Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional

Departamento de Ingeniería Eléctrica

Sección Computación

Reconstrucción de Volúmenes a partir de sus Mapas de Contornos

Tesis que presenta

Abigail Martínez Rivas

para obtener el Grado de

Maestro en Ciencias

en la Especialidad de

Ingeniería Eléctrica

Director de la Tesis

Dr. Luis Gerardo de la Fraga

México, D.F. **Noviembre 2005**

Resumen

A pesar del desarrollo tecnológico, la mayor parte de los datos de elevación de la tierra se encuentra en forma de cubiertas de contornos impresos en mapas. Éstos tienen la ventaja de que han sido desarrollados usando la comprensión humana de las superficies observadas, sin embargo tienen la desventaja de que no pueden ser convertidos a un formato digital útil.

En esta tesis desarrollamos la visualización de volúmenes reconstruidos a partir de mapas de contornos. Específicamente, se reconstruyeron terrenos (montañas, cerros, valles, etc.) a partir de sus mapas topográfico. La solución al problema se dividió en dos etapas: (1) procesar la imagen del mapa topográfico para obtener los puntos que definen a las líneas de contorno y (2) realizar una interpolación entre los contornos y generar la triangulación que defina la superficie del terreno.

En la primer etapa se aplicaron técnicas de procesamiento de imagen tales como adelgazamiento de regiones, seguimiento de puntos y mapas de distancias. En la segunda etapa se calcularon las construcciones geométricas del diagrama de Voronoi y la triangulación de Delaunay para obtener los puntos intermedios entre los contornos, es decir, los puntos del esqueleto. La triangulación se realiza sobre el conjunto completo de los puntos.

Los valores de elevación para los puntos que conforman los contornos están incluídos en los mapas topográficos. Los valores de elevación para los puntos del esqueleto generados se calculan a partir de las elevaciones de los contornos. Se realizó una interfaz gráfica para la visualización de las reconstrucciones realizadas empleando las herramientas de OpenGL y Qt.

Abstract

In spite of technological development, much of the world's elevation data is still in the form of contour lines printed in maps. The advantage of this maps is that they have been created using the human understanding of the observed surfaces, nevertheless their disadvantage is that they can't be converted to an useful digital format.

This thesis consists in the visualization of volumes reconstructed from its contours maps. Specifically, terrains (like mountains, hills, valleys, etc.) were reconstructed from their topographic map. The solution of this problem was divided in two stages: (1) to process the image of the topographic map in order to get the point defining the contour lines and (2) to compute an interpolation among the contours and generate the triangulation that define the terrain surface.

In the first stage image processing techniques were applied, such as thinning of regions, tracking of points and distance maps. In the second stage we compute the geometrical constructions of Voronoi diagram and Delaunay triangulation in order to get the medial points between each two contours, that is, the skeleton points. The triangulation is calculated with the complete set of points.

The elevation values of the points that define the contours are included in the topographic maps. The elevation values of the skeleton points are calculated from the contours elevations. We developed a graphical user interface using the OpenGL and Qt tools, to visualize the computed reconstructions.

Agradecimientos

A Dios,
porque me ha sostenido en Su seno
cada día de mi vida,
me ha demostrado Su amor en cada situación,
aunque grata o dificultosa,
y me ha dado todo cuanto soy y tengo.

A mis padres Adalberto y Maria de los Ángeles,
por su instrucción,
porque me han brindado todo su amor, su apoyo,
la oportunidad de alcanzar mis metas;
porque creen en mi.

A mis hermanos Laura, David, Priscila y Merari,
porque me dan su aliento siempre que lo necesito,
porque me han dejado saber
lo que realmente es importante en esta vida.

A mis amigos Claudia, Grettel, Juan Manuel y Francisco,
por todo el apoyo tanto moral como profesional
que me han brindado a lo largo de mi carrera,
porque sin ellos, muchos de mis logros
se hubieran visto afectados.

Al Dr. Luis Gerardo de la Fraga,
por su basta asesoría
en la realización de este trabajo,
por su apoyo moral, por su paciencia
y por todas sus contribuciones.

A Sofía Reza,
por la ayuda incondicional que me proporcionó
en mi estancia en el CINVESTAV.

Al CONACyT,
por haberme proporcionado
los recursos económicos necesarios
para la culminación satisfactoria de este trabajo.

Al CINVESTAV,
por haber prestado los recursos materiales
útiles en el desarrollo de esta tesis.

A todas aquellas personas que me han enseñado algo
y que han formado parte de mi desarrollo como persona y profesional.

¿Qué es el hombre, para que lo engrandezcas,
Y para que pongas en él Tu atención,
Y lo visites todas las mañanas,
Y todos los momentos lo pruebes?
Job 7:17-18

Índice general

Indice de figuras xii

Indice de cuadros xvi

1. Introducción 1
- 1.1. Antecedentes . . . 1
- 1.2. Sistema a desarrollar . . . 2
  - 1.2.1. Obtención del mapa de contornos . . . 3
  - 1.2.2. Interpolación de los contornos . . . 4
- 1.3. Organización de la tesis . . . 5

2. Marco Teórico 7
- 2.1. Modelos digitales de elevaciones . . . 7
  - 2.1.1. Contornos . . . 8
- 2.2. Vecinos en imágenes digitales . . . 9
  - 2.2.1. Imagen digital . . . 10
  - 2.2.2. Vecindad entre pixeles . . . 10
  - 2.2.3. Conexidad . . . 11
- 2.3. Mapa de distancias . . . 12
  - 2.3.1. Transformada de la distancia . . . 12
  - 2.3.2. Algoritmo . . . 13
- 2.4. Eje medio . . . 15
- 2.5. Reconstrucción poligonal . . . 16
  - 2.5.1. Condición de muestreo . . . 17
- 2.6. Diagrama de Voronoi y triangulación de Delaunay . . . 18
  - 2.6.1. Diagrama de Voronoi . . . 18
  - 2.6.2. Triangulación de Delaunay . . . 19

3. Procesamiento Digital de Imagen 21
- 3.1. Extracción de las líneas de contorno . . . 22
  - 3.1.1. Conversión de la imagen RGB a imagen binaria . . . 23
  - 3.1.2. Adelgazamiento de las líneas de contorno . . . 25
  - 3.1.3. Extracción de los pixeles por línea de contorno . . . 26
- 3.2. Muestreo . . . 30

3.2.1. Cálculo del eje medio 30
3.2.2. Cálculo del TCL 34
3.2.3. Cálculo del conjunto de muestreo 36

4. Interpolación de Contornos **39**
4.1. Corteza y esqueleto 40
4.1.1. Prueba *DentroDelCírculo* 40
4.1.2. Cálculo del diagrama de Voronoi y la triangulación de Delaunay 42
4.1.3. Cálculo del esqueleto 45
4.1.4. Reducción del número de puntos del esqueleto 47
4.1.5. Organización en grafos de los puntos del esqueleto 49
4.2. Cálculo de las elevaciones 54
4.2.1. Caso 1: Puntos entre dos contornos diferentes 56
4.2.2. Caso 2: Puntos en una cima 57
4.2.3. Caso 3: Puntos rodeados por secciones de contornos reentrantes 58

5. Interfaz Gráfica **61**
5.1. Entorno de programación 62
5.1.1. OpenGL 62
5.1.2. Qt 63
5.2. Estructuras de datos 63
5.2.1. Corteza 63
5.2.2. Esqueleto 64
5.2.3. Triangulación 66
5.3. Visualización del terreno 67
5.3.1. Selección de una curva de nivel 68
5.3.2. Asignación del color a cada vértice 69
5.4. Descripción de la interfaz gráfica 72

6. Conclusiones y Trabajo Futuro **77**
6.1. Conclusiones 77
6.2. Trabajo futuro 80

A. Algoritmo para la extracción de grafos del esqueleto **81**

B. Ejemplos de Aplicación **87**
B.1. Elipses 87
B.2. Cerros 89
B.3. Malinche 91

C. Densidad de Muestreo **95**
C.1. Valor de $r = 0.415$ 95
C.2. Valor de $r = 0.6$ 97
C.3. Valor de $r = 0.8$ 98

Índice de figuras

1.1. Imagen digitalizada del dibujo de las líneas de contorno de un cerro tomado del mapa topográfico E14B47 del INEGI. 4
1.2. Diagrama general del sistema 5

2.1. Una imagen física y su correspondiente imagen digital. 10
2.2. Ejemplo de una transformada de la distancia. 12
2.3. Máscaras para el ATD Chamfer 3-4 y 5-7-11. 13
2.4. Cálculo de la transformada de la distancia con el Chamfer 3-4. 15
2.5. Eje medio de una curva. 16
2.6. (a) Curvas que definen el dibujo de un ganzo. (b) Conjunto de puntos de muestra de las curvas. (c) Reconstrucción poligonal de las curvas. . 17
2.7. Diagrama de Voronoi y triangulación de Delaunay. 19

3.1. Diagrama general para el procesamiento de imagen. 21
3.2. (a) Una porción del mapa topográfico E14B47 del INEGI y (b) El dibujo de sus líneas de contorno. 22
3.3. Histograma en tonos de gris de una imagen con objetos obscuros sobre fondo claro. 24
3.4. Histograma en tonos de gris de la imagen de la figura 3.2(b). 24
3.5. (a) Imagen binaria del terreno de la figura 3.2, (b) Acercamiento a uno de los cerros. 25
3.6. (a) Líneas de contorno adelgazadas del terreno de la figura 3.2, (b) Acercamiento a uno de los cerros. 26
3.7. (a) Matriz que representa el orden que debe seguirse en la verificación del número de caminos que pueden recorrerse a partir de un pixel. (b) Conjunto de pixel para el que se debe hallar la secuencia que llevan. (c) Secuencia de los pixeles. Los pixeles grises representan el inicio de la trayectoria y el inicio de una ramificación, respectivamente. 28
3.8. Imagen binaria de las líneas de contorno adelgazadas y su mapa de distancias 31
3.9. Máscara usada para la detección de máximos en un mapa de distancias. 32
3.10. Imágenes del eje medio extraídas del mapa de distancias de la figura 3.8, usando diferentes umbrales. 33
3.11. Ejemplo numérico del mapa de distancias de la imagen de un rectángulo. 34

3.12. Mapa de distancias numérico del eje medio del rectángulo de la figura 3.11. La región resaltada representa la región de los pixeles que conforman el rectángulo. 35
3.13. Imagen del mapa de distancias del eje medio de la figura 3.10(f). . . . 35
3.14. Imagen del conjunto de muestreo las líneas de contorno de la figura 3.8(a), y un acercamiento. 37

4.1. Prueba *DentroDelCírculo*. 40
4.2. Gráfica de los vértices de Delaunay (+) y de Voronoi (×) del *conjunto E*. 45
4.3. Las líneas continuas representan la gráfica de las aristas de Delaunay y las líneas punteadas la gráfica de las aristas de Voronoi del *conjunto E*. 46
4.4. Gráfica de las aristas de la corteza y del esqueleto del *conjunto E*. . . 47
4.5. Gráfica de las aristas de la corteza y del esqueleto del *conjunto E* después de haber ejecutado la reducción de los puntos del esqueleto. . 48
4.6. (a) Gráfica de un esqueleto con dos curvas. La numeración representa la secuencia inicial (desordenada) de los vértices en la lista. (b) Lista de vértices y aristas del esqueleto en (a). 50
4.7. Las líneas gruesas representan la gráfica de la corteza y las delgadas la gráfica del esqueleto para las curvas de nivel de los cerros del mapa en la Fig. 3.2. 50
4.8. (a) Gráfica del esqueleto con los vértices ordenados de la figura 4.6. (b) Lista ordenada de los vértices del esqueleto en (a). 53
4.9. Representación en grafos del esqueleto de la figura 4.8(a). 54
4.10. Diferentes tipos de puntos en el esqueleto. Detalles en el texto. 56
4.11. Puntos del esqueleto que se encuentran entre dos contornos de corteza. El punto dentro de un círculo es un punto del esqueleto y los otros puntos son sus vecinos, los cuales pertenecen a diferentes contornos de la corteza. 57
4.12. Puntos del esqueleto que se encuentran en una cima. El punto dentro de un círculo es un punto del esqueleto y los demás puntos son sus vecinos en el contorno de corteza que representa la cima de la montaña. 58
4.13. Puntos del esqueleto que se encuentran rodeados por una sección de un contorno reentrante. 59

5.1. En rojo se muestra la corteza, en azul el esqueleto y en gris la triangulación del mapa de contornos de la figura 3.2. 67
5.2. Cajas que delimitan las curvas de nivel de una superficie creada a partir de elipses. 68
5.3. (a) Visualización de las curvas de nivel de la figura 5.1 después de haberles asignado valor de elevación y color. (b) Visualización del esqueleto con los valores de elevación calculados. 69

5.4. Ejemplos de triángulos realizados con OpenGL. El triángulo de la izquierda ha sido coloreado con un sólo color (coloreado liso) y el de la derecha con tres colores diferentes (coloreado degradado). 70
5.5. Relación de proporcionalidad para la asociación entre el valor de elevación de un punto con el nivel que le corresponde en la escala de interpolación de los colores. 71
5.6. Visualización de la superficie del terreno de la figura 5.1 (a) usando los colores de las curvas de nivel y (b) usando sólo dos colores. 72
5.7. Interfaz gráfica de la aplicación para la reconstrucción de terrenos. . . 73
5.8. Selección de curvas de nivel y asignación de su valor de elevación. En café se muestran las curvas que no tienen valor asignado, en rojo las que ya lo tienen y en verde a la que se está asignando. 74

A.1. (a) Esqueleto con ramificaciones. (b) Esqueleto con un ciclo. 84

B.1. (a) Imagen digitalizada de la superficie formada con elipses. (b) Imagen binaria correspondiente a la imagen en (a). 87
B.2. Corteza, esqueleto y triangulación de la superfice de la figura B.1. . . 88
B.3. Superfice de la figura B.1 coloreada usando los colores asignados de acuerdo a los valores de elevación. 88
B.4. Superfice de la figura B.1 coloreada usando sólo dos colores para la degradación. 89
B.5. Imágenes digitalizada y binaria del mapa de contornos del par de cerros en la región del estado de Puebla. 89
B.6. Corteza, esqueleto y triangulación de la superfice de la figura B.5. . . 90
B.7. Superfice de la figura B.5 coloreada usando los colores asignados de acuerdo a los valores de elevación. 90
B.8. Superfice de la figura B.5 coloreada usando sólo dos colores para la degradación. 91
B.9. Imágenes digitalizada y binaria del mapa de contornos la Malinche. . 91
B.10. (a) Corteza, esqueleto y triangulación de la superfice de la figura B.9. (b) Acercamiento al pico de la superficie. 92
B.11. (a) Superfice de la figura B.9 coloreada usando los colores asignados de acuerdo a los valores de elevación. (b) Acercamiento al pico de la superficie. 93
B.12. (a) Superfice de la figura B.9 coloreada usando sólo dos colores para la degradación. (b) Acercamiento al pico de la superficie. 94

C.1. Resultado del muestreo empleando un valor de $r = 0.415$. 96
C.2. Corteza (líneas gruesas) y esqueleto (líneas delgadas) del conjunto de puntos muestreados con $r = 0.415$. 96
C.3. Resultado del muestreo empleando un valor de $r = 0.6$. 97
C.4. Corteza (líneas gruesas) y esqueleto (líneas delgadas) del conjunto de puntos muestreados con $r = 0.6$. 98

C.5. Acercamiento a una región de la gráfica C.4. En la región marcada con el círculo se presenta una deformación tanto en la reconstrucción poligonal como en el esqueleto. 99
C.6. Resultado del muestreo empleando un valor de $r = 0.8$. 99
C.7. Corteza (líneas gruesas) y esqueleto (líneas delgadas) del conjunto de puntos muestreados con $r = 0.8$. 100
C.8. Acercamiento a una región de la gráfica C.7. Las regiones marcadas con un círculo represetan las zonas con deformación. 101

Índice de cuadros

2.1. Estructuras usuales para el almacenamiento de los MDE's. 9

3.1. La primera columna es el formato del archivo con la información de las líneas de contorno, la segunda columna es la explicación del formato. 29
3.2. Formato del archivo con la información del conjunto de muestreo de cada contorno. 38

4.1. Condiciones que deben cumplir los puntos en el plano, A, B, C y D, para pasar la prueba dentro del círculo. 41
4.2. Ejemplo un archivo de entrada para el programa que calcula el diagrama de Voronoi y la triangulación de Delaunay del conjunto de puntos muestreados que definen una elipse (*conjunto E*). 42
4.3. Archivo de salida del programa *qdelaunay* para el archivo de entrada del *conjunto E*. 43
4.4. Archivo de salida del programa *qvoronoi* para el archivo del *conjunto E*. 44
4.5. Formato del archivo con la información de los vértices y las aristas del esqueleto. 49
4.6. Codificación de los grafos del esqueleto de la figura 4.8(a). 53
4.7. Formato del archivo con la información de los vértices del esqueleto organizados en grafos. 55

Capítulo 1

Introducción

1.1. Antecedentes

Uno de los principales intereses de la topografía es la representación gráfica de las características de la superficie de un lugar o región en un mapa, indicando sus posiciones relativas y elevaciones. Para representar formaciones terrestres, a través del tiempo se han desarrollado metodologías que involucran el uso de contornos y mapas bidimensionales para representar topografía tridimensional.

Los contornos son líneas continuas que representan puntos con la misma altitud y son utilizados para representar elevaciones terrestres en planos bidimensionales. Entre más cercanos estén los contornos entre sí, mayor es la pendiente del terreno.

En topografía, los mapas de líneas de contorno constituyen aún la forma más común de representar datos de elevación de la superficie de la tierra y representan un auxiliar en estudios cuantitativos en las ciencias de la naturaleza: el clima a escala local, el movimiento y la acción del agua y, consecuentemente, los numerosos procesos biológicos condicionados por ellos, se encuentran estrechamente asociados a la forma y altitud de la superficie del terreno en los que se desarrollan [1].

A pesar del desarrollo de sistemas basados en los satélites modernos, una gran parte de la base de datos mundial de elevación del terreno está en forma de cubiertas de contornos, por las agencias cartográficas tradicionales (INEGI [2], p.e.). Estos mapas tienen la ventaja de que fueron desarrollados usando la comprensión humana de las formas de la tierra observadas, sin embargo tienen la desventaja de que no pueden ser convertidas fácilmente a un formato digital útil. Debido al alto costo de los métodos directos para la captura del terreno, a menudo se prefieren documentos cartográficos tales como mapas de contornos [3].

Por lo dicho anteriormente, el proyecto de reconstruir la superficie de un terreno a partir del mapa de contornos representa un problema de visualización atractivo.

Una aproximación empleada para reconstruir un terrero consiste en interpolar las líneas de contorno a un arreglo bidimensional de elevaciones [4]; este es un problema clásico de la cartografía computacional. Entonces, el problema de la reconstrucción de una superficie es equivalente a plantear un problema de interpolación. Para resolver el problema de interpolación, primero están las heurísticas tradicionales tales como extender líneas rectas en ocho direcciones partiendo de un punto de prueba hasta que intersequen ocho líneas de contorno, y después interpolar con un promedio de pesos, como se describe en [5]. En [6] se muestran métodos similares de pesos de distancia inversa usando a menudo "vecinos naturales". Estos métodos pueden trabajar si los contornos no están entrelazados.

Una segunda aproximación para resolver el problema de interpolación consiste en el uso de ecuaciones diferenciales parciales (EDPs) para modelar una superficie sujeta a ciertas restricciones. Se usa una EDP simple que representa el Laplaciano (de la superficie que interpola las curvas de nivel), o ecuación de flujo de calor, $z_{xx} + z_{yy} = 0$, donde $z_{xx} = \frac{\partial^2 z}{\partial x^2}$ y $z_{yy} = \frac{\partial^2 z}{\partial y^2}$. Los valores de elevación entonces, pueden hacerse corresponder con valores de temperaturas. La relevancia del flujo de calor es que, si se considera que las líneas de contorno están fijas a temperaturas conocidas, y se considera que la superficie conduce calor uniformemente, entonces cada punto en la superficie entre las líneas de contorno eventualmente se equilibrará a alguna temperatura, la cual se regresa a un valor de elevación [4].

Otro método completamente diferente es la interpolación de Voronoi de los puntos [7]. Aquí, se forma un diagrama de Voronoi con los puntos que forman los contornos. Un punto de prueba (punto de Voronoi) es insertado dentro del diagrama de los contornos, y se usan las áreas de los polígonos de Voronoi que vienen de sus vecinos para poner peso a las elevaciones de tales vecinos. La interpolación de Voronoi resuelve el problema de artefactos como superficies, pendientes o discontinuidades en las curvaturas, y elevación local extrema.

La triangulación de puntos en Redes Irregulares Trianguladas (RITs), con triangulación de Delaunay, es otro método relacionado [8]. Cualquier método de interpolación puede ser usado dentro de cada triángulo. Sin embargo, algunas veces un triángulo tendrá sus tres vértices en el mismo contorno, causando que sea horizontal, lo cual es indeseable.

1.2. Sistema a desarrollar

En este trabajo de tesis se desarrollarán la visualización y la manipulación de volúmenes reconstruidos a partir de su mapa de contornos. Específicamente, se reconstruirán terrenos (montañas, valles, etc.) a partir de su mapa topográfico.

El problema de la reconstrucción de terrenos puede verse como un caso especial

del problema general de la reconstrucción tridimensional a partir de secciones bidimensionales cruzadas. La reconstrucción aquí es un problema de interpolación de n contornos cada uno con una elevación específica. La característica de estos contornos es que no se intersecan entre ellos y que están anidados.

En este trabajo se propone que la solución del problema de reconstrucción tridimensional de un terreno sea dividida en dos partes principales:

1. procesar la imagen del mapa topográfico para obtener las líneas de contornos y

2. realizar una interpolación entre los contornos y generar la triangulación.

Después de haber generado la triangulación que defina la superficie del terreno, se realiza la visualización de la reconstrucción.

1.2.1. Obtención del mapa de contornos

La información de las curvas de nivel o contornos de determinada región puede ser obtenida principalmente en dos formatos: mapas topográficos impresos y datos vectoriales, ambos proporcionados por el INEGI [2]. En este trabajo optamos por hacer uso de los mapa topográficos impresos por que, además de ser un problema tratado en la literatura [9, 10, 11, 12], con este formato se facilita, en cierto sentido, el muestreo de las líneas de contorno (necesario para la segunda etapa del trabajo) empleando técnicas de procesamiento digital de imágenes.

El problema específico de extracción de las líneas de contorno a partir del mapa topográfico digitalizado es un problema difícil de resolver por la limitante global que representa su topología: el gran número de interrupciones en las líneas de contorno, las conexiones entre dos (o más) líneas de contorno y la baja densidad de información de elevación contenida en el mapa.

Estudios previos han mostrado que la geometría local de las líneas de contorno no proporciona evidencia suficiente para una reconstrucción automática de éstas. Para sobrellevar esta insuficiencia muchos artículos relacionados describen sistemas que involucran la intervención de un operador humano, quien generalmente resuelve la ambigüedad y/o corrige el resultado de la reconstrucción [10]. La tarea se simplifica trabajando con la imagen digitalizada de un dibujo con las líneas de contorno. En la figura 1.1 se presenta el ejemplo de una imagen de este tipo, con las líneas de contorno de un cerro.

