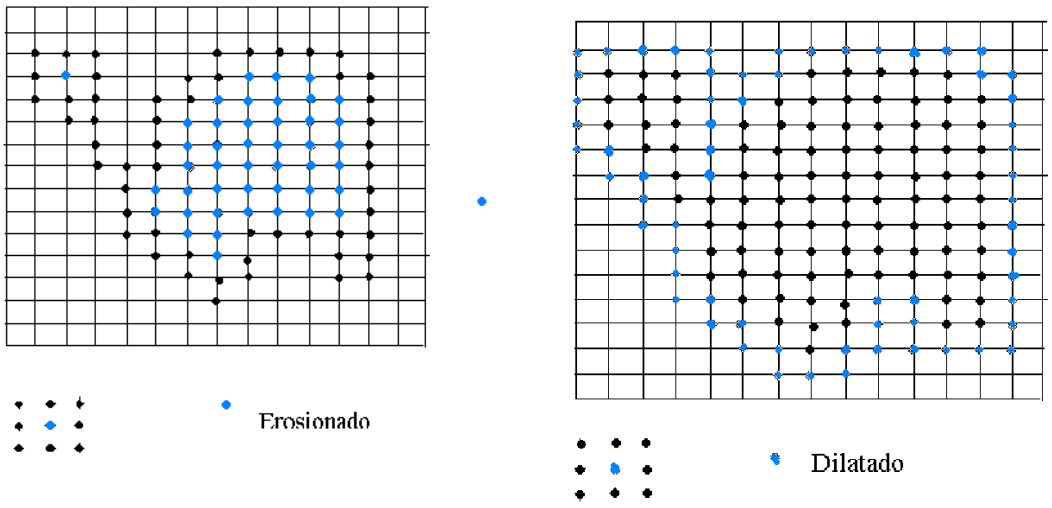
Figura 3.3. a) Erosionado, b) Dilatado.

La ***Figura 3.3. (a)*** y la ***Figura 3.3. (b)*** muestran un ejemplo de imagen erosiona y dilatada. Se utiliza un elemento estructural formado por un conjunto 3x3 píxeles.

Los puntos en color azul de la ***Figura 3.4. (b)*** pertenecen al conjunto erosionado de la ***Figura 3.4. (a)***, mientras que los puntos en color azul en ***Figura 3.4. (c),*** son aquellos que se agregan al conjunto original (***Figura 3.4. (a)***) para formar el dilatado. Es decir, los puntos negros más los puntos azules forman el dilatado.



(a)



(b) ***(c)***

Figura 3.4. a) Conjunto original, b) y c) Erosionado y dilatado de (a) con un elemento estructural 3x3 píxeles respectivamente.

Las definiciones de las transformaciones básicas en el caso de imágenes binarias, se pueden extender al caso de imágenes numéricas. De esta forma, la erosión y dilatación de una imagen *f* por un elemento estructural λB están dadas por;

$$\varepsilon_{\lambda B}(f)(x) = (f \Theta \lambda \check{B})(x) = \wedge\{f(y) : y \in \lambda \hat{B}_x\} \qquad \textbf{(4)}$$

$$\delta_{\lambda B}(f)(x) = (f \oplus \lambda \check{B})(x) = \vee\{f(y) : y \in \lambda \hat{B}_x\} \qquad \textbf{(5)}$$

La ***Figura 3.5.*** muestra ambas operaciones, la dilatación y la erosión, tanto en el caso de imágenes binarias como en el caso de imágenes numéricas.

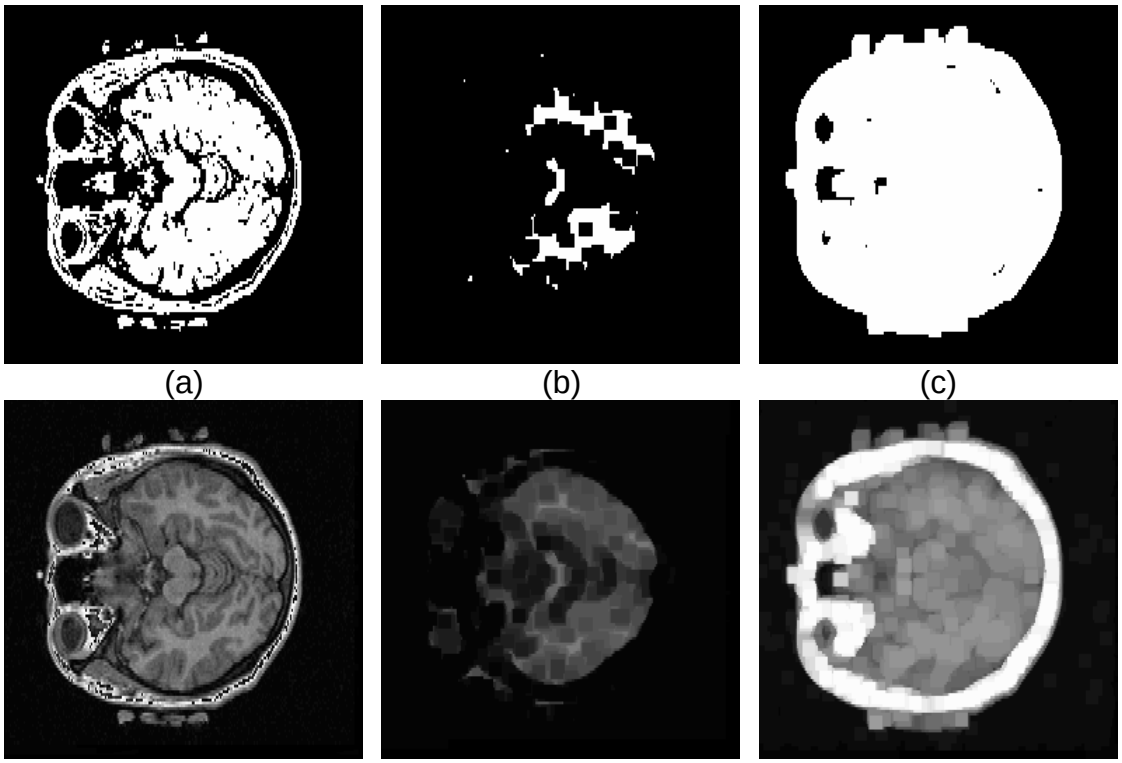


Figura 3.5. a) Imagen binaria, b), c) Erosionado y dilatado binario de imagen (a) respectivamente, d) Imagen original, e), f) Erosionado y dilatado numérico de imagen (d) respectivamente.

3. II.1.1. Propiedades de la erosión y la dilatación morfológicas.

Las siguientes propiedades son validas tanto para el caso binario, como para el caso numérico. De manera general, I representa una imagen binaria o una imagen numérica. El signo de desigualdad $\leq$ representa según sea el caso; $\subset$ la inclusión en el caso binario y $\leq$ el orden usual para el caso numérico.

1. Ambas transformaciones, la erosión y la dilatación, son transformaciones iterativas, es decir;

$$\varepsilon_{\lambda B} = \underbrace{\varepsilon_B \circ\circ\circ \varepsilon_B \varepsilon_B \varepsilon_B}_{\lambda \text{ veces}} (I) \qquad \textbf{(6)}$$

$$\delta_{\lambda B} = \underbrace{\delta_{\lambda B} \circ\circ\circ \delta_{\lambda B} \delta_{\lambda B} \delta_{\lambda B}}_{\lambda \text{ veces}} (I) \qquad \textbf{(7)}$$

2. Las dos transformaciones son crecientes:

Para cualesquiera dos imágenes I_1, I_2 con $I_1 \leq I_2$

$$\varepsilon_{\lambda B}(I_1) \leq \varepsilon_{\lambda B}(I_2) \qquad \textbf{(8)}$$

$$\delta_{\lambda B}(I_1) \leq \delta_{\lambda B}(I_2) \qquad \textbf{(9)}$$

3. No son transformaciones idempotentes:

$$\varepsilon_{\lambda B}\varepsilon_{\lambda B}(I_2) \neq \varepsilon_{\lambda B}(I_2) \qquad \textbf{(10)}$$

$$\delta_{\lambda B}\delta_{\lambda B}(I) \neq \delta_{\lambda B}(I) \qquad \textbf{(11)}$$

4. La erosión es un transformación anti-extensiva y la dilatación una transformación extensiva. Para cualesquier λ

$$\varepsilon_{\lambda B} \leq I \qquad \textbf{(12)}$$

$$\delta_{\lambda B}(I) \geq I \qquad \textbf{(13)}$$

5. La erosión y la dilatación son transformaciones duales con respecto a la complementación.

$$\varepsilon_{\lambda B}(I) = \left[\delta(I^C)\right]^C \qquad \textbf{(14)}$$

6. La dilatación es conmutativa, mientras que la erosión no lo es.

$$X \oplus B = B \oplus X \quad \textbf{(15)}$$
$$X \Theta B \neq B \Theta X \quad \textbf{(16)}$$

7. La dilatación y la erosión usando el origen como elemento estructurante están dadas por,

$$X \Theta \{o\} = X \quad \textbf{(17)}$$
$$X \oplus \{o\} = X \quad \textbf{(18)}$$

En general, para cualquier punto x tenemos,

$$X \Theta \{x\} = X_x \quad \textbf{(19)}$$
$$X \oplus \{x\} = X_x \quad \textbf{(20)}$$

Ejemplo (transformaciones iterativas). Sea el conjunto B dado por el elemento 3x3 píxeles.

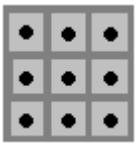


Figura 3.6. Conjunto B dado por el elemento 3x3 píxeles.

Al dilatar un elemento estructural 3x3 con el mismo, obtenemos un conjunto 5x5 que denotamos por 2B

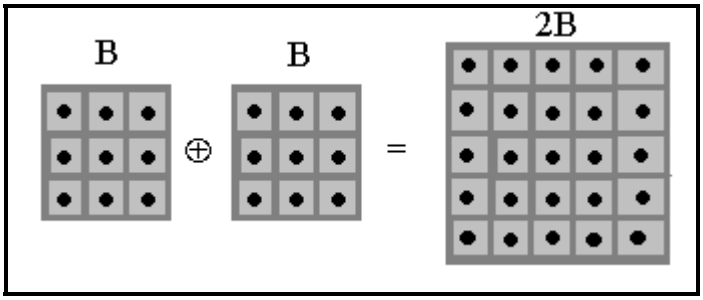


Figura 3.7. Conjunto 5x5 que denotamos por 2B.

Al dilatar 2 veces un elemento estructural 3x3, obtenemos un elemento estructural 7x7 píxeles.

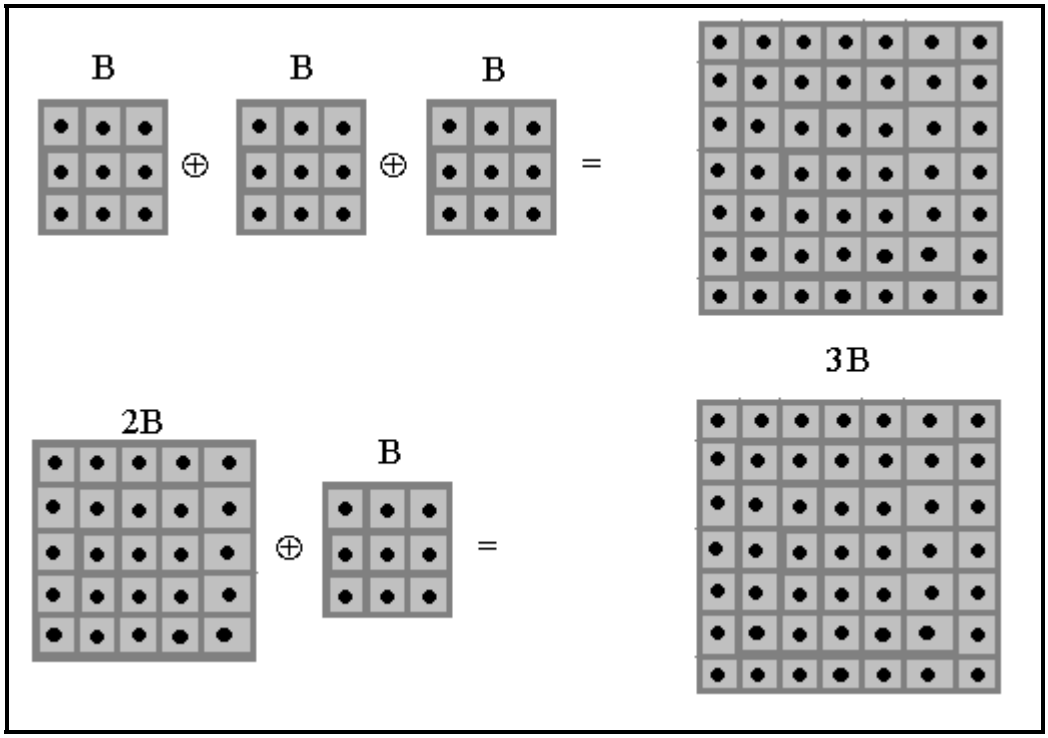


Figura 3.8. Elemento estructural 7x7 píxeles.

De manera general, n elementos estructurales básicos dilatados nos da un conjunto nB de (2n+1)x(2n+1) píxeles .

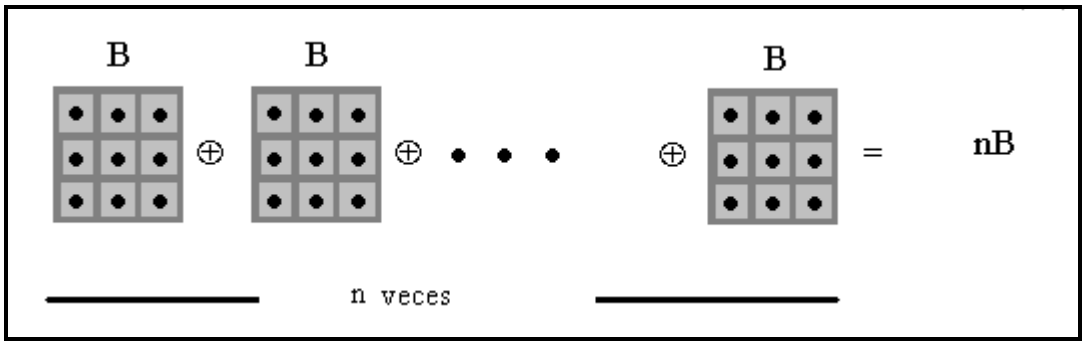


Figura 3.9. n elementos estructurales básicos dilatados nos da un conjunto nB de (2n+1)x(2n+1) píxeles.

Ejemplo. (Descomposición del elemento estructural básico). Consideremos los siguientes conjuntos,

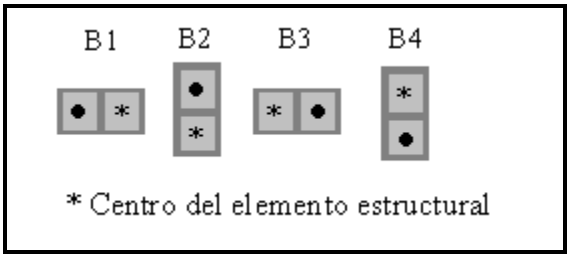


Figura 3.10. Descomposición del elemento estructural básico.

Inicialmente, en el paso ***(1)***, el conjunto ***X*** compuesto de un solo punto es dilatado por el conjunto ***B1***, el resultado obtenido (conjunto de dos puntos) es dilatado por el conjunto ***B2*** obteniendo un conjunto formado por 4 puntos como se muestra en el paso ***(2)***. Posteriormente, este conjunto es dilatado por el elemento ***B3***, con lo que se obtiene un conjunto de 6 puntos mostrado en el paso ***(3)***. Finalmente, este último conjunto de 6 puntos es dilatado por el elemento ***B4*** para obtener el mismo resultado que se tendría al dilatar el conjunto ***X*** por el elemento básico ***3x3*** píxeles.

3. II.2. Apertura y cerradura morfológicas.

Aunque la erosión y la dilatación permiten eliminar regiones de acuerdo a un criterio de tamaño (elemento estructural), estas transformaciones tienen el inconveniente de modificar considerablemente las estructuras que permanecen en la imagen de salida. En particular, los contornos de los objetos son en general modificados. Por otra parte, estas transformaciones no tienen una de las propiedades más interesantes de la MM; la idempotencia. Sin embargo, es posible construir nuevas transformaciones a partir de la dilatación y la erosión. Aún más, podemos decir que la mayor parte de las transformaciones en MM pueden expresarse en función de la dilatación y la erosión.

La combinación de ambas transformaciones permite crear dos nuevas transformaciones, las cuales además de ser transformaciones crecientes, son también transformaciones idempotentes. Es decir, son filtros morfológicos. En efecto, los filtros morfológicos básicos en MM son la apertura morfológica $\gamma_{\mu B}$ y la cerradura morfológica $\varphi_{\mu B}$ usando un elemento estructural μB, con B el elemento estructurante elemental (3x3 píxeles en este trabajo) conteniendo el origen, $\hat{B}$ su transpuesto ($\hat{B} = \{-x : x \in B\}$) y μ un factor de homotecia. De esta forma, el elemento estructural de tamaño μB estará formado por un conjunto de tamaño $(2\mu+1)x(2\mu+1)$ píxeles. La apertura y la cerradura morfológicas en el caso binario se expresan a partir de la dilatación $\delta_{\mu B}$ y la erosión $\varepsilon_{\mu B}$.

$$\gamma_{\lambda B}(X) = ((X \Theta \lambda \hat{B}) \oplus \lambda B) = \bigcup \{\lambda B_x : \lambda B_x \subset X\} \qquad \textbf{(21)}$$

$$\varphi_{\lambda B}(X) = ((X \oplus \lambda \hat{B}) \Theta \lambda B) = \bigcup \{\lambda B : \lambda B_x \subset X\}^C \qquad \textbf{(22)}$$

En la ***Figura 3.11.*** se ilustra el comportamiento de ambas transformaciones. Del punto de vista geométrico, diremos que la apertura de un conjunto X por un conjunto λB, es el área barrida por el elemento estructural cuando se encuentra totalmente incluida en la estructura X. En la ***Figura 3.11. (a)***, se puede observar que las regiones gris oscuro en la estructura X son eliminadas dado que el conjunto λB no puede penetrar al interior de estas regiones. Por otra parte, la cerradura morfológica de un conjunto X será el complemento del área barrida por el conjunto λB cuando este se encuentre completamente incluido en el complemento de X (ver ***Figura 3.11. (b)***).

En la ***Figura 3.12.*** y ***Figura 3.13.*** se ilustran la apertura y la cerradura morfológica, respectivamente, en el caso digital realizando dilatados y erosionados. El elemento estructural es un conjunto básico 3x3. En la ***Figura 3.14.*** se ilustra un ejemplo real de la apertura y cerradura morfológicas. Observe que la apertura tiende a desconectar regiones, mientras que a conectarlas la cerradura.

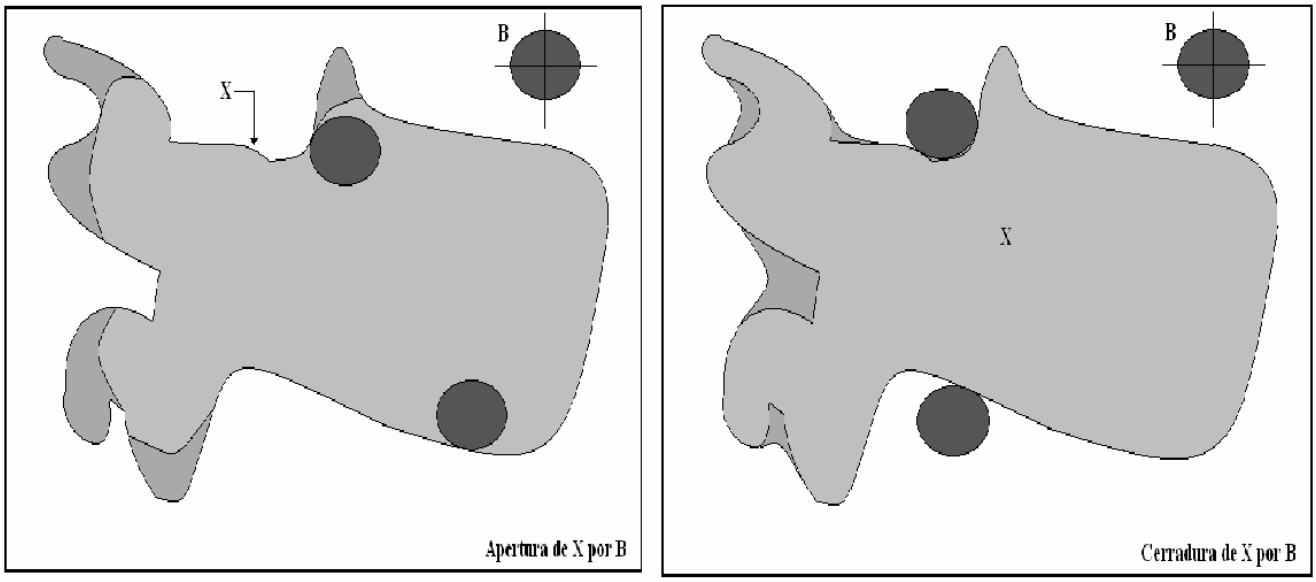


Figura 3.11. a) Apertura morfológica usando un disco como elemento estructural, b) Cerradura morfológica.

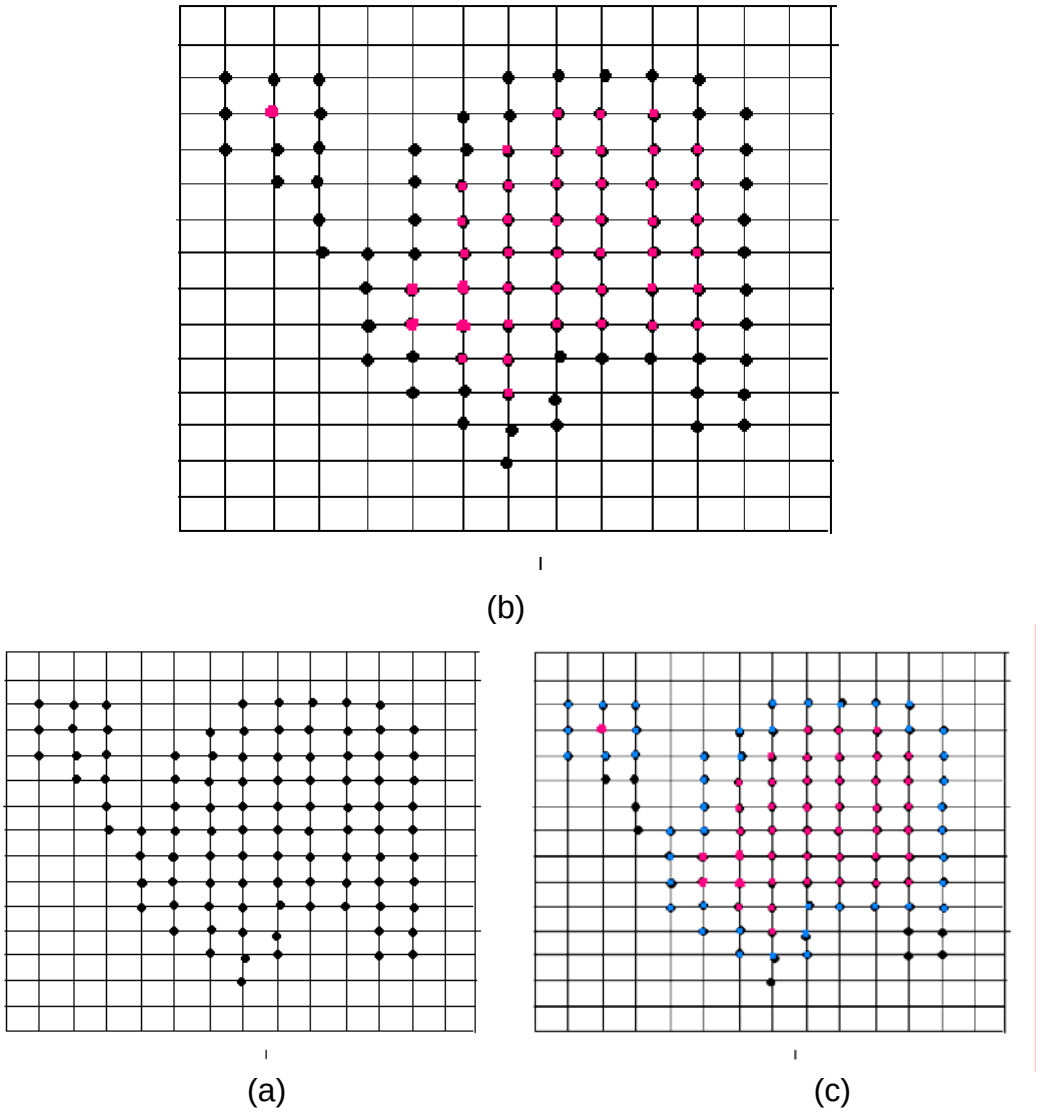
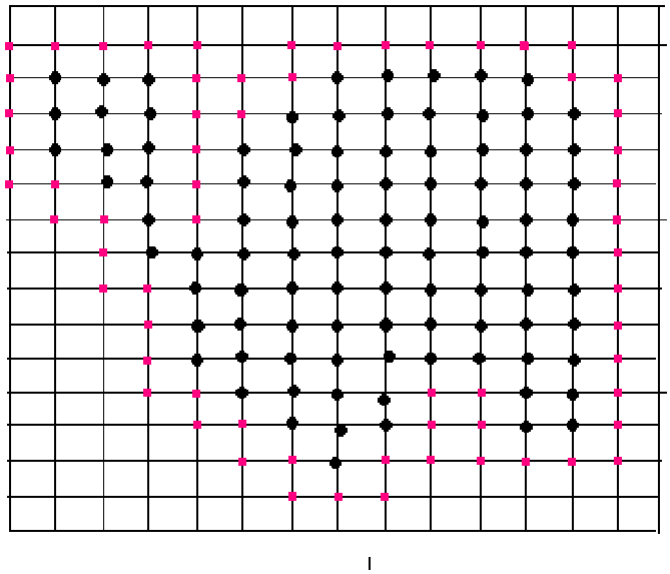
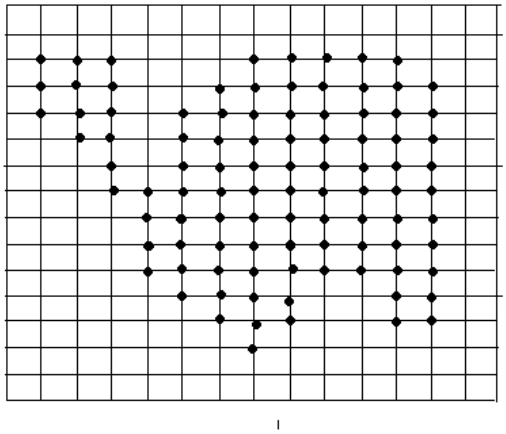
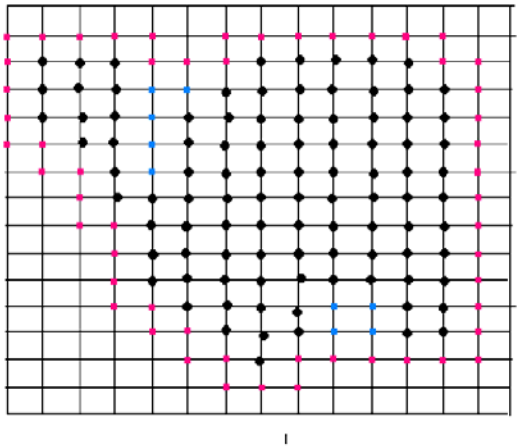


Figura 3.12. APERTURA MORFOLÓGICA. a) Imagen original, b) imagen erosionada (puntos color rojo), c) apertura de la imagen original o dilatado de la imagen erosionada (unión de los puntos color rojo y azul).