Figura 1.1: Imagen digitalizada del dibujo de las líneas de contorno de un cerro tomado del mapa topográfico E14B47 del INEGI.

1.2.2. Interpolación de los contornos

La segunda parte la han resuelto varios autores de distintas formas. Hasta ahora se han propuesto algunos métodos que resuelven este problema (Heurísticas, Ecuaciones Diferenciales Parciales, Interpolación de Voronoi, Triangulación de Delaunay, etc.), cada uno con ventajas y desventajas que dependen de la procedencia y forma de los datos [4].

Aquí, se ha resuelto el problema empleando las construcciones geométricas de la *corteza* y el *esqueleto*, basadas a su vez en dos construcciones geométricas fundamentales, a saber, el *diagrama de Voronoi* y la *triangulación de Delaunay*. Para este método se requiere contar con el conjunto de puntos muestra que definan las líneas de contorno. El conjunto de muestreo debe cumplir con ciertas condiciones de calidad.

Después de obtener la triangulación a partir de los contornos, se procede a la visualización de la reconstrucción.

En la figura 1.2 se presenta un diagrama que ilustra la estructura general del sistema.

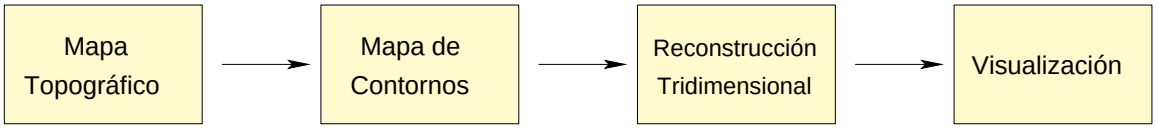


Figura 1.2: Diagrama general del sistema

1.3. Organización de la tesis

Como se mencionó en la sección 1.2, en este trabajo se resuelven dos problemas fundamentales:

1. la extracción de las líneas de contorno de un mapa topográfico y
2. la interpolación de los contornos.

La tesis se ha dividido en seis capítulos. En el segundo de ellos se definen una serie de conceptos básicos que se emplearán en los capítulos tres y cuatro con respecto al procesamiento de imagen y la interpolación de los contornos.

El capítulo tres está destinado a la explicación del proceso de la extracción de las líneas de contornos a partir del mapa topográfico. A la imagen digitalizada del mapa topográfico se le aplican diferentes algoritmos de procesamiento de imagen y como resultado se obtiene un conjunto de pixels que corresponde al muestreo de cada una de las líneas de contorno del mapa. En este capítulo también se describe el proceso de muestro de las líneas de contorno.

En el capítulo cuatro se plantea la metodología a seguir para la interpolación de los contornos. Se describe el método que se ha empleado para obtener el esqueleto y la corteza de un conjunto de puntos, cómo y por qué se emplean en el trabajo de la interpolación de los contornos, y la interpolación de los contornos en sí. Se plantea también cómo organizar la información para poder realizar un modelo para la reconstrucción del terreno.

En el capítulo cinco se explica cómo realizar la visualización del terreno a partir del modelo obtenido. Se describe la integración de todos los elementos de visualización y manipulación en una interfaz gráfica. La manipulación se realiza sobre la visualización; en este capítulo también se plantea de qué manera puede aplicarse tal manipulación. Se presentan además algunas imágenes correspondientes a los resultados obtenidos.

Finalmente, en el capítulo seis se presentan las conclusiones y el trabajo futuro.

Capítulo 2

Marco Teórico

Como se describió en el capítulo anterior, el propósito de este trabajo de tesis es llevar a cabo el proceso, junto con todas sus implicaciones, de llevar el mapa impreso de los contornos de un terreno (cerro, montaña, valle, etc.) a la reconstrucción tridimensional de éste. En este capítulo se expone una serie de conceptos y definiciones útiles en dicho proceso.

2.1. Modelos digitales de elevaciones

El tratamiento de los datos geográficos se ha visto grandemente afectado con la revolución informática que ha tenido presencia desde los años 60. El concepto tradicional de lo que es la cartografía, ha sufrido severas transformaciones. El manejo manual de la información cartográfica ha evolucionado gracias a la introducción de los *sistemas de información geográfica* (SIG's), definidos como sistemas informáticos diseñados para el manejo y análisis de información espacial.

El concepto del mapa impreso, en este contexto, se ha extendido considerablemente y se ha reconocido que un mapa puede ser representado también mediante un conjunto de datos numéricos en donde se encuentran reunidas las relaciones espaciales de los elementos de un terreno.

Un *modelo digital de elevaciones* (MDE) se define como una estructura numérica de datos que representa la distribución espacial de la altitud de la superficie del terreno [1]. Un MDE puede describirse de forma genérica de la siguiente manera:

$$z = \zeta(x, y) \tag{2.1}$$

donde z es la altitud del punto situado en las coordenadas x y y, y ζ es la función que relaciona la variable con su localización geográfica. Los valores de x y y generalmente corresponden con las abscisas y ordenadas de un sistema de coordenadas plano.

La ecuación 2.1 representa una superficie o campo escalar en la que la altitud es una variable continua. Dado que esta superficie está formada por un número infinito de puntos no es posible su modelización sin cierta pérdida de información, proceso equivalente al de generalización cartográfica en los mapas convencionales.

En la versión digital sería posible presentar, al menos teóricamente, la ecuación 2.1, que relaciona la altitud con la localización geográfica. Sin embargo, la complejidad del relieve hace que su representación matemática mediante funciones no tenga más que un significado simbólico.

En la práctica, las cotas correspondientes a una zona sólo pueden representarse mediante una ecuación cuando la parcela descrita es pequeña y su relieve muy simple. Este método puede ser utilizado para aplicaciones concretas, operando sobre zonas muy limitadas, pero en cuanto el relieve se complica o la superficie aumenta, el ajuste de una ecuación para su descripción se hace imposible.

Como alternativa, se han buscado soluciones para representar la altitud mediante conjuntos limitados de cotas, diseñando las estructuras de datos que buscan un equilibrio entre la facilidad de manejo y la descripción realista del relieve.

Existen diferentes formas de representar los modelos digitales de elevaciones de acuerdo a la estructura y organización de los datos. Clásicamente, las formas de presentar estos modelos se han dividido en dos, a saber, *vectorial* y *raster*. Los modelos *vectoriales* están basados en entidades (básicamente puntos y líneas) definidas por sus coordenadas. En los modelos *raster*, la representación de los datos se basa en la localización espacial sobre una retícula regular de puntos a los cuales se le asigna el valor de elevación. En el cuadro 2.1 se presenta un resumen de las estructuras más usuales empleadas para el almacentamiento de los MDE's.

En este trabajo se ha empleado la estructura de los *contornos*, puesto que presenta un acoplamiento directo con la extracción de los datos que se maneja en la fase de procesamiento digital de la imagen del mapa de contornos.

2.1.1. Contornos

En el modelo de contornos, la estructura base es un vector compuesto por un conjunto de pares de coordenadas (x, y) que describen la trayectoria de líneas isométricas, es decir, líneas correspondientes a las curvas de nivel de un mapa topográfico convencional. Cada vector tiene un número de elementos variable, y tiene asociada la información de elevación correspondiente con la curva de nivel que describe.

Una curva de nivel concreta queda definida, por tanto, mediante un vector orde-

VECTORIALES	*Contornos*	*Secuencial*	Las líneas se almacenan como cadenas de cotas.
		Analítica	Las líneas se almacenan como segmentos de Bézier, polinómicos, etc.
	Perfiles		Cadenas paralelas de cotas en línea con altitud variable.
	Triángulos		Red de triángulos irregulares (RIT).
RASTER	*Matrices*	*Regulares*	Cotas sobre una malla cuadrada de filas y columnas equidistantes.
		Escalables	Cotas sobre submatrices jerárquicas y de resolución variable.
	Polígonos		Cotas asignadas a retículas poligonales regulares (triángulos o hexágonos).

Cuadro 2.1: Estructuras usuales para el almacenamiento de los MDE's.

nado de puntos que se sitúan sobre ella a intervalos adecuados (no necesariamente iguales) para garantizar la exactitud necesaria del modelo. La localización espacial de cada elemento es explícita, pues concuerda con los valores de su coordenada.

Entonces, el MDE en su totalidad está constituído por el conjunto de las curvas de nivel que pasan por la zona representada, separadas generalmente por intervalos constantes de altitud.

2.2. Vecinos en imágenes digitales

En esta sección se presenta el concepto de imagen digital, los elementos que la componen y algunas relaciones básicas entre los pixeles de una imagen digital.

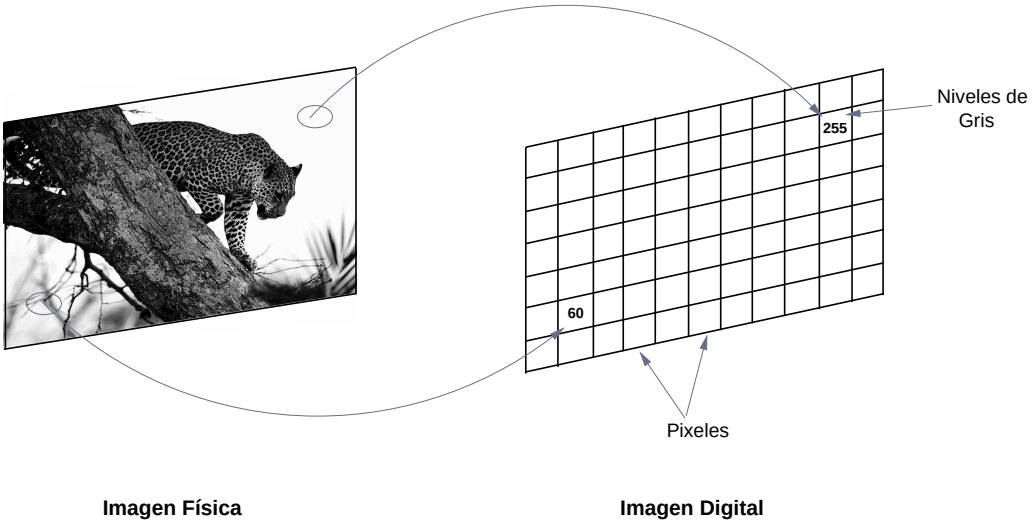


Figura 2.1: Una imagen física y su correspondiente imagen digital.

2.2.1. Imagen digital

La forma natural en la que se encuentran las imágenes físicas no es una representación idónea para su análisis por medio de una computadora. Puesto que las computadoras trabajan con datos numéricos, una imagen debe ser convertida a un formato numérico antes de poder ser procesada en ellas.

La figura 2.1 ilustra cómo una matriz de números puede representar a una imagen física, la cual es dividida en pequeñas regiones llamadas *elementos de la imagen* (*picture elements*) o *pixeles*. Entonces, la imagen es dividida en líneas horizontales de pixeles adyacentes. A la representación matricial de la imagen física se le conoce como *imagen digital*. Cada pixel tiene una ubicación o dirección en la imagen digital (renglón y columna) y un valor entero asociado llamado *nivel de gris*, el cual refleja el brillo de la imagen en ese punto.

Al proceso de conversión de la imagen física a digital se le llama *digitalización*. En este paso se genera un número entero para cada pixel que representa el brillo o la opacidad de la imagen en ese pixel.

2.2.2. Vecindad entre pixeles

Un pixel p en la coordenada (x, y) tiene cuatro vecinos *horizontales* y *verticales*, cuyas coordenadas están dadas por:

$$\begin{array}{ll}(x+1, y) & (x-1, y) \\ (x, y+1) & (x, y-1).\end{array}$$

A estos pixeles se les conoce como *4-vecinos* de p, y al conjunto que forman se le denota $N_4(p)$. Cada uno de los pixeles se encuentra a una distancia unitaria de (x, y).

Los cuatro vecinos en *diagonal* de p, conocidos como *d-vecinos* de p, tienen coordenadas:

$$(x+1, y+1) \quad (x+1, y-1)$$
$$(x-1, y+1) \quad (x-1, y-1)$$

y el conjunto de ellos es denotado por $N_D(p)$. Estos puntos junto con los 4-vecinos, son llamados los *8-vecinos* de p, y el conjunto es denotado por $N_8(p)$.

2.2.3. Conexidad

Para establecer si dos pixeles están conectados [13], debe determinarse de alguna manera si son adyacentes (por ejemplo, si son 4-vecinos) y si sus niveles de gris satisfacen un criterio de similitud específico (por ejemplo, si son iguales). En el caso de una imagen binaria con valores 0 y 1, dos pixeles pueden ser 4-vecinos, pero no se dice que estén conectados a menos de que tengan el mismo valor.

Sea V el conjunto de valores de nivel de gris usados para definir la conexidad; por ejemplo, en una imagen binaria, $V = \{1\}$ para la conexidad de pixeles con valor 1. Se consideran tres tipos de conexidad:

1. *Conexidad-4*. Dos pixeles p y q con valores en V están *4-conexos* si q está en el conjunto $N_4(p)$.
2. *Conexidad-8*. Dos pixeles p y q con valores en V están *8-conexos* si q está en el conjunto $N_8(p)$.
3. *Conexidad-m* (conexidad mixta). Dos pixeles p y q con valores en V están *m-conexos* si
   - *a)* q está en $N_4(p)$, o
   - *b)* q está en $N_D(p)$ y el conjunto $N_4(p) \bigcap N_4(q)$ está vacío.

Un pixel p es *adyacente* a un pixel q si están conectados. Se define *4-, 8-*, ó *m-adyacencia* dependiendo el tipo de conexidad especificada.

Un camino del pixel p con coordenadas (x, y) al pixel q con coordenadas (s, t) es una secuencia de pixeles con coordenadas

$$(x_0, y_0), (x_1, y_1), \cdots, (x_n, y_n)$$

donde $(x_0, y_0) = (x, y)$ y $(x_n, y_n) = (s, t)$, (x_i, y_i) es adyacente a (x_{i-1}, y_{i-1}), $1 \leq i \leq n$, y n es la longitud del camino. Pueden definirse *4-, 8-* ó *m-caminos* dependiendo del tipo de adyacencia especificada.

2.3. Mapa de distancias

En el capítulo 1 se establecieron dos etapas principales en este trabajo: procesamiento digital de imágenes e interpolación de contornos. La primera fase, puede describirse como el proceso de obtención del conjunto de coordenadas (pixeles) que describen a cada contorno dentro del mapa de contornos de un terreno determinado. En este proceso, se emplea una operación denominada *transformada de la distancia*, la cual se aplica a imágenes binarias. Cuando se ejecuta esta operación, se obtiene por resultado una imagen en tonos de gris. A la imagen resultante se le conoce como el *mapa de distancias* de la imagen original.

2.3.1. Transformada de la distancia

Considérese una imagen binaria constituida por pixeles característicos y no-característicos. Los pixeles característicos son los que conforman los puntos, aristas u objetos en la imagen y los no-característicos son los pixeles de fondo. Una transformada de la distancia es una operación que convierte esta imagen binaria en una imagen en tonos de gris, donde todos los pixeles tienen un valor correspondiente a la distancia hacia el pixel característico más cercano. En la figura 2.2 se muestra un ejemplo. Después de aplicar la transformada de la distancia a la imagen del ejemplo, todos los pixeles tienen un valor correspondiente a la distancia hacia el objeto en ella. La imagen resultante puede ser vista como una serie de contornos de distancia, donde cada contorno está formado por todos los pixeles equidistantes al objeto característico.

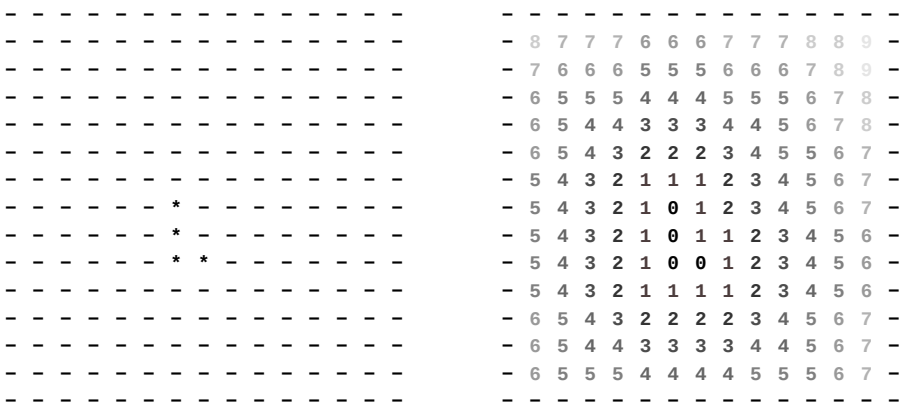


Figura 2.2: Ejemplo de una transformada de la distancia.
A la izquierda se encuentra una imagen binaria, con pixeles característicos (*) y no-característicos (-). A la derecha se observa la imagen resultante. La distancia euclidiana ha sido redondeada al valor entero más cercano.

Calcular la distancia de un pixel a un conjunto de pixeles característicos es, inherentemente, una operación global. A menos de que se trabaje con imágenes digitales

muy pequeñas, todas las operaciones globales son inadmisiblemente costosas. Por lo tanto, se hace necesario el uso de algoritmos en los que se considere sólo una vecindad pequeña a la vez, pero sin dejar de proporcionar aproximaciones razonables a la distancia euclidiana.

Se han desarrollando varios algoritmos para el cálculo de la transformada de la distancia. En adelante, estos algoritmos serán llamados ATD's. En este trabajo se ha empleado el algoritmo denominado *Chamfer 3-4* [14].

2.3.2. Algoritmo

Los ATD's que manejan sólo una porción pequeña de la imagen a la vez, se basan en la siguiente idea: las distancias globales en la imagen son aproximadas por medio de la propagación de distancias locales, es decir, distancias entre pixeles vecinos.

Una imagen binaria original, a la cual le será aplicada la transformada de la distancia, consiste de pixeles característicos con valor inicial de 0, y pixeles no-característicos con valor inicial en infinito, esto es, un número convenientemente grande que, por efectos prácticos, en este trabajo se ha considerado 255.

En la figura 2.3 se presentan las máscaras empleadas en el algoritmo. Cada casilla de las máscaras, tiene asignado un valor entero aproximado a la distancia local que será propagada sobre la imagen.

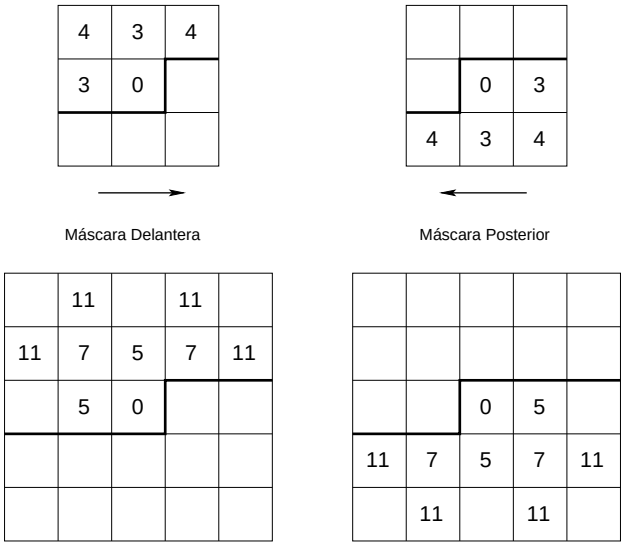


Figura 2.3: Máscaras para el ATD Chamfer 3-4 y 5-7-11.
En la parte superior se muestra la máscara de 3×3, empleada para el Chamfer 3-4; y en la parte inferior la de 5×5, para el Chamfer 5-7-11.

El algoritmo entonces inicia a partir de la imagen cuyos pixeles sólo tienen valores de 0 o 255. Las máscaras son pasadas sobre la imagen, una a la vez, de la siguiente manera: la máscara delantera, de izquierda a derecha y de arriba hacia abajo; y la máscara posterior, de derecha a izquierda y de abajo hacia arriba. En las barridas, las máscaras son colocadas de manera que la casilla central se sitúe sobre cada pixel de la imagen. Entonces, para cada casilla k de la máscara, se calcula la suma de su valor de distancia local, c_k, con el valor del pixel que se encuentra "debajo" de tal casilla. El nuevo valor del pixel "central" en la imagen (el pixel sobre el que se situó la casilla central), es el valor mínimo de las sumas calculadas anteriormente. Después de las dos pasadas, la transformada de la distancia se ha calculado completamente.

El algoritmo, entonces, queda como se muestra en el procedimiento 1. En la figura 2.4 se presenta un ejemplo de la ejecución de este algoritmo. Nótese que en las barridas, la casilla central de las máscaras nunca es situada en los bordes de la imagen.

Procedimiento 1 Cálculo de la transformada de la distancia

Entrada: La imagen binaria $imgIN$, de dimensiones $rows \times cols$ y las máscaras delantera, $FWDmask$, y posterior, $BCKmask$.

Salida: La transformada de la distancia de la imagen, $imgDT$.

{Los índices de la imagen van de 0 a $rows - 1$ en las filas y de 0 a $cols - 1$ en las columnas}

{Barrida hacia adelante}
for $i = 1$ to $rows - 2$ **do**
  for $j = 1$ to $cols - 2$ **do**
    for $k = 0$ to 1 **do**
      for $l = 0$ to 2 **do**
        $matrix_{k,l} = imgIN_{i-1+k,j-1+l} + FWDmask_{k,l}$
    $imgDT_{i,j} = \min(matrix)$

{Barrida hacia atrás}
for $i = rows - 2$ to 1 **do**
  for $j = cols - 2$ to 1 **do**
    for $k = 0$ to 1 **do**
      for $l = 0$ to 2 **do**
        $matrix_{k,l} = imgIN_{i+k,j-1+l} + BCKmask_{k,l}$
    $imgDT_{i,j} = \min(matrix)$

A este algoritmo se le denomina *Chamfer* [15], y puede ejecutarse con máscaras de dimensiones 3×3 (Chamfer 3-4) y 5×5 (Chamfer 5-7-11); ambas máscaras se muestran en la figura 2.3. Los valores de los coeficientes en las matrices, son el resultado de la mejor aproximación entera a las distancias locales óptimas, según se

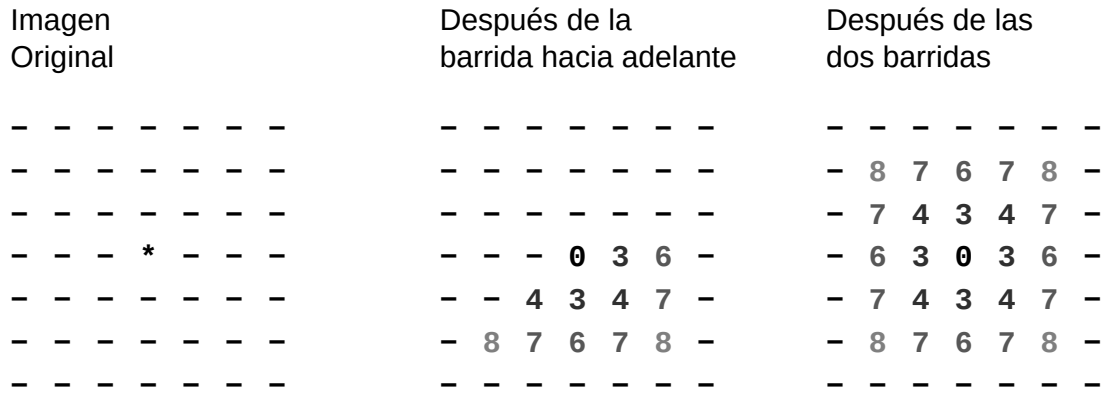


Figura 2.4: Cálculo de la transformada de la distancia con el Chamfer 3-4.

demostró en [14], en donde se plantea que tales aproximaciones fueron encontradas multiplicando todas las distancias locales por un factor constante k, y redondeándolas hacia el entero más cercano. Así pues, después de aplicar el ATD, los valores obtenidos deben ser divididos por k para recuperar la distancia euclidiana. De acuerdo a los autores, los valores de las distancias locales obtenidos por el ATD, aunque son ligeramente diferentes a los valores de la distancia euclidiana real, son aceptables para en el procesamiento de imágenes. La desviación máxima con respecto a la distancia euclidiana real es de 8 % para la máscara de 3×3, y de 2 % para la de 5×5.

Como se mencionó anteriormente, en este trabajo se ha empleado la máscara de 3×3 (Chamfer 3-4). La justificación de dicha elección se presentará en el capítulo 3, donde se muestre la aplicación de la transformada de la distancia en este trabajo.

2.4. Eje medio

En esta sección se revisará la definición de *eje medio*, la cual también será empleada en la fase de procesamiento de imagen, definida en el capítulo 1.

El *eje medio* de una curva F, es el lugar geométrico de los centros de todos los círculos en el plano que son tangentes a F en dos o más puntos distintos [13]. Esto también puede ser visto como el lugar geométrico de los puntos en el plano que tienen dos o más "puntos más cercanos" a F [16]. En la imagen de la figura 2.5 la línea gruesa representa el lugar geométrico de alguna curva F y las curvas delgadas representan el eje medio de F.

Como se mencionó en el capítulo 1, en el método de interpolación de los contornos empleado en este trabajo se requiere contar con el conjunto de puntos que definan a las curvas de nivel o líneas de contorno del terreno a reconstruir. Para encontrar este conjunto de puntos, es necesario calcular el eje medio de las curvas de nivel y la

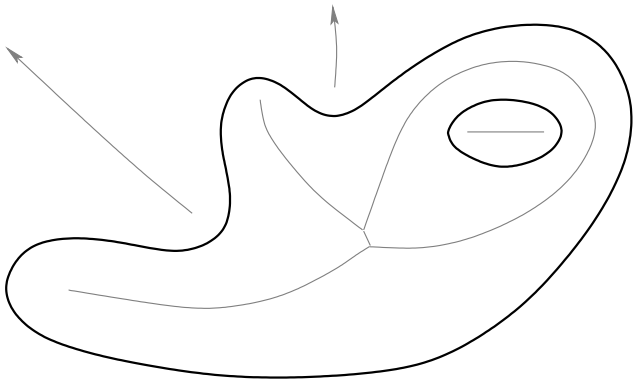


Figura 2.5: Eje medio de una curva.
Las curvas delgadas son el eje medio de la curva gruesa.

distancia de cada punto en las curvas hacia el punto más cercano en el eje medio de éstas. Este proceso será explicado con mayor detalle en la sección 3.2.

Existen diferentes métodos para el cálculo del eje medio (exactos, discretos, continuos) [17], en este trabajo hemos empleado un método discreto, con el cual el objeto (eje medio) es almacenado en una imagen binaria. Para calcularlo hemos empleado la transfomada de la distancia definida en la sección anterior. Este proceso será explicado en la sección 3.2.1.

El eje medio también es conocido como *esqueleto* [13]. En el capítulo 4, éste es calculado de una manera diferente, pues se requiere contar con un conjunto de puntos que lo definan para realizar la interpolación entre las curvas de nivel del terreno. La explicación detallada se presenta en la sección 4.1.

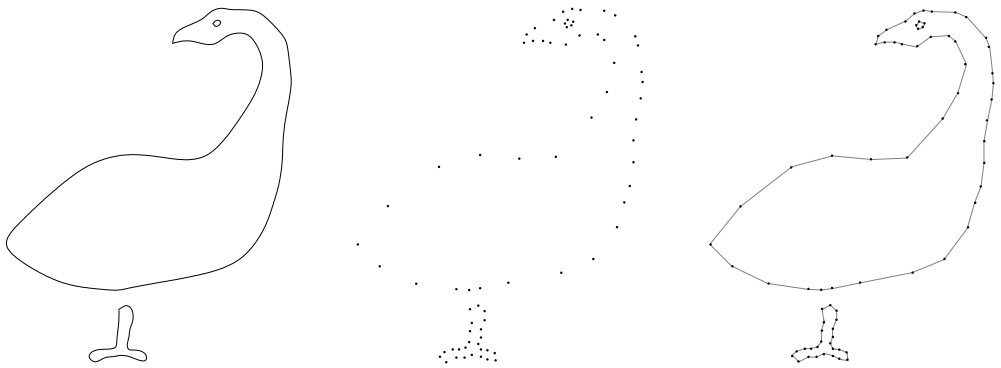
2.5. Reconstrucción poligonal

Como se ha venido mencionando, para la aplicación del método de interpolación elegido, se requiere contar con un conjunto de puntos de muestreo que definan a cada curva de nivel. En esta sección se define el concepto de *reconstrucción poligonal* y la condición que el conjunto de puntos muestreados debe cumplir para garantizar la reconstrucción.

En este trabajo se considerará una *curva suave* como una curva en el plano que no puede tener ramificaciones ni intersecciones consigo misma. Las curvas de nivel de los mapas de contornos cumplen con esta característica. Si una curva suave es muestreada con la densidad suficiente, el grafo sobre las muestras es una *reconstrucción poligonal* de la curva.

Sea F una *curva suave*, y sea $S \subset F$ un conjunto finito de puntos muestra sobre F. La *reconstrucción poligonal* de F a partir de S es un grafo que conecta cada par de puntos adyacentes a lo largo de F, y no otras. A esta reconstrucción poligonal también se le conoce como *corteza* [16].

En la figura 2.6(a) se presenta, a modo de ejemplo, el dibujo de un ganzo formado por tres curvas (ojo, cuerpo y pata), en la parte (b) el conjunto de muestreo para cada una de estas curvas y en la parte (c) la reconstrucción poligonal de cada curva. Nótese que se requiere un menor número de muestras en la espalda del ganso que en su cabeza y patas. Este ejemplo fue construido manualmente para ilustrar el proceso de reconstrucción poligonal.



(a) (b) (c)

Figura 2.6: (a) Curvas que definen el dibujo de un ganzo. (b) Conjunto de puntos de muestra de las curvas. (c) Reconstrucción poligonal de las curvas.

A continuación se indica la condición de muestreo, planteada en [16], para garantizar una reconstrucción poligonal adecuada. El proceso para la obtención de la corteza o reconstrucción poligonal de una curva a partir de su conjunto de muestreo se presenta en la sección 4.1.

2.5.1. Condición de muestreo

La condición de muestreo que debe cumplir S para garantizar una reconstrucción poligonal adecuada de F a partir de S, está basada en una función del *Tamaño de la Característica Local*, la cual, en cierto sentido, cuantifica el "nivel de detalle" local en un punto sobre F.

El *Tamaño de la Característica Local*, $TCL(p)$, de un punto $p \in F$ es la distancia euclidiana de p al punto más cercano en el eje medio de la curva [16].

Nótese que, puesto que se hace uso del eje medio, la definición del Tamaño de la Característica Local depende tanto de la curvatura en p como de la cercanía de ciertos rasgos característicos de la curva.

Entonces, la condición de muestreo queda como sigue: F está *r-muestreada* por un conjunto de puntos muestra S si cada punto $p \in F$ guarda una distancia no mayor a $r \times TCL(p)$ con una muestra $s \in S$, para valores de $r \leq 1$.

2.6. Diagrama de Voronoi y triangulación de Delaunay

Para calcular la reconstrucción poligonal de una curva a partir de su conjunto de muestreo se emplean las construcciones geométricas del diagrama de Voronoi y la triangulación de Delaunay. De acuerdo a [16], si en el proceso de muestreo de una curva se elige un valor de r adecuado, la triangulación de Delaunay contiene la reconstrucción poligonal de la curva, y el diagrama de Voronoi contiene su esqueleto, el cual es empleado en la fase de interpolación de contornos.

A continuación se presentan las definiciones de ambas construcciones.

2.6.1. Diagrama de Voronoi

Considérese el siguiente problema. Dado un conjunto S de N puntos en el plano, para cada punto p_i en S ¿cuál es el lugar geométrico de puntos (x, y) en el plano que están más cerca a p_i que a cualquier otro punto de S?.

Intuitivamente, puede observarse que la solución a este problema es una partición del plano en regiones, donde cada región es el lugar geométrico de los puntos (x, y) que son más cercanos a un punto que a cualquier otro en S. Ahora se analizará dicha partición del plano. Dados dos puntos p_i y p_j, el conjunto de puntos más cercanos a p_i que a p_j es justamente el *plano medio* que contiene a p_i, el cual está definido por la bisectriz perpendicular de $\overline{p_i p_j}$. Se denotará como $H(p_i, p_j)$ a este plano medio. El lugar geométrico de puntos más cercanos a p_i que a cualquier otro punto, el cual será denotado por $V(i)$, es la *intersección de $N-1$ planos medios*, y es una región poligonal convexa que no tiene más de $N-1$ lados, esto es,

$$V(i) = \bigcap_{i \neq j} H(p_i, p_j).$$

A $V(i)$ se le conoce como el *polígono de Voronoi asociado con* p_i.

Al conjunto de las N regiones, que dividen al plano en una red convexa, se le llama *diagrama de Voronoi*. Los vértices del diagrama se conocen como *vértices de Voronoi*, y a sus segmentos de línea como *aristas de Voronoi* [18].

2.6.2. Triangulación de Delaunay

Considérese nuevamente el conjunto S de N puntos en el plano. La cubierta convexa de S, es el conjunto convexo mínimo que contiene a S. Una *triangulación de* S, es el conjunto de regiones del plano que se producen cuando se unen los puntos de S, mediante líneas rectas que no se intersecan, de manera que cada región producida, dentro de la cubierta convexa de S, sea un triángulo.

En el diagrama de Voronoi, una línea o arista dual es aquella que pasa por un par de puntos de S cuyos polígonos de Voronoi comparten una arista. El conjunto de aristas duales del diagrama de Voronoi es una triangulación de S, y se le conoce como *triangulación de Delaunay* [18].

En la figura 2.7 se muestra la imagen del diagrama de Voronoi y la triangulación de Delaunay de un conjunto de puntos en el plano.

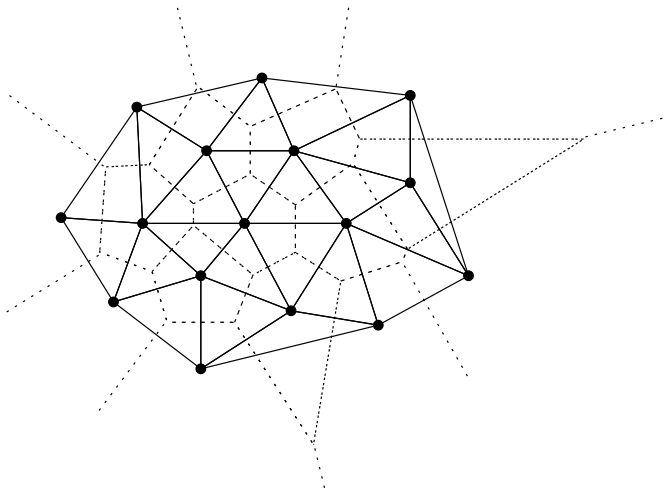


Figura 2.7: Diagrama de Voronoi y triangulación de Delaunay.

La triangulación de Delaunay también se ha empleado para generar una Red Irregular de Triángulos sobre el conjunto de puntos de muestreo junto con los puntos del

esqueleto para obtener la interpolación de los contornos.

La triangulación de Delaunay es un método que satisface el requerimiento de que el círculo que pasa por los tres nodos de un triángulo no contiene a ningún otro nodo. Este método para la triangulación de un conjunto de puntos (nodos) tiene varias ventajas sobre otros métodos:

- Los triángulos son lo más equiangulares posible, lo cual reduce los problemas de precisión numérica generados por triángulos delgados y largos.
- Se asegura que cualquier punto sobre la superficie esté tan cerca como sea posible a un nodo.
- La triangulación es independiente del orden en que sean procesados los puntos.

Por estas características, la triangulación de Delaunay ha sido la opción para que la mayoría de los usuarios e investigadores de Sistemas de Información Geográfica (SIGs) construyan Redes Irregulares Trianguladas [19].

Para el cálculo del ambas construcciones geométricas, en este trabajo se ha empleado la herramienta de software *QHull*, en donde se encuentra implementado el algoritmo *Quickhull* para calcular la cubierta convexa [20].

Capítulo 3

Procesamiento Digital de Imagen

Para obtener la reconstrucción tridimensional de un terreno partiendo del mapa impreso de los contornos (curvas de nivel) de éste, en primer lugar, es necesario extraer la información del mapa con la que se pueda definir a cada uno de los contornos en él. Esta tarea ha sido resuelta con técnicas de *procesamiento digital de imagenes*.

Como primera instancia, se requiere obtener el conjunto de todos los puntos (pixeles) que conforman a cada contorno. Ahora bien, no es necesario contar con el conjunto completo de los puntos para tener una definición de los contornos, puesto que en ciertas regiones de una curva se puede disminuir el número de puntos tomados, de modo que con el conjunto reducido de puntos pueda reconstruirse el contorno [16].

En términos generales, la imagen digitalizada del mapa de contornos debe ser procesada en dos etapas principales:

1. Extraer de la imagen digitalizada las líneas de contorno.

2. Muestrear la líneas de contorno.

En la figura 3.1 se muestra un diagrama que ilustra este proceso.

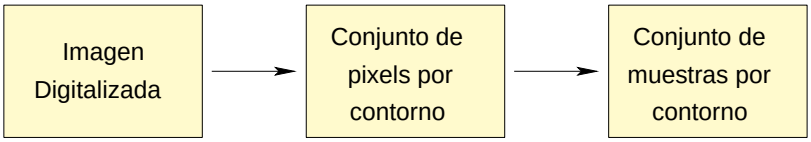
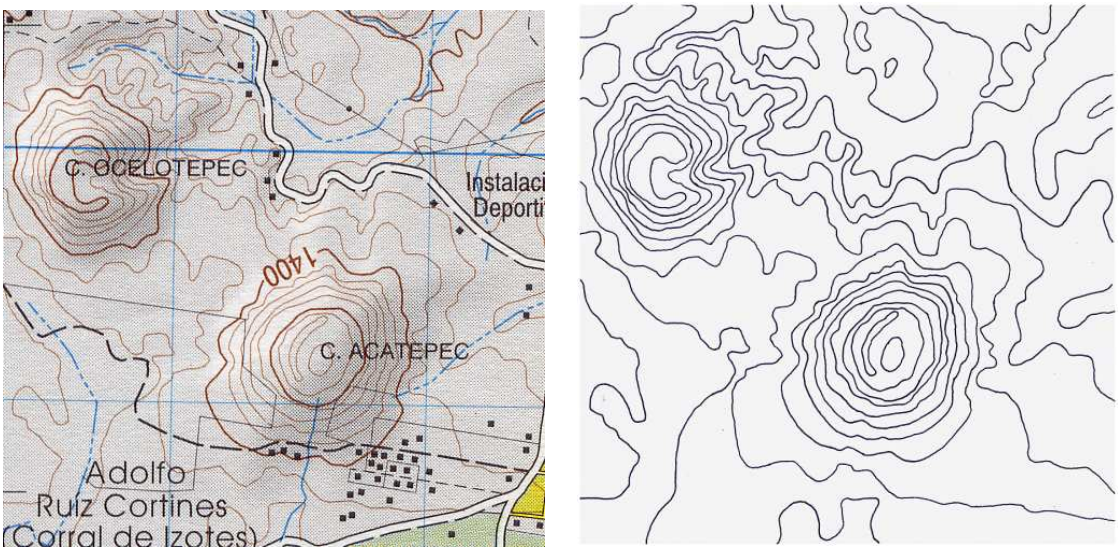


Figura 3.1: Diagrama general para el procesamiento de imagen.

3.1. Extracción de las líneas de contorno

La forma más común de adquirir información de las elevaciones de la tierra se encuentra en los mapa topográficos. En estos mapas se encuentran diversos tipos de datos a parte de las curvas de nivel del terreno, tales como delimitaciones entre estados, carreteras, ríos, leyendas, etcétera; cada uno representado con diferentes simbologías y colores, lo cual provoca interrupciones entre las líneas de contorno, como puede verse en la figura 3.2(a). Procesar este tipo de mapas para obtener aisladas las curvas de nivel del terreno de forma automática, no es un problema de solución sencilla [10].



(a) (b)

Figura 3.2: (a) Una porción del mapa topográfico E14B47 del INEGI y (b) El dibujo de sus líneas de contorno.

Con la finalidad de simplificar un poco el problema, en este trabajo partimos del dibujo de las líneas de contorno, resultado de calcar y completar con un pincel negro las curvas de nivel presentes en un mapa topográfico sobre una hoja blanca. Para digitalizar las imágenes se ha empleado un escaner Epson Perfection 1260.

En los mapas topográficos comerciales impresos, independientemente de la escala, hay regiones en las que la separación entre las curvas de nivel es muy pequeña, de modo que el calcado de las líneas se complica si se desea que no haya intersecciones ni puntos de contacto entre ellas. Por tal motivo, primeramente, se digitalizaron los mapas topográficos a una resolución convenientemente alta para evitar este tipo de problemas. Después los mapas digitalizados fueron impresos y posteriormente calcados.

En la figura 3.2(a) se muestra la imagen del mapa topográfico E14B47 del INEGI, el cual fue digitalizado a 96 dpi (*dots per inch*), y en la figura 3.2(b) la imagen del dibujo de las líneas de contorno. Estos últimos fueron digitalizados a 96 y 150 dpi, sin embargo en las imágenes a 96 dpi hay regiones en las que la distancia entre las curvas es de uno o dos pixeles, por lo que se optó por trabajar con las imágenes a 150 dpi.

Una vez que se cuenta con el dibujo de las líneas de contorno del terreno a reconstruir, el procedimiento para la extracción de las líneas es el siguiente:

1. Obtener la imagen binaria.
2. Adelgazar las líneas de contorno al grueso de un pixel.
3. Extraer todos los pixeles que conforman cada línea de contorno.

3.1.1. Conversión de la imagen RGB a imagen binaria

La imagen del dibujo de la líneas de contorno es una imagen en RGB (en tres canales de colores: rojo, verde y azul), con la posibilidad de 256^3 colores diferentes en ella. Es necesario convertir esta imagen a una imagen binaria, en donde puedan identificarse únicamente dos tipos de pixeles: pixeles de línea de contorno, o bien, *pixeles característicos* y pixeles de fondo o *pixeles no característicos*, y así facilitar el proceso de extracción de los pixeles de las líneas de contorno. Los pixeles característicos se han tomado como pixeles negros y los no característicos como blancos.

Para realizar la conversión, debe hallarse un valor de umbral para determinar cuáles pixeles de la imagen RGB deben tomarse como pixeles blancos y cuáles otros como pixeles negros. Hallar este valor en el dominio de la imagen en RGB (256^3 posibilidades de colores) es más difícil que hacerlo en el dominio de una imagen en tonos de gris (sólo 256 tonos). Para convertir una imagen RGB a tonos de gris se emplea la siguiente fórmula [13]:

$$Y = 0.299 \times R + 0.587 \times G + 0.114 \times B \tag{3.1}$$

donde Y es el tono de gris correspondiente al pixel de color (R, G, B). Y es el componente de luminosidad del modelo YIQ usado para la transmisión de video por la televisión comercial, el cual es proporcional a la cantidad de luz percibida por los ojos. Nosotros tomamos este componente como la imagen en tonos de gris de la imagen en color.

Histograma en tonos de gris

El histograma en tonos de gris es una función que muestra, por cada tono de gris, el número de pixeles en la imagen que tienen ese tono de gris. El eje de las abscisas

representa los tonos de gris, mientras que el eje de las ordenadas representa la frecuencia de ocurrencia.

Esta función es una herramienta útil para la selección del umbral en imágenes que contienen objetos obscuros sobre un fondo claro [15]. Los pixeles obscuros dentro del objeto, producirán un pico en la parte izquierda del histograma de la imagen y los pixeles de fondo producirán un pico en la parte derecha, tal como se muestra en la figura 3.3.

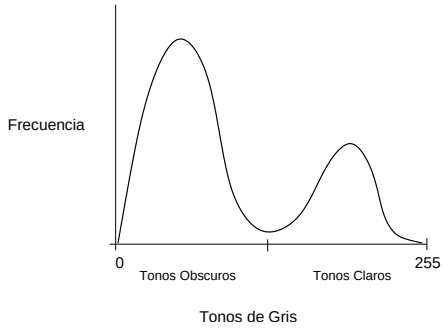


Figura 3.3: Histograma en tonos de gris de una imagen con objetos obscuros sobre fondo claro.

Los relativamente pocos pixeles de tonos medios de gris que se encuentran alrededor del objeto producen el valle entre los dos picos. Un valor de umbral elegido en el área del valle producirá un límite razonable para el proceso de conversión de la imagen en RGB a binaria. En la figura 3.4 se presenta el histograma en tonos de gris de la imagen de la figura 3.2(b).

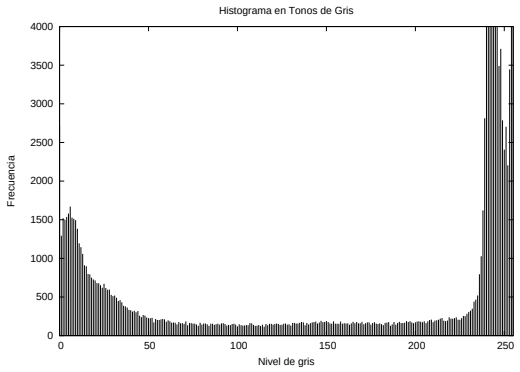


Figura 3.4: Histograma en tonos de gris de la imagen de la figura 3.2(b).

Imagen binaria

Una vez que se ha determinado un valor de umbral, para obtener la imagen binaria simplemente debe leerse cada pixel de la imagen RGB, aplicarle la fórmula de la ecuación 3.1 y si el resultado es menor o igual al umbral, entonces en la imagen binaria debe grabarse un pixel negro, en caso contrario, un pixel blanco. La figura 3.5 muestra el resultado de la conversión de la imagen de la figura 3.2(b).