(b)





(a) (c)

Figura 3.13. CERRADURA MORFOLÓGICA. a) Imagen original, b) imagen dilatada (unión de los puntos color negro y rojo), c) Cerradura de la imagen original o erosionada de la imagen dilatada (unión de los puntos color negro y azul).

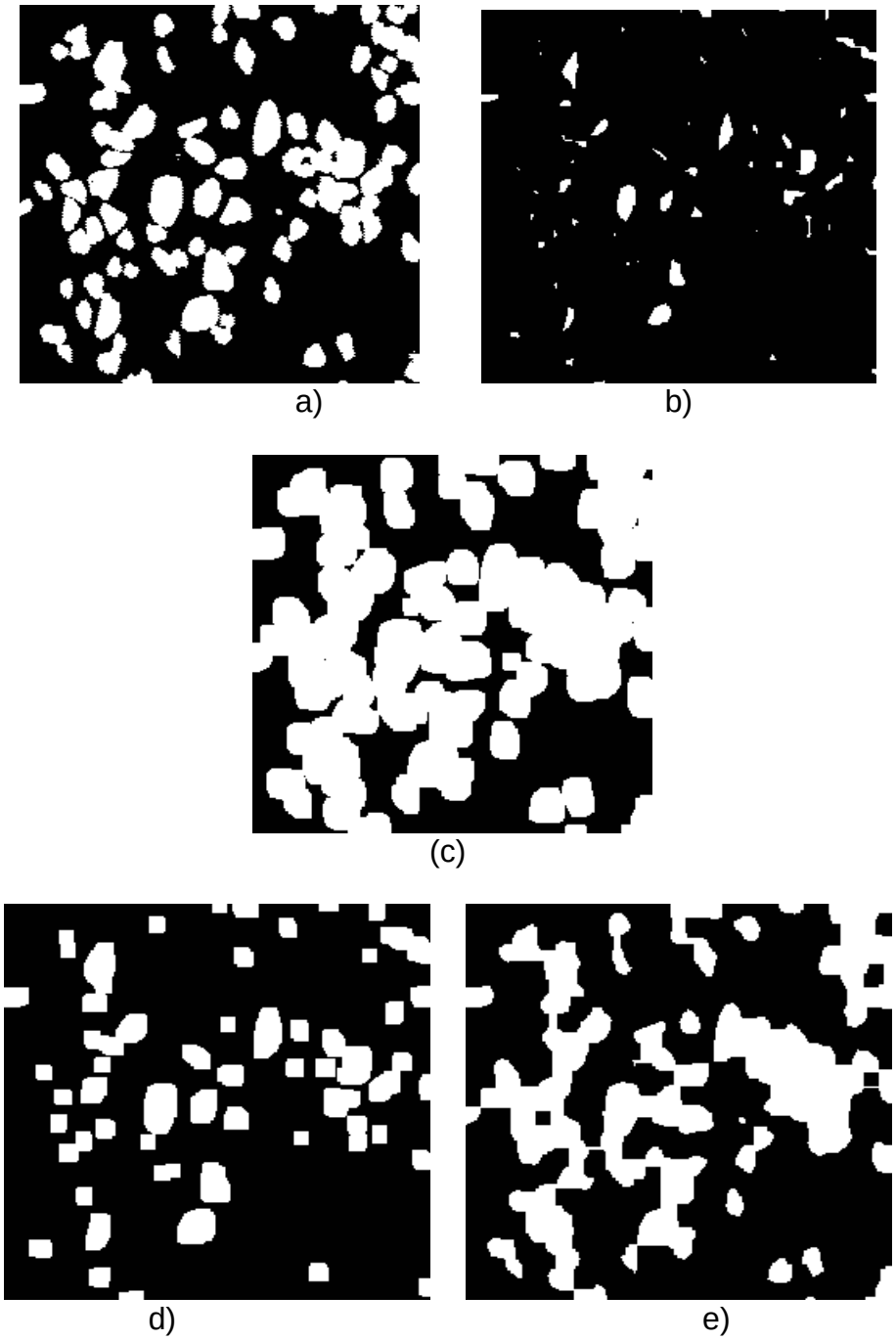


Figura 3.14 a) Imagen binaria original, b) Erosionado tamaño 6 (11x11 píxeles), c) Dilatado tamaño 6, d) Apertura morfológica tamaño 6, e) Cerradura morfológica tamaño 6.

En el caso de imágenes numéricas tenemos definiciones similares a caso binario. La apertura y cerradura morfológicas están dadas, respectivamente, por las siguientes ecuaciones;

$$\gamma_{\mu B}(f)(x) = \delta_{\mu \breve{B}}(\varepsilon_{\mu B}(f))(x) \quad \textbf{(23)}$$

$$\varphi_{\mu B}(f)(x) = \varepsilon_{\mu \breve{B}}(\delta_{\mu B}(f))(x) \quad \textbf{(24)}$$

La erosión y la dilatación se expresan por las siguientes ecuaciones.

$$\varepsilon_{\mu B}(f(x)) = \wedge\{f(y); y \in \mu\breve{B}_x\} \quad \textbf{(25)}$$

$$\delta_{\mu B}(f(x)) = \vee\{f(y); y \in \mu\breve{B}_x\} \quad \textbf{(26)}$$

Donde $\vee$ y $\wedge$ son los operadores supremo e ínfimo. En el caso Z^2 (Z es el conjunto de los enteros) o en imágenes digitales el supremo es sustituido por el valor máximo, mientras que el ínfimo por el valor mínimo.

Como se vio en la primera sección, podemos simplemente calcular el mínimo y el máximo en cada punto de la imagen dentro de la región definida para calcular la erosión y la dilatación. Posteriormente, aplicar estas operaciones por composición para obtener las aperturas y cerraduras. Sin embargo vamos a tomar otro camino para tratar de dar una interpretación geométrica como en el caso binario. Consideremos la siguiente definición:

Propiedad II.2.1. La erosión y la dilatación morfológicas conmutan con la umbralización. Para toda imagen f tenemos que,

$$X_t(\varepsilon_{\mu B}(f)) = \varepsilon_{\mu B}(X_t(f)) \quad \textbf{(27)}$$

$$X_t(\delta_{\mu B}(f)) = \delta_{\mu B}(X_t(f)) \quad \textbf{(28)}$$

Observe que, aunque no se hace diferencia en la notación, la erosión y la dilatación en la parte izquierda de las ecuaciones son transformaciones sobre funciones, mientras que las del lado derecho son transformaciones sobre

conjuntos. Las ecuaciones expresan que el conjunto obtenido por el umbral X_t sobre la erosión y la dilatación de una función (imagen numérica) es igual que la erosión y la dilatación de conjunto obtenido por el umbral X_t de la función. De esta forma podemos expresar la erosión y la dilatación sobre una función por las expresiones;

$$\varepsilon_{\mu B}(f)(x) = \vee\{t: \ x \in \varepsilon_{\mu B}(X_t(f))\} \quad \textbf{(29)}$$

$$\delta_{\mu B}(f)(x) = \vee\{t: \ x \in \delta_{\mu B}(X_t(f))\} \quad \textbf{(30)}$$

Puesto que la apertura y cerradura son obtenidas por composición de erosiones y dilataciones, de manera similar tenemos las expresiones

$$\gamma_{\mu B}(f)(x) = \vee\{t: \ x \in \gamma_{\mu B}(X_t(f))\} \quad \textbf{(31)}$$

$$\varphi_{\mu B}(f)(x) = \vee\{t: \ x \in \varphi_{\mu B}(X_t(f))\} \quad \textbf{(32)}$$

Esto significa, que se puede utilizar la misma interpretación geométrica de la apertura y cerradura sobre conjuntos para el caso de funciones. La ***Figura 3.11.*** ilustra del punto de vista geométrico, como en el caso binario, la apertura y cerradura morfológica. Observe que, en el caso de la apertura, el elemento estructural es barrido al interior de la función (ver ***Figura 3.11. (a)***).

Toda el área barrida por dicho elemento forma parte de la función obtenida por la apertura. En el caso de la cerradura tomamos el complemento del área barrida por el elemento estructural como se observa en la ***Figura 3.13. (c)*** La propiedad 5 es más general, y se cumple en el caso digital, para toda transformación creciente e invariante en traslación.

La ***Figura 3.16. (b), (c)*** ilustran la apertura y cerradura en el caso binario, mientras que la ***Figura 3.16. (e), (f)*** muestran el caso numérico.

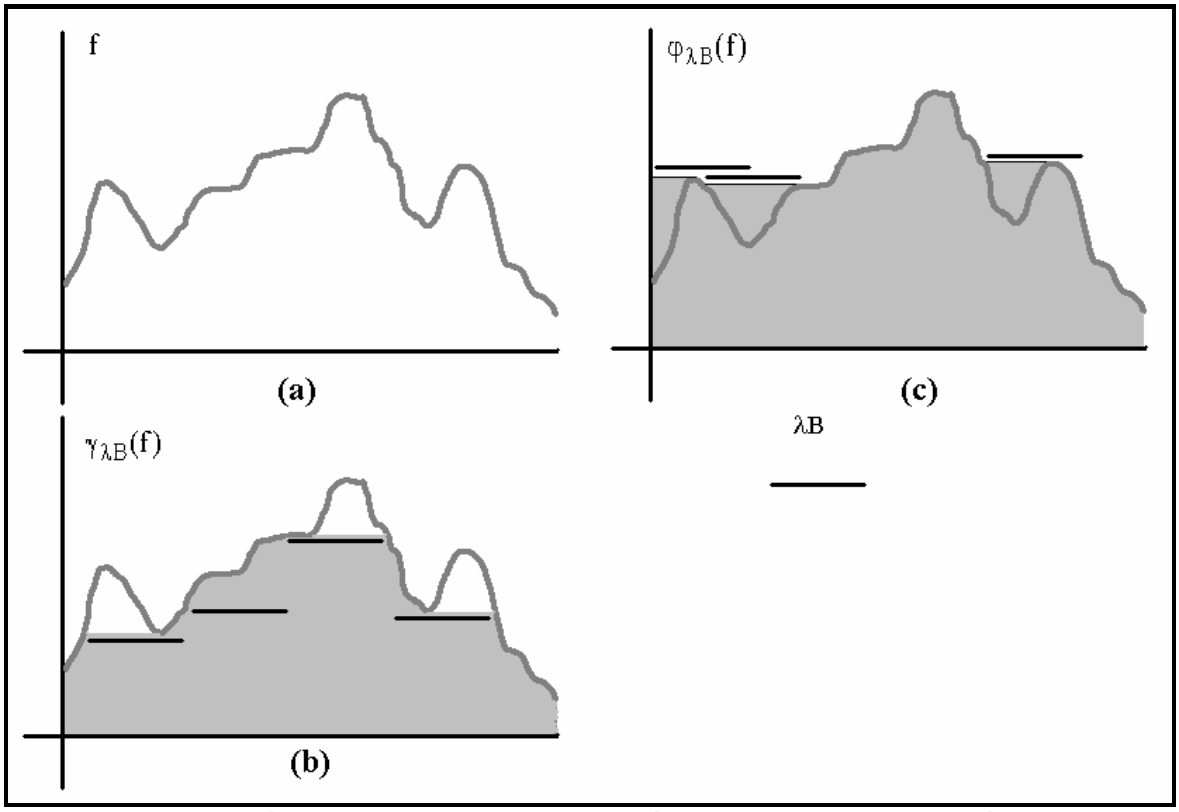


Figura 3.15. a) Función original, b) Apertura morfológica, c) Cerradura morfológica.

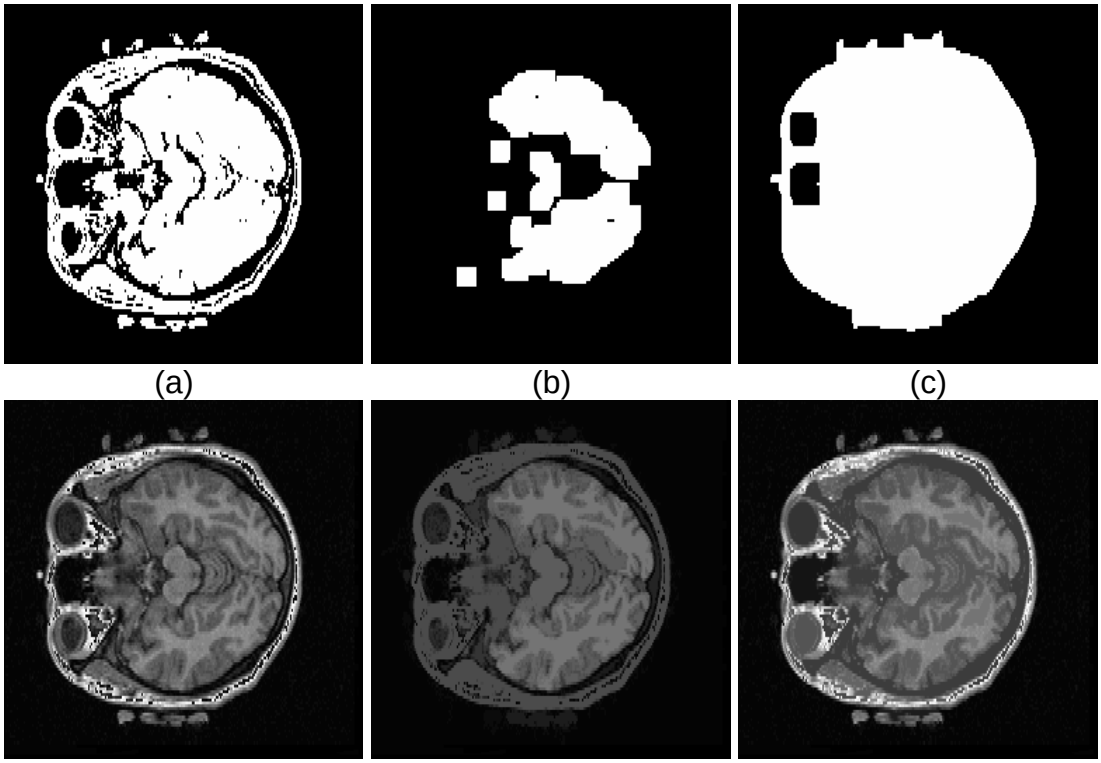


Figura 3.16. a) Imagen binaria, b) Apertura morfológica binaria tamaño 6, c) Cerradura morfológica binaria tamaño 6, d) Imagen numérica, e) Apertura morfológica numérica tamaño 6, f) Cerradura morfológica numérica tamaño 6.

3. II.2.1. Propiedades de las aperturas y las cerraduras morfológicas.

Las siguientes propiedades son validas tanto para el caso binario, como para el caso numérico.

Propiedad II.2. Ambas transformaciones, la apertura y la cerradura, son transformaciones idempotentes, es decir, para toda imagen f,

$$\gamma_{\lambda B}(f) = \gamma_{\lambda B}\gamma_{\lambda B}(f) \qquad \textbf{(33)}$$

$$\varphi_{\lambda B}(f) = \varphi_{\lambda B}\varphi_{\lambda B}(f) \qquad \textbf{(34)}$$

Propiedad II.3. Ambas transformaciones, la apertura y la cerradura, son transformaciones crecientes:

Para cualesquiera dos imágenes f, g con $f \leq g$

$$\gamma_{\lambda B}(f) \leq \gamma_{\lambda B}(g) \quad \textbf{(35)}$$

$$\varphi_{\lambda B}(f) \leq \varphi_{\lambda B}(g) \quad \textbf{(36)}$$

Propiedad II.4. La apertura es un transformación anti-extensiva y la cerradura una transformación extensiva. Para cualesquier λ

$$\gamma_{\lambda B}(f) \leq f \quad \textbf{(37)}$$

$$\varphi_{\lambda B}(f) \geq f \quad \textbf{(38)}$$

Propiedad II.5. La apertura y la cerradura son transformaciones duales con respecto a la complementación.

$$\gamma_{\lambda B}(f) = \left[\varphi(f^{C})\right]^{C} \quad \textbf{(39)}$$

Propiedad II.6. Para cualesquiera dos parámetros λ, μ, tenemos

$$\gamma_{\lambda B}\gamma_{\mu B}(f) = \gamma_{\max\{\lambda,\mu\}B}(f) \quad \textbf{(40)}$$

$$\varphi_{\lambda B}\varphi_{\mu B}(f) = \varphi_{\max\{\lambda,\mu\}B}(f) \quad \textbf{(41)}$$

Una última propiedad interesante de las aperturas morfológicas es que conmutan con anamorfosis; se dice que toda función monótona creciente es una anamorfosis. Por ejemplo, la función logaritmo; para todo $t1 \leq t2$ tenemos que $\log(t1) \leq \log(t2)$.

Propiedad II.7. La apertura y la cerradura conmutan con las anamorfosis. Sea α una anamorfosis, tenemos que;

$$\alpha(\gamma_{\lambda B}(f(x))) = \gamma_{\lambda B}(\alpha(f(x))) \quad \textbf{(42)}$$

$$\alpha(\varphi_{\lambda B}(f(x))) = \varphi_{\lambda B}(\alpha(f(x))) \quad \textbf{(43)}$$

Es evidente que la erosión y la dilatación conmutan también con las anamorfosis.

3. II.3. Gradientes morfológicos.

Sea f una función definida en R2,

El gradiente Morfológico de f esta definido por:

$$grad(f) = \lim_{r\to 0}\left(\frac{\delta_{rB}(f) - \varepsilon_{rB}(f)}{r}\right) \quad \textbf{(44)}$$

En donde rB es una bola de radio r

El Gradiente externo de f por:

$$grade(f) = \lim_{r\to 0}\left(\frac{\delta_{rB}(f) - (f)}{r}\right) \quad \textbf{(45)}$$

Y el Gradiente interno de f por:

$$gradi(f) = \lim_{r\to 0}\left(\frac{(f) - (\varepsilon_{rB}(f)}{r}\right) \quad \textbf{(46)}$$

En el caso discreto, se definen como:

Gradiente Morfológico

$$grad_{\mu B}(f)(x) = \delta_{\mu B}(f)(x) - \varepsilon_{\mu B}(f)(x) \quad \textbf{(47)}$$

Gradiente Externo

$$grade_{\mu B}(f)(x) = \delta_{\mu B}(f)(x) - (f)(x) \quad \textbf{(48)}$$

Gradiente Interno

$$gradi_{\mu B}(f)(x) = (f)(x) - \varepsilon_{\mu B}(f)(x) \quad \textbf{(49)}$$

El elemento estructural B es un cuadrado, hexágono, línea, etc.; de tamaño 1.

3. II.4. Transformaciones por reconstrucción (Apertura y cerradura por reconstrucción).

Los filtros morfológicos básicos presentas algunos inconvenientes, por ejemplo en una erosión las características no deseadas son eliminadas, pero a un precio muy alto, las estructuras restantes son modificadas significativamente.

Los filtros por reconstrucción ofrecen no modificar estas estructuras restantes. Los filtros por reconstrucción se construyen a partir de una imagen de referencia y una imagen marcadora (o marcador).

El marcador crece al interior de la imagen referencia al aplicar las transformaciones conocidas como transformaciones geodésicas; la dilatación y la erosión geodésicas.

3. II.4.1. Caso Binario:

Estas transformadas se construyen a partir de las transformadas geodesicas definidas por:

Dilatado Geodésico

$$\delta_X^1(Y) = X \cap \delta_B(Y) \qquad \textbf{(50)}$$

En donde $Y \subset X$

Erosionado Geodésico

$$\varepsilon_X^1(Y) = X \cup \varepsilon_B(Y) \qquad \textbf{(51)}$$

En donde $X \subset Y$

Donde B es el elemento estructural más pequeño 3x3, Y es el marcador y X es la referencia.

Para obtener un dilatado geodésico de tamaño m, se dilata m veces geodésicamente de tamaño 1,

$$\delta_X^m(Y) = \delta_X^1 \delta_X^1 \underbrace{\cdots}_{m\,veces} \delta_X^1 (Y) \qquad \textbf{(52)}$$

Para obtener las transformadas por reconstrucción dilatamos geodésicamente hasta la estabilidad o idempotencia

$$\rho_X(Y) = \mathrm{Re}\, c(X,Y) = \lim_{n\to\infty} \delta_X^n(Y) \qquad \textbf{(53)}$$

Y la reconstrucción dual esta dada por

$$\rho_X^*(Y) = \mathrm{Re}\, c^*(X,Y) = \lim_{n\to\infty} \varepsilon_X^n(Y) \qquad \textbf{(54)}$$

Dado que $\delta_X^M(Y) = \left[\varepsilon_{X^c}^m(Y^c)\right]^c$ **(55)**

Usando la dualidad con respecto a la complementación tenemos de esta forma

$$\rho_X(Y) = \left[\rho_{X^c}^*(Y^c)\right]^c \qquad \textbf{(56)}$$

$$\rho_X^*(Y) = \left[\delta_{X^c}(Y^c)\right]^c \qquad \textbf{(57)}$$

La reconstrucción $\delta_X(Y) = \left[\rho_{X^c}^*(Y^c)\right]^c$ elimina las componentes conexas blancas de la imagen referencia que no están marcadas por Y.

Mientras que la reconstrucción dual $\rho_X^*(Y) = \left[\delta_{X^c}(Y^c)\right]^c$ elimina las componentes negras que no están marcadas por Y.

SI $Y = \varepsilon_{\mu B}(X)$ tenemos que

$$\rho_X(Y) = \rho_X(\varepsilon_{\mu B}(X)) = \tilde{\gamma}_{\mu B}(x) \quad \textbf{(58)}$$

Es la apertura por reconstrucción tamaño μB .

De la misma forma cuando

SI $Y = \delta_{\mu B}(X)$ tenemos que

$$\rho_X^*(Y) = \rho_X^*(\delta_{\mu B}(X)) = \tilde{\varphi}_{\mu B}(x) \quad \textbf{(59)}$$

Es la cerradura por reconstrucción tamaño μB.

Usando estas nuevas aperturas y cerraduras por reconstrucción se puede generar nuevas transformaciones

Filtros alternados

$$\theta(X) = \tilde{\varphi}[\gamma_{\mu B}(X)] \quad \textbf{(60)}$$

$$\theta^*(X) = \tilde{\gamma}[\tilde{\varphi}_{\mu B}(X)] \quad \textbf{(61)}$$

Donde $\theta(X) \neq \theta^*(X)$

3. II.4.2. Caso Numérico.

Las transformaciones por reconstrucción en el caso de imágenes numéricas, es una extensión directa del caso binario. Consideremos dos funciones f y g.

En el Caso numérico la dilatación y erosión geodesia están dadas por

$$\delta_f^1(g)(x) = f(x) \wedge \delta_B(g)(x) = \min\{f(x), \delta_B(g)(x) \quad \textbf{(62)}$$

$$\varepsilon_f^1(g)(x) = f(x) \vee \varepsilon_B(g)(x) \quad \textbf{(63)}$$

Las transformaciones por reconstrucción utilizando dilataciones y erosiones geodésicas, expresadas respectivamente por $R(f,g)$ y $R^*(f,g)$, de f a partir de g se definen por:

$$R(f,g) = \lim_{n\to\infty} \delta_f^n(g) = \underbrace{\delta_f^1 \delta_f^1 \cdots \delta_f^1}_{\text{Hasta estabilidad}}(g) \quad \textbf{(64)}$$

$$R^*(f,g) = \lim_{n\to\infty} \varepsilon_f^n(g) = \underbrace{\varepsilon_f^1 \varepsilon_f^1 \cdots \varepsilon_f^1}_{\text{Hasta estabilidad}}(g) \quad \textbf{(65)}$$

Al igual que en el caso binario, cuando la función g es igual a la erosión $\varepsilon_\lambda(f)$ o a la dilatación $\delta_\lambda(f)$) de la imagen original obtenemos la apertura y cerradura por reconstrucción:

$$\tilde{\gamma}_\lambda(f) = \lim_{n\to\infty} \delta_f^n(\varepsilon_\lambda(f)) = \underbrace{\delta_f^1 \delta_f^1 \cdots \delta_f^1}_{\text{Hasta estabilidad}}(\varepsilon_\lambda(f)) \quad \textbf{(66)}$$

$$\tilde{\varphi}_\lambda(f) = \lim_{n\to\infty} \varepsilon_f^n(\delta_\lambda(f)) = \underbrace{\varepsilon_f^1 \varepsilon_f^1 \cdots \varepsilon_f^1}_{\text{Hasta estabilidad}}(\delta_\lambda(f)) \quad \textbf{(67)}$$

3. II.5. Filtrado morfológico.

Una transformación T se dice creciente si para dos conjuntos X, Y tal que $X \subset Y \rightarrow T(X) \subset T(Y)$, una transformación T es idempotente si y sólo si $\mathrm{T(T(f))} = \mathrm{T(f)}$. Toda transformación que es creciente e idempotente se le llama filtro morfológico

3. II.6. Segmentación de imágenes por medio de LDA (*Watershed* en ingles).

La segmentación de imágenes es una de los problemas de procesamiento de imágenes más interesantes. El objetivo principal consiste en extraer las regiones de interés de la imagen. El método debe permitir la introducción de criterios para obtener las regiones deseadas (niveles de gris, contraste, tamaño, textura, etc.). Una típica tarea de segmentación consiste en la partición de la imagen aérea (Por satélite) en áreas Urbanas y áreas rurales, de la misma forma

En morfología matemática, la técnica de marcadores y línea divisora de aguas (LDA , o Watershed en Inglés) es el método tradicional de segmentación de imágenes. En efecto, la segmentación de imágenes basada en la LDA ha probado ser un método eficiente. Su principal problema consiste en la sobre-segmentación producida por el LDA en el caso que se aplique directamente sobre la imagen a segmentar.

La solución para prevenir esta sobre-segmentación consiste en seleccionar un conjunto de marcadores que señalan las regiones de interés. Estos marcadores se imponen como los únicos mínimos de la imagen a segmentar. Después la LDA es aplicada a esta imagen.

Es decir la técnica de marcadores y LDA se puede dividir en dos etapas como se ilustra en la ***Figura 3.17.***

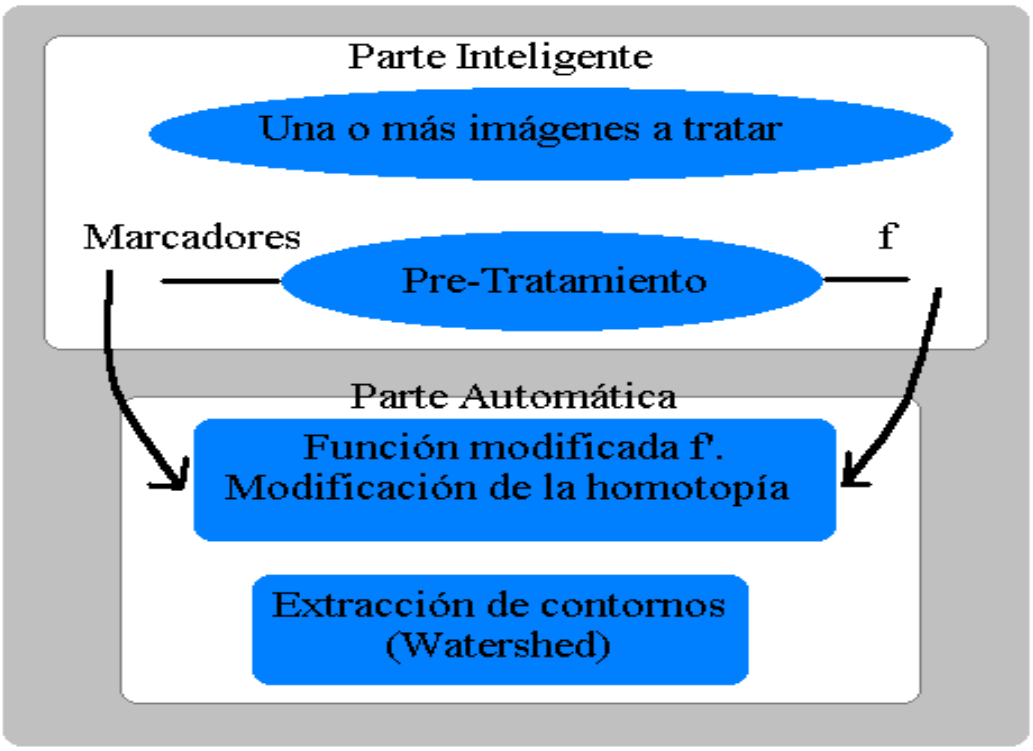


Figura 3.17. Las dos etapas en que se divide la técnica de marcadores y LDA

Una etapa inteligente donde se tiene una o más imágenes a tratar, sobre estas imágenes se aplican las diferentes transformaciones morfológicas para detectar el conjunto de marcadores y generar una función que frecuentemente es el gradiente de la imagen. La segunda etapa es automática y es donde se impone los mínimos y se aplica la LDA.

La LDA Aparece en 1978 por D. Digabel y C. Lantuéjoul. Las líneas divisoras de aguas son los bordes de las vertientes (*catchment basins)* de una superficie topográfica imaginaria. También se le conoce como las líneas divisoras de aguas. Cualquier imagen en niveles de gris puede ser vista como una

superficie topográfica asociamos cada punto de la imagen con una elevación proporcional al nivel de gris de la imagen.

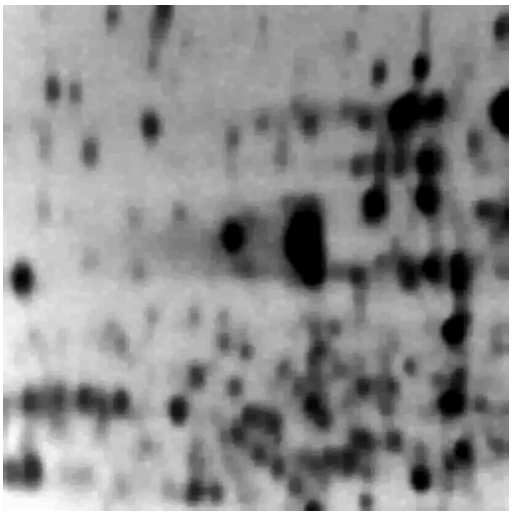
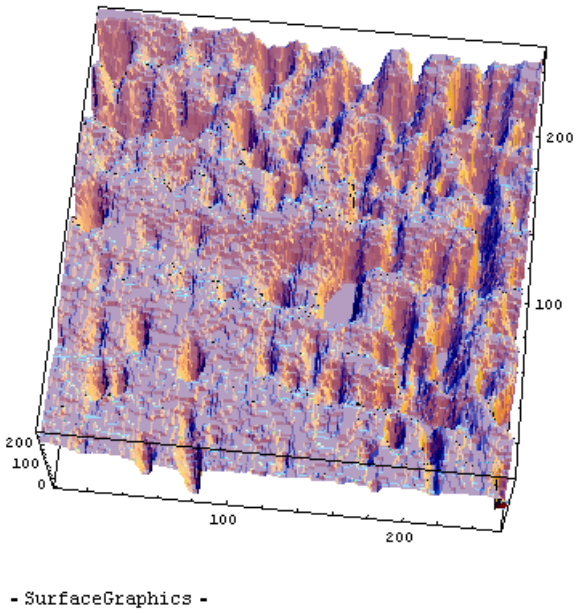


Imagen de niveles de gris



Grafica en forma de superficie de la imagen de niveles de gris

Figura 3.18. a) Imagen original gel de eletroforesis b) Imagen de gel de eletroforesis vista como relieve topográfico

En efecto, existen varios algoritmos para realizar LDA, Uno de los mejores y más intuitivos de la transformada LDA es una basada en la simulación de la inundación. Se considera a la imagen en niveles de gris de entrada como una superficie topográfica. El problema es para producir las líneas divisoras sobre la superficie. Para lograr esto las vertientes de la superficie son asociadas a las regiones marcadas.

Cada marca es asociada a un color, hacia un uniforme promedio sobre la imagen completa. Cuando el agua de distintos colores sube y podría juntarse, se construye un dique o presa para prevenir que se junten. La inundación podría alcanzar une estado en que solamente sea visibles la parte superior de la presa o dique eso se conoce con líneas divisoras de los WS (Watershed), las regiones coloreadas son los catchment basins (CB) asociados a cada marcador.

3. III. Estructura de datos y algorítmica.

3. III.1. Colas y pilas.

3. III.1.1. Pilas.

3. III.1.1.1. Introducción.

Una Pila es una estructura de datos en donde todas las inserciones y eliminaciones se realizan en un solo extremo denominado cima de la pila, una analogía es una pila de platos, una de cajas, una de libros, etc., la implementación se realiza a través de apuntadores debido a que estas crecen y decrecen dinámicamente. Se les denomina lista LIFO (Last-Input First-Output).

3. III.1.1.2 Tipo de dato Abstracto Pila.

Una Pila es una lista ordenada de elementos en la que todas las inserciones y supresiones se realizan por un solo extremo denominado cima de la pila. En una pila el ultimo elemento añadido es el primero en salir, por esa razón, se les denomina también listas LIFO.

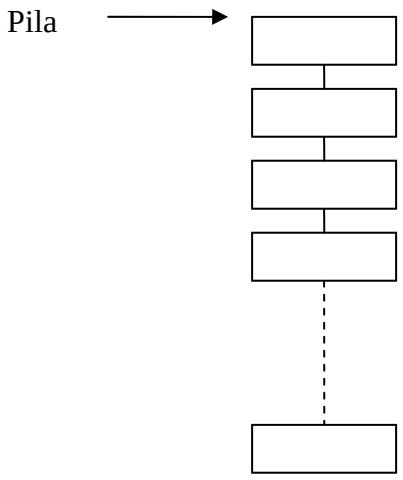


Figura 3.19. Representación de una Pila

Las operaciones básicas que definen al tipo de dato abstracto pila son mostradas en la ***tabla 1***.

Pilavacia(P)	Crea una pila sin elementos
Esvacia(P)	Devuelve verdadero si la pila es vacia
Cima(P)	Devuelve el elemento que esta en la cima de la pila P
Suprime(P)	Elimina el elemento que esta en al cima de la pila P
Sacar(X, P)	Devuelve el elemento cabeza en X y lo suprime de la pila P
Meter(X, P)	Añade el elemento X en la Pila P

Tabla 1. Operaciones Básicas de una pila

3. III.1.2 Colas.

3. III.1.2.1 Introducción.

En la vida cotidiana el concepto de cola es muy usado, existen colas en consultas al doctor, en la entrada al cine, pago de impuestos, en el banco, etc.,

En una aplicación de informática una cola es una lista, en la cual todas las inserciones de elementos a la lista se realizan por un extremo y las eliminaciones por el otro extremo.

Las colas también son conocidas como estructuras FIFO (First-In First-Out)

Algunas aplicaciones de las colas en el mundo de la computación son colas de impresión, acceso de almacenamiento a disco, el uso del CPU, etc., Nosotros la utilizaremos para diversas operaciones en un grafo tales como el recorrido en Anchura, En la división del Grafo, en la Fusión de dos subgrafos, Tambien se utiliza en la creación de las vertientes de la imagen, en la generación del MST.

3. III.1.2.2 Tipo de dato abstracto Cola.

Una cola es una lista ordenada de elementos, en la cual las eliminaciones de estos elementos se realizan en un solo extremo llamado frente o principio de la cola, y los nuevos elementos son añadidos por el otro extremo llamado final de la cola o fondo.

En esta estructura de datos el primer elemento en entrar es el primero en salir, por esta razón se les conoce también como FIFO (First-In First-Out).

Las operaciones básicas que definen el tipo de dato cola son:

QCrear(Q)	Crea la cola Q como estructura vacia
Qvacia	Nos indica si la cola esta vacía
Frente	Devuelve el Frente de la Cola
QBorrar	Elimina el frente de la cola
QAnula(Q)	Borra todos los elementos de la cola Q
Quitar(X,Q)	Elimina y devuelve el frente de la cola
QPoner(X,Q)	Añade un nuevo elemento a la cola

Tabla 2. Operaciones básicas que definen el tipo de dato cola

La ***Figura 3.20.*** muestra una estructura cola en su estructura original y después de eliminar y añadir un elemento.

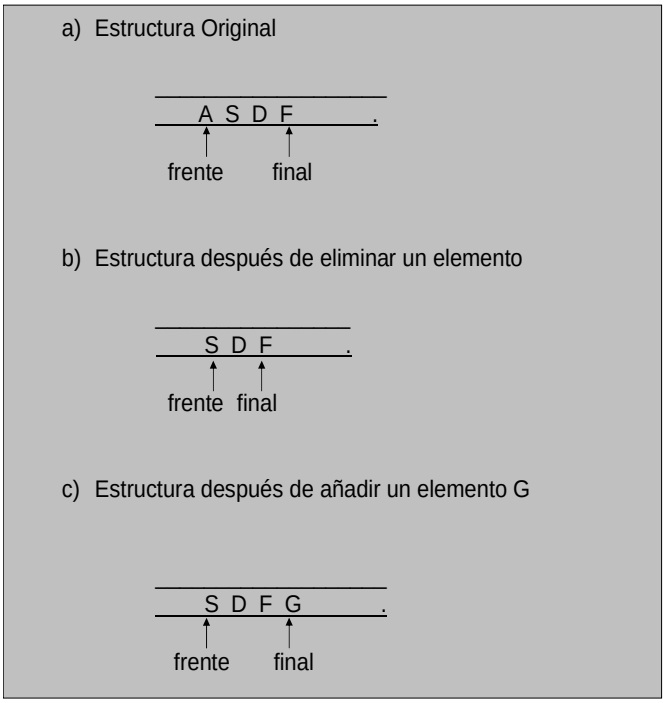


Figura 3.20. Representación de una Cola

3. III.2 Grafos.

3. III.2.1 Introducción.

La teoría de grafos tiene un inicio preciso: Un artículo publicado en 1736 por el matemático suizo Leonhard Euler (1707-1783). La idea principal en que se apoya su trabajo surgió del problema conocido como los siete puentes de Königsberg.

El problema consiste en encontrar la posibilidad de un trayecto, alrededor de los puentes, de tal forma que cruce solamente una vez cada uno de ellos. El mapa de abajo muestra la disposición de siete puentes y dos islas en la ciudad de Königsberg.

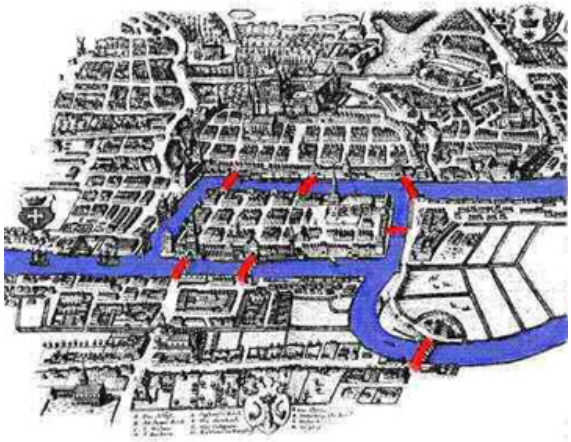


Figura 3.21. Siete puentes y dos islas en la ciudad de Königsberg.

En 1736, el matemático suizo radicado en San Petersburgo Leonhard Euler publicó "Solutio Problematis ad Geometriam Situs Pertinentis", un artículo en el que resolvía el problema en el caso general. Este trabajo es considerado como el nacimiento de la Teoría de Grafos, utilizada hoy en día en una multiplicidad de aplicaciones, y también uno de las primeras apariciones de una «nueva geometría» en la que importan sólo las propiedades estructurales de un objeto y no sus medidas. A esto se refieren las palabras «geometriam situs» en el título de Euler, palabras que hoy se traduce como topología.

3. III.2.2 Definiciones .

Definición 3. Un grafo ***G*** consiste de un conjunto finito de puntos llamados nodos o vértices y un conjunto finito de líneas o curvas llamadas aristas. Cada arista conecta dos nodos distintos o a un vértice consigo mismo; en este último caso se denomina circuito, lazo o loop.

Definición 3.1. Sea V un conjunto finito no vació, y sea $E \subseteq VxV$. El par (V, E) es un grafo dirigido sobre V, Donde V es el conjunto de vértices, o nodos y E es su conjunto de aristas. Para denotar un grafo escribimos ***G= (V, E).***

Definición 4. Sean x, y vértices de un grafo no dirigido G= (V, E). Un ***camino x-y*** en G es una sucesión alternada finita (sin lazos o loop) $x = x_0, e_1, x_1, e_2, x_2, e_3, ..., e_{n-1}, x_{n-1}, e_n, x_n = y$ de vértices y aristas de G que comienzan en el vértice ***x*** y terminan en el vértice ***y*** y que contiene las n aristas $e_i = \{x_{i-1}, x_i\}$ donde $1 \leq i \leq n$

La longitud de un camino es n, el número de aristas que hay en el camino.

Cualquier camino x-y donde x=y (y n>1) es un camino cerrado. En caso contrario el camino es abierto. Un camino puede repetir aristas y vértices.

Definición 5. (Recorrido y camino simple) Consideremos un camino x-y en un grafo no dirigido G= (V, E).

Si no se repite ninguna arista en el camino x-y, entonces el camino es un recorrido x-y. Un recorrido x-x es un circuito.

Cuando ningún vértice del camino x-y se presenta más de una vez, el camino es un camino simple x-y. El termino ciclo se usa para describir un camino simple cerrado x-x.

Los conceptos anteriores se resumen en la ***tabla 3.***

Vértice(s) repetidos	Arista(s) repetidas	Abierto	Cerrado	Nombre
Si	Si	Si		Camino
Si	Si		Si	Camino (Cerrado)
Si	No	Si		Recorrido
Si	No		Si	Circuito
No	No	Si		Camino simple
No	No		Si	Ciclo

Tabla 3. Definiciones Grafos: Recorrido y camino simple

Definición 6. (conexo) Sea G=(V, E) un grafo no dirigido. Decimos que g es conexo si existe un camino simple entre cualesquiera dos vértices distintos de G. Un grafo que no es conexo es disconexo.

Definición 7. (Componentes) Para cualquier grafo G= (V, E), el numero de componentes de G se denota con k(G). Como se muestra en la ***Figura 3.22.***

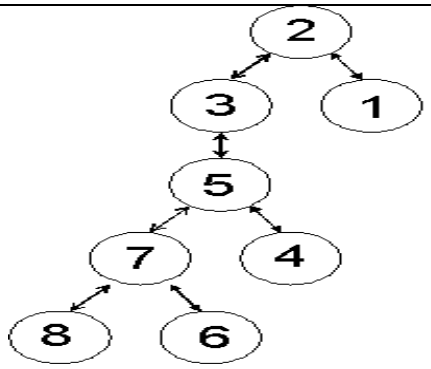
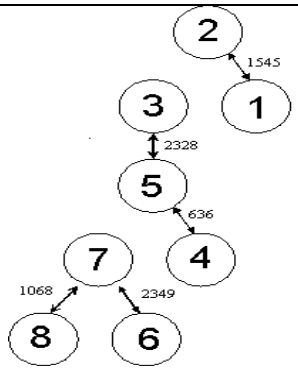
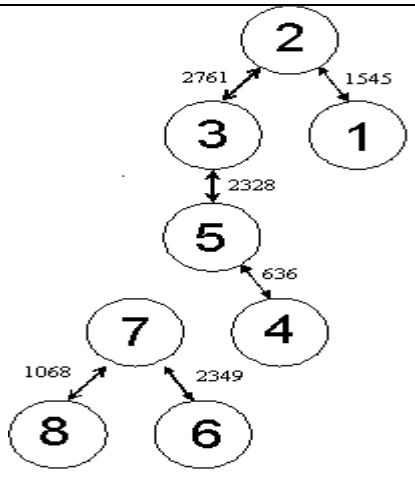
G	*k(G)*
	1
	3
	2

Figura 3.22. Componentes de un grafo G.

Definición 8. (subgrafo) Si G= (V, E) es un grafo, entonces $G_1 = (V_1, E_1)$ es un subgrafo de G si $\phi \neq \subseteq V_1$ y $E_1 \subseteq E$, donde cada arista de E_1 es incidente con los vértices de V_1.

Definición 9. (subgrafo recubridor) Dado un grafo G= (V, E), sea $G_1 = (V_1, E_1)$ un subgrafo de G. si $V_1 = V$, entonces G_1 es un subgrafo recubridor de G.

Definición 10. (grafo completo) Sea V un conjunto de n vértices. El grafo completo sobre V, que se denota K_n, es un grafo no dirigido sin lazos tal que para toda $a, b \in V, a \neq b$, existe una arista {a,b}.

Definición 11. (grado de vértice) Sea G un grafo no dirigido, Para cualquier vértice v de G, el grado de v, que se denota por grad(v), es el numero de aristas en G que son incidentes con v. En este caso, un lazo en un vértice v se considera como dos aristas incidentes en v.

Definición 12. (grafos planos) Un grafo G es plano si podemos dibujar G en el plano de modo que sus aristas se intersequen solo en los vértices de G. Este dibujo de G se conoce como una inmersión de G en el Plano.

Aquellos grafos que pueden dibujarse de tal manera que sus aristas solo se intersequen en nodos se denominar grafos planares

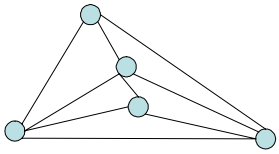


Figura 3.23. Grafos planares

Los grafos no planares son aquellos que no son dibujables sus aristas sin que se intersecten

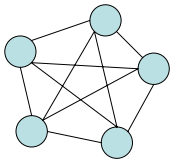


Figura 3.24. Grafos no planares

Un grafo completo es aquel en el cual todos sus nodos están conectados a todos los demás. Es por eso que si tales nodos tienen n nodos entonces tienen n(n-1)/2 aristas. Es decir; el nodo n tiene n-1 aristas que conectan a los n-1 nodos, el nodo n-1 tiene n-2 aristas, así sucesivamente el nodo 1 tiene 0 aristas. Entonces el problema se reduce a sumar la siguiente serie

$$0+1+2+3+4+\ldots+(n-3)+(n-2)+(n-1)=n(n-1)/2. \quad \textbf{(68)}$$

Si una arista conecta los nodos U y W, diremos que es incidente a U y a W.

El grado de un nodo G es el numero de aristas que son incidentes a G, cada loop o lazo es contado 2 veces puesto que tiene 2 extremos en G.

Si el grado de G es par, se dirá que G es un nodo par de igual forma si el grado de G es impar, G será un nodo impar.

Definición 13. (Subdivisión Elemental de Un grafo) Sea G= (V, E) un grafo no dirigido sin lazos, tal que E no es vació, Una División Elemental de G resulta cuando eliminamos una arista e={u, w} de G y entonces las aristas {u, v} , {v, w} se añaden a G-e , donde $v \in V$

Los grafos no dirigidos sin lazos $G_1 = (V_1, E_1)$ y $G_2 = (V_2, E_2)$ son homeomorfos si son isomorfos o si ambos pueden obtenerse del mismo grafo no dirigido sin lazos H por una sucesión de divisiones elementales.

3. III.3 Árboles.

3. III.3.1 Introducción.

Los árboles fueron utilizados por primera vez en 1847 por Gustav Kirchhoff (1824-1887) en un trabajo de redes eléctricas, posteriormente fueron desarrollados y definidos de nuevo por Arthur Cayley (1821-1895). Algunos tipos de árboles son importantes en el estudio de las estructuras de datos, ordenaciones, teoría de codificación, segmentación de imágenes, etc,

3. III.3.2 Definiciones, propiedades.

Definición 14. (Árbol) Sea G = (V, E) un grafo no dirigido y sin lazos. El Grafo G es un árbol si G es conexo y no contiene ciclos.

Preposiciones equivalentes para un grafo no dirigido G = (V, E) sin lazos:

- ***G*** es un árbol
- ***G*** es conexo, pero si se elimina cualquiera de sus aristas, g quedara desconectado en dos subgrafos que son árboles.
- ***G*** no contiene ciclos y $|V|=|E|+1$.
- ***G*** es conexo y $|V|=|E|+1$
- ***G*** no contiene ciclos y si a, b $\in$ V con {a, b} $\notin$ E, entonces el grafo que se obtiene de añadir la arista {a, b} a G tiene precisamente un ciclo.

3.III.3.3 Algoritmo de Búsqueda en profundidad.

Sea $G=(V,E)$ un grafo no dirigido conexo, sin lazos, tal que $|V|=n$ y donde los vértices están ordenados como $v_1, v_2, v_3, ..., v_n$. Para encontrar el árbol recubridor en profundidad, ordenado con raíz, aplicamos el siguiente

algoritmo, donde usaremos la variable v para guardar el vértice que se analiza en un momento dado.

Algoritmo de Búsqueda en Profundidad.	
Paso	*Actividad*
1	Se asigna v_1 a la variable v y se inicializa T como el árbol que consta de solamente este vértice,(será la raíz de árbol recubridor que se va a desarrollar)
2	Seleccionamos el subindice más pequeño i, $2 \leq i \leq n$, tal que $\{v, v_1\} \in E$ y v_1 no ha sido visitado todavía Si no se encuentra tal subíndice, entonces se va al paso 3. en caso contrario se hace lo siguiente: • Añadimos la arista $\{v, v_1\}$ al arbol T. • Asignamos v_1 a v . • Regresamos al Paso 2.
3	Si $v = v_1$, el árbol T es el árbol recubridor (ordenado, con raíz) del orden dado.
4	Si $v \neq v_1$, retrocedemos desde v . Si u es el padre del vértice asignado a v en T, entonces asignamos u a v y regresamos la paso 2.

3. III.3.4 Búsqueda en anchura.

Un segundo método para buscar los vértices de un grafo no dirigido conexo sin lazos es la búsqueda de anchura.

Sea $G = (V, E)$ un grafo no dirigido conexo, sin lazos, tal que $|V| = n$ y donde los vértices están ordenados como $v_1, v_{2,} v_{3,} ...., v_n$. Para encontrar el árbol recubridor en anchura T de G para el orden dado, esta el siguiente algoritmo.