(a) (b)

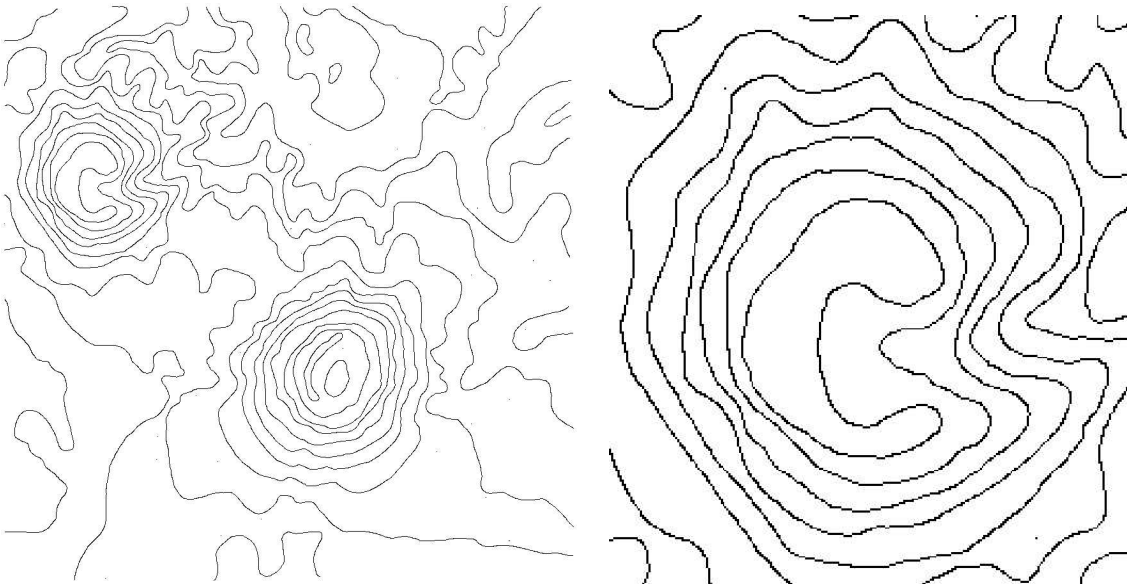
Figura 3.5: (a) Imagen binaria del terreno de la figura 3.2, (b) Acercamiento a uno de los cerros.

3.1.2. Adelgazamiento de las líneas de contorno

Como puede observarse en la figura 3.5(b), las líneas de contorno de la imagen son muy gruesas (de dos a cuatro pixeles) y necesitan ser adelgazadas al grueso de un pixel, para quedarse exclusivamente con la secuencia de pixeles que describen el contorno.

Se ha empleado un algoritmo de adelgazamiento de regiones, el cual se aplica sobre imágenes binarias y funciona desgastando el contorno de las regiones de pixeles correspondientes a los objetos hasta dejarlas al grueso de un pixel. En este trabajo se utilizó la biblioteca *scimagen* [21], donde se encuentra programado este algoritmo.

En la figura 3.6 se presenta la imagen de las líneas de contorno adelgazadas de la imagen binaria mostrada en la figura 3.5(b).



(a) (b)

Figura 3.6: (a) Líneas de contorno adelgazadas del terreno de la figura 3.2, (b) Acercamiento a uno de los cerros.

3.1.3. Extracción de los pixeles por línea de contorno

La extracción de los pixeles que forman cada línea de contorno se realiza por rastreo, esto es, se hace un recorrido por renglones de la imagen de las líneas de contorno adelgazadas y a partir de donde se encuentra el primer pixel característico, se hace una búsqueda en la vecindad de éste para hallar el siguiente pixel de la línea de contorno. Esto es equivalente a encontrar las componentes conexas de un grafo. El proceso se repite con cada pixel hallado hasta terminar con todos los pixeles del contorno. Cada pixel se elimina de la imagen después de haber registrado su posición. En el procedimiento 2 se presenta el algoritmo con el que se realiza este proceso, el cual entrega a la salida una lista de líneas de contorno $\mathcal{L}$ (una matriz) con i líneas y j pixeles por cada línea. La expresión $\mathcal{L}_{i,j}$ se usa para referir al pixel j de la línea i.

El algoritmo funciona como sigue: una vez que se encuentra un pixel que pertenece a un contorno, éste se agrega a la lista de pixeles que conforma la lista de contornos. Después, con la función *checa_siguiente_punto()*, se verifican tres cosas:

- si el pixel es terminal (ramas = 0),
- si a partir de ese pixel sólo hay un camino o rama por recorrer (ramas = 1) ó

Procedimiento 2 Extracción de las líneas de contorno

Entrada: La imagen binaria de las líneas de contorno adelgazadas.
Salida: La lista de líneas de contorno $\mathcal{L}$ con i líneas y j pixeles por cada una.

```
i ← 0 {Contador del número de contornos}
for cada pixel p en la imagen de entrada do
  if p es negro then {pixel de una línea}
    j ← 0
  else
    continue
  termina_contorno ← FALSE
  flag ← TRUE
  repeat
    ramas ← checa_siguiente_punto( p )
    if flag = TRUE then
      L_{i,j} ← p
      j++
    borro p {Se pone a blanco}
    if ramas = 0 then
      if pila_vacia( ) then
        termina_contorno ← TRUE
      else
        p ← pop( )
        flag ← FALSE
    else if ramas = 1 then
      p ← ir_al_siguiente_punto( p )
    else
      push( p )
      p ← ir_al_siguiente_punto( p )
  until termina_contorno = TRUE
  i++
```

- si a partir de ese pixel puede recorrerse más de un camino o rama (ramas > 1)

Para realizar esto, en la función *checa_siguiente_punto()* se toma de la imagen una matriz de 3×3 con el punto a verificar en el centro de ella, a la cual llamaremos *matriz de región*. El número de caminos que pueden seguirse a partir del punto central de la matriz está dado por el número de transiciones de pixel blanco a pixel negro cuando la matriz se recorre de la casilla uno a la ocho en el orden que se muestra en la figura 3.7(a). En la figura 3.7(b) se presenta, a modo de ejemplo, un conjunto de pixeles para los que debe hallarse la secuencia que llevan; y en la figura 3.7(c) la secuencia que siguen, representada por la línea trazada sobre los pixeles.

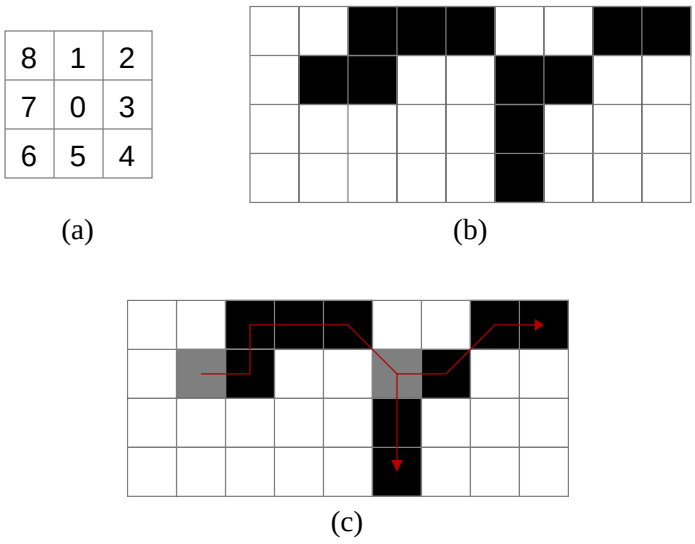


Figura 3.7: (a) Matriz que representa el orden que debe seguirse en la verificación del número de caminos que pueden recorrerse a partir de un pixel. (b) Conjunto de pixel para el que se debe hallar la secuencia que llevan. (c) Secuencia de los pixeles. Los pixeles grises representan el inicio de la trayectoria y el inicio de una ramificación, respectivamente.

Tomemos los pixeles grises en la figura 3.7(c) como ejemplo para la verificación del número de caminos que pueden seguirse a partir de ellos. Al pixel que se encuentra más a la izquierda lo llamaremos *pixel uno* y al otro *pixel dos*. En el caso del *pixel uno*, el número de transiciones de pixel blanco a pixel negro es sólo una, de acuerdo a la verificación de la matriz de la figura 3.7(a), y representa el número de caminos que pueden seguirse a partir de ese pixel, pues, si bien tiene dos posibles caminos por seguir (a su derecha o en diagonal), sólo el camino hacia su derecha es óptimo, por que si se siguiera el camino en diagonal, el pixel a su derecha tendría que almacenarse en una pila para registrarlo posteriormente, y la secuencia de los pixeles se alteraría. Finalmente el pixel verificado se borra de la imagen.

Si el pixel actual sólo tiene un camino posible por recorrer, el nuevo pixel simplemente será el que entregue la función *ir_al_siguiente_punto()*; para realizar esta función debe tomarse en cuenta que el algoritmo de adelgazamiento empleado en la sección 3.1.2, entrega los contornos 4-conexos [13], entonces tiene preferencia un pixel que sea 4-vecino a uno que sea d-vecino (en la diagonal). Por lo tanto, en la función *ir_al_siguiente_punto()* se verifica en primer lugar la 4-vecindad del pixel en su *matriz de región*, y en caso de hallarse un pixel negro allí, ese pixel será el siguiente en la secuencia. Si no hay pixeles en la 4-vecindad, entonces se busca en la d-vecindad.

Si el pixel tiene ramificaciones, entonces se coloca en una *pila* y como él ya forma parte de la lista, se pone una bandera para que en la siguiente iteración no se agregue

de nuevo en la lista de contornos, pero sí para que se revisen sus vecinos.

Eliminación de ruido

En la figura 3.6 pueden notarse algunas manchas pequeñas, es decir, regiones de pixeles negros que no pertenecen a ningún contorno. Estos pixels pueden considerarse como ruido en la imagen inicial (imagen RGB) que pasó a la imagen binaria en el proceso de conversión. Para remover el ruido, después de la extracción de las líneas de contorno de la imagen, sólo se descartan las líneas de menor longitud (número de pixeles).

La salida del programa que extrae las líneas de contorno y elimina el ruido, es un archivo que contiene las dimensiones de la imagen, el número de líneas de contorno, el número de pixeles en cada una y la lista de las coordenadas de los pixeles. El formato del archivo se presenta en la primera columna del cuadro 3.1, en la segunda columna se presenta la explicación del formato; lo mismo ocurre en el resto del capítulo.

$rens$	$cols$	Dimensiones de la imagen: número de filas y columnas
n		Número de líneas de contorno
p_1		Número de puntos en la línea de contorno 1
x_{11}	y_{11}	Lista de puntos (pixeles) que conforman a la línea de contorno 1
x_{12}	y_{12}	
$\vdots$	$\vdots$	
x_{1p_1}	y_{1p_1}	Fin de la lista de puntos para el contorno 1
p_2		Número de puntos en la línea de contorno 2
x_{21}	y_{21}	Lista de puntos (pixeles) que conforman a la línea de contorno 2
x_{22}	y_{22}	
$\vdots$	$\vdots$	
x_{2p_2}	y_{2p_2}	Fin de la lista de puntos para el contorno 2
$\vdots$		
p_n		Número de puntos en la línea de contorno n
x_{n1}	y_{n1}	Lista de puntos (pixeles) que conforman a la línea de contorno n
x_{n2}	y_{n2}	
$\vdots$	$\vdots$	
x_{np_2}	y_{np_2}	Fin de la lista de puntos para el contorno n

Cuadro 3.1: La primera columna es el formato del archivo con la información de las líneas de contorno, la segunda columna es la explicación del formato.

3.2. Muestreo

Después de haber obtenido la secuencia de pixeles que forman cada contorno, debe hacerse un muestreo sobre el conjunto de pixeles; pues basta con tener un subconjunto de puntos muestreados con una densidad adecuada, para obtener una reconstrucción poligonal de cada contorno, tal como se mencionó en la sección 2.5.

El muestreo que se realice debe cumplir con ciertas condiciones, pues si bien, ciertamente es imposible construir un algoritmo que reconstruya cualquier curva a partir de cualquier conjunto de muestras; se necesita que este conjunto cumpla con ciertas características de calidad. La condición de muestreo que aquí se sigue, es la que se planteó en la sección 2.5.1, en donde básicamente se propone que cada punto p que sea parte de la curva debe guardar una distancia máxima de $r \times TCL(p)$ con alguna muestra s del conjunto de muestreo, donde $r \leq 1$ y la función $TCL(p)$ es la distancia euclidiana del punto p al eje medio de la curva. De acuerdo al trabajo [22], un límite adecuado para los valores que puede tomar r es 0.42, con lo cual se garantiza que en la reconstrucción poligonal de las curvas muestreadas no se introduzcan aristas indeseables. El valor para r que hemos empleado en este trabajo es 0.415.

Así pues, para calcular el conjunto de muestreo de las curvas de contorno, en primer lugar se requiere calcular el eje medio de éstas, después calcular la función TCL y finalmente muestrear.

3.2.1. Cálculo del eje medio

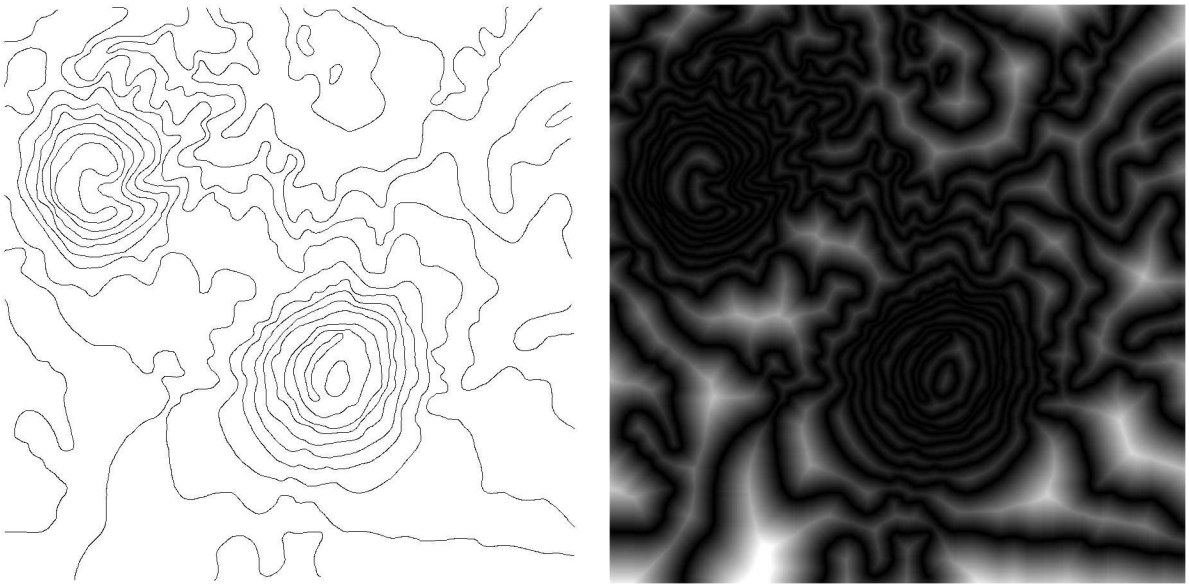
El eje medio de las curvas de contorno se ha obtenido utilizando los mapas de distancias. Para el cálculo de estos mapas se ha empleado el Chamfer 3-4 descrito en la sección 2.3.2.

Mapa de distancias

Puesto que el eje medio de una curva puede verse como la región de los puntos que tienen al menos dos puntos más cercanos a la curva (sección 2.4) y, en un mapa de distancias el valor de cada pixel representa la distancia que hay de él hacia el pixel característico más cercano, entonces es posible producir la imagen del eje medio de los contornos a partir de su mapa de distancias.

Se calcula entonces el mapa de distancias de la imagen binaria de las líneas de contorno adelgazadas. En la figura 3.8 se presenta el ejemplo de un mapa de distancias, el cual es una imagen en tonos de gris. Recordemos que en una imagen el tonos de gris más obscuro se representa con el valor de 0, y el tono más claro con 255. Es por esta razón que en la imagen del mapa de distancias las regiones más cercanas a

los pixeles característicos los tonos de gris son más obscuros, y en las regiones más alejadas los pixeles son más claros, como puede notarse en la figura.



(a) (b)

Figura 3.8: Imagen binaria de las líneas de contorno adelgazadas y su mapa de distancias

De esta manera, el eje medio de las curvas de contorno se obtiene de su mapa de distancias al extraer los puntos máximos locales, esto es, los puntos más claros, que son los puntos que guardan una distancia máxima con respecto a alguno de los pixeles que forman parte un contorno.

Extracción del eje medio

Para extraer los puntos máximos de la imagen del mapa de distancias, se ha usado un filtro pasa altas [13]. En terminología de imágenes digitales, un filtro es una máscara que se pasa sobre cada pixel de la imagen y para cada uno de ellos devuelve un valor. La salida del filtro se calcula como sigue:

$$R = \sum_{i=1}^{n} c_i p_i$$

donde c_i es el valor del coeficiente i de la máscara, p_i el nivel de gris del pixel debajo de la máscara en la posición i, y n el número de localidades de la máscara.

Los filtros pasa altas deben tener coeficientes positivos cerca del centro y coeficientes negativos en la periferia. La máscara que hemos usado es la que se muestra en la figura 3.9. Nótese que la suma se sus coeficientes es cero. Así, cuando la máscara sea pasada sobre un área donde haya niveles de gris constantes o poco variables, la salida será cero o un valor pequeño.

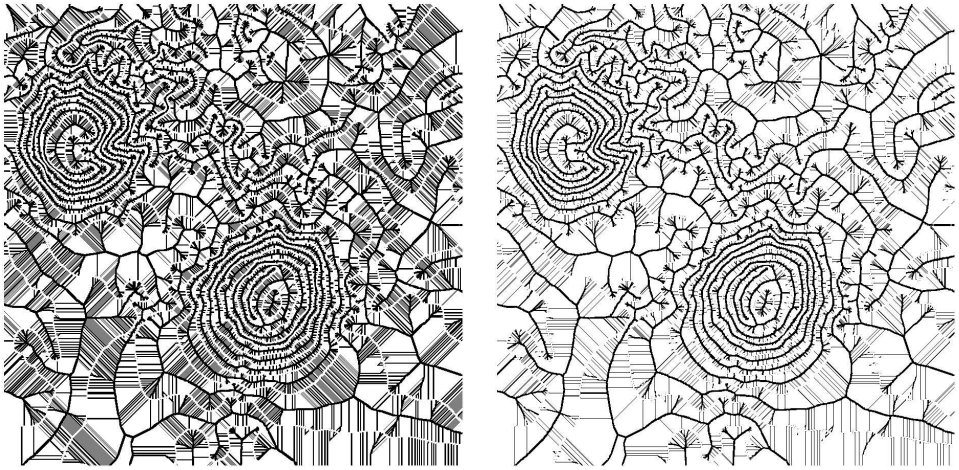
−1	−1	−1	−1	−1
−1	1	1	1	−1
−1	1	8	1	−1
−1	1	1	1	−1
−1	−1	−1	−1	−1

Figura 3.9: Máscara usada para la detección de máximos en un mapa de distancias.

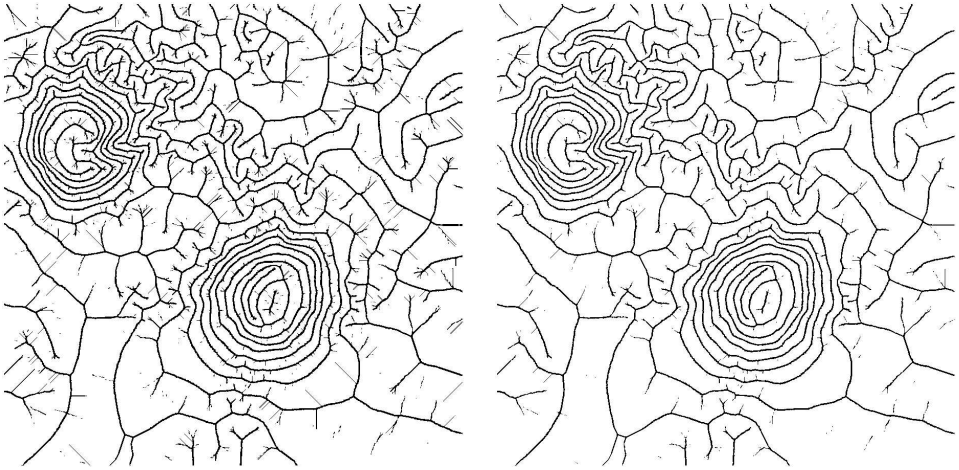
Entonces, se pasa el filtro sobre la imagen del mapa de distancias, y si la salida del filtro para el pixel i es mayor que determinado valor de umbral, entonces en la posición i de la imagen de salida se escribe un pixel negro, o uno blanco en caso contrario. En la figura 3.10 se muestran ejemplos de imágenes obtenidas después de filtrar la imágen del mapa de distancias con diferentes umbrales.

Para obtener una buena aproximación del eje medio, el valor de umbral empleado puede variar dependiendo de cada imagen, sin embargo, de acuerdo a las pruebas realizadas, encontramos que el empleo un umbral entre 20 y 25 produce una imagen del eje medio admisible.

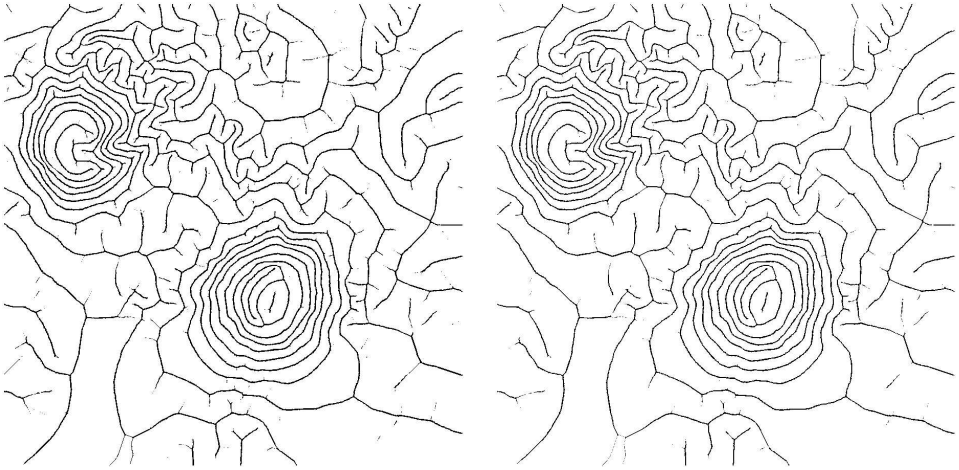
Es importante notar que las líneas del eje medio obtenido son sólo una buena aproximación al eje medio exacto, pues las líneas no son continuas en todos los puntos, y no son del grueso de un pixel. Sin embargo, esto no tiene repercusiones significativas, pues lo que se requiere del eje medio es conocer la distancia de los pixeles de la curva a él, con el fin de determinar qué tan separadas deben estar las muestras una de otra. Ahora bien, no se requiere una presición tan exacta en la distancia entre las muestras, sólo se necesita que no sea mayor que $2(r \times TCL(p))$, y puesto que el grueso de las líneas del eje medio es mayor a un pixel, la distancia que se calcule de la curva al eje medio será relativamente menor a la real, de manera que el error presente no afecta la condición de separación entre las muestras. Es también por esta razón que para el cálculo del mapa de distancias se eligió emplear el Chamfer 3-4 aunque presenta un error del 8 % (ver sección 2.3.2), en lugar del Chamfer 5-7-11 con un error del 2 %.



(a) umbral = 0 (b) umbral = 5



(c) umbral = 10 (d) umbral = 15



(e) umbral = 20 (f) umbral = 25

Figura 3.10: Imágenes del eje medio extraídas del mapa de distancias de la figura 3.8, usando diferentes umbrales.