Algoritmo de búsqueda en anchura.	
Paso	*Actividad*
1	Insertamos el vértice v_1 en la cola Q, y se inicializa T como el árbol que consta de solamente este vértice v_1
2	Eliminamos los vértices del frente Q. Al eliminar un vértice v , consideramos v_i para cada $2 \leq i \leq n$. Si la arista $\{v, v_1\} \in E$ y v_1 no ha sido visitado, agregamos la arista a T. Si examinamos todos los vértices que estaban en Q y no obtenemos aristas nuevas, el árbol T (Generado hasta ese momento) es el árbol recubridor (ordenado con raíz) del orden dado.
3	Insertamos los vértices adyacentes a cada v (del paso 2) en el final de al cola Q, según el orden en que fueron visitados por primera vez. Después regresamos la paso 2.

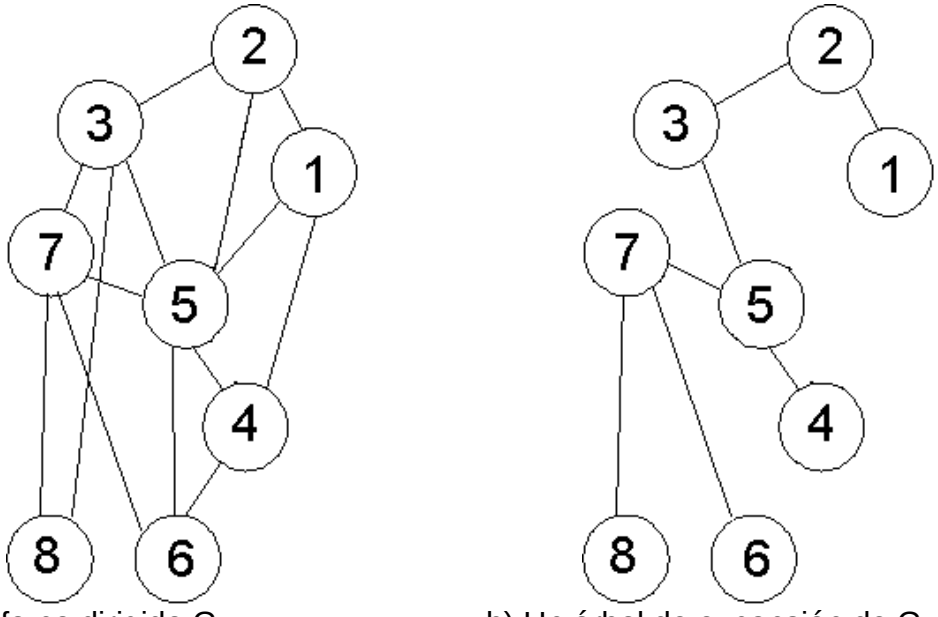
3. III.4 Minimum spanning tree.

Se les conoce como Árboles de expansión de coste mínimo, el problema del árbol de expansión de coste mínimo se aplica sobre grafos no dirigidos,

Una red conectada es una red tal que cualquier par de vértices de esta puede se unido mediante un camino. Un Árbol en una red es un subconjunto G' del grafo G que es conectado y sin ciclos. Los árboles tienen dos propiedades importantes

Todo árbol de n vértices o nodos contiene exactamente n-1 arcos o aristas. Si se añade un arco a un árbol, entonces resulta un ciclo

Un árbol de expansión es un árbol que contiene a todos los nodos de una red, de aquí se concluye que buscar un árbol de expansión de una red es averiguar si la red es conectada.



a) Grafo no dirigido G b) Un árbol de expansión de G

Figura 3.25. Grafo no dirigido G y Un árbol de expansión de G.

Dado un grafo no dirigido ponderado y conexo, encontrar su árbol de expansión cuya suma de los costos de las aristas sea mínima. A este árbol se le conoce como árbol de expansión de coste mínimo (o *Minimum Spanning Tree*, por simplicidad utilizaremos MST para referirnos a el.)

Existen algoritmos que nos permiten encontrar un MST, tales como el Algoritmo de Kruskal y el Algoritmo de Prim.

3. III.4.1 Algoritmo de Kruskal.

Sea $G = (V, E)$ un grafo no dirigido conexo, sin lazos, tal que $|V| = n$ y cada arista e tiene asignado un numero real positivo $p(e)$, Para encontrar un árbol recubridor optimo (MST) para G, Existe el siguiente algoritmo.

Paso	Descripción.
1	Hacemos el contador i =1 y seleccionamos una arista e_1 en G, tal que $p(e_1)$ sea lo más pequeño posible.
2	Para $1 \le i \le n-2$, si hemos seleccionado las aristas $e_1, e_2, ..., e_i$, entonces seleccionamos la arista e_{i+1} de las aristas restantes en G de modo que: a) $p(e_{i+1})$ sea lo más pequeño posible b) El subgrafo de G determinado por las aristas $e_1, e_2, ..., e_i, e_{i+1}$ (y los vértices incidentes) no contenga ciclos.
3	Incrementamos i en 1, i=i+1. Si i =n-1, el subgrafo de G determinado por las aristas $e_1, e_2, ..., e_i, e_{n-1}$ es conexo, con n vértices y n-1 aristas y es un árbol recubridor optimo para G Si $i \prec n-1$, regresamos al paso 2.

3. III.4.2 Algoritmo de Prim.

Sea $G = (V, E)$ un grafo ponderado no dirigido, sin lazos, Para encontrar un árbol recubridor optimo para G, aplicamos el siguiente algoritmo.

Paso	Descripción
1	Hacemos el contador i =1 y colocamos un vértice arbitrario $v_1 \in V$ en el conjunto P. Definimos $N = V - \{v_1\}$ y $T = \phi$.
2	Para $1 \leq i \leq n-1$, donde $\|V\| = n$, sean $P = \{v_1, v_2, ..., v_i\}$, $T = \{e_1, e_2, ..., e_{1-i}\}$, y N =V - P. Añadimos a T la arista más corta (la arista de peso minima) de G que conecta un vértice x en P con un vértice y(= v_{i+1}) en N. Colocamos y en P y lo eliminamos de N.
3	Incrementamos el contador en 1. Si i =n, el subgrafo de G determinado por las aristas $e_1, e_2, ...e_{n-1}$ es conexo, con n vértices y n-1 aristas y es un árbol optimo para G. Si $i \prec n$, regresamos al paso 2.

3. III.5 Uso de una FIFO jerárquica en LDA y vertientes.

Veremos en esta parte como se realiza la LDA, La ***Figura 3.26. (a)*** *(Original)* nos muestran los niveles de gris de la imagen original en los puntos x, y indicados por las columnas marcadas en amarillo y la tabla de la ***Figura 3.26. (b)*** *(mínimos)* nos muestra los puntos que serán tomados como mínimos impuestos (a partir de donde iniciamos la inundación), están marcados con verde.

Original

1	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	4	4	4	4	4	5	6	5	5	5	5	5	5	7
2	5	3	3	3	5	5	6	5	4	4	4	4	6	7
3	5	3	2	4	4	6	7	6	4	4	4	4	6	7
4	5	4	2	3	5	6	7	6	4	4	4	4	5	7
5	5	3	3	3	5	6	7	6	5	6	6	5	5	7
6	5	5	5	5	5	6	6	6	6	6	6	6	6	7
7	6	6	6	6	6	6	6	7	8	7	7	7	7	7
8	7	7	7	7	7	7	7	9	9	9	9	9	9	9
9	8	8	8	8	8	8	8	8	8	8	8	8	8	8
10	9	9	9	9	9	9	9	8	7	7	7	7	7	7
11	10	10	10	10	10	10	9	8	7	6	6	6	6	7
12	11	11	11	11	11	10	9	8	7	6	5	5	6	7
13	12	12	12	12	12	10	9	8	7	6	6	6	6	7
14	13	13	13	12	11	10	9	8	7	7	7	7	7	7
15	14	14	13	12	11	10	9	8	8	8	8	8	8	8

a)

Minimos

1	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	4	4	4	4	4	5	6	5	5	5	5	5	5	7
2	5	3	3	3	5	5	6	5	4	4	4	4	6	7
3	5	3	2	4	4	6	7	6	4	4	4	4	6	7
4	5	4	2	3	5	6	7	6	4	4	4	4	5	7
5	5	3	3	3	5	6	7	6	5	6	6	5	5	7
6	5	5	5	5	5	6	6	6	6	6	6	6	6	7
7	6	6	6	6	6	6	6	7	8	7	7	7	7	7
8	7	7	7	7	7	7	7	9	9	9	9	9	9	9
9	8	8	8	8	8	8	8	8	8	8	8	8	8	8
10	9	9	9	9	9	9	9	8	7	7	7	7	7	7
11	10	10	10	10	10	10	9	8	7	6	6	6	6	7
12	11	11	11	11	11	10	9	8	7	6	5	5	6	7
13	12	12	12	12	12	10	9	8	7	6	6	6	6	7
14	13	13	13	12	11	10	9	8	7	7	7	7	7	7
15	14	14	13	12	11	10	9	8	8	8	8	8	8	8

b)

Figura 3.26. a) Imagen original, b) Mínimos de la imagen original.

Necesitamos también generar otra tabla en la cual etiquetamos los mínimos, aunque realmente para efectos de mostrar el proceso de inundación, cada mínimo se mostrara con un color diferente sobre la imagen original y se empezara a inundar y se ira mostrando como se utiliza una FIFO jerárquica para controlar la inundación.

Una FIFO Jerárquica es una lista de varias FIFO's en donde cada una de las FIFO's tiene un peso o valor (En nuestro caso es el nivel de gris de la imagen), Debido a este valor pueden ser ordenadas de menor a mayor.

En la ***Figura 3.27.*** se muestra como los diferentes puntos de las vertientes son vaciados a una FIFO jerárquica.

Como primer paso o inicialización de la FIFO jerárquica, se crea una FIFO jerárquica vacía FJ, Luego se toman los puntos que representa el primer mínimo, como se sabe que su nivel de gris es 4, se añade una FIFO a FJ con valor 4 y se meten todos los puntos. De la misma forma se toman los puntos del mínimo 2, cuyo valor en nivel de gris es 2, y se agregan a la FIFO con valor 2 insertada previamente en FJ antes de la FIFO con valor 4, para poder conservar el orden jerárquico. Finalmente los puntos del mínimo 3 son agregados a la FIFO con valor 5 adicionada después de la FIFO 4.

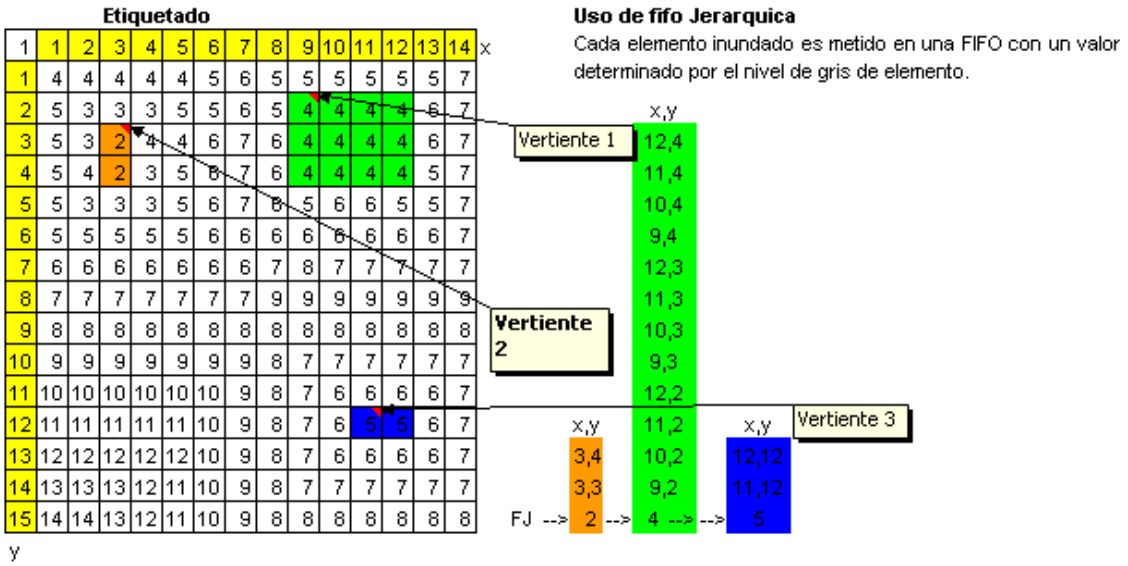


Figura 3.27. Uso de Fifo Jerárquica.

Lo que sigue es procesar los elementos de la FIFO con menor valor que exista en la FJ, hasta que esta quede vacía. El procesar un elemento de una FIFO significa inundar a sus elementos vecinos, lo cual esta indicado por colores según la vertiente. En este caso se comienza con todos los elementos de la vertiente 2. Cuando un elemento es inundado es metido a la FIFO que le corresponda según su valor.

EL primer punto a procesar es el 3,3 ver figura siguiente.

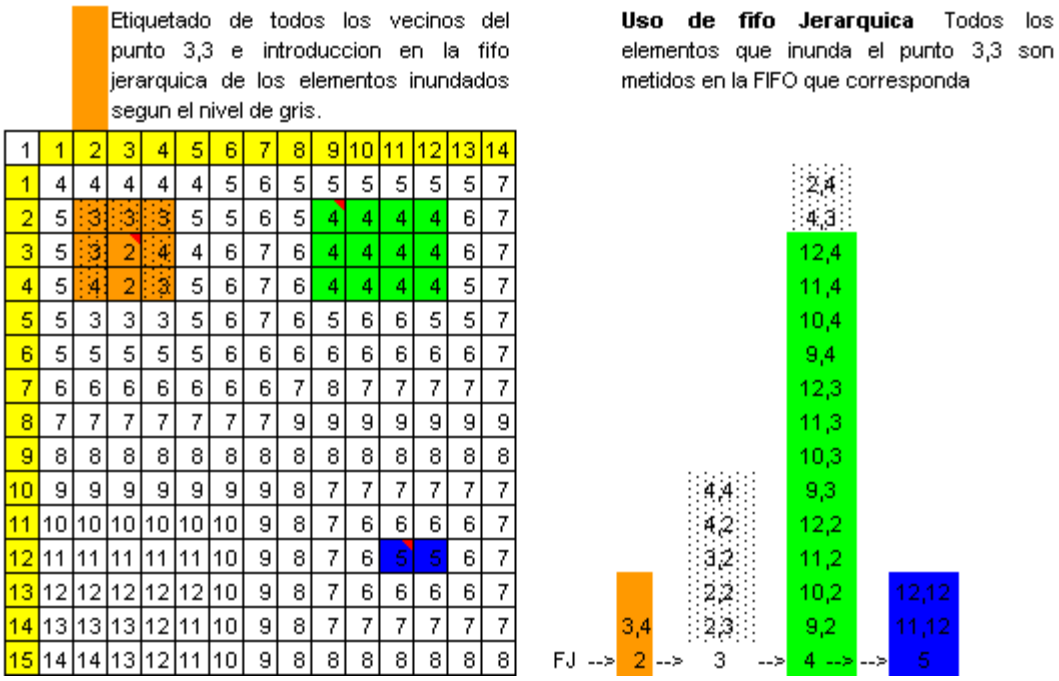


Figura 3.28. Uso de una FIFO jerárquica.

De la misma forma se procesa el siguiente elemento de la FIFO 2, que es el 3,4.

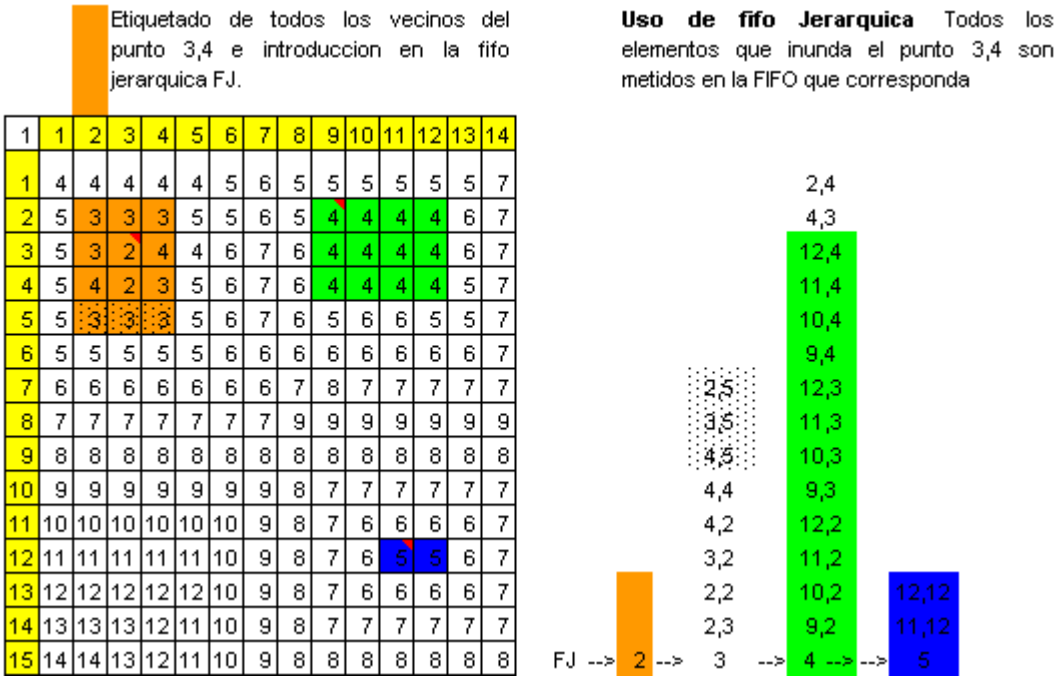


Figura 3.29. Uso de una FIFO jerárquica.

Hasta aquí ya terminamos con todos los elementos de la FIFO con valor 2 y lo que sucedió en la imagen fue que inundamos todos los vecinos de los elementos que pertenecían a esta FIFO 2 y los mandamos a sus FIFO correspondiente.

Lo que sigue ahora es tomar la siguiente FIFO que no este vacía y tenga el valor menor, en este caso el la FIFO con valor 3, vamos a inundar directamente sobre la tabla todos los elementos que tengan valor 3, que son los mismos que están en la FIFO con valor 3, y así sucesivamente con las demás FIFOs, poniendo atención de construir un dique para que no se junten las vertientes durante el proceso de inundación, en nuestro caso lo marcaremos con rojo.

La búsqueda de los elementos marcados a inundar se realizara sobre la tabla empezando por el sentido de los punto (1,1), (1,2) y terminando con los (13,15), (14,15) que es el mismo en que se introducen en las FIFOS.

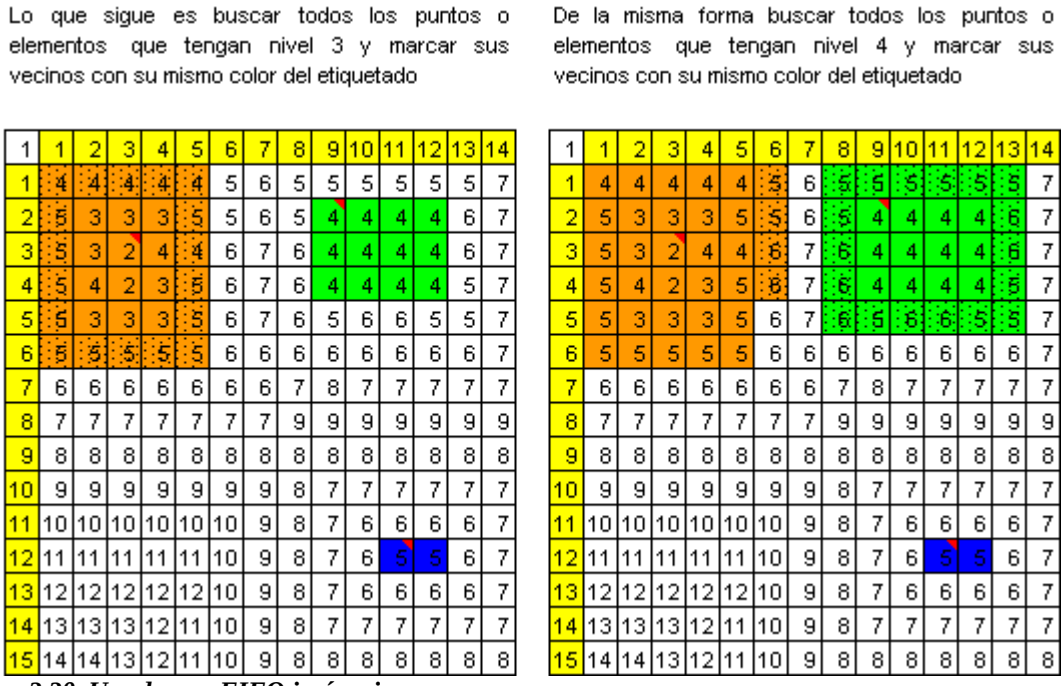


Figura 3.30. Uso de una FIFO jerárquica.

Ahora buscar el valor minimo de los marcados, que es 5 y repetir el proceso anterior, no permitir que se junten 2 vertientes . En el caso que se quieran juntar marcarlos con mask, en este caso sera el rojo

1	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	4	4	4	4	4	5	6	5	5	5	5	5	5	7
2	5	3	3	3	5	5	6	5	4	4	4	4	6	7
3	5	3	2	4	4	6	7	6	4	4	4	4	6	7
4	5	4	2	3	5	6	7	6	4	4	4	4	5	7
5	5	3	3	3	5	6	7	6	5	6	6	5	5	7
6	5	5	5	5	5	6	6	6	6	6	6	6	6	7
7	6	6	6	6	6	6	6	7	8	7	7	7	7	7
8	7	7	7	7	7	7	7	9	9	9	9	9	9	9
9	8	8	8	8	8	8	8	8	8	8	8	8	8	8
10	9	9	9	9	9	9	9	8	7	7	7	7	7	7
11	10	10	10	10	10	10	9	8	7	6	6	6	6	7
12	11	11	11	11	11	10	9	8	7	6	5	5	6	7
13	12	12	12	12	12	10	9	8	7	6	6	6	6	7
14	13	13	13	12	11	10	9	8	7	7	7	7	7	7
15	14	14	13	12	11	10	9	8	8	8	8	8	8	8

Ahora buscar el valor minimo de los marcados, que es 6 y repetir el proceso anterior, no permitir que se junten 2 vertientes .

1	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	4	4	4	4	4	5	6	5	5	5	5	5	5	7
2	5	3	3	3	5	5	6	5	4	4	4	4	6	7
3	5	3	2	4	4	6	7	6	4	4	4	4	6	7
4	5	4	2	3	5	6	7	6	4	4	4	4	5	7
5	5	3	3	3	5	6	7	6	5	6	6	5	5	7
6	5	5	5	5	5	6	6	6	6	6	6	6	6	7
7	6	6	6	6	6	6	6	7	8	7	7	7	7	7
8	7	7	7	7	7	7	7	9	9	9	9	9	9	9
9	8	8	8	8	8	8	8	8	8	8	8	8	8	8
10	9	9	9	9	9	9	9	8	7	7	7	7	7	7
11	10	10	10	10	10	10	9	8	7	6	6	6	6	7
12	11	11	11	11	11	10	9	8	7	6	5	5	6	7
13	12	12	12	12	12	10	9	8	7	6	6	6	6	7
14	13	13	13	12	11	10	9	8	7	7	7	7	7	7
15	14	14	13	12	11	10	9	8	8	8	8	8	8	8

Ahora buscar el valor minimo de los marcados, que es 7 y repetir el proceso anterior, no permitir que se junten 2 vertientes .

1	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	4	4	4	4	4	5	6	5	5	5	5	5	5	7
2	5	3	3	3	5	5	6	5	4	4	4	4	6	7
3	5	3	2	4	4	6	7	6	4	4	4	4	6	7
4	5	4	2	3	5	6	7	6	4	4	4	4	5	7
5	5	3	3	3	5	6	7	6	5	6	6	5	5	7
6	5	5	5	5	5	6	6	6	6	6	6	6	6	7
7	6	6	6	6	6	6	6	7	8	7	7	7	7	7
8	7	7	7	7	7	7	7	9	9	9	9	9	9	9
9	8	8	8	8	8	8	8	8	8	8	8	8	8	8
10	9	9	9	9	9	9	9	8	7	7	7	7	7	7
11	10	10	10	10	10	10	9	8	7	6	6	6	6	7
12	11	11	11	11	11	10	9	8	7	6	5	5	6	7
13	12	12	12	12	12	10	9	8	7	6	6	6	6	7
14	13	13	13	12	11	10	9	8	7	7	7	7	7	7
15	14	14	13	12	11	10	9	8	8	8	8	8	8	8

Ahora buscar el valor minimo de los marcados, que es 8 y repetir el proceso anterior.

1	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	4	4	4	4	4	5	6	5	5	5	5	5	5	7
2	5	3	3	3	5	5	6	5	4	4	4	4	6	7
3	5	3	2	4	4	6	7	6	4	4	4	4	6	7
4	5	4	2	3	5	6	7	6	4	4	4	4	5	7
5	5	3	3	3	5	6	7	6	5	6	6	5	5	7
6	5	5	5	5	5	6	6	6	6	6	6	6	6	7
7	6	6	6	6	6	6	6	7	8	7	7	7	7	7
8	7	7	7	7	7	7	7	9	9	9	9	9	9	9
9	8	8	8	8	8	8	8	8	8	8	8	8	8	8
10	9	9	9	9	9	9	9	8	7	7	7	7	7	7
11	10	10	10	10	10	10	9	8	7	6	6	6	6	7
12	11	11	11	11	11	10	9	8	7	6	5	5	6	7
13	12	12	12	12	12	10	9	8	7	6	6	6	6	7
14	13	13	13	12	11	10	9	8	7	7	7	7	7	7
15	14	14	13	12	11	10	9	8	8	8	8	8	8	8

Figura 3.31. Uso de una FIFO jerárquica.

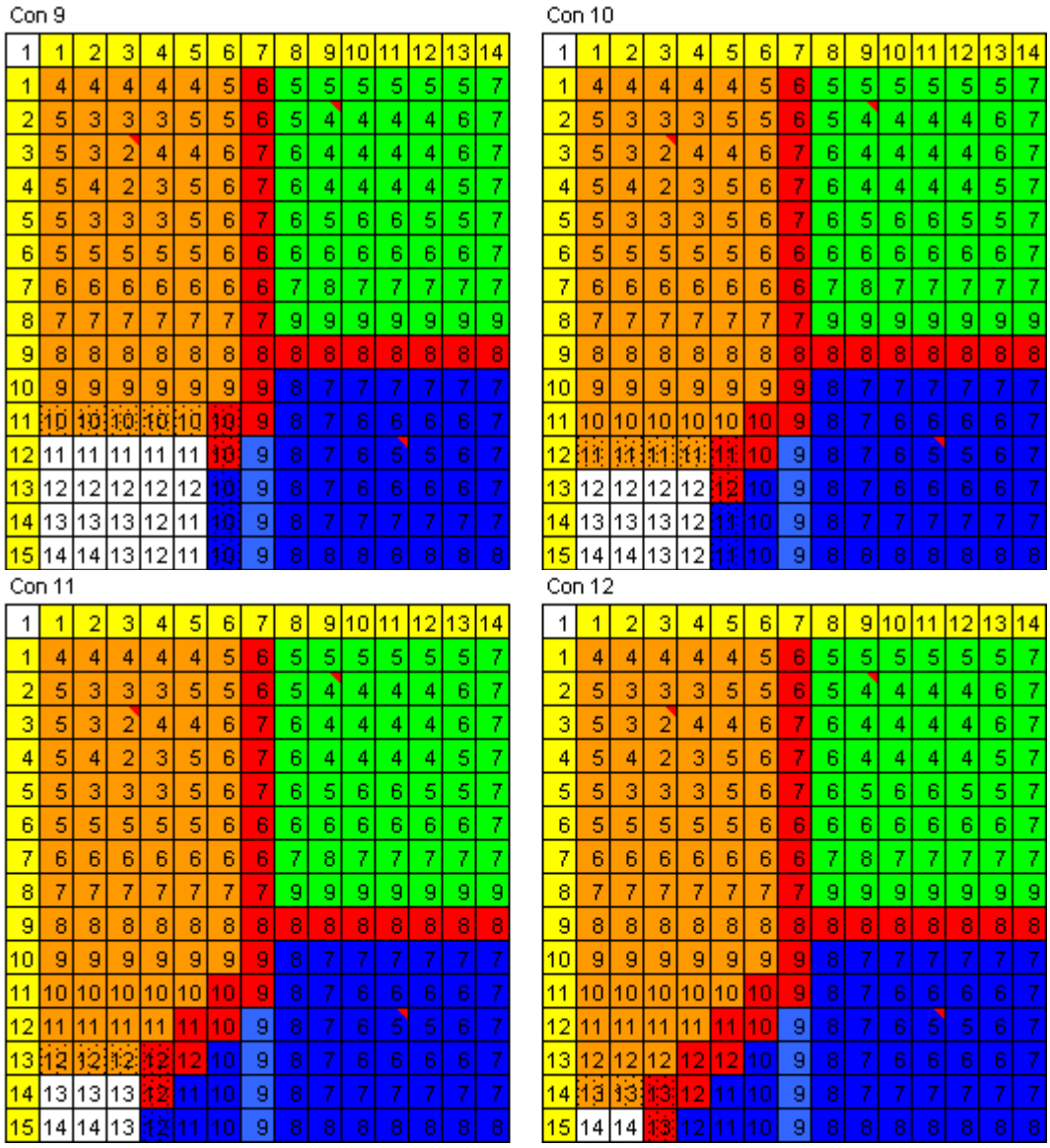


Figura 3.32. Uso de una FIFO jerárquica.

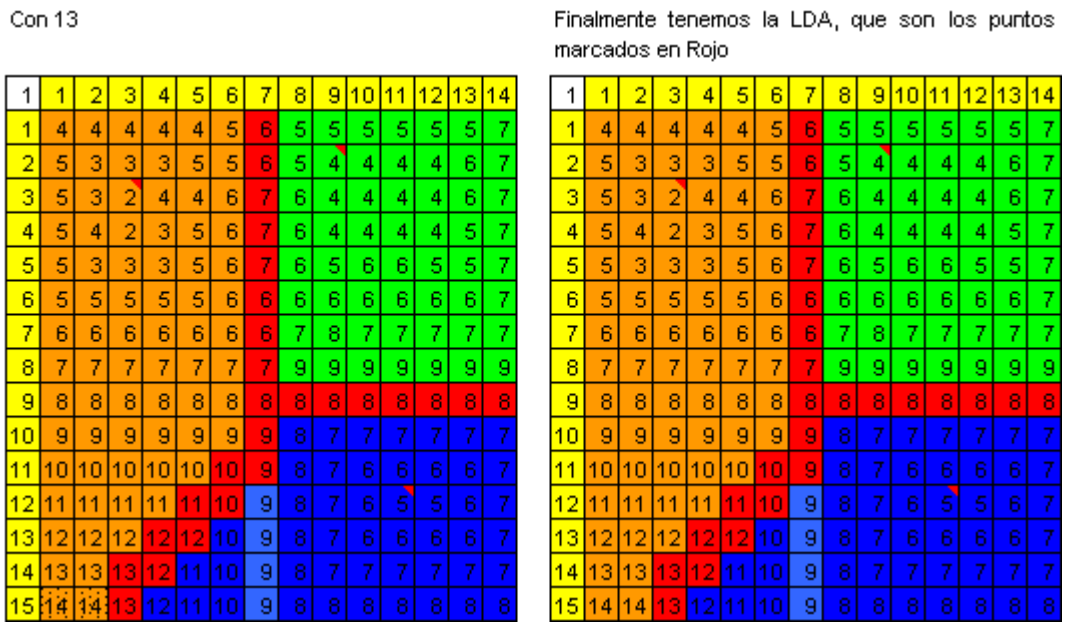


Figura 3.33. Uso de una FIFO jerárquica.

3. III.5.1 Algoritmo para LDA y Vertientes, uso de una FIFO jerárquica.

Procedimiento **Watershed (ime, mask, ims)**

Imagen de Entrada Numérica ***ime***
Imagen Binaria ***mask*** Conteniendo los mínimos de ime
Imagen Binaria de Salida ***ims*** (Para el Watershed)

Imagen de Trabajo ***imwl*** (Para las Vertientes)
Etiquetado de los mínimos de ***imwl ←mask***

// Inicializar Fifo Jerárquica

Meter puntos de contorno de los mínimos etiquetados en la fifo jerárquica a la jerarquía de ***ime*** de la imagen de entrada

Para Todo Punto ***p*** de la Imagen ***imwl*** Hacer

Si ***imwl***[***p***] diferente de ***cero*** hacer
ban_ =0xffff;
Para Todo 8 Vecino ***q*** de ***p*** Hacer
ban_ =ban_ & imwl[q] // & intersección bit x bit
Fin Para Todo

Si ***ban_*** es igual a cero
Fifoj ← ***p*** (en su jerarquía dada por ime[p])
Fin Si
Fin Si
Fin Para Todo

Mientras la ***fifoj*** no este vacía hacer:
P ← ***fifoj*** // Extraer coordenada
Para todo 8 vecino ***q*** de ***p*** hacer //de imwl
Si ***imwl***[***q***] igual a cero //no esta inundado todavia

Para todo 8 vecino ***q'*** de ***q*** hacer
// checar si es punto de watershed
Si ***imwl***[***q'***] <> de ***imwl***[***p***] y ***mwl***(***q'***) <>0 hacer
ims[***q***]=255 //Watershed //**aquí se tocan 2 vertientes distintas**
Fin Si
Fin Para todo

imwl[q] ← ***imwl[p]*** //Vertientes
fifoj ← ***q;***
Fin Si
Fin Para todo
Fin Mientras

3 METODOLOGÍA

SEGUNDA PARTE:

EL ALGORITMO PROPUESTO.

3. IV. Algoritmo multi-escala de segmentación interactiva de imágenes.

3. IV.1. Aproximación Multi-escala.

Las transformaciones básicas multi-escala en morfología matemática están formadas por las aperturas y cerraduras por reconstrucción. Aún más, la aperturas y cerraduras por reconstrucción forman una pirámide de operadores conexos. Esta propiedad interesante explica los efectos de simplificación de una imagen sin introducir nuevos contornos. De esta forma, una pirámide $\{\Psi_\lambda\}$ formada por operadores conexos incrementa el tamaño de las zonas planas conforme el tamaño λ (elemento estructural) incrementa su tamaño; las zonas planas se fusionan.

La clásica pirámide en morfología matemática esta formada por las granulometrías basadas en aperturas por reconstrucción, para $\lambda < \mu$, se tiene que la partición de $\tilde{\gamma}_\lambda(f)$ generada por la noción de zona plana es más fina que aquella generada por $\tilde{\gamma}_\mu(f)$. Para generar un método de procesamiento multi-escala, algunas propiedades son requeridas. Entre ellas, la causalidad y la preservación de los contornos, son las propiedades más importantes. La causalidad implica que las escalas menos finas son únicamente originadas por lo que sucede en las escalas más finas. La Figura 35 ¿ilustra el efecto de simplificación de una imagen usando la apertura por reconstrucción? $\tilde{\gamma}_\lambda(f)$. Las imágenes contienen cada vez menos detalles: algunas estructuras se preservan; otras son removidas de una escala a la otra. Por otra parte, si el objetivo principal es la segmentación de imágenes se requiere la preservación de contornos; los contornos deben permanecer si ningún desplazamiento. Sin embargo, aunque se tiene una jerarquía de representaciones a nivel segmentación de imágenes, es muy difícil o complejo tratar de extraer a partir de una representación de este tipo las diferentes componentes de la imagen a sus escalas. Se requiere tener la información codificada en forma adecuada sobre una estructura de datos, de tal forma que permita suprimir regiones de manera simple o que nos entregue una

imagen con un número de regiones. Este tipo de estructura permitiría generar un algoritmo interactivo de segmentación de imágenes. En las secciones siguientes, desarrollaremos un método de segmentación multi-escala basado en un MST.

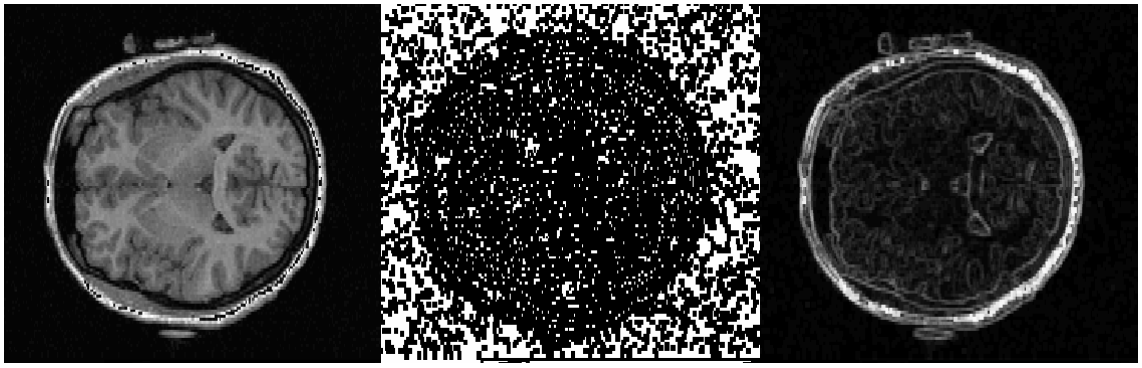


Figura 3.34. (a)Original

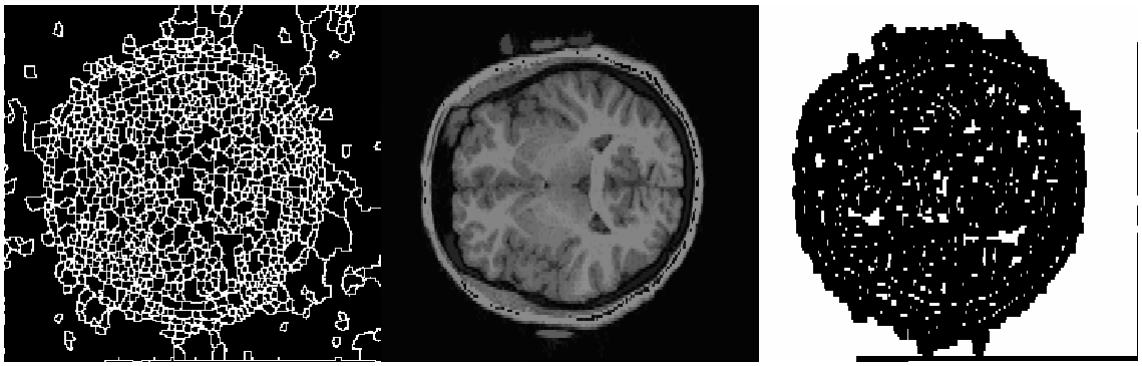


Figura 3.34. (b)Apertura 2

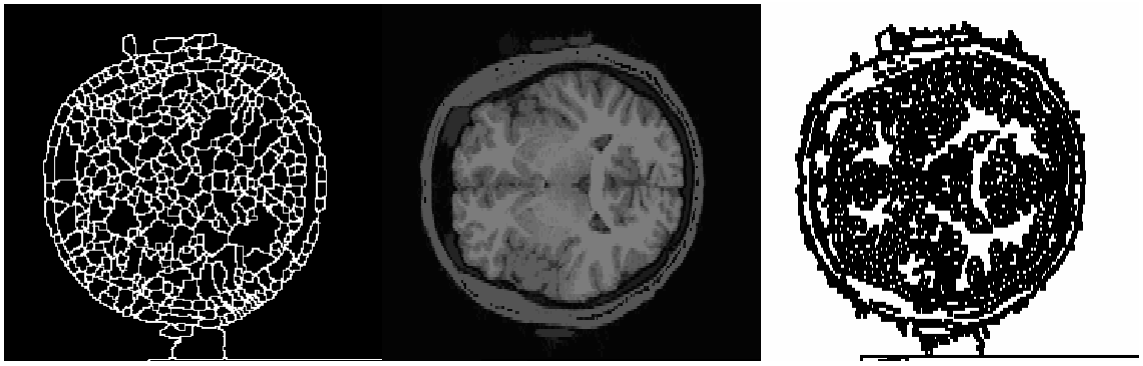


***Figura 3.34. (c)*Apertura 5**

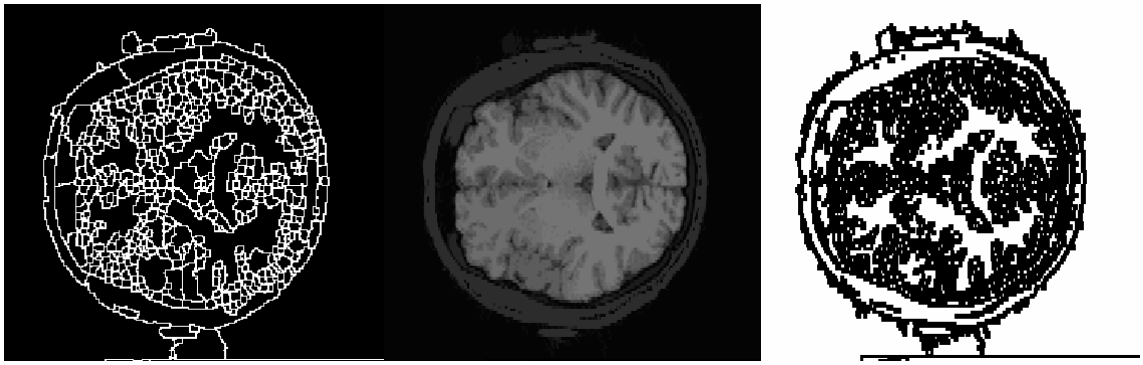
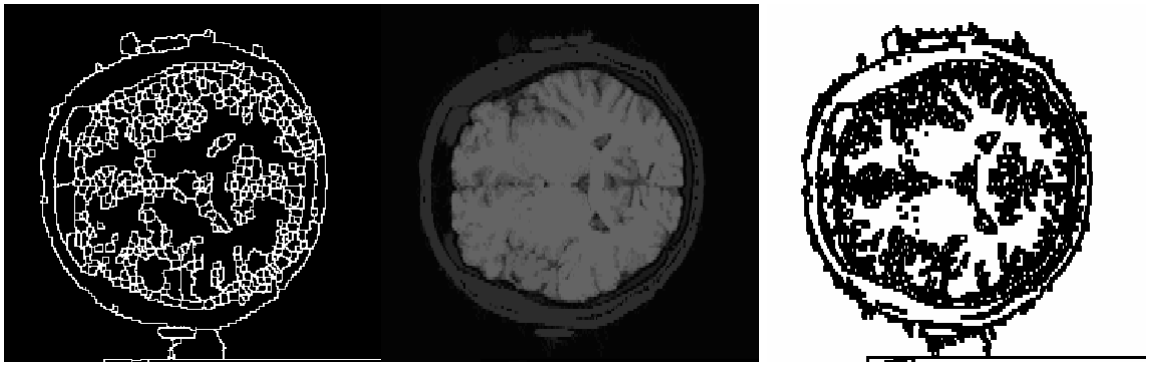


Figura 3.34. (d)Apertura 8



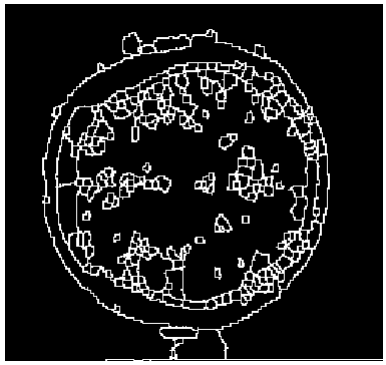


Figura 3.34. (e)Apertura11

Figura 3.34. Aperturas aplicadas a (a) imagen original.

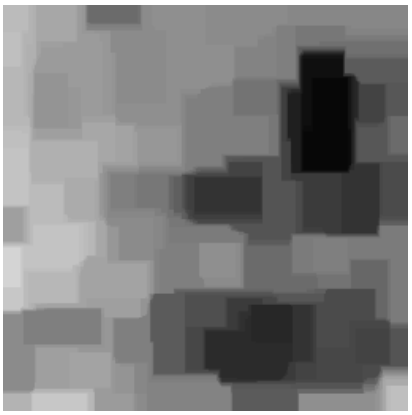
3. IV.2. Generación del MST.

En los capítulos anteriores se describieron las estructuras de datos principales. En particular, el grafo y su estructura asociada, el MST generado a partir de los algoritmos de Prim o Kruskal. Sin embargo, para generar el método interactivo, no utilizaremos el algoritmo de prim o Kruskal para obtener nuestro MST asociado a la imagen. Esto debido a que funcionan sobre un grafo ponderado no dirigido sin lazos, y lo que vamos a hacer es crear el MST en el proceso de inundación,

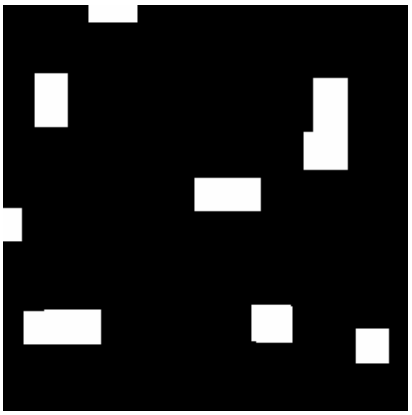
Debido a que el grafo MST se genera durante el proceso de inundación del Watershed, veamos como es el proceso.

En esta primera parte se desea obtener un grafo MST que represente a todas las vertientes de la imagen original, este grafo será utilizado posteriormente para realizar las particiones o segmentaciones de la imagen original y fusión de regiones.

Como primer paso se calculan todos los mínimos de la imagen original, los cuales tendrán cada uno un nodo correspondiente en el MST creado al final.

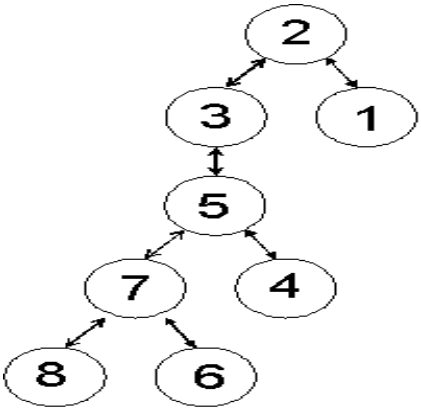


a) Imagen original



b) Mínimos de la imagen Original

c) Vertientes de la Imagen Original



d) grafo asociado a las vertientes

Figura 3.35. Mínimos, Vertientes y grafo asociado a una imagen,

En la imagen de la ***Figura 3.35 (b)*** se observan todos los mínimos de la imagen original, un mínimo puede estar formado por varios puntos, para nuestro

proceso se tomara un punto de cada mínimo el cual será el punto representativo de la vertiente.

En la ***Figura 3.36.*** Cada mínimo representativo (cx) de la imagen original esta asociado a un nodo en memoria (nx) .

Estos nodos se crean durante el proceso de inundación de las vertientes. En la sección anterior se ilustró el proceso de inundación a partir de los mínimos de una imagen. Cada mínimo representa una clase dada que generará un elemento de la partición. De esta forma, por cada mínimo existe un nodo, las aristas que conectan a los nodos se determinan de la siguiente manera:

Cuando se tocan (En el proceso de inundación, ver Algoritmo para LDA y Vertientes Uso de una Fifo Jerárquica) dos vertientes se calcula un parámetro (área, volumen o altura), (para nuestro caso estamos trabajando con el área), en ambas vertientes y se etiquetan la aristas (Existen realmente 2 aristas entre los nodos, una para apuntar del nodo ***a*** al ***b*** y otra para apuntar del nodo ***b*** al ***a*** entre los nodos con el menor valor del parámetro calculado. Ver *Proceso de valorado de aristas entre nodos* que se menciona más adelante.

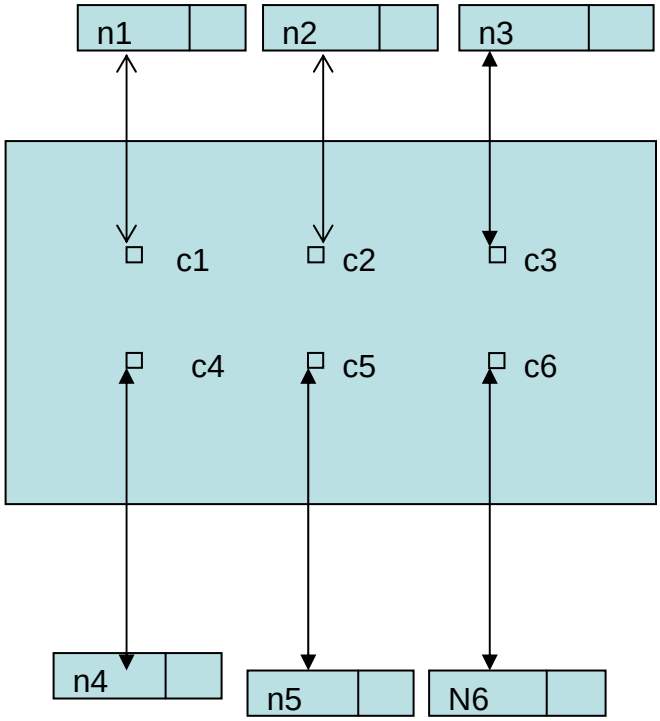


Figura 3.36. Los mínimos representativos.

Inicialmente todos los mínimos representativos (***c1, c2,.., c6***)) de las vertientes tienen una clase distinta (la clase determina a que subárbol pertenece dicho nodo), debido a que cada mínimo representa un elemento de la partición de la imagen de vertientes de la imagen (***Figura 3.35. (c***)

3. IV.3. Calculo de área, Altura y Volumen de vertientes en imágenes.

3. IV.3.1. Cálculo de área de vertientes en imágenes:

Cuenta los puntos de la superficie de las vertientes, de la imagen de vertientes que se va formando en el proceso de inundación, En una altura o nivel de agua determinada (Justo en el momento en que se conocen las 2 vertientes que aun no se han conocido).

Cuando se dice que dos vertientes se conocen se refiere a que al agregar un nivel más de agua se fusionarían,

3. IV.3.2. Cálculo de la altura de las vertientes en imágenes:

Justo en el momento en que dos vertientes se conocen durante el proceso de inundación, se pueden calculan la altura determinada por la diferencia entre el nivel del agua actual menos el nivel de agua original.

3. IV.3.3. Cálculo del volumen de las vertientes en imágenes:

Como tercera opción también se puede calcular el volumen de las vertientes para etiquetar las aristas de las vertientes que se van conociendo durante el proceso de inundación, es cual esta determinado por la suma de las alturas de todos los puntos de una región dada.

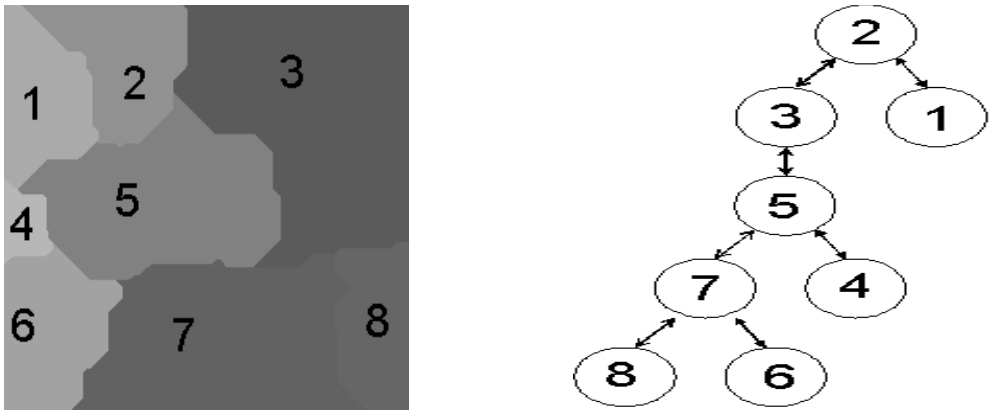
3. IV.4. Proceso de valorado de aristas entre nodos.