12	11	10	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	10	11	12
11	8	7	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	7	8	11
10	7	4	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	4	7	10
9	6	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	6	9
9	6	3	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	0	3	6	9
9	6	3	0	3	6	6	6	6	6	6	6	6	6	6	6	6	3	0	3	6	9
9	6	3	0	3	6	9	9	9	9	9	9	9	9	9	9	6	3	0	3	6	9
9	6	3	0	3	6	9	12	12	12	12	12	12	12	12	9	6	3	0	3	6	9
9	6	3	0	3	6	9	12	15	15	15	15	15	15	12	9	6	3	0	3	6	9
9	6	3	0	3	6	9	12	15	18	18	18	18	15	12	9	6	3	0	3	6	9
9	6	3	0	3	6	9	12	15	15	15	15	15	15	12	9	6	3	0	3	6	9
9	6	3	0	3	6	9	12	12	12	12	12	12	12	12	9	6	3	0	3	6	9
9	6	3	0	3	6	9	9	9	9	9	9	9	9	9	9	6	3	0	3	6	9
9	6	3	0	3	6	6	6	6	6	6	6	6	6	6	6	6	3	0	3	6	9
9	6	3	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	0	3	6	9
9	6	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	6	9
10	7	4	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	4	7	10
11	8	7	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	7	8	11
12	11	10	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	10	11	12

Figura 3.11: Ejemplo numérico del mapa de distancias de la imagen de un rectángulo.

3.2.2. Cálculo del TCL

Hasta este punto, se ha obtenido la imagen del eje medio de las curvas de contorno. Para realizar el muestreo se requiere encontrar la distancia de cada punto en la curva hacia su eje medio, en otras palabras, se requiere conocer el TCL de cada punto sobre la curva.

Si calculamos el mapa de distancias de la imagen del eje medio de las curvas, obtendremos resultados semejantes a los de la sección anterior, excepto que las regiones más claras de la imagen que aquí se obtenga, correspodenderán con las regiones de los pixeles que definen a las curvas.

Considérese el ejemplo de la figura 3.11, en donde se muestra el mapa de distancias numérico de la imagen de un rectángulo. La región de 0's define al rectángulo, porque la distancia al pixel característico más cercano a cada uno de los pixeles en esa región es cero, pues ellos mismo son los pixeles característicos. El resto de los números representan la distancia de cada pixel en la imagen hacia el pixel característico más cercano. Lo mismo ocurre con el mapa de la figura 3.13, el cual es el mapa de distancias del eje medio del rectángulo. La sección resaltada represeta la región de los pixel del rectángulo del que se obtuvo el eje medio.

16	15	14	13	12	13	14	15	16	19	20	20	19	16	15	14	13	12	13	14	15	16
15	12	11	10	9	10	11	12	15	16	19	19	16	15	12	11	10	9	10	11	12	15
14	11	8	7	6	7	8	11	12	15	16	16	15	12	11	8	7	6	7	8	11	14
13	10	7	4	3	4	7	8	11	12	15	15	12	11	8	7	4	3	4	7	10	13
12	9	6	3	0	3	4	7	8	11	12	12	11	8	7	4	3	0	3	6	9	12
13	10	7	4	3	0	3	4	7	8	11	11	8	7	4	3	0	3	4	7	10	13
14	11	8	7	4	3	0	3	4	7	8	8	7	4	3	0	3	4	7	8	11	14
15	12	11	8	7	4	3	0	3	4	6	6	4	3	0	3	4	7	8	11	12	15
16	15	12	11	8	7	4	3	0	3	3	3	3	0	3	4	7	8	11	12	15	16
19	16	15	12	11	8	7	4	3	0	0	0	0	3	4	7	8	11	12	15	16	19
16	15	12	11	8	7	4	3	0	3	3	3	3	0	3	4	7	8	11	12	15	16
15	12	11	8	7	4	3	0	3	4	6	6	4	3	0	3	4	7	8	11	12	15
14	11	8	7	4	3	0	3	4	7	8	8	7	4	3	0	3	4	7	8	11	14
13	10	7	4	3	0	3	4	7	8	11	11	8	7	4	3	0	3	4	7	10	13
12	9	6	3	0	3	4	7	8	11	12	12	11	8	7	4	3	0	3	6	9	12
13	10	7	4	3	4	7	8	11	12	15	15	12	11	8	7	4	3	4	7	10	13
14	11	8	7	6	7	8	11	12	15	16	16	15	12	11	8	7	6	7	8	11	14
15	12	11	10	9	10	11	12	15	16	19	19	16	15	12	11	10	9	10	11	12	15
16	15	14	13	12	13	14	15	16	19	20	20	19	16	15	14	13	12	13	14	15	16

Figura 3.12: Mapa de distancias numérico del eje medio del rectángulo de la figura 3.11. La región resaltada representa la región de los pixeles que conforman el rectángulo.

Por lo dicho anteriormente, el mapa de distancias del eje medio de una curva puede ser usado como una tabla de consulta para el TCL de los puntos en la curva, basta con consultar en el mapa la posición del pixel de la curva deseado para obtener la distancia de él hacia el punto más cercano en el eje medio. En la figura 3.13 se presenta la imagen del mapa de distancias del eje medio de la figura 3.10(f).

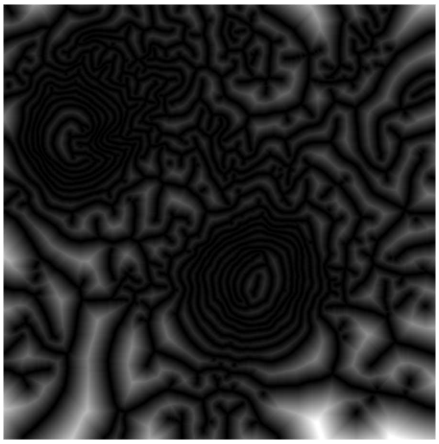


Figura 3.13: Imagen del mapa de distancias del eje medio de la figura 3.10(f).

3.2.3. Cálculo del conjunto de muestreo

El muestreo de las líneas de contorno se realizó empleando el algoritmo que se presenta en el procedimiento 3, el cual toma como uno de sus parámetros de entrada la lista de listas, $\mathcal{L}$, de los pixeles que conforman a los contornos; entonces con la expresión $\mathcal{L}_{i,j}$ se hace referencia al pixel j del contorno i. A la salida entrega también una lista de listas, $\mathcal{R}$, de los pixeles que conforman el conjunto de muestreo de cada contorno; al igual que en $\mathcal{L}$, la expresión $\mathcal{R}_{i,k}$ hace referencia al punto k del conjunto de muestreo del contorno i.

Procedimiento 3 Muestreo de las líneas de contorno

Entrada: La lista de los pixeles que forman los contornos, $\mathcal{L}$, con i contornos y j pixeles por contorno, la tabla de consulta del TCL de los puntos de las líneas de contorno a muestrear y el factor de muestreo $r \leq 1$.

Salida: La lista del conjunto de puntos r-muestreados, $\mathcal{R}$, con k puntos muestreados por cada contorno i.

```
for cada contorno i do
    k ← 0 {Contador de muestras por contorno}
    p ← L_{i,0} {primer punto del contorno i}
    R_{i,k} ← p
    for cada punto j en L_i, j > 0 do
        p_medio ← L_{i,j}
        medida ← r * TCL(p_medio)
        d ← distanciaEuclidiana( p, p_medio )
        if d ≥ medida then
            d ← 0
            while d ≤ medida y haya puntos en el contorno do
                j ← j + 1
                p ← pixel L_{i,j}
                d ← distanciaEuclidiana( p_medio, p )
            k ← k + 1
            R_{i,k} ← p
```

Lo que realiza el algoritmo es lo siguiente: el primer punto de un contorno pertenecerá a su conjunto de puntos de muestreo. Para los puntos siguientes, se obtiene la *medida característica* del punto actual ($r * \mathrm{TCL}(p)$) y la distancia euclidiana entre la muestra anterior y el punto en cuestión. Si la distancia es mayor o igual a la medida del punto, entonces se ha encontrado un punto, p_{medio}, aproximadamente situado a la mitad entre dos muestras, y se procede a calcular la distancia euclidiana entre p_{medio} y los puntos que le siguen a él con la finalidad de hallar el punto cuya distancia con el punto p_{medio} sea mayor o igual a la *medida característica* de p_{medio}. Tal punto será el

siguiente en el conjunto de muestreo. El proceso se repite hasta terminar todos los contornos.

Al programa donde se implementó el algoritmo de muestreo se le pasa como argumento el archivo que contiene la lista de los pixeles, cuyo formato se presentó en el cuadro 3.1, y la imagen del mapa de distancias del eje medio de las líneas de contorno. Como salida, escribe un archivo que contiene la lista del conjunto de puntos muestreados, con el formato que se presenta en el cuadro 3.2, y también se obtiene la imagen de las líneas muestreadas como la que se muestra en la figura 3.14, en donde puede observarse que en las regiones de la curva con mayor detalle, la distancia entre las muestras en menor que en las regiones de menor detalle.

Nótese que algunos de los datos almacenados en el archivo del conjunto de puntos son la altura del contorno, el color por defecto y una indicación para determinar si el contorno es una curva abierta o cerrada. La utilidad de estos datos se explicará en los capítulos posteriores. El programa escribe por defecto en el archivo un valor de 0 para la alturas de cada contorno, y color (142, 0, 0).

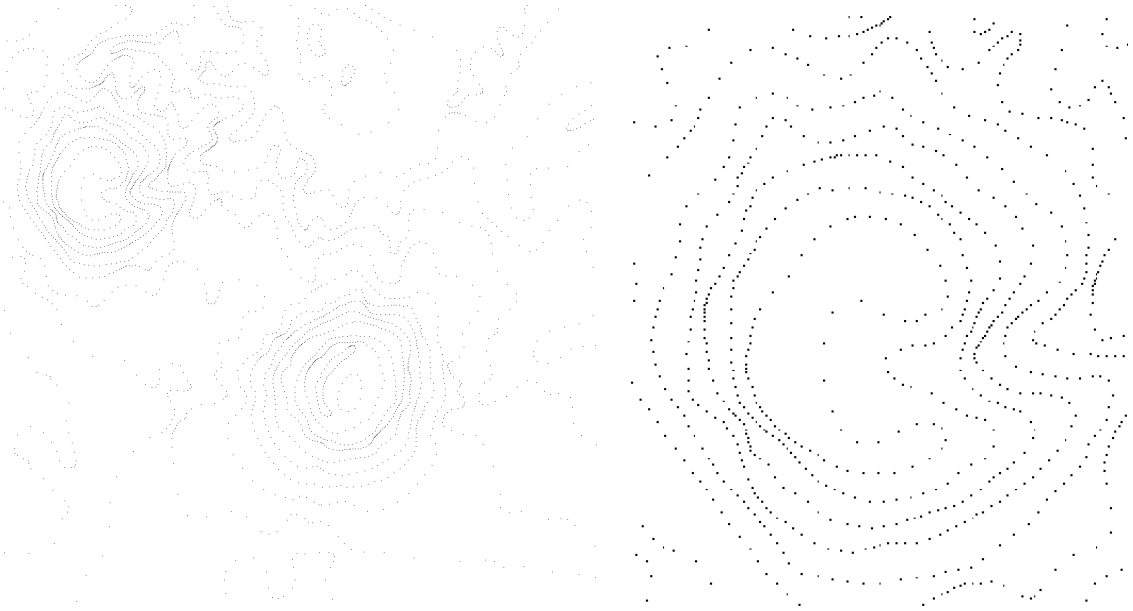


Figura 3.14: Imagen del conjunto de muestreo las líneas de contorno de la figura 3.8(a), y un acercamiento.

max			Valor máximo entre el número de filas y el número de columnas de la imagen
n			Número de líneas de contorno
p_1			Número de puntos en el conjunto de muestreo del contorno 1
h_1			Altura del contorno 1
R	G	B	Color por defecto de cada contorno
x_{11}	y_{11}		Lista de los puntos del 1er. conjunto de muestreo
x_{12}	y_{12}		
$\vdots$	$\vdots$		
x_{1p_1}	y_{1p_1}		Fin de la lista
ac_1			Indicador de que la curva de contorno 1 es abierta o cerrada, 1 o 0 respectivamente.
p_2			Número de puntos en el conjunto de muestreo del contorno 2
h_2			Altura del contorno 2
R	G	B	Color por defecto de cada contorno
x_{21}	y_{21}		Lista de los puntos del 2do. conjunto de muestreo
x_{22}	y_{22}		
$\vdots$	$\vdots$		
x_{2p_1}	y_{2p_1}		Fin de la lista
ac_2			Indicador de que la curva de contorno 2 es abierta o cerrada, 1 o 0 respectivamente.
$\vdots$			
p_n			Número de puntos en el conjunto de muestreo del contorno n
h_n			Altura del contorno n
R	G	B	Color por defecto de cada contorno
x_{n1}	y_{n1}		Lista de los puntos del n-ésimo conjunto de muestreo
x_{n2}	y_{n2}		
$\vdots$	$\vdots$		
x_{np_n}	y_{np_n}		Fin de la lista
ac_n			Indicador de que la curva de contorno n es abierta o cerrada, 1 o 0 respectivamente.

Cuadro 3.2: Formato del archivo con la información del conjunto de muestreo de cada contorno.

Capítulo 4

Interpolación de Contornos

Una vez que se cuenta con el conjunto de puntos que define a cada contorno, la forma más simple de realizar la reconstrucción del terreno es asignando alturas a cada contorno y realizar la triangulación de Delaunay de todos ellos. Sin embargo, esto resulta en una reconstrucción de mala calidad, debido a la aparición de *triángulos planos* y *triángulos grandes* entre los contornos, lo que provoca que en la visualización del terreno los cambios entre contornos no sean suaves.

Basándonos en el trabajo [23], hemos empleado una metodología para la reconstrucción de terrenos a patir del conjunto de puntos que definen los contornos, la cual está basada en las construcciones geométricas de la corteza y el esqueleto [16] sobre el conjunto de puntos. En forma general, se trata de agregar los puntos del esqueleto al conjunto de puntos de los contornos y construir una malla de triángulos sobre el conjunto aumentado.

La metodología empleada es la siguiente:

1. Los puntos muestreados de las líneas de contorno forman la *corteza*.
2. Se calcula el esqueleto.
3. Se reduce el número de puntos que conforman el esqueleto.
4. Se organizan los puntos del esqueleto en grafos.
5. Se realiza la triangulación de Delaunay del conjunto total de puntos (corteza y esqueleto) para obtener un modelo de la superficie del terreno.
6. Se asignan alturas a cada vértice.
7. Se visualiza el terreno.

En este capítulo se tratan los puntos del 1 al 6. El punto 7 se tratará en el siguiente capítulo.

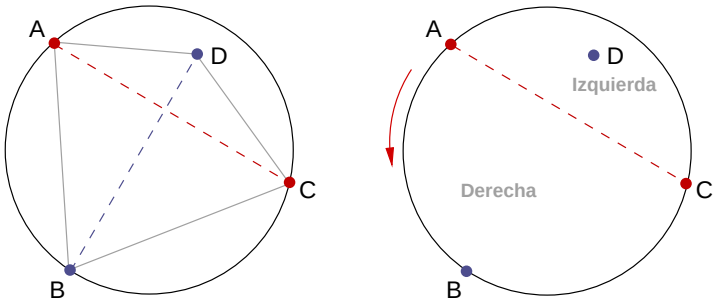


Figura 4.1: Prueba *DentroDelCírculo*.

4.1. Corteza y esqueleto

Como se mencionó en el primer punto de la metodología, el conjunto de puntos muestreados forman la corteza, pues los puntos se encuentran ordenados en secuencia según se fueron obteniendo al rastrear las líneas de contorno en la imagen binaria, tal y como se explicó en la sección 3.1.3 en la página 26. Entonces, basta con formar una arista entre cada dos puntos consecutivos del conjunto de muestreo para obtener la construcción de la corteza, esto es, unir el punto uno con el dos, el dos con el tres, el tres con el cuatro y así sucesivamente.

El cálculo del esqueleto se realizó de acuerdo al procedimiento descrito en [22], para lo cual se requiere de una prueba estándard llamada prueba *DentroDelCírculo* [24].

4.1.1. Prueba *DentroDelCírculo*

Esta prueba es aplicada a cuatro puntos en el plano, como se muestra en la figura 4.1(a). El predicado *DentroDelCírculo*(A, B, C, D) se define como verdadero si y sólo si el punto D cae dentro de la región del plano delimitada por el círculo que pasa por los puntos A, B y C, y además cae a la izquierda de dicho círculo cuando éste es recorrido en la dirección de ABC, como se ilustra en la figura 4.1(b).

En particular, esto implica que D debe caer dentro del círculo ABC si los puntos A, B y C definen un triángulo orientado en el sentido contrario a las manecillas del reloj, y cae fuera si definen un círculo orientado en el sentido de las manecillas del reloj. En caso de que A, B y C sean colineales, se interpreta la línea como un círculo añadiendo un punto al infinito. Si A, B, C y D son cocirculares, el predicado se interpreta como falso [24].

La forma que empleamos para hacer esta prueba está basada en las coordenadas

de los puntos:

$$DentroDelCirculo(A,B,C,D) = \begin{vmatrix} 1 & x_A & y_A & x_A^2+y_A^2 \\ 1 & x_B & y_B & x_B^2+y_B^2 \\ 1 & x_C & y_C & x_C^2+y_C^2 \\ 1 & x_D & y_D & x_D^2+y_D^2 \end{vmatrix} > 0 \qquad (4.1)$$

donde $A = (x_A, y_A)$, $B = (x_B, y_B)$, $C = (x_C, y_C)$ y $D = (x_D, y_D)$. La prueba de esta aseveración se encuentra en [24].

La prueba *DentroDelCírculo* posee propiedades interesantes que deben tomarse en consideración. Si A, B, C y D definen un cuadrilátero orientado en el sentido contrario a las manecillas del reloj y *DentroDelCírculo*(A, B, C, D) es verdadero, entonces *DentroDelCírculo*(C, B, A, D) es falso, de manera que invertir la orientación, invierte el valor del predicado. Esta propiedad no es conveniente para nuestros propósitos, de manera que no es suficiente aplicar la ecuación 4.1, sino que además es necesario determinar el sentido de los puntos. Para resolver este problema, debe evaluarse el determinante de 3×3

$$\Delta = \begin{vmatrix} x_A & y_A & 1 \\ x_B & y_B & 1 \\ x_C & y_C & 1 \end{vmatrix}$$

el cual corresponde al doble del área con signo del triángulo ABC, donde el signo es positivo si y sólo si ABC forman un ciclo en el sentido contrario a las manecillas del reloj [18].

Entonces, cuatro puntos en el plano, A, B, C y D, cumplen con la prueba dentro del círculo si cumplen con alguna de las condiciones establecidas en el cuadro 4.1.

Orden de recorrido	**Sentido contrario a las manecillas del reloj**	**Prueba** *DentroDelCírculo*
$ABCD$	Verdadero	Verdadero
	Falso	Falso
$CBAD$	Verdadero	Verdadero
	Falso	Falso

Cuadro 4.1: Condiciones que deben cumplir los puntos en el plano, A, B, C y D, para pasar la prueba dentro del círculo.

4.1.2. Cálculo del diagrama de Voronoi y la triangulación de Delaunay

Para calcular el esqueleto, primeramente se requiere calcular el diagrama de Voronoi y la triangulación de Delaunay, para lo cual se empleó el paquete *qhull* [20]. Qhull es código de dimensión general para el cálculo de cubiertas convexas, triangulaciones de Delaunay, diagramas de Voronoi, entre otras. Entre el conjunto de programas que conforman qhull, hay dos que son de nuestro interés: *qdelaunay* y *qvoronoi*, los cuales sirven para calcular la triangulación de Delaunay y el diagrama de Voronoi, respectivamente.

Para cualquiera de las dos construcciones que se desee calcular, el programa requiere un archivo de entrada como el que se muestra en el cuadro 4.2. La lista de puntos que ahí se presentan, corresponde al conjunto de puntos de muestreo tomado a partir de una elipse, la cual cumple con la definición de *curva suave* que se presentó en la sección 2.5 en la página 16. Este conjunto cumple con las condiciones de muestreo previamente establecidas. En el resto del capítulo llamaremos *conjunto E* a este conjunto de puntos.

2		Dimensión de los puntos
14		Número de puntos
44	25	Lista de puntos
59	26	
72	31	
82	40	
84	53	
78	64	
67	71	
53	74	
39	73	
27	68	
17	59	
15	46	
21	35	
33	28	Fin de la lista

Cuadro 4.2: Ejemplo un archivo de entrada para el programa que calcula el diagrama de Voronoi y la triangulación de Delaunay del conjunto de puntos muestreados que definen una elipse (*conjunto E*).

Triangulación de Delaunay

Para calcular la triangulación de Delaunay se debe ejecutar una instrucción como la que sigue:

```
qdelaunay i < archivoEntrada.txt > Delaunay.txt
```

El archivo que entrega a la salida es como el que se muestra en el cuadro 4.3.

12			Número de triángulos generados
8	0	7	Lista de los índices de los vértices
0	1	7	que forman cada triángulo
9	10	11	
12	9	11	
2	3	4	
5	2	4	
6	2	5	
1	6	7	
2	6	1	
13	9	12	
8	13	0	
9	13	8	Fin de la lista

Cuadro 4.3: Archivo de salida del programa *qdelaunay* para el archivo de entrada del *conjunto E*.

En este archivo se encuentra la lista de triángulos generados, donde en cada línea se encuentran los índices, con respecto al archivo de entrada, de los vértices que forman cada triángulo. Por ejemplo, el primer triángulo está formado por los puntos 8, 0 y 7 del archivo de entrada, esto es, los puntos $(39, 73)$, $(44, 25)$ y $(53, 74)$.

Diagrama de Voronoi

Para calcular el diagrama de Voronoi, como ya se mencionó, se emplea el programa *qvoronoi*, el cual tiene una opción para generar además la lista de aristas duales con respecto a la triangulación de Delaunay. Se ejecuta una instrucción como la siguiente:

```
qvoronoi p Fv < archivoEntrada.txt > Voronoi.txt
```

Con las opciones *p* y *Fv* se le indica al programa que se desea obtener la lista de vértices de Voronoi, y la lista de aristas duales (aristas de Delaunay y aristas de Voronoi). El archivo de salida es como el que se muestra en el cuadro 4.4.

2	Dimensión de los puntos
12	Número de vértices de Voronoi
47.70384047267356 49.64623338257016	Lista de vértices
49.91735537190083 49.2396694214876	
34.45535714285715 49.66071428571428	
34.625 49.56818181818182	
64.54464285714286 49.33928571428572	
64.375 49.43181818181818	
61.35185185185185 49.98148148148148	
55.15596330275229 49.89449541284404	
57.42660550458716 49.49082568807339	
37.26027397260274 49.08904109589041	
44.7280701754386 49.33625730994152	
41.64705882352941 49.74705882352941	Fin de la lista de vértices
25	Número de correspondencias entre aristas de Voronoi y Delaunay
4 0 8 1 11	Lista de aristas duales
4 0 7 1 2	
4 0 1 0 2	
4 0 13 0 11	
⋮	
4 9 12 4 10	
4 9 13 10 12	
4 10 11 0 3	
4 11 12 0 4	
4 12 13 0 10	Fin de la lista de aristas duales

Cuadro 4.4: Archivo de salida del programa *qvoronoi* para el archivo del *conjunto E*.