Siempre que dos vertientes o conjunto de vertientes se conocen y tiene distinta clase, durante el proceso de inundación se realiza el siguiente procedimiento; (Conjunto de vertientes se refiere a que 2 o más mínimos están fusionados en una misma clase)

a) Dos aristas son agregada entre los nodos a, b correspondientes a las vertientes, una arista apunta del nodo a al b y la otra apunta del nodo b al a.
b) Se calcula Área, Volumen o Altura de ambas vertientes y se asigna el valor más pequeño de las vertientes que se conocen, a la aristas agregada en el punto a).
c) En una imagen diferente, Ambas clases de vertientes se fusionan en una sola clase. Igual se etiquetan los nodos del MST involucrados.

Estos pasos se realizan sobre los nodos del MST que representan a cada uno de los mínimos de la imagen original, los mínimos de la imagen representan a su vez a una partición de la imagen de vertientes.

Al final se obtiene un grafo MST asociado a las vertientes como el de la ***Figura 3.37. (b)*** en el cual cada nodo está representando una vertiente de la ***Figura 3.37. (a).***



a) b)

Figura 3.37. Un grafo MST asociado

Por ejemplo: (Tomado y adaptado del articulo A “Toolbox for Interactive Segmentación Based on Nested Partitions”)

Supongamos que tenemos 4 vertientes con los siguientes mínimos asociados.

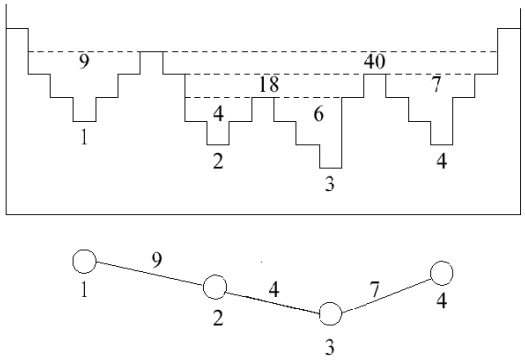
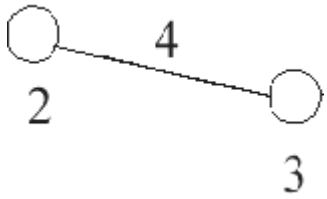


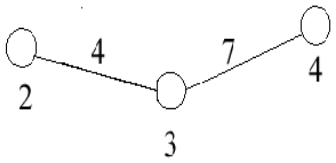
Figura 3.38. Vertientes con mínimos asociados.

Veamos paso a paso como se obtiene el grafo.

En el proceso de inundación primero se conocen las vertientes 2 y 3, el Volumen en ese nivel del agua es de 4 y 6 respectivamente, se crean 2 nodos y la arista entre estos nodos. Como 4 < 6 tenemos el siguiente grafo inicial. Ambos nodos se fusionan en una clase



Después la vertiente 4 conoce a las vertientes 2 y 3, Los Volúmenes 7 y 18 respectivamente, Los 3 nodos se fusionan en una misma clase.



Finalmente la vertiente 1 conoce a la vertiente formada por las vertientes 2,3 y 4 con valores igual a 9 y 40 respectivamente, quedando finalmente el grafo MST asociado.

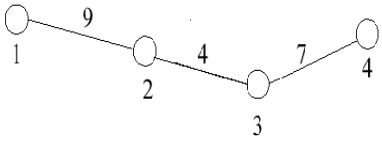


Figura 3.39. Pasos para obtener un grafo asociado

3. IV.5. Recorrido del grafo MST.

La operación de recorrer una estructura consiste en visitar (procesar) cada uno de los nodos a partir de uno dado. Existe 2 formas: Recorrido en profundidad y recorrido en anchura.

El recorrido en profundidad se denomina así porque la dirección de visita es hacia adelante mientras que sea posible, al contrario que el recorrido en anchura, que primero visita todos los nodos posibles en amplitud.

El recorrido principal que se esta utilizando para procesar nuestra grafo creada es el de anchura, esto debido a que parece ser el más adecuado a nuestra aplicación, en la cual nos interesan más las componentes conexas vecinas.

El recorrido en anchura consiste en visitar el nodo de partida A y a continuación visitar los nodos adyacentes que no estén ya visitados, de igual manera los nodos adyacentes.

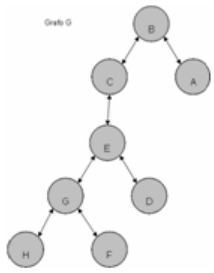
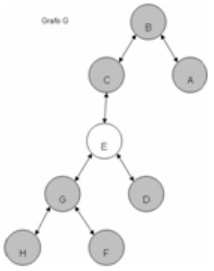
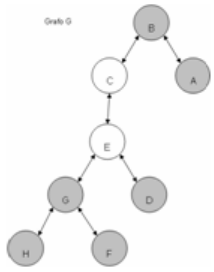
Los pasos para el recorrido de anchura son los siguientes:

1. Visitar un nodo dado E.
2. Meter en la cola W el nodo E y marcarlo como procesado.
3. Repetir los pasos 4 y 5 hasta que la cola W este vacía.
4. Retirar el nodo frente de la cola (W), y Visitarlo
5. Meter en la cola W todos los nodos adyacentes del nodo retirado de W que no estén procesados y marcarlos como procesados
6. Fin

3. IV.5.1. Ejemplo de recorrido en anchura.

Queremos Visitar todos los nodos del grafo G.

Ver ejemplo en la siguiente ***Figura 3.40 Ejemplo de recorrido en anchura***.

Consecutivo	Paso del Algoritmo (Recorrido en Anchura)	Grafo	<-Elementos en Cola W	Nodos		Comentario
				Procesados	Visitados	
1	1. Visitar un nodo dado E.					Se puede enpezar desde cualquier nodo, al final se visitan todos
2	2. Meter en la cola W el nodo E y marcarlo como procesado.		**E**	**E**		Lo que se ira marcando de blanco en el grafo sera los nodo visitados.
3	3. Repetir los pasos 4 y 5 hasta que la cola W este vacía.		E	E		Solo se pondra el grafo, si cambia respecto al anterior
4	4. Retirar el nodo frente de la cola (W), y Visitarlo			E	**E**	Se visita nodo E
5	5 Meter en la cola W todos los nodos adyacentes del nodo retirado de W que no estén procesados y marcarlos como procesados		**C, G, D**	**E, C, G, D**	E	Los nodos abyacentes de E son C,G, y D y se agregan a la Cola W
6	4. Retirar el nodo frente de la cola (W), y Visitarlo		**G, D**	E, C, G, D	**E, C**	Se visita nodo C
7	5 Meter en la cola W todos los nodos adyacentes del nodo retirado de W que no estén procesados y marcarlos como procesados		**G, D, B, A**	**E, C, G, D, B, A**	E, C	

Recorrido en anchura (***Figura 3.40. Continuación…***)

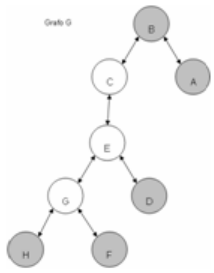
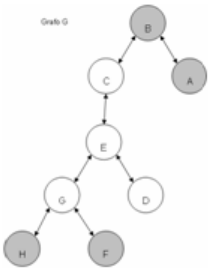
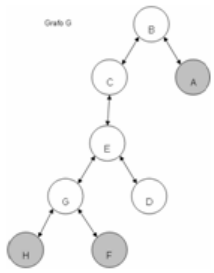
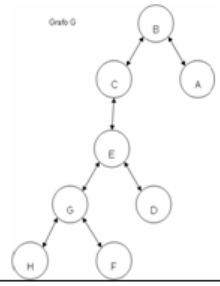
Consecutivo	Paso del Algoritmo (Recorrido en Anchura)	Grafo	<-Elementos en Cola W	Nodos		Comentario
				Procesados	Visitados	
8	4. Retirar el nodo frente de la cola (W), y Visitarlo		**D, B, A**	**E, C, G, D, B, A**	**E, C,G**	
9	5 Meter en la cola W todos los nodos adyacentes del nodo retirado de W que no estén procesados y marcarlos como procesados		**D, B, A, H,F**	**E, C, G, D, B, A,H,F**	E, C,G	
10	4. Retirar el nodo frente de la cola (W), y Visitarlo		**B, A, H,F**	E, C, G, D, B, A,H,F	**E, C,G,D**	
11	5 Meter en la cola W todos los nodos adyacentes del nodo retirado de W que no estén procesados y marcarlos como procesados		B, A, H,F	E, C, G, D, B, A,H,F	E, C,G,D	Ya todos los nodos estan procesados, por lo que no existen nodos a agregar a la cola, El paso 5 ya no lo pondemos mas en este ejemplo para no hacelo largo
12	4. Retirar el nodo frente de la cola (W), y Visitarlo		**A, H,F**	E, C, G, D, B, A,H,F	**E, C,G,D,B**	Este paso 4 consiste en sacar los elementos de la cola W y Visitarlos, por lo que al ejecutarse 3 veces mas queda como sigue abajo.
13	4. Retirar el nodo frente de la cola (W), y Visitarlo			E, C, G, D, B, A,H,F	**E, C, G, D, B, A, H, F**	Todos los nodos han sido visitados
14	6 Fin					

Figura 3.40. Ejemplo de recorrido en anchura de un grafo

3. IV.6. División de MST.

Como se menciono anteriormente el grafo que se genero por medio de la inundación de vertientes esta asociado a la vertientes de la imagen original. Cada nodo esta asociado a una vertiente y todos los nodos están unidos formando un árbol de expansión mínimo (MST).

Lo que se pretende explicar es como se realiza el proceso de división del árbol de expansión mínimo para lograr segmentarlo en n sub árboles lo que implicaría que la imagen se segmente también en n regiones conexas.

Este proceso puede ser aplicado a cada uno de las sub árboles generados en el punto anterior. Es decir particionar en *m* particiones a una de las particiones generadas anteriormente.

3. IV.6.1. División del grafo MST en dos particiones.

Empezaremos por el proceso más simple, el de dividir el MST asociado a la imagen de vertientes original en 2, esto implica que también la imagen original se divida en 2, estaremos trabajando en esta parte con la imagen de vertientes obtenida de la imagen original, y el grafo generado MST.

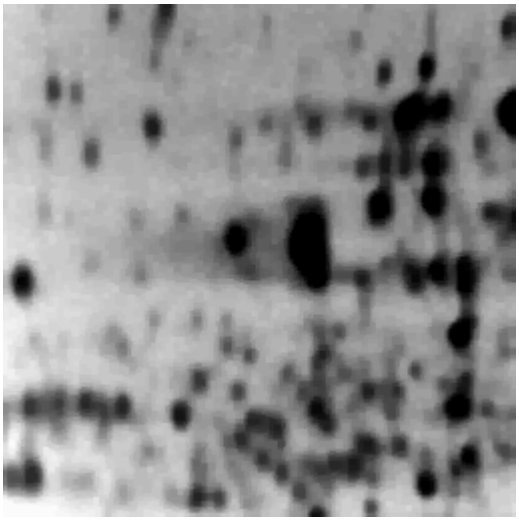


Figura 3.41.. Imagen original gel de eletrofolisis

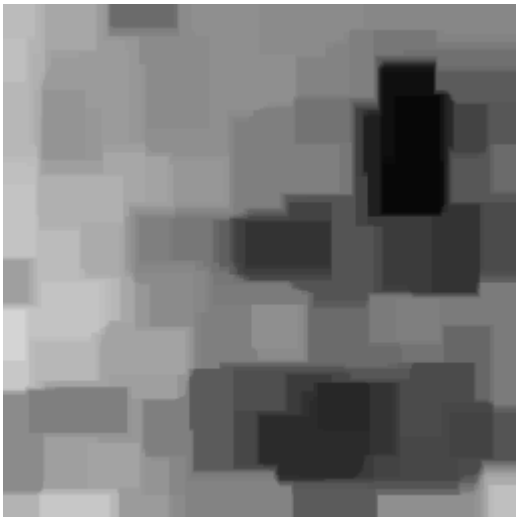


Figura 3.42. Se le aplico apertura y cerradura de tamaño 10 a la imagen original

Deseamos segmentar nuestra imagen ***Figura 3.42.*** en 1 región, La imagen de vertientes contiene a 8 particiones, lo cual implica que nuestro MST debe contener 8 nodos, y como deseamos una sola partición, La imagen completa tendría un grafo asociado MST de 8 nodos, en donde cada nodo contiene una variable llamada subárbol con el mismo valor, en este caso 1. Para efectos de mostrar mejor a que subárbol pertenece determinado nodo lo etiquetaremos con el valor de la variable subárbol.

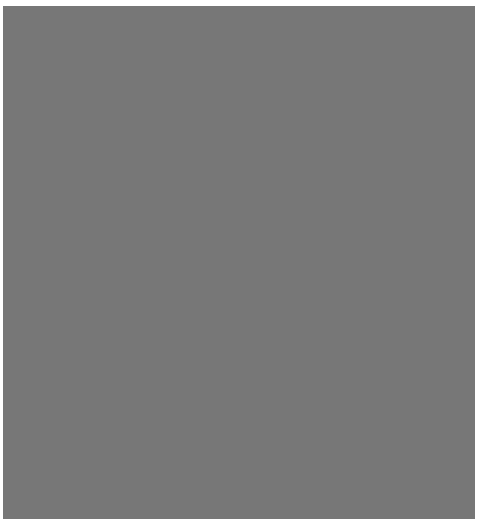


Figura 3.43. Imagen A la que se le aplico una segmentación de 1, es el nivel más áspero de la imagen con solo 1 partición

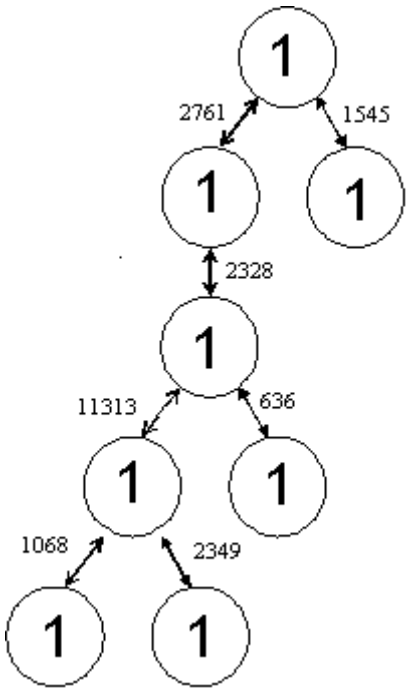


Figura 3.44. MST asociado a la Figura 3.43., el numero 1 en cada uno de los nodos es una etiqueta que indica que el nodo pertenece a la partición o subárbol 1.

Aquí vemos que hay 3 nodos etiquetados con 3, Lo cual significa que los 3 nodos están fusionados una sola región, lo mismo para los 5 nodos etiquetados con 1.

Figura 3.45. (a). *Imagen de vertientes con 2 segmentaciones*

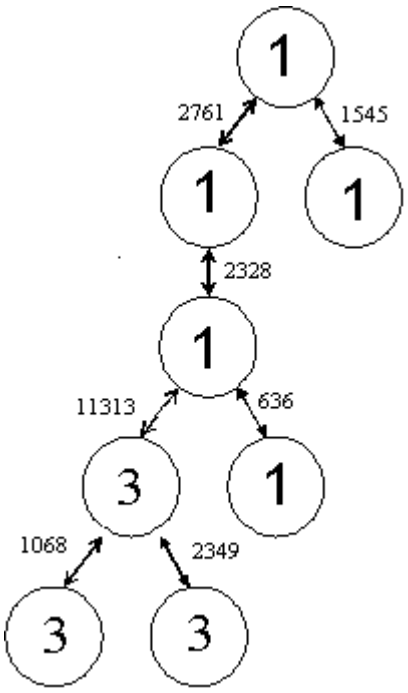


Figura 3.45. (b). *grafo MST asociado a las regiones de la* ***Figura 3.45. (a),***

Figura 3.45. Imagen de vertientes con 2 segmentaciones y su grafo asociado.

Si deseamos todas las regiones de nuestra imagen (8 regiones para este caso), el MST asociado se forma de 8 nodos etiquetados de manera distinta cada uno (por ejemplo 1, 2, 3, 4, 5, 6, 7,8)

Figura 3.46. Imagen de vertientes de la Figura 3.42., con 8 Regiones

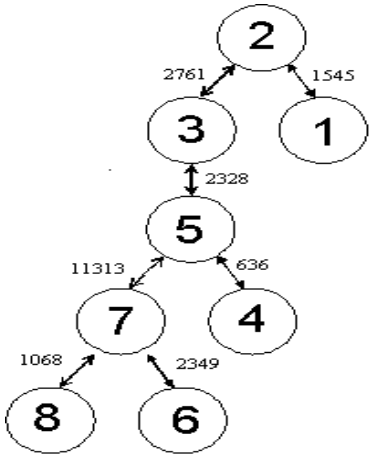


Figura 3.47.. MST asociado a la Figura 3.46., con 8 Regiones

El proceso de segmentar o dividir en dos un grafo MST cuyos nodos están etiquetados con el mismo valor se resume en los siguientes pasos (Realmente el grafo MST no se rompe, solo estamos etiquetando los nodos del MST según la clase o partición que pertenecen por esta razón estamos usando la variable subárbol contenida en cada uno de los nodos):

a) Identificar la arista que tenga el valor más grande de todas las aristas del grafo MST, en este caso seria la arista que esta valorada con 11313, e Identificar los nodos que están unidos por esta arista, supongamos que son los nodos **u** y **v** los que están unidos por la arista con valor **11313**.

b) Marcar a **u** como procesado, hacer un recorrido en anchura a partir del nodo **v** e ir procesando (Etiquetar con un valor *z* distinto de la etiqueta de *u*) los demás nodos.

El aplicar los pasos anteriores al grafo de la ***Figura 3.46.*** se obtuvo el grafo de la ***Figura 3.45.***

Si volvemos aplicar el mismo proceso de segmentar en 2, a uno de los subárboles obtenidos en la ***Figura 3.45.***, por ejemplo el subárbol que contiene a los nodos etiquetados con 1 se obtiene los resultados mostrados en la ***Figura 3.49.*** y ***Figura 3.50.***

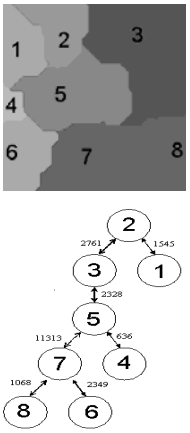


Figura 3.48. Imagen de Vertientes y su grafo MST Asociado

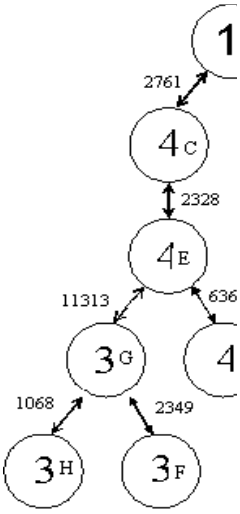


Figura 3.49. MST Asociado a Figura 3.50.

Figura 3.50. Imagen con 3 particiones, se particiona la imagen en 2, y de las 2 particiones resultantes la partición superior en 2.

3. IV.7. Representación del grafo en memoria.

3. IV.7.1 Nodo del MST.

Un nodo podría ser una estructura del siguiente tipo:

```
struct nodo_g
{
int x,y;                          (Es el punto x,y en la imagen que representa el minino regional de una vertiente.)
unsigned char mixmax, procesado;  (Nos indica la clase o subárbol a la que pertenece un nodo, y para controlar si el nodo ha sido procesado)
nodo_apun_g *primero;             Estructura que nos sirve para apuntar al nodo del grafo MST, Son las aristas en el MST.
nodo_g *siguiente;                Siguiente Nodo del MST.
}
```

3. IV.7.2. Aristas en el grafo.

Esta estructura representa a las aristas en el grafo

```
typedef struct nodo_apun nodo_apun_g;

struct nodo_apun
{
    nodo_g *nodo_grafo;
    nodo_apun *siguiente;
    long    area;
};
```

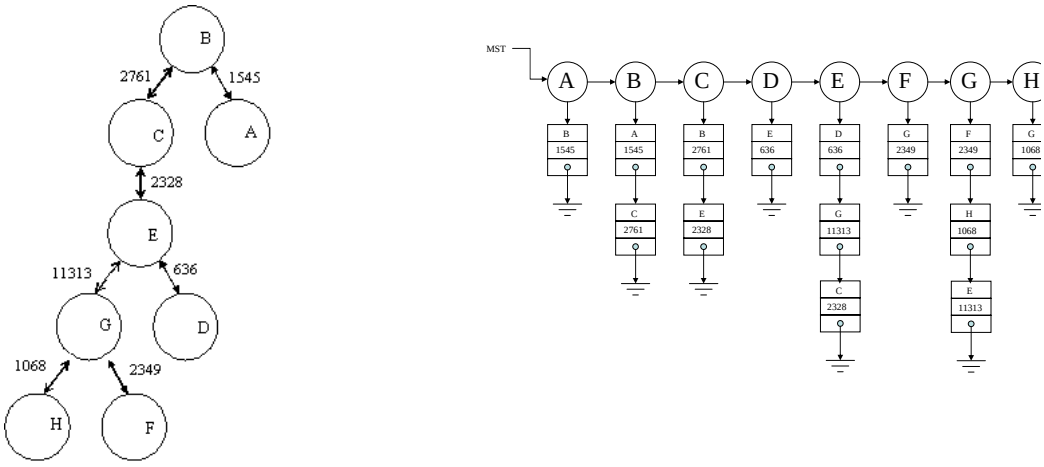
Las variables utilizadas son:

Variable	Uso
*nodo_grafo	Apuntador al nodo del grafo,
*siguiente	Apuntador al siguiente nodo
area	Es el valor de la arista

Tabla 4 Las variables utilizadas en Aristas en el Grafo

Se debe de representar un número finito de nodos y aristas. Se puede hacer de varias formas, de una manera secuencial mediante un arreglo o de manera dinámica por medio de una lista de adyacencia

Las listas de adyacencia son una estructura multienlazada formada por una lista directorio cuyos nodos representan los vértices del grafo y de cada uno de los nodos de la lista directorio emerge una lista enlazada cuyos nodos, a su vez, representan los arcos con vértice origen el del nodo de la lista del directorio y destino indicado.



Grafo

Nótese el doble sentido de todas las aristas en el grafo y su representación en la Lista de Adyacencia.

Figura 3.51. Representación del MST Mediante lista de adyacencia.

3. IV.8. división de un grafo MST en n particiones.

Usaremos algunas características o propiedades de un nodo del grafo creado; las variables *subarbol* y *procesado*. Y la variable de control *ArbolNuevo*

La variable *ArbolNuevo* se incrementa cada que se genera una nueva partición o subárbol y se utiliza para ir indicando el nuevo subárbol al que ahora pertenecen algunos nodos que se están segmentando.

La variable ***procesado*** no sirve para tomar el control de los nodos ya visitados y/o procesados:

Cuando se realiza un primer recorrido al **MST** se marcan los nodos como procesados utilizando la variable ***procesado= 1***, si se vuelve realizar otro recorrido al MST se marcan los nodos con 2 por medio de la variable *procesado*, y así sucesivamente para cada uno de los recorridos.

La variable subarbol nos indica a que subárbol, clase o partición pertenece determinado nodo. Adicionalmente el grafo cuando se crea por primera vez tiene marcado todos los nodos con el **valor 1**, lo cual nos indica que todos los nodos pertenecen al **MST 1**, esto utilizando la variable **subarbol=1**.

En la ***Figura 51***. los valores de la propiedad *subarbol* podrían ser:

Nodo	**Valor de la propiedad o variable subarbol**
A	1
B	1
C	4
D	4
E	4
F	4
G	3
H	3

Tabla 5. Valores de la propiedad subarbol

Los valores mostrados en la ***tabla 5*** nos indican que:

a) Los nodos A y B pertenecen al subárbol 1, de la misma forma :

b) Los nodos C, D, E y F al subárbol 4 y

c) Los nodos G al H al subárbol 3

Así aplicando los puntos vistos anteriormente, el algoritmo para dividir un MST en n subárboles seria el siguiente:

Supongamos que se ha creado el siguiente grafo valorado en el proceso inundación de vertientes:

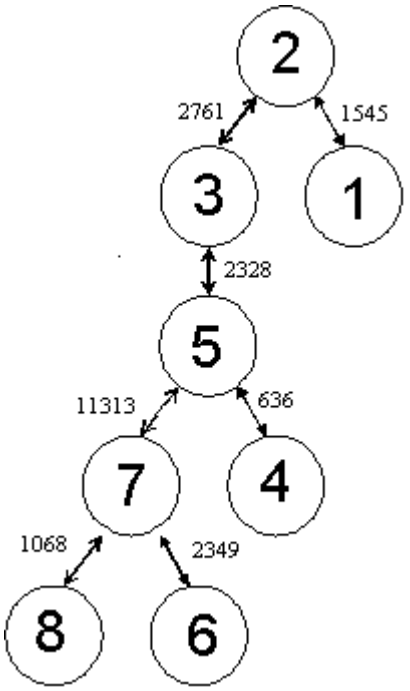


Figura 3.52. Grafo valorado

Lo primero es crear una función que dado un grafo o subárbol genere una lista ordenada decreciente de los valores de las aristas (***Tabla 6***) en donde los nodos pertenezcan al mismo subárbol que se desea dividir en n subárboles.

Si deseamos particionar el MST de la **Figura 3.52.**, en n particiones;

La función nos debe de generar una lista como la que sigue de acuerdo el número de particiones deseado:

Numero de Elemento (k) en la lista	Valor de la arista a eliminar	Numero de particiones deseado (n = k+1)
1	11313	2
2	2761	3
3	2349	4
4	2328	5
5	1545	6
6	1068	7
7	636	8

Tabla 6. Particiones deseadas.

Se observa de la ***Tabla 6*** que el número de particiones *n* que se desean del mst deben ser de 2 hasta m+1, en donde *m* indica el número total de aristas en el grafo, Y *k* va desde 1 hasta *m*.

Si se desea generar *n* particiones hay que tomar los primeros *k = n-1* elementos de nuestra lista de la ***Tabla 6***. Todas las aristas que contengan los k valores tomados serán eliminadas del grafo generando de esta manera *n = k+1* particiones.

Sabiendo la utilización de esta lista veamos ahora como seria la segmentación del MST.

Se utiliza como base el mismo recorrido de anchura visto anteriormente con algunas modificaciones.

Como una primer parte:

Se hace un recorrido de anchura en el árbol y se genera la lista de las aristas en orden descendente con los k elementos, podría utilizarse cualquier otro recorrido el objetivo es obtener las lista de la ***tabla 6***.

Como segunda Parte:

parte realizar lo siguiente (Nótese que se utilizan ***2 FIFOS W y W1***):

Lo que se hace en este paso es etiquetar cada uno de los nodos como en la ***Tabla 1***, de acuerdo a las particiones deseadas de un subárbol.

1. Visitar un nodo dado A del Arbol X a Segmentar
2. Meter en la ***FIFO W1*** el nodo A y marcarlo como procesado.
3. Repetir los pasos 4 y 5 hasta que la FIFO W1 este vacía.
4. Retirar el nodo frente de la ***FIFO W1***, Meterlo en la ***FIFO W*** e Incrementar la variable Arbol nuevo en 1
5. Repetir los pasos 6 y 7 hasta que la FIFO W este vacía.
6. Retirar el nodo frente de la ***FIFO W***. etiquetar el nodo con el nuevo valor de subarbol que pertenece y marcar el nodo como procesado
7. Meter en la ***FIFO X*** todos los nodos adyacentes del nodo retirado en el paso 6 de la FIFO W que no estén procesados y pertenezcan al mismo subárbol que se esta segmentando,

   La ***FIFO X*** puede ser ***W*** o ***W1*** dependiendo de:

   Si el Valor de la arista hacia el nodo adyacente esta contenido en la lista de las aristas, Entonces meter en la FIFO W1, en otro caso meter en la FIFO W
8. Fin.

3. IV.8.1 Probar el Algoritmo.

Veamos el algoritmo con un ejemplo más general. Supongamos que tenemos el siguiente grafo asociado a una imagen de vertientes, el cual contiene

los nodos desde A hasta H, con arista valoradas (desde *a* hasta *g*) como se muestran en la figura. Como sabemos cada nodo contiene la variable *subárbol* la cual nos indica a que subárbol pertenece cada uno de los nodos, para este caso supongamos que tiene el valor *X* como se muestra en la ***Figura 55.***

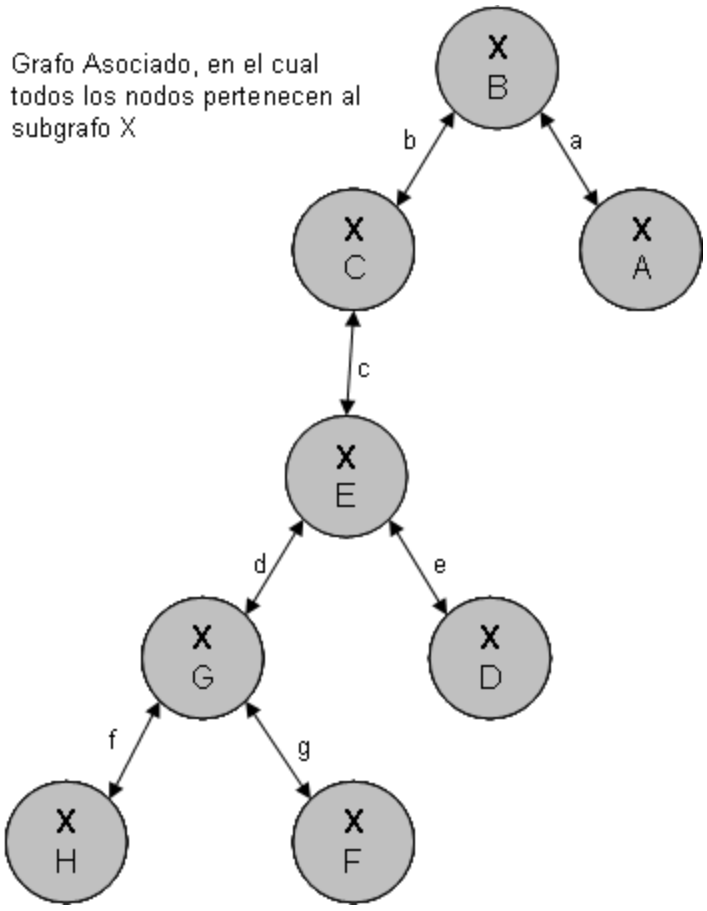


Figura 3.53. Un Grafo Asociado X.

Lo que deseamos es segmentar el grafo en 3 regiones (por ejemplo). Como un primer paso se debe determinar la lista ordenada en forma descendente de las aristas, supongamos que los elementos de esta lista quedan ordenados como *lista (d,b),* Las aristas con valor d y b son las que debemos de quitar para que nuestro grafo X se separa en tres subgrafos, entonces lo que tendríamos al final del proceso es una grafo asociado como el de la ***Figura 3.54.***

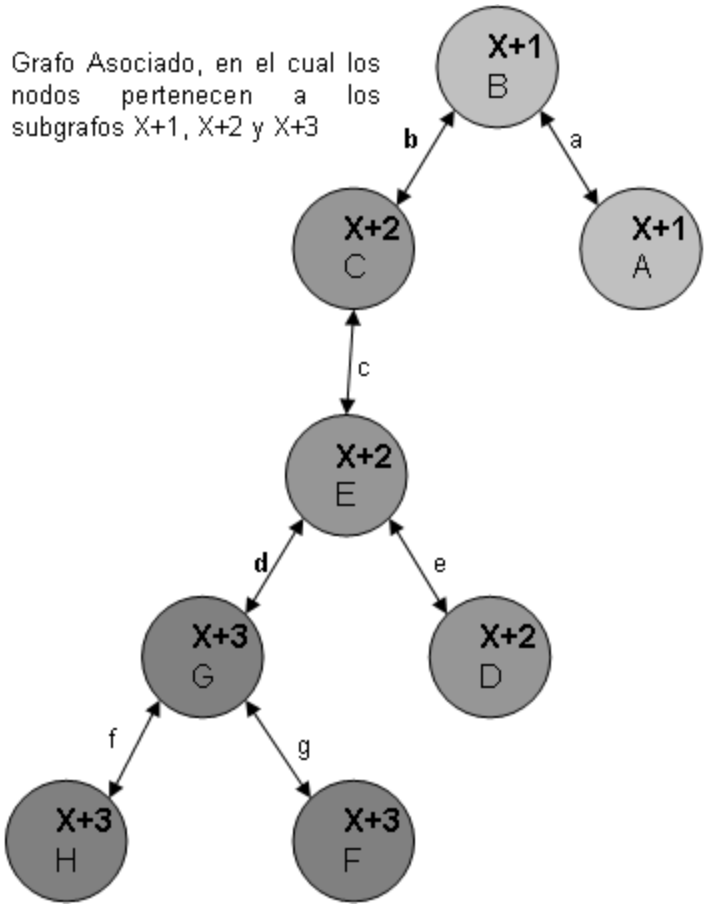


Figura 3.54. Un grafo asociado con 3 subgrafos.

En este grafo se observa que existen 3 subgrafos, llamados X+1, X+2 y X+3.

En la ***Figura 3.55..*** se ilustra como se llega a la figura final deseada:

Figura 3.55. Probar Algoritmo.

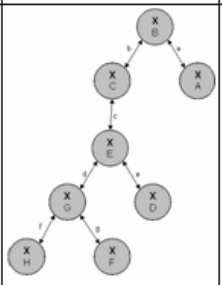
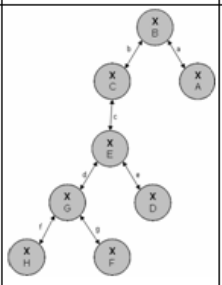
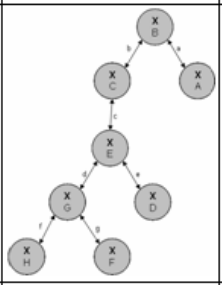
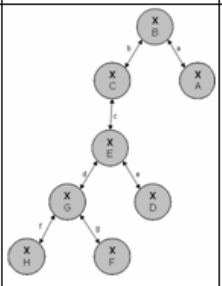
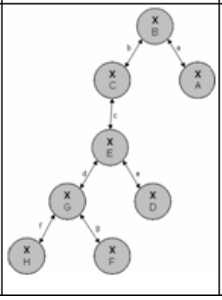
	Paso del Algoritmo	Grafo	Nodo en		Arbol		Nodos		Comentario
			W	W1	Actual	Nuevo	Por Procesar	Procesados	
1	1. Visitar un nodo dado ***A*** del Arbol ***X*** a Segmentar				**X**	X	A, B, C, D, E, F, G, H		En el campo ***Arbol actual*** se guarda el valor del arbol que se esta segmentando, y en el campo ***nodos por procesar*** los nodos que aun no se han procesado.
2	2. Meter en la FIFO W1 el nodo A.			**A**	X	X	A,B, C, D, E, F, G, H		
3	3. Repetir los pasos 4 y 5 hasta que la FIFO W1 este vacía			A	X	X	A,B, C, D, E, F, G, H		
4	4. Retirar el nodo frente de la FIFO W1, Meterlo en la FIFO W e Incrementar la variable Arbol nuevo en 1		**A**		X	**X+1**	A,B, C, D, E, F, G, H		Cuando se crea por primera vez el MST todos los nodos pertenecen al arbol X=1, cada nuevo subarbol creado se incrementa en uno, para hacerlo mas general el nuevo subarbol siempre sera el consecutivo del
5	5. Repetir los pasos 6 y 7 hasta que la FIFO W este vacía.		A		X	X+1	A,B, C, D, E, F, G, H		

Figura 3.55. (Continuación...)

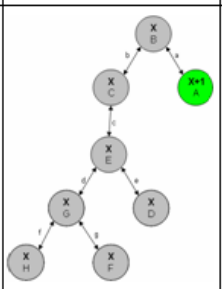
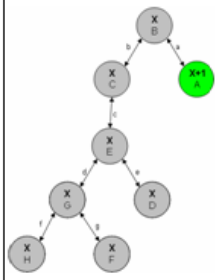
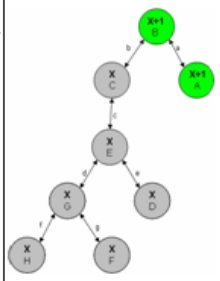
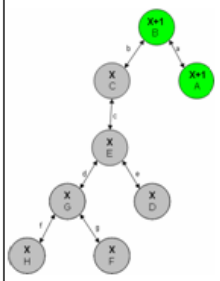
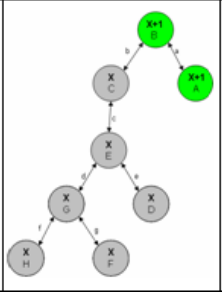
	Paso del Algoritmo	Grafo	Nodo en		Arbol		Nodos		Comentario
			W	W1	Actual	Nuevo	Por Procesar	Procesados	
6	6. Retirar el nodo frente de la FIFO W. Procesar el nodo etiquetándolo con el nuevo valor de subarbol que pertenece.				X	X+1	**B, C, D, E, F, G, H**	**A**	El Color en verde indica que el nodo A, ha sido procesado.
7	7.Meter en la FIFO X todos los nodos adyacentes del nodo retirado en el paso 6 de la FIFO de W que no estén procesados y pertenezcan al mismo subárbol que se esta segmentando, La FIFO X puede ser W o W1 dependiendo de: Si el Valor de la arista hacia el nodo adyacente esta contenido en la lista de las aristas (d,b) Entonces meter en la FIFO W1, en otro caso meter en la FIFO W.		**B**		X	X+1	B, C, D, E, F, G, H	A	La lista de las aristas calculada en la primera parte contiene los elementos ***d*** y ***b,*** El unico con valor de arista a y no pertenece a ningun elemento de la lista. Por lo tanto el nodo B se agrega a la FIFO W
8	6. Retirar el nodo frente de la FIFO W. Procesar el nodo etiquetándolo con el nuevo valor de subarbol que pertenece.				X	X+1	**C, D, E, F, G, H**	**A, B**	Como en el paso Anterior, La FIFO W no esta vacia ejecutar paso 6 y 7 otra vez.
9	7.Meter en la FIFO X todos los nodos adyacentes del nodo retirado en el paso 6 de la FIFO de W que no estén procesados y pertenezcan al mismo subárbol que se esta segmentando, La FIFO X puede ser W o W1 dependiendo de: Si el Valor de la arista hacia el nodo adyacente esta contenido en la lista de las aristas (d,b) Entonces meter en la FIFO W1, en otro caso meter en la FIFO W.			**C**	X	X+1	C, D, E, F, G, H	A, B	Como la arista entre B y C es b, y esta en la lista entonces se pone el la FIFO W1. Por otra parte como la FIFO W esta vacia se ejecuta paso 4 y 5 hasta que W1 este vacia segun paso 3
10	4. Retirar el nodo frente de la FIFO W1, Meterlo en la FIFO W e Incrementar la variable Arbol nuevo en 1		**C**		X	**X+2**	C, D, E, F, G, H	A, B	

Figura 3.55. (Continuación...)

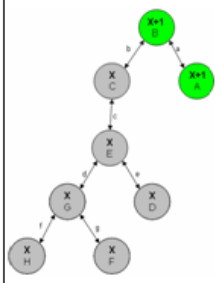
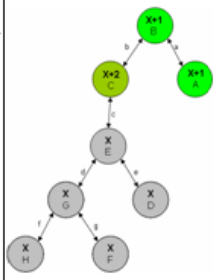
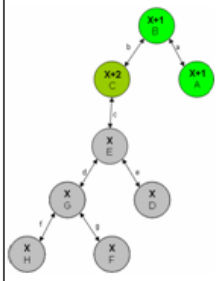
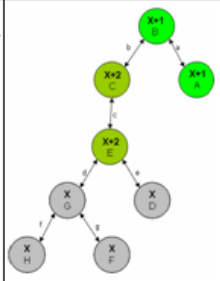
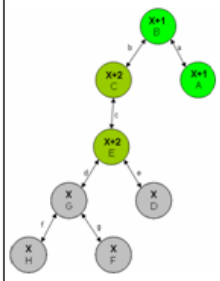
	Paso del Algoritmo	Grafo	Nodo en		Arbol		Nodos		Comentario
			W	W1	Actual	Nuevo	Por Procesar	Procesados	
11	5. Repetir los pasos 6 y 7 hasta que la FIFO W este vacía.		**C**		X	X+2	C, D, E, F, G, H	A, B	
12	6. Retirar el nodo frente de la FIFO W. Procesar el nodo etiquetándolo con el nuevo valor de subarbol que pertenece.				X	X+2	**D, E, F, G, H**	**A, B, C**	
13	7.Meter en la FIFO X todos los nodos adyacentes del nodo retirado en el paso 6 de la FIFO de W que no estén procesados y pertenezcan al mismo subárbol que se esta segmentando, La FIFO X puede ser W o W1 dependiendo de: Si el Valor de la arista hacia el nodo adyacente esta contenido en la lista de las aristas (d,b) Entonces meter en la FIFO W1, en otro caso meter en la FIFO W.		**E**		X	X+2	D, E, F, G, H	A, B, C	Repetir pasos 6 y 7, ya que W no esta vacia
14	6. Retirar el nodo frente de la FIFO W. Procesar el nodo etiquetándolo con el nuevo valor de subarbol que pertenece.				X	X+2	**D, F, G, H**	**A, B, C,E**	
15	7.Meter en la FIFO X todos los nodos adyacentes del nodo retirado en el paso 6 de la FIFO de W que no estén procesados y pertenezcan al mismo subárbol que se esta segmentando, La FIFO X puede ser W o W1 dependiendo de: Si el Valor de la arista hacia el nodo adyacente esta contenido en la lista de las aristas (d,b) Entonces meter en la FIFO W1, en otro caso meter en la FIFO W.		**D**	**G**	X	X+2	D, F, G, H	A, B, C,E	Repetir pasos 6 y 7, ya que W no esta vacia

Figura 3.55. (Continuación...)

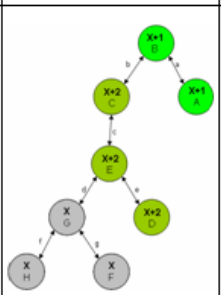
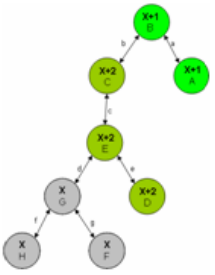
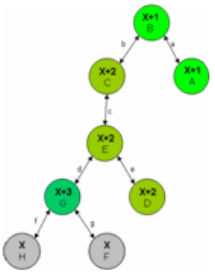
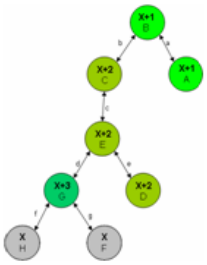
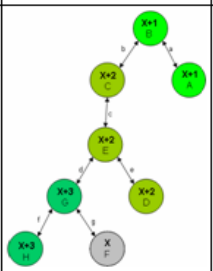
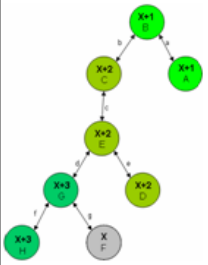
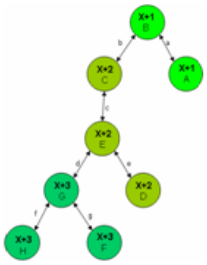
	Paso del Algoritmo	Grafo	Nodo en		Arbol		Nodos		Comentario
			W	W1	Actual	Nuevo	Por Procesar	Procesados	
16	6. Retirar el nodo frente de la FIFO W. Procesar el nodo etiquetándolo con el nuevo valor de subarbol que pertenece.			G	X	X+2	**F, G, H**	**A, B, C,E,D**	
17	7. Meter en la FIFO X todos los nodos adyacentes del nodo retirado en el paso 6 de la FIFO de W que no estén procesados y pertenezcan al mismo subárbol que se esta segmentando, La FIFO X puede ser W o W1 dependiendo de: Si el Valor de la arista hacia el nodo adyacente esta contenido en la lista de las aristas (d, b) Entonces meter en la FIFO W1, en otro caso meter en la FIFO W			G	X	X+2	F, G, H	A, B, C,E,D	Como la FIFO W esta vacia repetir pasos 4 y 5 hasta que la FIFO W1 este vacia
18	4. Retirar el nodo frente de la FIFO W1, Meterlo en la FIFO W e Incrementar la variable Arbol nuevo en 1		**G**		X	**X+3**	F, G, H	A, B, C,E,D	
19	5. Repetir los pasos 6 y 7 hasta que la FIFO W este vacía.		**G**		X	X+3	F, G, H	A, B, C,E,D	
20	6. Retirar el nodo frente de la FIFO W. Procesar el nodo etiquetándolo con el nuevo valor de subarbol que pertenece.				X	X+3	**F, H**	**A, B, C,E,D,G**	

Figura 3.55. (Continuación...)

	Paso del Algoritmo	Grafo	Nodo en		Arbol		Nodos		Comentario
			W	W1	Actual	Nuevo	Por Procesar	Procesados	
21	7. Meter en la FIFO X todos los nodos adyacentes del nodo retirado en el paso 6 de la FIFO de W que no estén procesados y pertenezcan al mismo subárbol que se esta segmentando, La FIFO X puede ser W o W1 dependiendo de: Si el Valor de la arista hacia el nodo adyacente esta contenido en la lista de las aristas (d, b) Entonces meter en la FIFO W1, en otro caso meter en la FIFO W		**H,F**		X	X+3	F, H	A, B, C,E,D,G	Como W no es vacia repetir paso 6 y 7
22	6. Retirar el nodo frente de la FIFO W. Procesar el nodo etiquetándolo con el nuevo valor de subarbol que pertenece.		**F**		X	X+3	**F**	**A, B, C,E,D,G,H**	
23	7. Meter en la FIFO X todos los nodos adyacentes del nodo retirado en el paso 6 de la FIFO de W que no estén procesados y pertenezcan al mismo subárbol que se esta segmentando, La FIFO X puede ser W o W1 dependiendo de: Si el Valor de la arista hacia el nodo adyacente esta contenido en la lista de las aristas (d, b) Entonces meter en la FIFO W1, en otro caso meter en la FIFO W		**F**		X	X+3	F	A, B, C,E,D,G,H	Repetir pasos 6 y 7, ya que W no esta vacia
24	6. Retirar el nodo frente de la FIFO W. Procesar el nodo etiquetándolo con el nuevo valor de subarbol que pertenece.				X	X+3		**A, B, C, E, D, G, H, F**	
25	7. Meter en la FIFO X todos los nodos adyacentes del nodo retirado en el paso 6 de la FIFO de W que no estén procesados y pertenezcan al mismo subárbol que se esta segmentando, La FIFO X puede ser W o W1 dependiendo de: Si el Valor de la arista hacia el nodo adyacente esta contenido en la lista de las aristas (d, b) Entonces meter en la FIFO W1, en otro caso meter en la FIFO W				X	X+3		A, B, C, E, D, G, H, F	Como W y W1 es vacia sigue paso 8 FIN . En la proxima segmentacion o fusion en que se necesite otro atiqueta de nodos para formar un Arbol nuevo se utilizara X+4 y asi sucesivamente.
26	8. Fin								

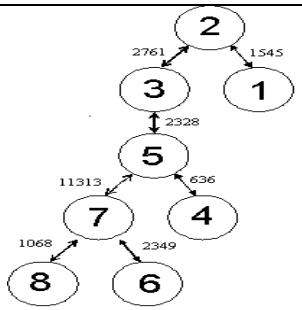
3. IV.9. Fusionar regiones.

3. IV.9.1. Fusionar 2 regiones marcadas.

Veremos la manera en que se fusionan 2 subárboles.

Una condición necesaria para que dos subárboles puedan ser fusionados es que exista una arista entre ellos, esto para poder seguir conservando el MST original.

Veamos algunos ejemplos de regiones que se pueden fusionar en la siguiente ***Tabla 7***:

Grafo Representativo MST	**Imagen de Vertientes**	**Nodos o regiones que se pueden fusionar**
		Aquí se puede fusionar la región 2 con 1, región 2 con 3, región 3 con 5, región 5 con 4, región 5 con 7, región 7 con 6, y región 7 con 8. Cualquier otra con fusión no es posible ya que se rompe el MST.

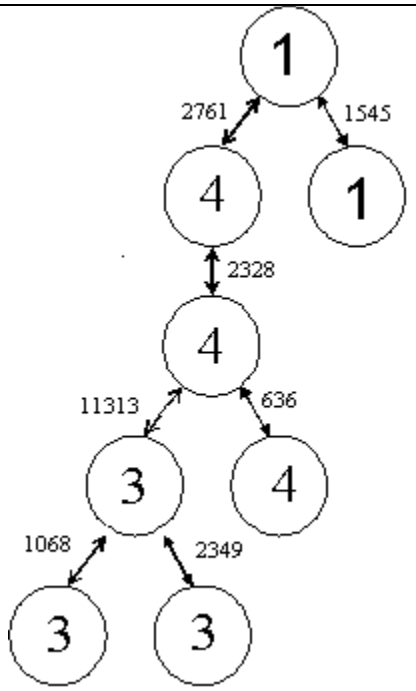
		En este caso si se pueden fusionar la regiones 1 y 4. De igual forma la región 3 y 4, No así las regiones 1 y 3

Tabla 7 Regiones que se pueden fusionar

La forma de indicar las regiones a fusionar es seleccionar cualquier punto de cada una de las regiones y realizar su fusión. En nuestro caso el método para seleccionar una región 1 es usando el ratón con un clic y soltar el clic en la segunda región.

Veamos un ejemplo: Nuestra imagen original con 3 particiones seria como el segundo caso de la de la ***Tabla 7***, se marcan para fusionar la region1 y la region4, la imagen quedaría:

Figura 3.56. Imagen Original con todas sus vertientes

Figura 3.57. Imagen con 3 Zonas

Figura 3.58. Imagen con dos zonas o regiones, region5 (que resulto de fusionar la región 1 y 4) y región 3. Ver tabla7.

3. IV.9.2. Algoritmo Para fusionar:

Para fusionar 2 regiones o pasar a un nivel menos fino de granularidad, necesitaremos una función que no indique si 2 regiones están conectadas.

a) ¿Adyacentes?.

Utilizaremos la función llamada *pruebaVecino* para determinar cuando 2 subgrafos están conectados.

Subárbol y subárbol1 son las etiquetas que determinar a que subgrafo pertenece cada nodo, si la función *pruebaVecino* regresa verdadero están conectados, si regresa falso no existe conexión entre ambos subgrafos.

Podría agregarse la línea “if subárbol=subarbol1 return false”, al inicio de la función, para el caso que se tomen 2 nodos pertenecientes al mismo subgrafo y así evitar el proceso.

A continuación se muestra el algoritmo de la función ***pruebaVecino***.

1. Visitar un nodo dado A de cualquiera de los subgrafos a fusionar.
2. Meter en la cola el nodo A y marcarlo como procesado.
3. Repetir los pasos 4 y 5 hasta que la cola este vacía.
4. Retirar el nodo frente de la cola (W), visitar W.
5. Revisar en cada uno de los nodos adyacentes de W que no estén procesados y que pertenezcan a cualquiera de los dos subgrafos a fusionar, ***IF*** subárbol=subarbol1 ***return true ELSE*** Meterlo en la cola y marcarlo como procesados.
6. ***return False***

b) Fusiona MST.