En este archivo, el formato de las líneas en la lista de aristas duales es como sigue: el primer número simplemente es el número de índices, los siguientes dos números corresponden al los índices de los vértices de Delaunay del archivo de entrada que forman una arista, y los últimos números corresponden a los índices de los vértices de Voronoi listados posteriormente en el mismo archivo, los cuales forman la arista dual a la arista de Delaunay. Por ejemplo, la línea

```
4 0 8 1 11
```

indica que se listan 4 vértices, y que la arista de Delaunay que va del punto 0 al punto 8 del archivo de entrada, es dual con la arista de Voronoi que va del punto 1 al punto 11 en la lista de vértices de Voronoi, esto es, que hay un par de aristas duales D y V, con vértices $(44, 25)$ y $(39, 73)$, y $(47.703, 49.646)$ y $(44.728, 49.336)$, respectivamente.

Es importante remarcar que los índices de los vértices de Voronoi en la lista de aristas duales van desde 0 hasta 12, es decir, se han considerado 13 vértices de Voronoi cuando en la lista sólo se aparecen 12. La razón es que el vértice 0 es un vértice al infinito, considerado en qhull como $(-10.101, -10.101)$ y no incluído en la lista de vértices de Voronoi, entonces, la numeración de esta lista comienza en 1, y no en 0 como en la lista de vértices de Delaunay, la cual se encuentra en el archivo de entrada para el programa.

En la figura 4.2 se muestra una gráfica con los vértices de Delaunay marcados con $(+)$ y los vértices de Voronoi marcados con $(\times)$. En la figura 4.3 se muestran en líneas continuas las aristas de Delaunay y en líneas punteadas las aristas de Voronoi. Los números más grandes representan la numeración de acuerdo al archivo que se presentó en el cuadro 4.2, y los números más pequeños siguen la numeración de la lista de vértices de Voronoi según su aparición en el archivo presentado en el cuadro 4.4.

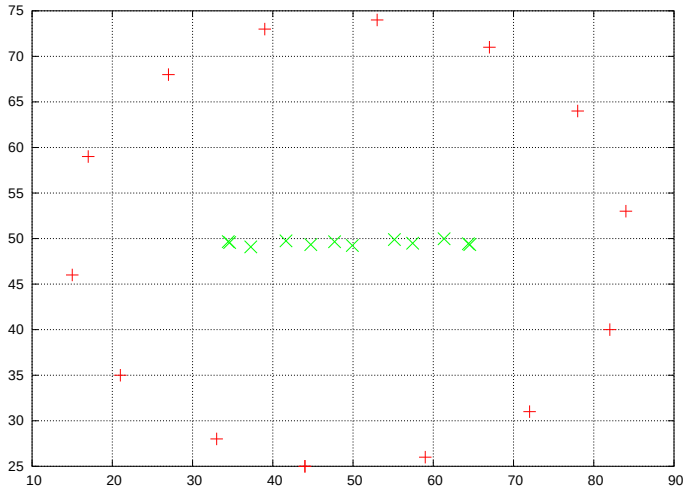


Figura 4.2: Gráfica de los vértices de Delaunay $(+)$ y de Voronoi $(\times)$ del *conjunto E*.

4.1.3. Cálculo del esqueleto

Una vez que se cuenta con las construcciones de la triangulación de Delaunay, el diagrama de Voronoi y la lista de aristas duales, se sigue el procedimiento 4 para hallar el esqueleto.

Lo que se hace en este algoritmo es que para cada arista de Delaunay (p, q) con su arista dual de Voronoi (r, s), se hace la "prueba dentro del círculo" sobre los cuatro puntos, en la cual se hacen las verificaciones establecidas en el cuadro 4.1. En esta prueba se determina si el punto s se encuentra dentro o fuera del círculo que pasa

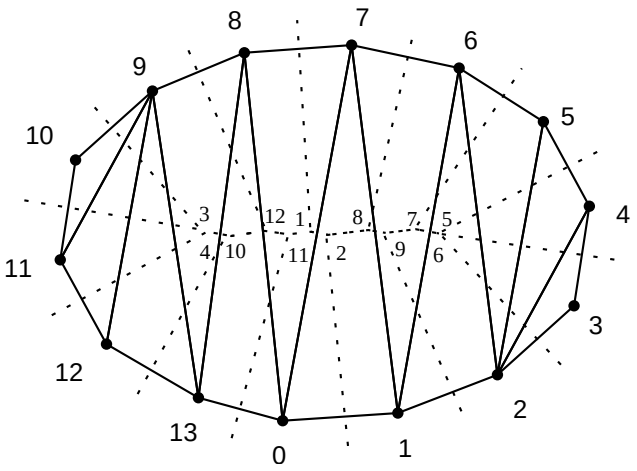


Figura 4.3: Las líneas continuas representan la gráfica de las aristas de Delaunay y las líneas punteadas la gráfica de las aristas de Voronoi del *conjunto E*.

Procedimiento 4 Cálculo de la corteza y el esqueleto

Entrada: Las listas de parejas de aristas duales: lista de aristas de Delaunay, $\mathcal{LD}$, y lista de aristas de Voronoi, $\mathcal{LV}$; n aristas en cada una de ellas. Las aristas de las listas son duales a pares.

Salida: La lista $\mathcal{LC}$ de c aristas que forman la corteza, y la lista $\mathcal{LE}$ de e aristas que forman el esqueleto.

$c \leftarrow 0$
$e \leftarrow 0$
for $i = 0$ hasta n **do**
  $(p, q) = \mathcal{LD}_i$
  $(r, s) = \mathcal{LV}_i$
  $verifica \leftarrow$ PruebaDentroDelCirculo(p, r, q, s)
  if $verifica$ =FUERA **then**
    $\mathcal{LC} \leftarrow \mathcal{LC} + (p, q)$
    $c \leftarrow c + 1$
  else
    $\mathcal{LE} \leftarrow \mathcal{LE} + (r, s)$
    $e \leftarrow e + 1$

por (p, r, q), asumiendo que este círculo está orientado en el sentido contrario a las manecillas del reloj, según se explicó en la sección 4.1.1.

Si la prueba indica que el punto se encuentra *fuera*, entonces ese círculo muestra que la *arista de Delaunay* (p, q) debe ser incluída en la *corteza*. Por otro lado, si s

cae *dentro*, entonces la *arista de Voronoi* (r, s) debe ser incluída en el *esqueleto*. En la figura 4.4 se muestra la gráfica de las aristas de la corteza y del esqueleto del conjunto de puntos que se ha venido tratando en el capítulo.

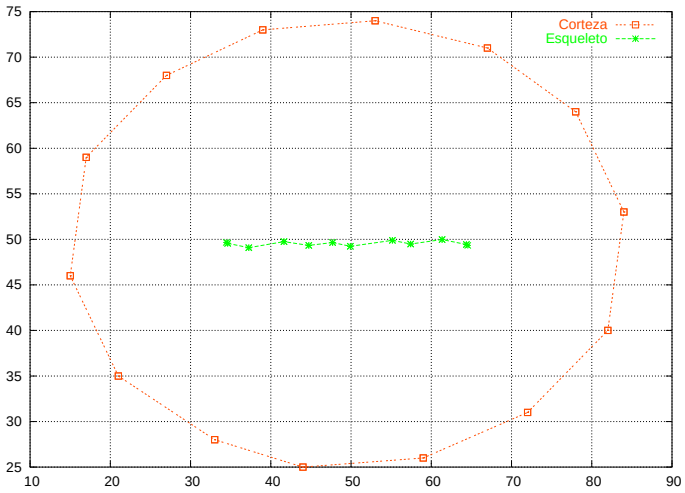


Figura 4.4: Gráfica de las aristas de la corteza y del esqueleto del *conjunto E*.

4.1.4. Reducción del número de puntos del esqueleto

Como se planteó en la metodología, para obtener el modelo de la superficie del terreno se realiza la triangulación del conjunto completo de puntos (corteza y esqueleto); evidentemente, mientras mayor sea el número de puntos mayor será el número de triángulos en el modelo. Ahora bien, ya se ha asegurado que la distancia entre los puntos de la corteza sea óptima con respecto al detalle local de las curvas de nivel; el inconveniente es que el conjunto de puntos del esqueleto no cumple con esta condición, pues algunos de los puntos se encuentran muy cercanos entre sí. Esto provoca un aumento innecesario de triángulos en zonas del terreno donde no se requiere.

En la gráfica de la figura 4.4 puede apreciarse que los puntos del esqueleto están más cercanos que los puntos de la corteza, como por ejemplo los puntos $(64.544, 49.339)$ y $(64.375, 49.431)$. Por esta razón, después de haber obtenido los vértices y aristas del esqueleto, debe efectuarse una reducción entre los vértices del esqueleto que se encuentren muy cercanos entre sí.

Hasta ahora contamos con una lista $\mathcal{LV}$, de vértices de Voronoi, y una lista $\mathcal{LA}$, de parejas de índices de la lista $\mathcal{LV}$, que forma las aristas del esqueleto. Para realizar la reducción de los puntos, se hace un recorrido en la lista de aristas del esqueleto, $\mathcal{LA}$, y se calcula la distancia euclidiana entre sus vértices. Si la distancia es menor a cierto valor de umbral, se calcula el punto medio entre los vértices de la arista, se

eliminan ambos vértices de la lista $\mathcal{LV}$ y se agrega el nuevo punto a ésta. También se verifica en la lista $\mathcal{LA}$ si hay otras aristas que estén formadas con alguno de los puntos eliminados y, de ser así, se sustituye el índice del punto eliminado por el índice del nuevo punto.

El valor de umbral que utilizamos es el menor valor de la *medida característica local* ($r\text{TCL}(p)$) que sea hallado en la imagen en el proceso de muestreo. La construcción que estamos obteniendo al hacer la reducción en realidad es una aproximación del esqueleto, y algunos de los puntos en ella no son vértices de Voronoi exactos. Sin embargo, esta nueva aproximación del esqueleto no implica ningún problema para los efectos de visualización, pues los puntos del esqueleto sólo se necesitan para realizar la interpolación entre los contornos y, como se mencionó, no se requiere demasiada precisión.

Recordemos que el valor $r\text{TCL}(p)$ representa la máxima distancia entre un punto p de la curva de nivel con algún punto muestra, lo cual significa que el par de puntos muestra entre los que se encuentra p, guardan una distancia de a los más $2r\text{TCL}(p)$, tal y como se describió en la sección 3.2.3. De manera que si tomamos el menor valor de $r\text{TCL}(p)$ como la medida para reducir los pares de puntos del esqueleto, la aproximación que se obtiene del esqueleto es aceptable para nuestros propósitos.

En la figura 4.5 se muestra la gráfica de la corteza y el esqueleto del *conjunto E* después de haber aplicado la reducción a los puntos del esqueleto.

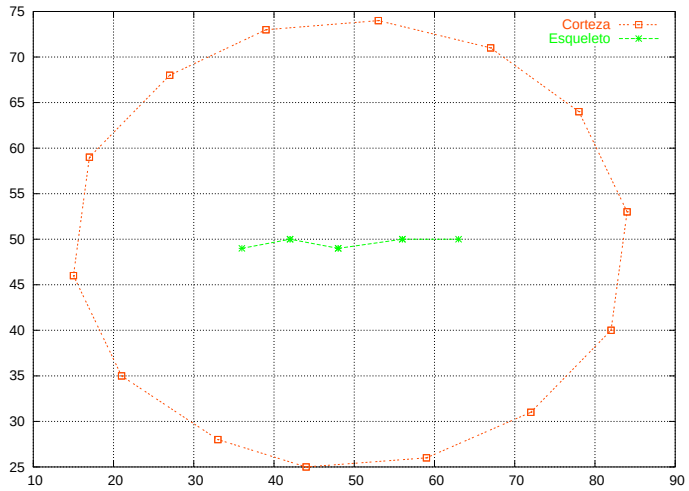


Figura 4.5: Gráfica de las aristas de la corteza y del esqueleto del *conjunto E* después de haber ejecutado la reducción de los puntos del esqueleto.

El programa que ejecuta el procedimiento 4 también realiza la reducción de los

puntos, y se obtiene como salida un archivo con el formato que se presenta en el cuadro 4.5. En ese archivo se listan los vértices y las aristas del esqueleto.

nv		Número de vértices del esqueleto
x_0	y_0	Lista de vértices
x_1	y_1	
$\vdots$		
x_{nv}	y_{nv}	
na		Número de aristas del esqueleto
v_{11}	v_{12}	Lista de aristas, se indican los índices de los vértices que las forman
v_{21}	v_{22}	
$\vdots$		
v_{na1}	v_{na2}	

Cuadro 4.5: Formato del archivo con la información de los vértices y las aristas del esqueleto.

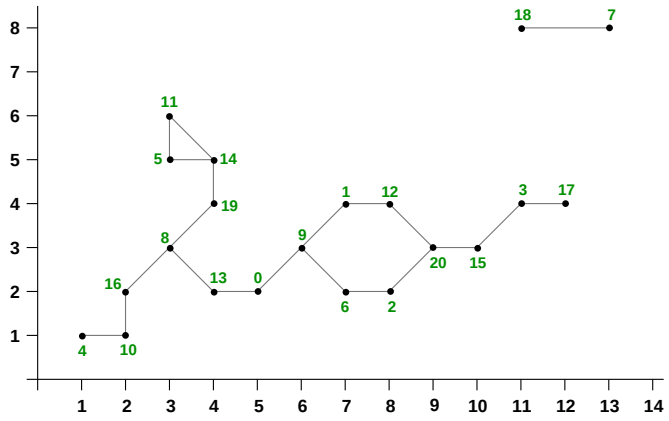
4.1.5. Organización en grafos de los puntos del esqueleto

La lista de vértices del esqueleto con la que se cuenta hasta este punto no está ordenada, es decir, los puntos no llevan ninguna secuencia que defina la trayectoria del esqueleto. Lo mismo sucede con la lista de aristas, que aunque indica los pares de puntos que forman a cada una de ellas, su secuencia tampoco define ninguna trayectoria. Esta estructura de datos no facilita el manejo de la información. Por esta razón se requiere idear una nueva estructura para el esqueleto.

Para una mejor apreciación de la estructura actual del esqueleto, considérese la gráfica del esqueleto con dos curvas que se presenta en la figura 4.6(a), la cual ha sido construída a partir de las listas de vértices y de aristas de la tabla de la figura 4.6(b). La primera lista indica las coordenadas de todos los vértices del esqueleto, la segunda lista muestra las parejas de vértices que forman las aristas. Por ejemplo, el primer elemento indica que hay una arista entre los vértice 13 y 0, el segundo, entre los vértices 4 y 10, y así sucesivamente. Nótese que ninguna de las dos listas llevan secuencia alguna, tal como se ilustra en la figura 4.6(a).

Un ejemplo práctico es el que se presenta en la figura 4.7. Las líneas gruesas representan la corteza de los cerros cuyo mapa se empleó en el capítulo anterior, las líneas claras indican el esqueleto. Se observa que el esqueleto está conformado por diferentes curvas, cada una con ramificaciones, e incluso con ciclos en algunas de ellas.

Dada la forma que tienen las curvas del esqueleto, en la nueva estructura de datos que proponemos cada curva será tratada como un grafo, con nodos, ramas y troncos.



(a)

Lista de vértices		Lista de aristas	
5, 2	3, 6	13, 0	8, 13
7, 4	8, 4	4, 10	5, 14
8, 2	4, 2	8, 19	6, 2
11, 4	4, 5	5, 11	14, 11
1, 1	10, 3	1, 12	9, 6
3, 5	2, 2	3, 17	12, 20
7, 2	12, 4	18, 7	1, 9
13, 8	11, 8	15, 3	15, 20
3, 3	4, 4	10, 16	0, 9
6, 3	9, 3	19, 14	2, 20
2, 1		8, 16	

(b)

Figura 4.6: (a) Gráfica de un esqueleto con dos curvas. La numeración representa la secuencia inicial (desordenada) de los vértices en la lista. (b) Lista de vértices y aristas del esqueleto en (a).

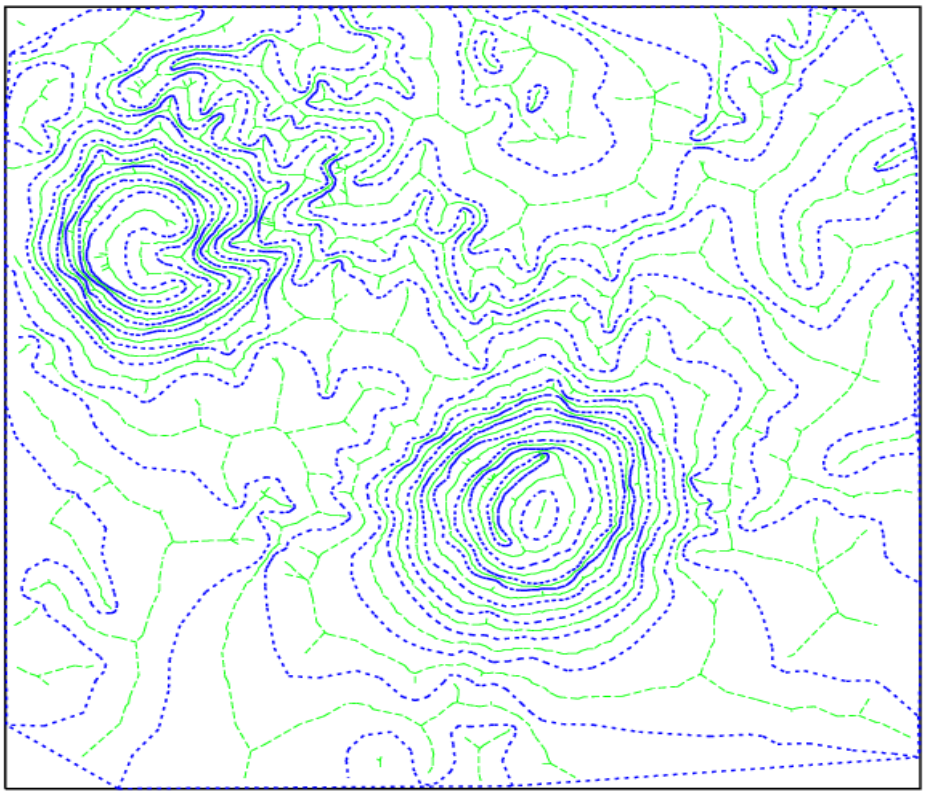


Figura 4.7: Las líneas gruesas representan la gráfica de la corteza y las delgadas la gráfica del esqueleto para las curvas de nivel de los cerros del mapa en la Fig. 3.2.

Entonces el esqueleto completo será un conjunto de grafos.

A continuación se presenta una serie de definiciones concernientes a los grafos, que serán empleadas en lo sucesivo.

Nodo: Es el índice de un vértice en la lista de vértices.

Nodo terminal: Es un nodo extremo en el grafo, es decir, un nodo que sólo pertenece a una arista.

Nodo ramificación: (O simplemente *ramificación*), es un nodo que pertenece a más de dos aristas.

Rama: Es una secuencia de nodos, desde un nodo terminal hasta una ramificación, o bien, entre dos nodos terminales.

Tronco: Es una cadena de nodos cuyas terminaciones son ramificaciones.

Así entonces, en la figura 4.6(a), los nodos $4, 17, 18$ y 7, son nodos terminales. Los nodos $8, 9, 20$ y 14 son nodos ramificación. Las secuencias de nodos (4-10-16-8) y (18-7), entre otras, son ramas. Y finalmente, uno de los troncos en el grafo es la secuencia de nodos (14-11-5).

Los puntos del esqueleto han sido organizados en grafos para facilitar el acceso a los datos necesarios en el cálculo de sus valores de elevación. Tales datos son el grafo y el nodo inicial de la rama o tronco al que pertenece cada punto, según se explicará en la siguiente sección. Evidentemente, la búsqueda de estos datos se dificulta si se mantiene la estructura anterior.

Como se vió en la figura 4.7, es posible que se presenten ciclos en los grafos del esqueleto; en tales casos, dos o más troncos comparten tanto el nodo inicial como el final, pero debido a que los puntos intermedios de los troncos son distintos, basta con almacenar el nodo que le sigue al nodo inicial y el número de nodos intermedios para obterner una representación única de cada tronco.

Para la representación del esqueleto, entonces, se sustituirá la lista de aristas por una lista de ramas y troncos por cada grafo del esqueleto. La codificación para las ramas y los troncos será una cuádrupla, de la forma (i, s, n, t), donde:

- i es el nodo inicial,
- s indica el nodo siguiente a i,
- n es el número de nodos que siguen a s y
- t indica el nodo terminal.

Para poder emplear la nueva estructura de datos, la lista de vértices necesita ser reorganizada, de modo que su secuencia describa la trayectoria de los grafos. La reorganización de la lista de vértices puede hacerse al mismo tiempo en que se van extrayendo los grafos, para lo cual es necesario realizar búsquedas en la lista de aristas.

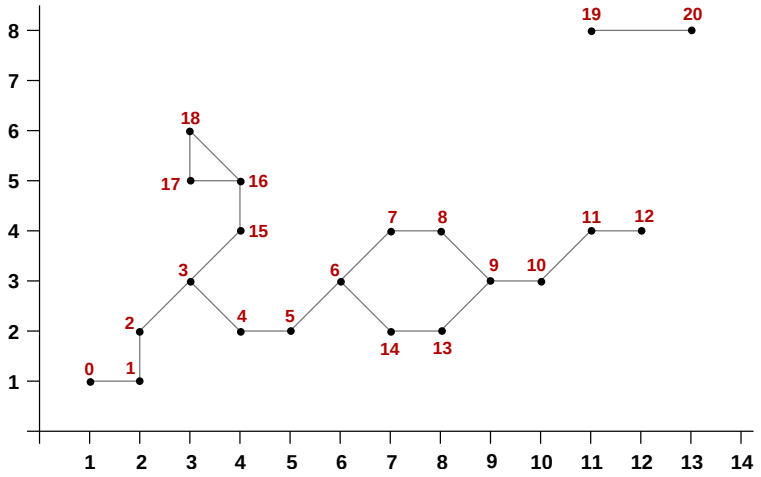
Llamemos a la lista de vértices, $\mathcal{LV}$, y a la lista de aristas, $\mathcal{LA}$. El procedimiento general para la extracción de los grafos es el siguiente:

1. Se inicia ordenando la lista $\mathcal{LA}$ como se explica en el apéndice A.
2. Para iniciar el recorrido de un grafo, se busca en $\mathcal{LA}$ una arista, a, que contenga un nodo terminal, v.
3. El nodo encontrado se agrega a una nueva lista de vértices, $\mathcal{L}$, y se elimina de la lista $\mathcal{LV}$. La lista $\mathcal{L}$ será la versión ordenada de $\mathcal{LV}$.
4. Se toma en v_{sig} el otro nodo de la arista a, la cual también es eliminada de su correspondiente lista $\mathcal{LA}$.
5. Se busca entonces otra arista, a_{sig}, asociada al nodo v_{sig}.
6. Se repiten los pasos del 3 al 5 para v_{sig} y a_{sig} hasta terminar con todos los nodos del grafo.
7. Se repiten los pasos del 2 al 6 hasta terminar con todos los grafos del esqueleto.

Cuando se hace la búsqueda de la siguiente arista, a_{sig}, puede ser que se encuentren dos o más de ellas que inicien con el mismo nodo, lo cual indica que dicho nodo es una ramificación y apartir de él pueden recorrerse al menos dos caminos. Entonces el nodo debe ser agregado a una pila para poder recorrer el otro camino posteriormente. Cuando se terminen de recorrer todas las aristas del primer camino, se sacan los nodos de la pila y se continúa el recorrido. Tal caso se presenta en el nodo 17 de la figura 4.6(a); una vez que se ha llegado a él, se saca el nodo 20 de la pila, el cual fue previamente introducido a ella, y se continúa el recorrido con el vértice 2.

Para encontrar un nodo terminal, en el paso 2, se realiza un conteo de las ocurrencias que tiene cada vértice en la lista de aristas. Entonces, un nodo terminal es aquel que tiene tan sólo una ocurrencia. Puede presentarse el caso en que en el esqueleto aparezca un grafo cerrado. En tal caso, evidentemente no se encontrará ninguna arista que contenga un nodo terminal, de modo que, para iniciar el recorrido de tal grafo, se toma la primer arista que se encuentre en la lista $\mathcal{LA}$. Después de que un vértice se ha agregado a la nueva lista $\mathcal{L}$, además de ser eliminado de la lista vieja $\mathcal{LV}$, también se reduce en uno el número de ocurrencias que éste tiene en la lista de aristas.

En la figura 4.8(a) se muestra la gráfica del esqueleto presentado en la figura 4.6 después de haber reorganizado la lista de vértices. En la parte (b) de la figura se muestra una tabla con la lista ordenada de vértices.



(a)

Lista de vértices	
1, 1	11, 4
2, 1	12, 4
2, 2	8, 2
3, 3	7, 2
4, 2	4, 4
5, 2	4, 5
6, 3	3, 5
7, 4	3, 6
8, 4	11, 8
9, 3	13, 8
10, 3	

(b)

Figura 4.8: (a) Gráfica del esqueleto con los vértices ordenados de la figura 4.6. (b) Lista ordenada de los vértices del esqueleto en (a).

La codificación de los grafos se obtiene durante el proceso, guardando un registro de los nodos inicial, siguiente, final y el número de nodos en el tronco o rama, siempre que se encuentre una ramificación o se llegue al final de una cadena de aristas. La codificación de los grafos del esqueleto de la figura 4.8(a) se presenta en el cuadro 4.6, y su representación gráfica en la figura 4.9.

Grafo	**No. de ramas/troncos**	**Codificación**			
		i	***s***	***n***	***t***
0	7	0	1	1	3
		3	4	1	6
		6	7	1	9
		9	10	1	6
		9	12	1	14
		3	15	0	16
		16	17	1	16
1	1	19	19	0	20

Cuadro 4.6: Codificación de los grafos del esqueleto de la figura 4.8(a).

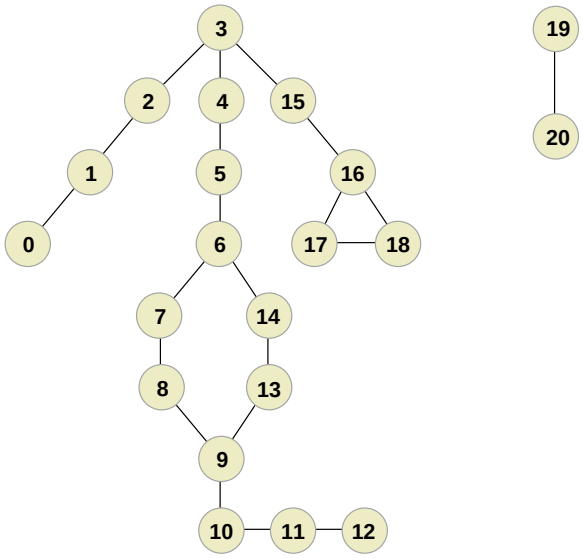


Figura 4.9: Representación en grafos del esqueleto de la figura 4.8(a).

El procedimiento que realiza todo este proceso se encuentra en el apéndice A, en donde se explica con mayor detalle. La salida del programa donde se implementa este procedimiento, es un archivo con el formato mostrado en el cuadro 4.7. Nótese que en la lista de vértices se incluye también una coordenada z, la cual inicialmente tiene un valor de cero (elevación cero). En la siguiente sección se explicará cómo calcular los valores de elevación para cada punto del esqueleto.

4.2. Cálculo de las elevaciones

Hasta ahora, contamos con el conjunto de puntos que definen los contornos de un terreno, esto es, los puntos de la corteza, y además hemos calculado y organizado los puntos del esqueleto de tales contornos.

Para obtener una reconstrucción tridimensional del terreno, se realiza la triangulación de Delaunay sobre el conjunto completo de los puntos (corteza y esqueleto), y se asignan valores de elevación a ellos. Los valores de elevación para los puntos de la corteza se conocen del mapa topográfico. A partir de esos valores se calcula la elevación para los puntos en el esqueleto.

El valor de elevación de cada punto en el esqueleto se calcula de acuerdo a su ubicación entre los contornos. Identificamos tres casos diferentes:

1. Puntos del esqueleto que se encuentran entre dos contornos diferentes,

nv				Número total de vértices en el esqueleto
x_1	y_1	z_1		Lista de vértices
x_2	y_2	z_2		
$\vdots$				
x_{nv}	y_{nv}	z_{nv}		
g				Número de grafos
$\boldsymbol{r_1}$				Número de ramas y troncos en el grafo 1
i_{11}	s_{11}	n_{11}	t_{11}	Lista de ramas y troncos del grafo 1
i_{12}	s_{12}	n_{12}	t_{12}	
$\vdots$				
i_{1r_1}	s_{1r_1}	n_{1r_1}	t_{1r_1}	
$\boldsymbol{r_2}$				Número de ramas y troncos en el grafo 2
i_{21}	s_{21}	n_{21}	t_{21}	Lista de ramas y troncos del grafo 2
i_{22}	s_{22}	n_{22}	t_{22}	
$\vdots$				
i_{2r_1}	s_{2r_1}	n_{2r_1}	t_{2r_1}	
$\vdots$				
$\boldsymbol{r_g}$				Número de ramas y troncos en el grafo g
i_{g1}	s_{g1}	n_{g1}	t_{g1}	Lista de ramas y troncos del grafo g
i_{g2}	s_{g2}	n_{g2}	t_{g2}	
$\vdots$				
i_{gr_g}	s_{gr_g}	n_{gr_g}	t_{gr_g}	

Cuadro 4.7: Formato del archivo con la información de los vértices del esqueleto organizados en grafos.

2. Puntos que se encuentran en una cima y

3. Puntos rodeados por secciones de contornos reentrantes.

La figura 4.10 es la gráfica de una porción de la corteza y el esqueleto del mapa de los cerros con el que hemos estado trabajando. En ella se ilustran cada uno de los tres casos de puntos en el esqueleto. Las líneas gruesas representan la corteza y la líneas delgadas el esqueleto.

Para identificar en qué caso caen los puntos del esqueleto, nos hemos auxiliado en la triangulación de Delaunay del conjunto completo de los puntos, esto es, puntos de corteza y puntos de esqueleto. En la triangulación, cada punto p pertenece a varios triángulos, de modo que el conjunto de puntos con los que p forma aristas en la triangulación, forman el conjunto de *puntos vecinos* de p. Los diferentes casos de puntos del esqueleto se identifican por sus puntos vecinos.

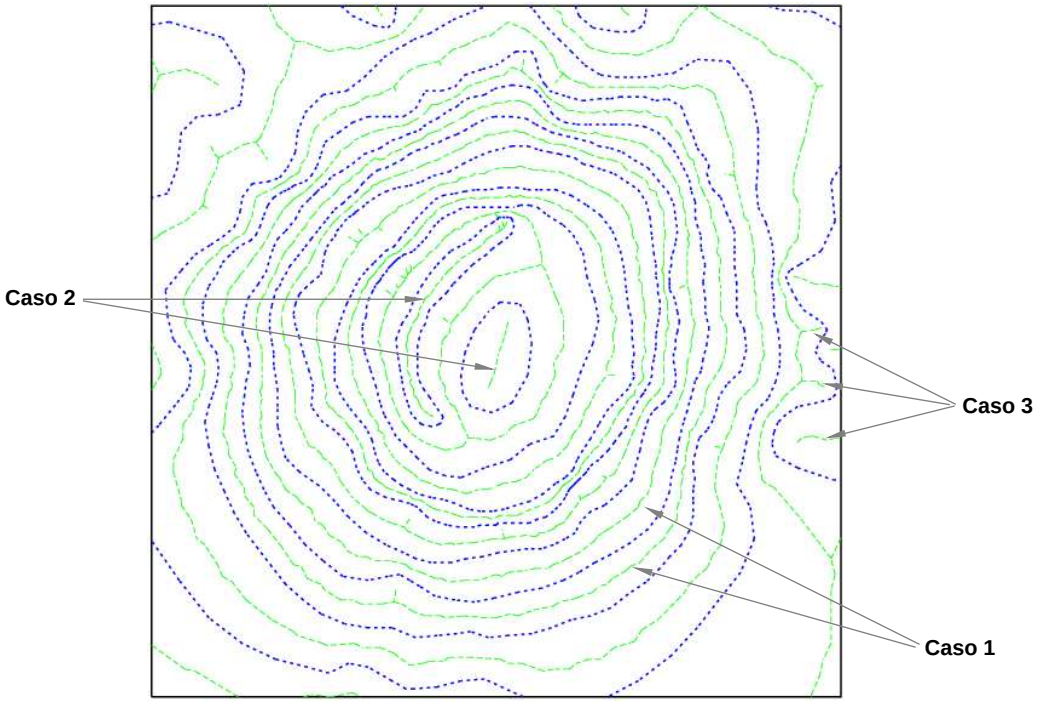


Figura 4.10: Diferentes tipos de puntos en el esqueleto. Detalles en el texto.

4.2.1. Caso 1: Puntos entre dos contornos diferentes

Este tipo de puntos se identifica porque en el conjunto de sus puntos vecinos, hay puntos de corteza que pertenecen a contornos diferentes, tal y como se muestra en la figura 4.11. La secuencia de los puntos del esqueleto de este tipo puede verse como un contorno intermedio.

El valor de elevación para esta clase de puntos se calcula simplemente como un promedio de las alturas de los contornos entre los que se encuentran. Se calcula como sigue:

$$z_p = \frac{z_1 + z_2}{2} \tag{4.2}$$

donde z_p es el valor de elevación para el punto p que se encuentra entre dos contornos y z_1 y z_2 son los valores de elevación de los puntos de los contornos entre los que se encuentra p.

Se calcula entonces z_p y se asigna a todos los nodos del grafo, al que pertenece p, que se clasifique en este caso de puntos.

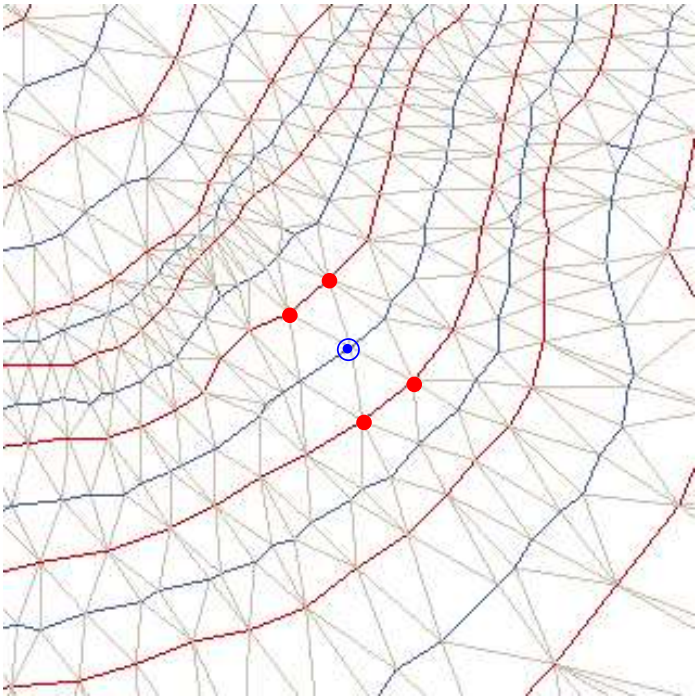


Figura 4.11: Puntos del esqueleto que se encuentran entre dos contornos de corteza. El punto dentro de un círculo es un punto del esqueleto y los otros puntos son sus vecinos, los cuales pertenecen a diferentes contornos de la corteza.

4.2.2. Caso 2: Puntos en una cima

Estos puntos pueden identificarse porque todos los vecinos de corteza que tienen pertenecen a un sólo contorno, y además todos los puntos del esqueleto que formen parte del grafo al que éstos pertenecen, cumplen con esta característica. La figura 4.12 ilustra este caso.

Recordemos que los vértices del esqueleto son vértices de Voronoi y los vértices de la corteza son vértices de Delaunay, entonces cada vértice del esqueleto es el centro de un círculo que toca al menos tres vértices de la corteza [3, 18], como puede observarse en la figura 4.12.

Sea p cualquier punto del esqueleto en una cima. El valor de elevación para p, de acuerdo a [23], está en función del radio del círculo que no contiene a ningún punto de la corteza, que está centrado en p y pasa por lo menos por tres puntos de la corteza. A los círculos que cumplan estas características, los llamaremos *círculos vacíos*. Entonces el valor de elevación para este tipo de puntos se calcula como sigue:

$$z_p = z_1 + r_p \tag{4.3}$$

donde z_p es el valor de elevación para el punto p, z_1 es el valor de elevación de contorno

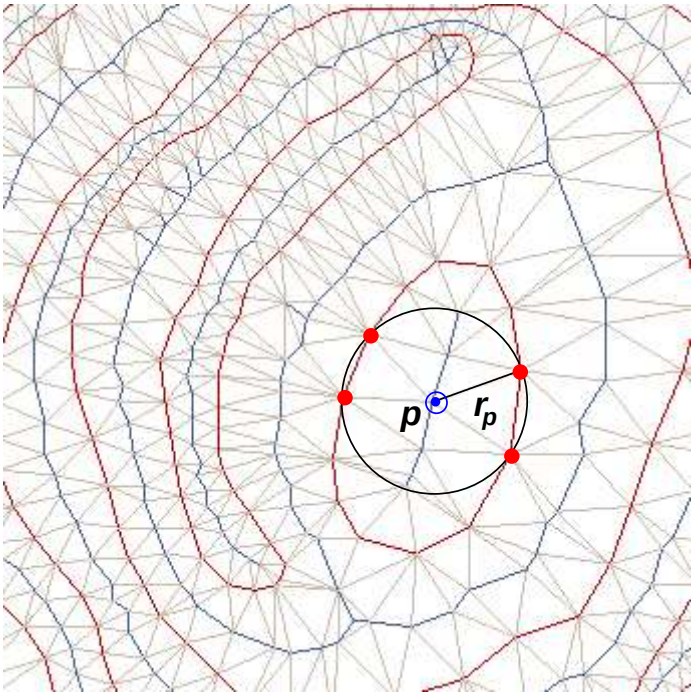


Figura 4.12: Puntos del esqueleto que se encuentran en una cima. El punto dentro de un círculo es un punto del esqueleto y los demás puntos son sus vecinos en el contorno de corteza que representa la cima de la montaña.

cima y r_p es el valor del radio del círculo descrito.

Para calcular el valor de r_p se busca en el conjunto de vecinos de p, el vecino de corteza más cercano; el valor de la distancia euclidiana entre p y tal vecino, será el valor para r_p.

4.2.3. Caso 3: Puntos rodeados por secciones de contornos reentrantes

Los puntos de este tipo se identifican por tener vecinos en un sólo contorno de corteza y, a diferencia del caso anterior, no todos los puntos del grafo al que ellos pertenecen, deben cumplir con esta característica. En la figura 4.13 se ilustra este caso.

El radio del círculo vacío centrado en cualquier punto p del esqueleto en estas secciones, comparado con el radio del círculo vacío del punto del esqueleto que inicia la ramificación en el esqueleto medio principal (contorno medio), proporciona el valor de elevación para el punto p, como una proporción de las elevaciones del contorno

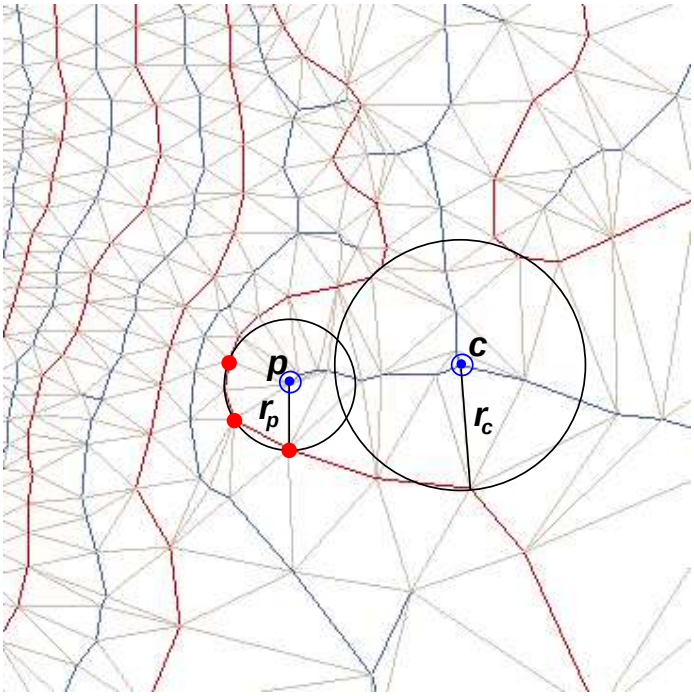


Figura 4.13: Puntos del esqueleto que se encuentran rodeados por una sección de un contorno reentrante.

medio y del que forma la reentrante [3].

El valor de elevación para un punto p en una reentrante, se calcula como sigue:

$$\begin{aligned} z_p &= \left(\frac{z_1 + z_2}{2}\right)\left(\frac{r_p}{r_c}\right) + \left(1 - \frac{r_i}{r_c}\right) z_1 \\ &= z_1 - \left(\frac{r_p}{r_c}\right)\left(\frac{z_1 - z_2}{2}\right) \end{aligned} \tag{4.4}$$

donde z_p es el valor de elevación para el punto p, z_1 y z_2 son los valores de elevación de los contornos entre los que se encuentra el esqueleto medio al que se encuentra conectada la sección reentrante, r_p es el radio del círculo vacío centrado en p, y r_c en el radio del círculo centrado en el punto que inicia la ramificación. Véase la figura 4.13.

El valor de r_p será la distancia euclidiana de p a su vecino de corteza más cercano. Para calcular el valor de r_c se busca en el conjunto de vecinos del punto c el vecino de corteza más cercano y el valor de r_c será la distancia euclidiana entre ellos. Para puntos como c (puntos que inician una ramificación) es posible que de acuerdo a la triangulación no tengan vecinos de corteza, entonces debe encontrarse alguno de los puntos de la corteza por el que pase el círculo centrado en c descrito anteriormente. Para encontrar dicho punto, se busca entre los vecinos de los vecinos de c (vecinos

de esqueleto) el vecino de corteza más cercano; entonces el valor del radio r_c será el valor de la distancia euclidiana entre el punto encontrado y el punto c.

Capítulo 5

Interfaz Gráfica

A lo largo de los capítulos anteriores, se ha explicado el proceso para la obtención de la información necesaria para poder realizar el modelo de la reconstrucción tridimensional de un terreno. Este capítulo concierne a la descripción de la realización de la visualización de dicho modelo, lo cual corresponde con el último paso de la metodología planteada en el capítulo 4.

La visualización del terreno consiste en el dibujado de sus tres elementos: las curvas de nivel (corteza), el esqueleto y la triangulación o malla de triángulos. La visualización es integrada a una interfaz gráfica con la cual el usuario puede manipular lo siguiente:

- Visualizar cualesquiera de los tres elementos del terreno.
- Asignar valores de elevación a las curvas de nivel.
- Calcular los valores de elevación para los puntos del esqueleto, de acuerdo a los valores de elevación de las curvas de nivel introducidos.
- Asignar diferentes colores a las curvas de nivel.
- Visualizar la superficie del terreno de dos maneras:
  - degradando los colores entre las curvas de nivel y
  - degradando sólo con dos colores.
- Realizar acercamientos y alejamientos sobre regiones específicas del terreno.
- Rotar la escena sobre los ejes x y y.

5.1. Entorno de programación

Para el desarrollo de la interfaz gráfica de usuario hemos empleado el paquete de herramientas de Qt y para el dibujado del terreno en tres dimensiones utilizamos las funciones de la bibliotéca gráfica de OpenGL (en realidad se usó la versión libre de OpenGL: Mesa [25]).

5.1.1. OpenGL

OpenGL es una interfaz de software (API) para hardware gráfico escrita originalmente en *C*. Esta biblioteca se concibió para programar en máquinas nativas *Silicon Graphics* bajo el nombre de GL (Graphics Library). Posteriormente se consideró la posibilidad de extenderla a cualquier tipo de plataforma, con lo que se llegó al término *Open Graphics Library*, es decir, OpenGL (los creadores de Mesa pidieron permiso a Silicon Graphics para usar la misma especificación del API).

La interfaz consiste de un conjunto de procedimientos y funciones que permiten producir imágenes en color de alta calidad, especialmente de objetos tridimensionales.

OpenGL tiene incorporadas las funciones para el dibujado de un pequeño grupo de primitivas:

- puntos,
- líneas,
- triángulos,
- cuadriláteros y
- polígonos en general,

a partir de las cuales es posible construir objetos mucho más elaborados.

OpenGL también proporciona otras funciones para la manipulación de los objetos dibujados:

- para realizar transfomaciones geométricas (rotación, translación y escalamiento),
- para el dibujado de objetos con color y con textura,
- para dar efectos de iluminación a la escena,
- para dar efectos de sombreado a los objetos.

5.1.2. Qt

Qt es una caja de herramientas de C++ para el desarrollo de Interfaces Gráficas de Usuario (GUI, por sus siglas en inglés) multiplataforma. Qt sigue una filosofía del "mismo código, distintos compiladores"; del mismo código fuente se generan varios ejecutables dependiendo del compilador que se use en la plataforma. Qt es un producto de *Troll Tech* [26].

Qt permite el desarrollo de prototipos rápidos y es gratuito bajo Linux, aparte de que tiene una documentación excelente. Qt introduce el mecanismo de señales−ranuras para el intercambio de mensajes entre objetos que es bastante intuitivo, en comparación al mecanismo de retrollamadas del desarrollo en XWindows y Motif. Las señales son enviadas por un objeto a la ranura de otro. Este mecanismo provoca un sobreuso del procesador porque hay que usar un metacompilador (llamado MOC, Meta Object Compiler, en Qt) para traducir el código de C++ con señales y ranuras a código C++ simple.