Es un recorrido de anchura empezando desde un nodo cualquiera perteneciente a uno de los 2 subgrafos(X, Y) a fusionar, se etiqueta cada nodo con un nuevo valor que representa el nuevo subgrafo resultante de la fusión de los dos grafos a fusionar.

Si se pueden fusionar aplicar los pasos siguientes (Determinar por medio de la función *pruebaVecino* del punto anterior)

Función **Fusiona**.

1. Visitar un nodo dado A de cualquiera de los subgrafos a fusionar.
2. Meter en la cola el nodo A y marcarlo como procesado.
3. Repetir los pasos 4 y 5 hasta que la cola este vacía.
4. Retirar el nodo frente de la cola (W), visitar W. procesar el nodo etiquetándolo con el nuevo valor de subárbol que pertenece(es el subgrafo resultante de la fusión de subgrafo y subgrafo1).
5. Meter en la cola todos los nodos adyacentes de W que no estén procesados y que pertenezcan a cualquiera de los dos subgrafos a fusionar, y marcarlos como procesados.

En resumen este algoritmo genera otro subgrafo nuevo ***Z***, que contiene todos los nodos de los grafos a fusionar **X, Y.**

4. RESULTADOS Y DISCUSIÓN

4. RESULTADOS Y DISCUSION.

4. I. Introducción.

4. I.1. Adaptación de Imagen.

Para obtener mejores resultados, se debe procesar la imagen original de tal forma que al final nos proporcione las particiones más adecuadas. Se hace la adaptación de tamaño o talla **n** a una imagen de entrada según la siguiente función.

Se ejecuta una apertura por reconstrucción, Después una cerradura por reconstrucción, esto para eliminar pequeñas regiones. Después calcula todos los mínimos y máximos y los guarda en img3 como mínimos. Calcula el gradiente de la imagen de entrada u original y finalmente le impone mininos guardados en img3. La imagen adaptada queda finalmente en img2.

roy22.***im_Ngreyopenbuild***_(img1,img1,m_nEditTalla);
roy22.***im_Ngreyclosebuild***_(img1,img1,m_nEditTalla);

roy22***.im_minima***_(img1,img2);
roy22.***im_maxima_(***img1,img3);
roy22.***im_greysup_(***img2,img3,img3);

roy22.***im_Ngreyerode***_(ImgEntra,img1,1);
roy22.***im_Ngreydilate***_(ImgEntra,img2,1);
roy22.***im_greysub***_(img2,img1,img2);

roy22.im***_MarkersHomotopyChange***_(img2,img3);

El número de particiones en el nivel más fino varía dependiendo de la talla (Tamaño del elemento estructurante) que se utilice. A Menor talla mayor el número de particiones de la imagen en el nivel más fino.

4. I.2. Tiempos de respuesta.

La graficas de las ***Figura 4.1.*** y ***Figura 4.2.*** muestran el tiempo que utilizó la computadora para generar el grafo MST y el tiempo para segmentar una imagen después de ya estar creado el grafo respectivamente. Para el caso de la ***Figura 4.1.*** se grafica contra el numero de particiones que contiene la imagen en su nivel más fino, en la ***Figura 4.2.*** contra **n** particiones o regiones en que se solicito segmentar una región.

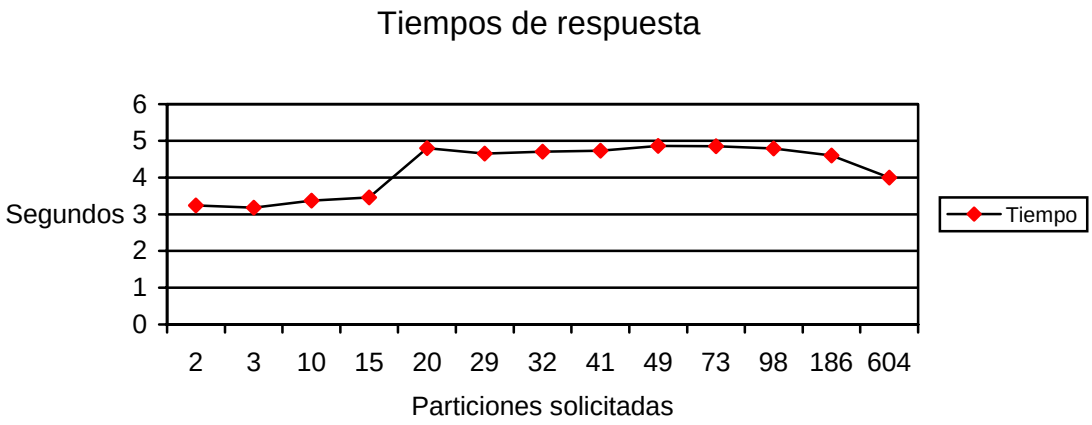


Figura 4.1. Tiempo en segundos utilizado por la computadora para generar un grafo con un total de n particiones.

El tiempo de respuesta para generar un grafo MST oscila entre 20 y 50 segundos dependiendo de número de particiones de la imagen en su nivel más fino.

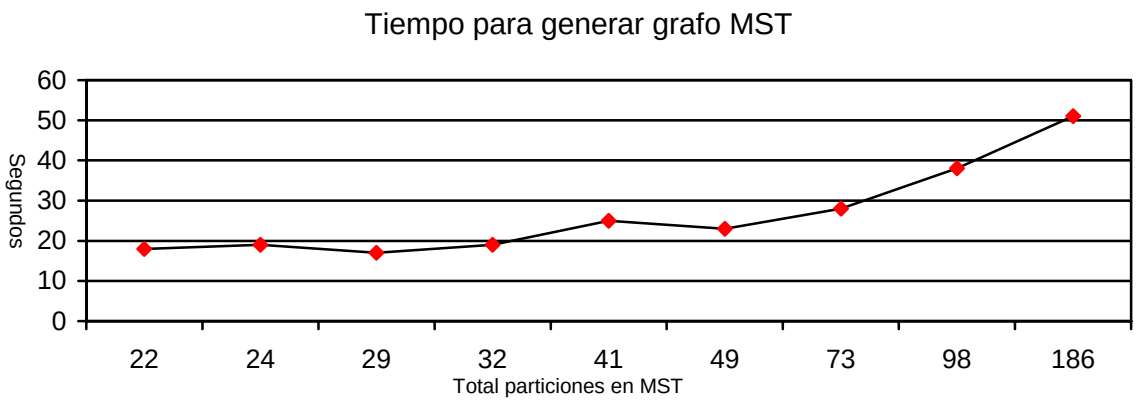


Figura 4.2. Tiempo en segundos que utiliza la computadora para particionar una región en n regiones, Utilizando el grafo MST.

El tiempo de respuesta para segmentar una región o partición en ***n*** particiones oscila entre 3 y 5 segundos dependiendo del número ***n*** de particiones solicitadas, esto utilizando el MST una vez ya creado.

Las diversas pruebas se realizaron sobre la imagen de la ***Figura 4.3.*** Se utilizó una computadora con Pentium II a 350 Mhz, en la actualidad existen computadoras con velocidades de procesador mayor a 3 Ghz, por lo que los tiempos de respuesta serian mucho mejor.

4. II. Imágenes de Rostros: Lena.

En este ejemplo se hace una adaptación de imagen de tamaño 5, y se piden 49 particiones de la imagen mostrada. ***Figura 4.3.***

Figura 4.3. Lena

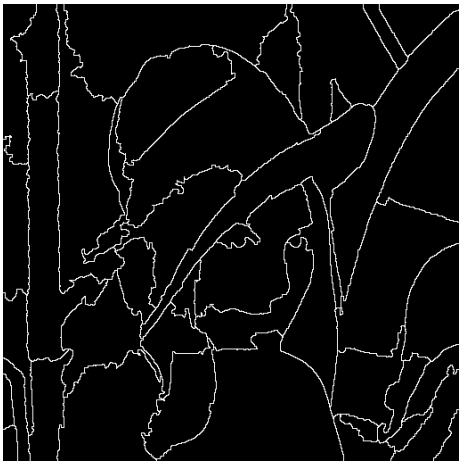


Figura 4.4. Hay 63 Particiones Iniciales de la Figura 4.3.

Se empiezan a fusionar y dividir las particiones de la ***Figura 4.3***. y obtenemos la segmentación que se muestra en la ***Figura 4.4.***, a esta se le hace un mask a la imagen original y se obtiene la imagen de la ***Figura 4.5.***

Figura 4.5. Particiones obtenidas al fusionar y dividir algunas regiones de la Figura 4.4.

Figura 4.6. mask de Figura 4.3. y Figura 4.5.

4. III. Imágenes Partículas: Gel de electroforesis

Se le aplica a la imagen original de la ***Figura 4.7.*** una adaptación de imagen de tamaño 2, se piden 143 regiones, y se obtiene la segmentación de la ***Figura 4.8.***

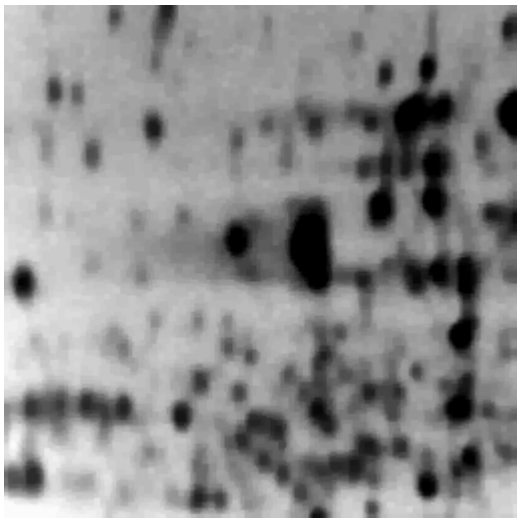


Figura 4.7. Gel de electroforesis.

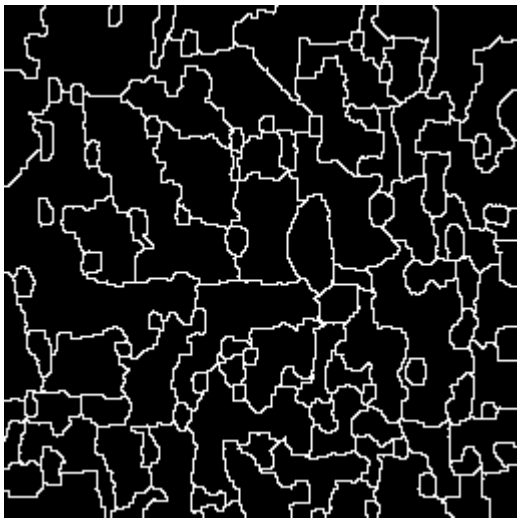


Figura 4.8. Segmentación de Figura 4.7., en 143 regiones

Al Fusionar regiones de la ***Figura 4.8.*** y aplicar a la segmentación un mask con ***Figura 4.7.*** se obtiene la ***Figura 4.9.***

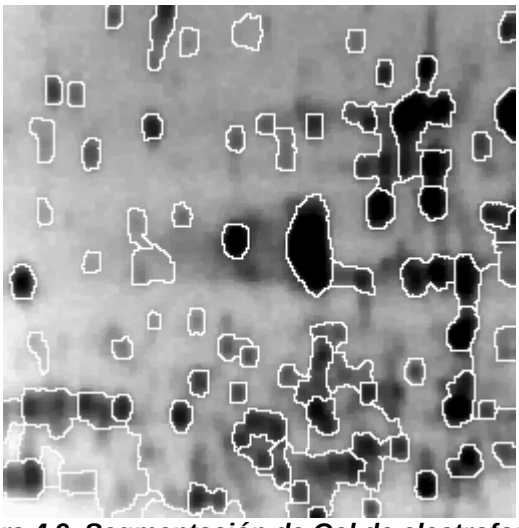


Figura 4.9. Segmentación de Gel de electroforesis

4. IV. Imágenes de resonancia magnética: Imágenes de cerebro

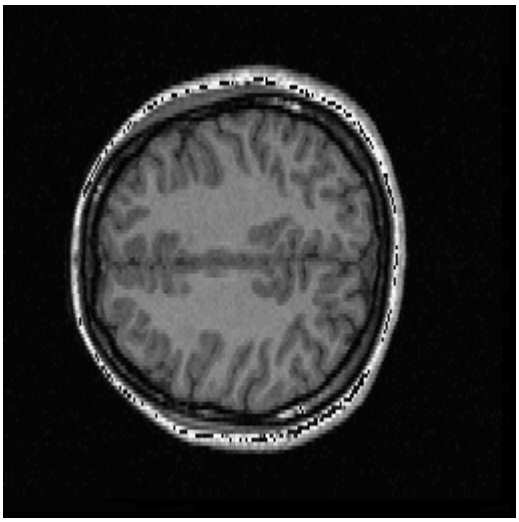


Figura 4.10. Imagen Original Cerebro

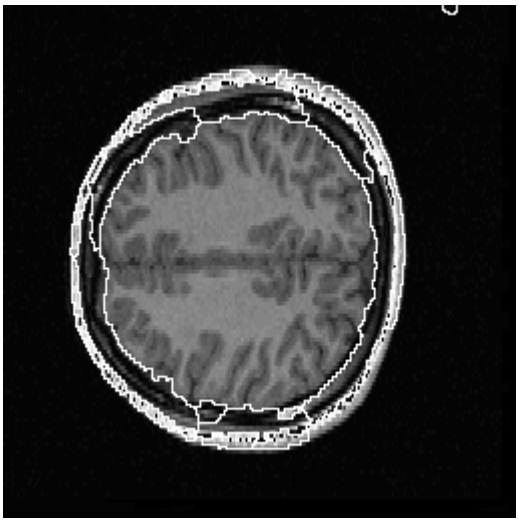


Figura 4.11. Formación de una sola Partición para la masa encefálica

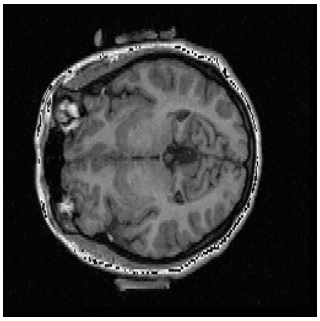


Figura 4.12. Imagen Original

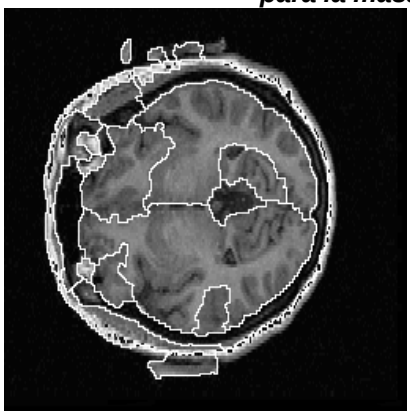


Figura 4.13. 22 particiones de Figura 4.12.

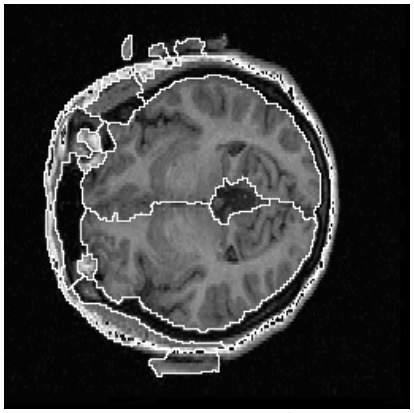


Figura 4.14. Fusión de regiones, separación de hemisferios

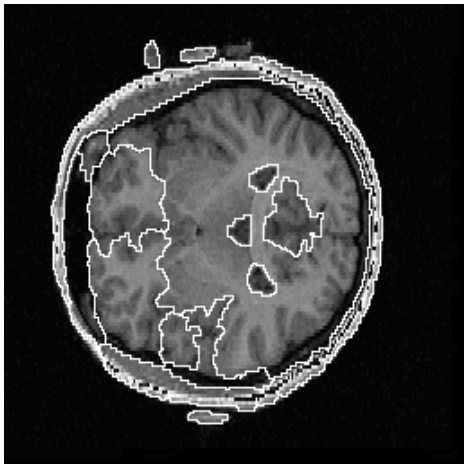


Figura 4.15. Hay 16 particiones

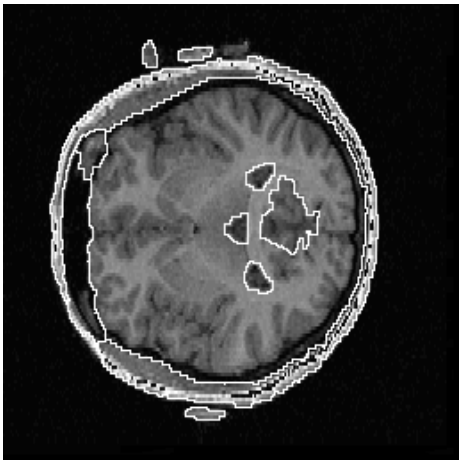


Figura 4.16. Fusión de regiones en masa encefálica

4. V. Imágenes de huesos.

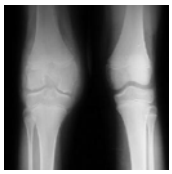
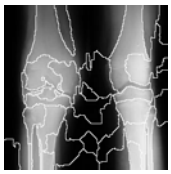


Imagen original


66 regiones

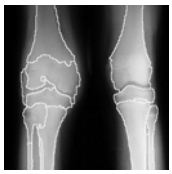

Fusionar para separar huesos de las piernas

Figura 4.17. Imágenes de huesos, fusión y segmentación.

4. VI. Imagen de Hoja de jitomate.

Segmentación de la enfermedad Phytophthora infestans, se solicita talla 3 en la adaptación de la imagen, el numero de regiones de la imagen en su nivel más fino es de 141, se solicitan 3 regiones hasta tener todas la particiones de la región de la hoja a revisar, se fusionan algunas regiones y se obtiene el resultado mostrado.


Imagen Original


El pico de la hoja y a la izquierda muestra la partición más afectada.

Figura 4.18. Enfermedad Phytophthora infestans en la hoja del jitomate

Conclusiones

La imagen debe de ser procesada (aplicación de filtros, aperturas cerraduras, etc.,) antes de iniciar el proceso de segmentación, para obtener mejores resultados. La función de adaptación ideal es aquella que nos genere las particiones exactas de los objetos de la imagen en su nivel más fino, para que por medio del algoritmo se puedan fusionar o dividir en objetos que contengan otros objetos. Seria muy difícil que el software decidiera que particiones deberían de estar agrupadas para formar un objeto, que nivel de granularidad es la deseada, que partición se desea tener mayor detalle, etc. por lo que es necesario que una persona interactué con el sistema y empiece a fusionar y dividir regiones para dejar solamente los objetos o particiones deseadas, dejar cada una de las particiones de la imagen en el nivel de resolución deseado. Con el algoritmo propuesto podemos hacer lo anterior. Por otra parte el algoritmo permite etiquetar el ***MST*** con un parámetro (***Área, Volumen o Altura***). En las pruebas realizadas sobre las imágenes se utilizó el ***área*** con resultados muy satisfactorios, se podría utilizar fácilmente cualquiera de los otros 2 parámetros en ciertas imágenes en las que no se obtengan resultados de segmentación adecuados.

Con la segmentación por medio de LDA tradicional, se puede producir una sobre segmentación de la imagen, la solución tradicional es la utilización de marcadores sobre la imagen [B. Marcotegui, F. Zanoguera, 2002], pero muchas veces este problema se pasa a las regiones marcadas. Con el algoritmo propuesto se piden **n** regiones iniciales en que se desea particionar la imagen y utilizando el MST genera las **n** particiones más significativas de la imagen, dejando al usuario la elección de particionar o fusionar más regiones.

Una ventaja de este método propuesto es que permite re segmentar aún más otra región ya segmentada, así como fusionar regiones que de acuerdo a nuestro criterio debe de estar como una sola región.

Otras ventajas de utilizar el algoritmo propuesto junto con el MST son:

Todas las operaciones de división y fusión de imágenes y sus regiones son hechas sobre el grafo MST y la respuesta de segmentación es inmediata. Solamente se efectúa una vez el cálculo de las líneas divisoras de aguas en el cual durante el proceso de inundación se va generando el grafo MST. Podemos generar particiones específicas sin tener que recalcular y guardar otras particiones con otros niveles más o menos finos.

Algunas áreas de aplicación son digitalización de huellas, Imágenes satelitales, medicina, milicia, imágenes óseas, imágenes neurológicas,

BIBLIOGRAFÍA.

B. Marcotegui and F. Zanoguera. 2002. “IMAGE EDITING TOOLS BASED ON MULTI-SCALE SEGMENTATION”, H. Talbot, R. Beare (Eds): Processings of ISMM2002, Redistribution rights reserved CSIRO Publishing. ISBN 0 643 06804 X.

Crespo, J., *et al* 1993. “Morphological Connected Filters and Intra-Region Smoothing for Image Segmentation”, Ph. D. Thesis, Georgia Institute of Technology, USA, 1993.

Crespo, J., Schafer *et al* 1994. R.W. In Mathematical Morphology and Its Applications to Image Processing, J. Serra and P. Soille, Eds., Kluwer Academic Publishers, pp. 85-92, (1994).

Crespo, J., Serra, J., Schafer et al 1995. R.W., “Theoretical aspects of morphological filters by reconstruction”, Signal Process., 47, 201-225 (1995).

Heijmans, et al 1997. H.J.A.M., “Connected morphological operators for binary images” Techical Report CWI No. PNA-R9708, April (1997).

Mendiola-Santibañez, J. D., Terol-Villalobos, I.R., et al 2001a. “Mapeos de Contraste Morfológicos sobre Particiones usando la Noción de Zona Plana” , VI Taller Iberoamericano de Reconocimiento de Patrones (TIARP’2001), Editado por el CIC en la serie Sistemas Computacionales, páginas 51-62, México (2001a).

Mendiola-Santibañez, J. D., Terol-Villalobos, I.R., et al 2001b. “Propagación de marcadores a través del gradiente morfológico para la segmentación de IRM ”, VI Taller Iberoamericano de Reconocimiento de Patrones (TIARP’2001), Editado por el CIC en la serie Sistemas Computacionales, páginas 85-96, México (2001b).

Mendiola-Santibañez, J. D., Terol-Villalobos, I.R., et al 2002. “Morphological contrast enhancement using connected transformations”, aceptado para aparecer en SPIE Image Processing: Algorithms and Systems, 12 páginas, San José California USA (2002).

Meyer, F., et al 1998a. “From connected operators to levelings”, In Mathematical Morphology and Its Applications to Image and Signal Processing, H. Heijmans and J. Roerdink Eds., Kluwer, pp. 191-198, (1998).

Meyer, F., et al 1998b. “Levelings”, In Mathematical Morphology and Its Applications to Image and Signal Processing, H. Heijmans and J. Roerdink Eds., Kluwer, pp. 199-206, (1998).

Meyer, F., Beucher S., *et al* 1990. “Morphological Segmentation”, *J. Visual Comm. Image Represent*., 1, 21-46, 1990.

Meyer, F., Maragos, P., 2000. ”Non-linear Scale-Space Representation with Morphological Levelings”, Journal of Visual Comm. Image Represent.,11, 245-265, (2000).

Ronse, C. et al 1998. “Set-Theoretical Algebraic Approaches to Connectivity in Continuous or Digital Spaces”, J. of Mathematical Imaging and Vision, 8, 41-58, (1998).

Luc Vicent at al 1997. “Current Topics in Applied Morphological Image Analysis”, To appear in Current Trends in Stochastic Geometry and its Applications, W.S. Kendall, 0.E. Barndorff-Nielsen, M.C. Van Lieshout, editors, Chapman Hall, London, (1997).

Salembier, P., Serra, J. et al 1995. “Flat zones filtering, connected operators and filters by reconstruction” IEEE Trans. Image Processing, 4, 1153-1160, (1995).

Salembier, Ph. and Oliveras, A. et al 1996. “Practical extensions of connected operators ” In Mathematical Morphology and Its Applications to Image and Signal Processing, P. Maragos, R.W. Schafer, and M.K. Butt Eds., Kluwer, pp. 97-110, (1996).

Salembier, Ph., Garrido, L., et al 2000. “Connected operators based on region-tree pruning”, In Mathematical Morphology and Its Applications to Image and Signal Processing, Goutsias, J., Vincent, L., Bloomberg, D.S. Eds., Kluwer, pp. 167-178, (2000).

Serra, J. et al 1988. Image Analysis and Mathematical Morphology, J. Serra, Ed., Vol. II, Academic, New York (1988).

Serra, J., et al 2000. “Connections for sets and functions”, Fundamenta Informaticae, 41, 147-186, (2000).

Terol-Villalobos, I., R., 1995. “Morphological slope filters”, Proc. SPIE Intelligent Robots and Computer Vision XIV: Algorithms, Techniques, Active Vision, and Materials Handling, 2588, 712-722, (1995).

Terol-Villalobos, I. R., 1996a. “Nonincreasing filtres using morphological gradient criteria”, Optical Engineering, 35, 3172-3182 , (1996a).

Terol-Villalobos, I., R., 1996b. “Morphological nonincreasing filtres for image segmentation and enhancement”, Proc. SPIE Intelligent Robots and Computer Vision XIV: Algorithms, Techniques, Active Vision, and Materials Handling, 2904, 557-566, (1996b).

Terol-Villalobos, I. R. and Cruz-Mandujano, et al 1998a. “Contrast enhancement and image segmentation using a class of morphological nonincreasing filters”, J.A. Journal of Electronic Imaging, 7, 641-654, (1998a).

Terol-Villalobos, I. R., et al 1998b. “Toggle mappings and some related transfortions: A study of contrast enhancement”, in Mathematical Morphology and Its Applications to Image and Signal Processing, H.J.A.M. Heijmans and J.B.T.M. Roerdink , Eds., Kluwer Academic Publishers, The Netherlands, pp. 11-18, (1998b).

Terol-Villalobos, I.R., et al 2001. “Morphological Image Enhancement and Segmentation”, en Advances in Imaging and Electron Physics, Editor Peter W. Hawkes, Vol. 118, Capítulo IV, páginas 207-273, Academic Press, septiembre, (2001).