Los componentes para diseñar una interfaz gráfica se llaman *widgets*. Algunos widgets comunes son: botones, barras de scroll, cuadros de diálogo, menús, botones con imágenes (o iconos), ventanas, etc. La biblioteca de Qt provee toda la funcionalidad para el dibujo de widgets, sus estilos de dibujo, y para controlar los eventos de teclado y ratón, además de poder diseñar nuevos widgets inexistentes, como podría ser una manija circular (como las que controlan el volumen en un radio). El programador se enfoca a la funcionalidad y la comunicación entre los diferentes widgets.

5.2. Estructuras de datos

Para el dibujado del terreno se establecieron estructuras de datos en C, apropiadas para el manejo de cada uno de estos elementos. Los datos para el llenado de estas estructuras se toman de los archivos cuyos formatos se presentaron en los capítulos 3 y 4, con la información de los contornos (cuadro 3.2), del esqueleto (cuadro 4.7) y de la triangulación (cuadro 4.3).

5.2.1. Corteza

La estructura en código C para la corteza o las curvas de nivel es la siguiente:

```
int n_contours;
typedef struct CONTOUR {
     int n_points;
     float *x;
     float *y;
     float  z;
     int open_close;
```

```
     float color[3];
};
CONTOUR *contours;
```

Esta estructura se utilizó como sigue:

- En `n_contours` se guarda el número de contornos o curvas de nivel.
- Dentro de la estructura `CONTOUR` se almacena la información de cada curva de nivel:
  - En `n_points` se guarda el registro del número de puntos que conforman la curva.
  - En `*x` y `*y` se mantiene la lista de coordenadas de los puntos.
  - En `z` se guarda el valor de elevación del contorno (de todos los puntos).
  - `open_close` es una bandera que indica si la curva es abierta o cerrada, para saber si se debe dibujar una arista entre los vértices primero y último o no.
  - Finalmente, en `color[3]` se mantiene el color con el que será dibujada la curva de nivel, que por omisión es (142, 0, 0), como se dijo al final de la sección 3.2.3.

5.2.2. Esqueleto

La estructura de datos para el esqueleto es un tanto más extensa que la estructura para la corteza. Primero tiene que definirse una estructura para los puntos del esqueleto, pues se requiere conocer más información sobre ellos a parte de simplemente sus coordenadas. La estructura en C para los puntos del esqueleto es la siguiente:

```
int n_points;
typedef struct SKEL_POINT {
     float x;
     float y;
     float z;
     int kind;
     int contours[2];
     int n_neighbors;
     int *v_neighbors;
     int graph;
};
SKEL_POINT *points;
```

Su uso es el siguiente:

- En `n_points` simplemente se guarda el número total de puntos en todo el esqueleto.
- La información de cada punto se mantiene en la estructura `SKEL_POINT`:
  - En `x`, `y` y `z` se guarda el valor de la coordenada del punto. Nótese que para los puntos del esqueleto es necesario tener un valor de elevación independiente, a diferencia de los puntos de la corteza que tienen un valor de elevación por contorno.
  - En la variable `kind` se almacena el tipo de punto del esqueleto al que pertenezca de acuerdo a la clasificación establecida en la sección 4.2.
  - En el arreglo `contours[2]` se guardan los números de los contornos entre los que se encuentra el punto; en caso de que el punto sólo se encuentre entre un sólo contorno, la segunda posición del arreglo se pone a -1.
  - En `n_neighbors` se mantiene el número de vecinos que tiene el punto en la triangulación.
  - En `*v_neighbors` se guarda la lista de los índices de sus vértices vecinos.
  - Finalmente en `graph` se guarda el índice del grafo al que pertenece el punto.

Esta estructura se planeó así para facilitar la búsqueda de los vecinos de cada punto en el esqueleto, pues es necesario para el cálculo de las elevaciones según se explicó en la sección 4.2.

Después, se requiere otra estructura para la información correspondiente a los grafos, las ramas y los troncos:

```
int n_graphs;
typedef struct BRANCH {
     int begin;
     int next;
     int n_inter;
     int end;
};
typedef struct GRAPH {
     int n_branches;
     BRANCH *branches;
     int kind;
};
GRAPH *graphs;
```

Se usa como sigue:

- En `n_graphs` se guarda el número de grafos encontrados en el esqueleto.
- La estructura `BRANCH` se tiene para almacenar la codificación tanto de ramas como troncos de los grafos, según se describió en la sección 4.1.5; no se hace distinción alguna entre ramas y troncos, en lo posterior nos referiremos a cualquiera de los dos como rama.
- La estructura `GRAPH` se tiene para el almacenamiento de cada uno de los grafos:
  - En `n_branches` se guarda el número de ramas en el grafo.
  - El arreglo `*branches` es para mantener la lista de las ramas.
  - En `kind` se guarda el tipo de grafo de acuerdo al tipo de puntos que contenga. La clasificación es:

    `1` - para los grafos que contengan exclusivamente puntos con vecinos de corteza en un sólo contorno y

    `2` - para el resto de los grafos.

5.2.3. Triangulación

La estructura de la triangulación es la más sencilla de las tres, pues sólo se necesita una variable para almacenar el número de triángulos y tres arreglos para la lista de índices de los vértices que forman a cada triángulo:

```
int n_triangles;
int *vertex[3];
```

Cada triángulo i está formado por los vértices cuyos indices son `vertex[0][i]`, `vertex[1][i]` y `vertex[2][i]`. Los vértices pueden ser tanto de corteza como de esqueleto, esto es, que los índices pueden ser tanto de la lista de vértices de la corteza como del esqueleto.

5.3. Visualización del terreno

Como se mencionó anteriormente, la visualización del terreno consiste en el dibujado de la corteza, el esqueleto y la malla de triángulos. En la figura 5.1 se presentan los resultados del dibujado de estos elementos para el mapa de los cerros con el que se trabajó en el capítulo 3; en rojo se muestra la corteza, en azul el esqueleto y en gris la triangulación.

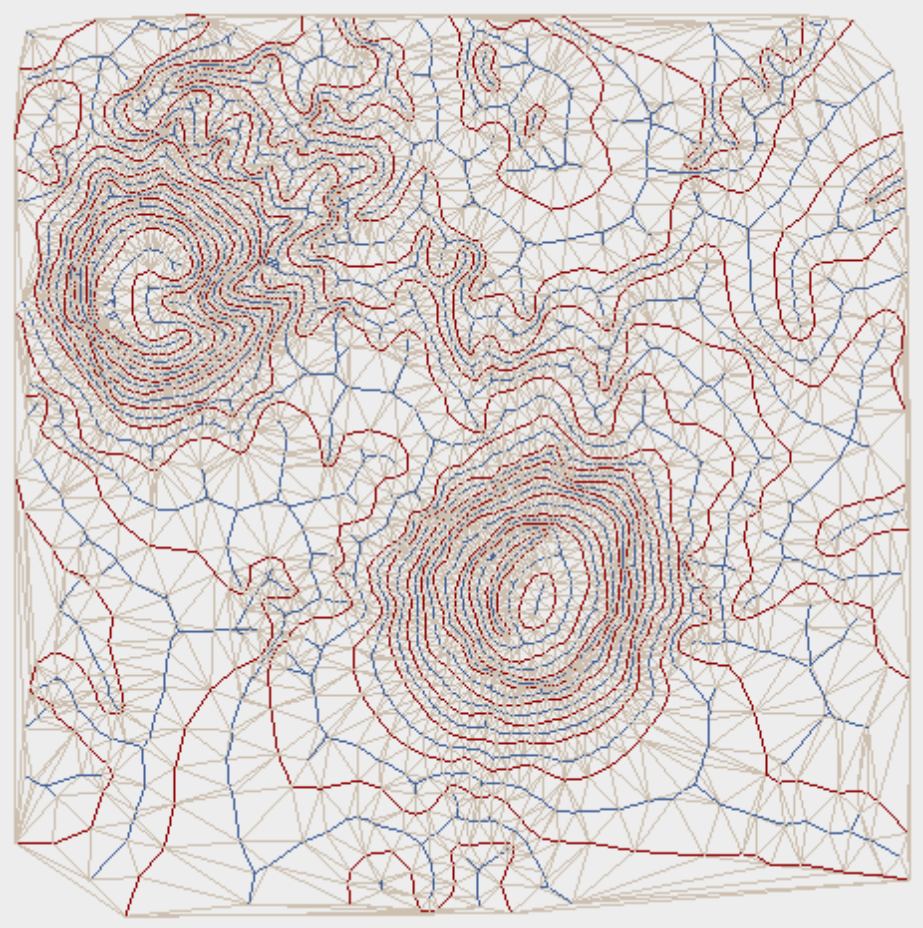


Figura 5.1: En rojo se muestra la corteza, en azul el esqueleto y en gris la triangulación del mapa de contornos de la figura 3.2.

En la visualización se contemplan dos aspectos más: la selección de una curva de nivel, para poder asignarle el valor de elevación, y el degradado de colores que se aplica a los triángulos dibujados para presentar una superficie suave. En esta sección se describirá cómo se resolvieron estos dos problemas.

5.3.1. Selección de una curva de nivel

El objetivo de la selección de curvas es permitir al usuario asignar valores de elevación a las curvas de nivel. Para cada curva se tiene el registro de las coordenadas inferior izquierda y superior derecha de la caja que encierra a todos los puntos de la curva. En la figura 5.2 se presenta el ejemplo de las cajas que delimitan las curvas de nivel del mapa de contornos de una superfice ficticia creada a partir de elipses.

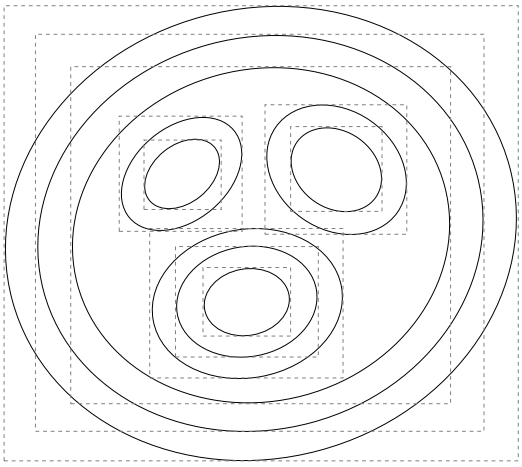
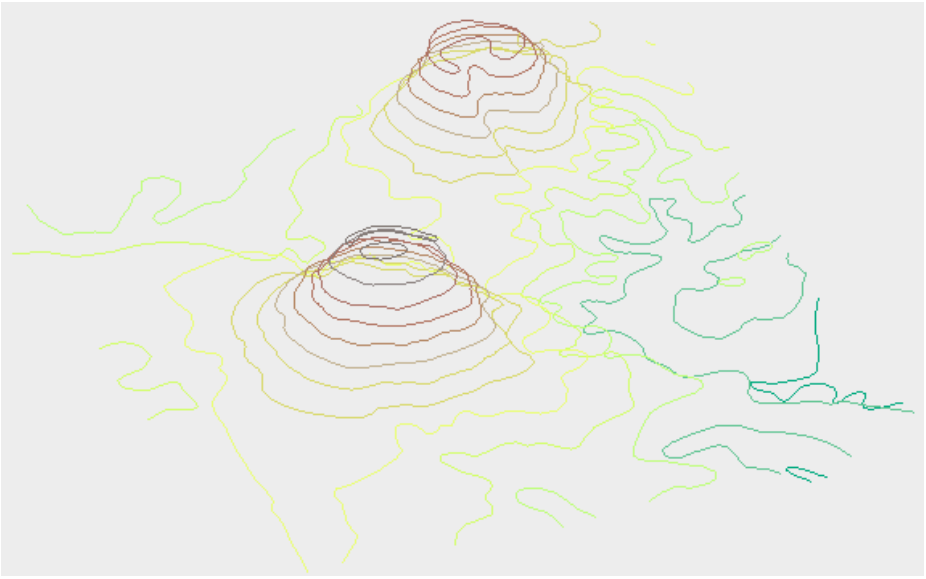


Figura 5.2: Cajas que delimitan las curvas de nivel de una superficie creada a partir de elipses.

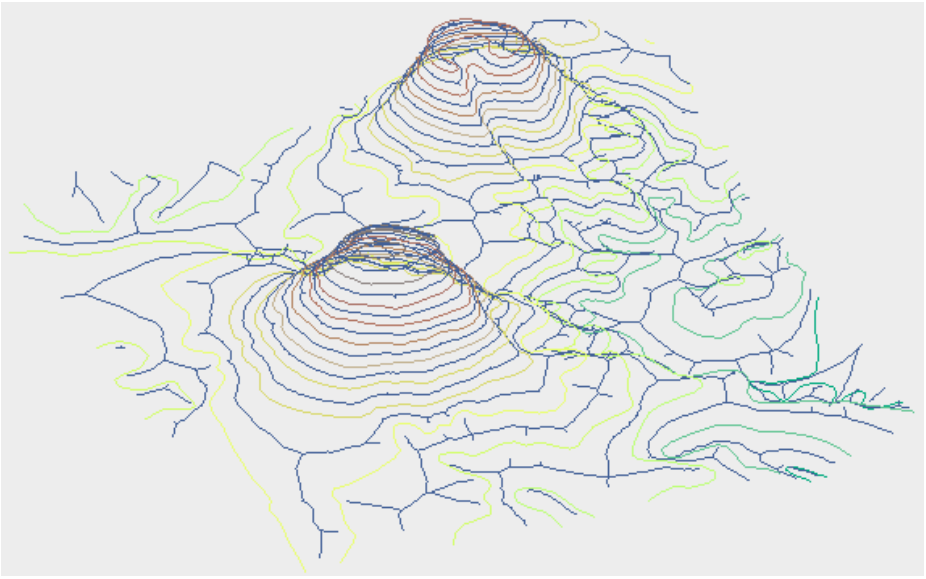
Para determinar si una curva ha sido seleccionada con el ratón, se toma la coordenada de la posición de éste al momento de dar click y se busca la caja de menor área que contenga a esa coordenada. La curva asociada a la caja hallada es la curva seleccionada.

Los valores de elevación asignados a las curvas de nivel se han empleado para calcular la elevación de los vértices del esqueleto, como se explicó en la sección 4.2 en la pág. 54, y se emplearán para asignar diferentes colores a las curvas, como se describirá en la sección 5.4 donde se explicará la integración de todos los elementos de la visualización en una interfaz gráfica.

En la figura 5.3(a) se presenta el dibujo de la corteza después de haber asignado los colores y los valores de elevación a las curvas de nivel, en la parte (b) de la figura se muestra el dibujo del esqueleto después de calcular los valores de elevación para cada punto con respecto a los valores de elevación de las curvas de nivel.



(a)



(b)

Figura 5.3: (a) Visualización de las curvas de nivel de la figura 5.1 después de haberles asignado valor de elevación y color. (b) Visualización del esqueleto con los valores de elevación calculados.

5.3.2. Asignación del color a cada vértice

Los efectos de la visualización de la superficie del terreno dependen de la manera en cómo se colorea cada triángulo de la malla. Si los triángulos son coloreados con un solo color, los cambios en la superficie no lucen suaves. Para evitar este tipo de inconvenientes, cada triángulo en la malla se ha coloreado utilizando interpolación de colores, es decir, con degradación.

OpenGL brinda la posibilidad de realizar este tipo de coloreado sobre los objetos dibujados. Para dibujar un objeto en OpenGL basta con indicarle el tipo de objeto (punto, línea o polígono) y la lista de vértices que lo conforman. El color por defecto que emplea OpenGL para el trazado y coloreado de objetos es negro. Para cambiar el color de los objetos, éste debe ser especificado por vértice. Si se desea obtener un coloreado degradado en los polígonos, debe especificarse un color diferente para cada vértice del polígono y OpenGL realiza el degradado automáticamente. En la parte izquierda de la figura 5.4 se muestra el ejemplo de un triángulo con todos sus vértices especificados en color verde; en el triángulo de la parte derecha se especificaron colores diferentes para cada vértice.

Figura 5.4: Ejemplos de triángulos realizados con OpenGL. El triángulo de la izquierda ha sido coloreado con un sólo color (coloreado liso) y el de la derecha con tres colores diferentes (coloreado degradado).

Entonces para obtener una visualización de la superfice coloreada con cambios suaves, deben asignarse diferentes colores a cada vértice de la triangulación, esto es, a cada vértice de la corteza y del esqueleto.

Se realizaron dos modalidades para la visualización de la superficie del terreno: degradando entre los colores de las curvas de nivel y degradando sólo entre dos colores definidos. En ambos casos, se requiere asignar colores a los vértices de acuerdo a su valor de elevación.

En el primer caso, se conocen los colores que tienen las curvas de contorno y, por consiguiente, se conoce el color de todos los puntos de la corteza. Para los puntos del esqueleto debe calcularse, de acuerdo a su valor de elevación, un color intermedio entre los colores asignados a las curvas entre las que se encuentra cada punto.

En el segundo caso, se asignan dos colores diferentes a los puntos de mayor y de menor elevación en la triangulación. Para el resto de los puntos, al igual que en el caso anterior, se calcula un color intermedio de acuerdo a su elevación.

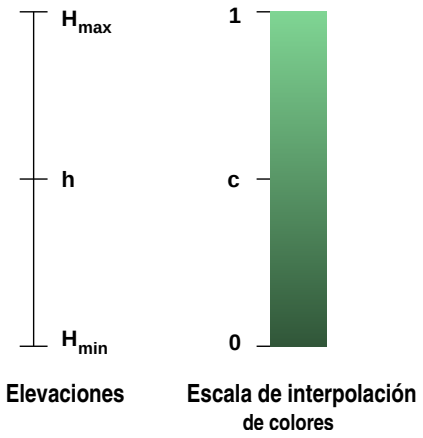
Para obtener un color intermedio entre los colores (R_1, G_1, B_1) y (R_2, G_2, B_2) se

emplea la siguiente fórmula:

$$\begin{aligned} R &= R_1 + \big(R_2 - R_1\big)c \\ G &= G_1 + \big(G_2 - G_1\big)c \\ V &= V_1 + \big(V_2 - V_1\big)c \end{aligned} \tag{5.1}$$

donde (R, G, B) es el nuevo color y c es el nivel en la escala de interpolación que va de 0 (máxima opacidad) a 1 (máxima luminosidad): si $c = 0$ el nuevo color será (R_1, G_1, B_1) y si $c = 1$ será (R_2, G_2, B_2).

Para obtener el nivel en la escala de interpolación que le corresponde a cada punto de acuerdo a su valor de elevación, se plantea una relación de proporcionalidad como se muestra en la figura 5.5.



$$\frac{h - H_{min}}{H_{max} - H_{min}} = \frac{c - 0}{1 - 0}$$

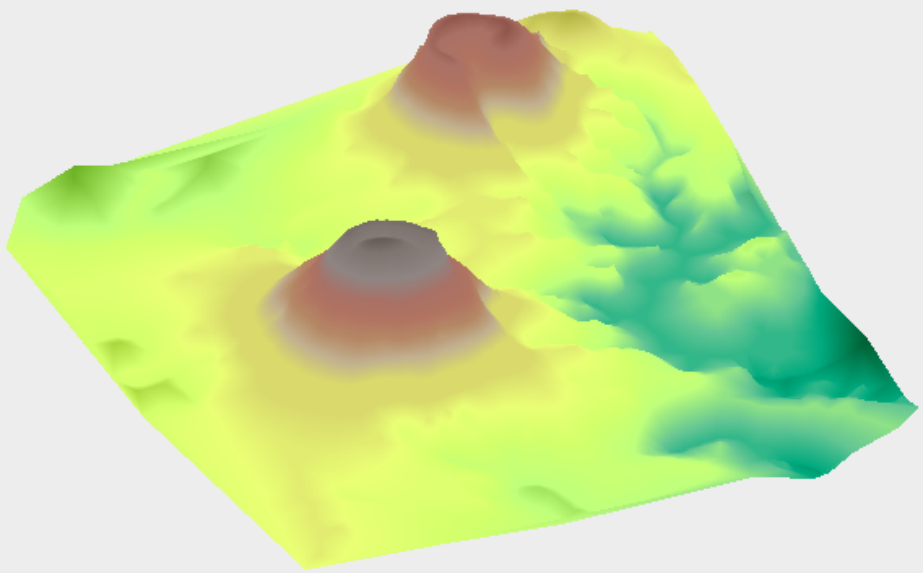
Figura 5.5: Relación de proporcionalidad para la asociación entre el valor de elevación de un punto con el nivel que le corresponde en la escala de interpolación de los colores.

Entonces, el nivel en la escala de interpolación entre los colores (R_1, G_1, B_1) y (R_2, G_2, B_2) que le corresponde al punto con valor de elevación h está dado por:

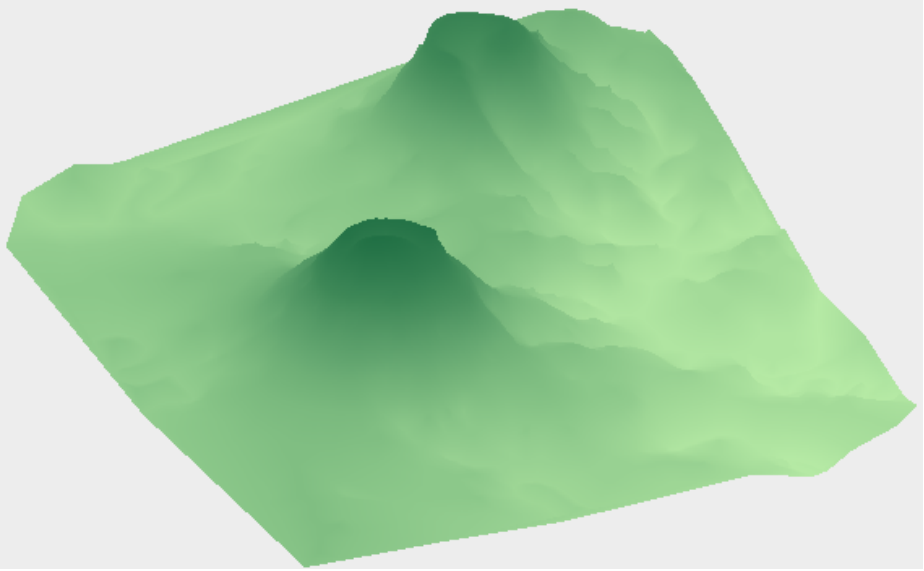
$$c = \frac{h - H_{min}}{H_{max} - H_{min}} \tag{5.2}$$

donde H_{max} es el valor de elevación de los puntos con color (R_1, G_1, B_1) y H_{min} de los puntos con color (R_2, G_2, B_2).

En la figura 5.6(a) se muestra el resultado de colorear la triangulación de acuerdo a los colores de las curvas de nivel introducidos en la sección anterior, en la parte (b) se presenta el resultado de colorear usando sólo dos colores.



(a)



(b)

Figura 5.6: Visualización de la superficie del terreno de la figura 5.1 (a) usando los colores de las curvas de nivel y (b) usando sólo dos colores.

5.4. Descripción de la interfaz gráfica

Como se mencionó al inicio del capítulo, en la interfaz gráfica se integra la visualización del terreno junto con todas las funciones para la manipulación de éste. La imagen de la figura 5.7 ilustra la pantalla de la aplicación, donde son señalados cada uno de sus componentes. Cada componente tiene asociadas ciertas funciones correspondientes a la visualización y la manipulación del terreno. Esta sección concierne a la explicación de cada componente.

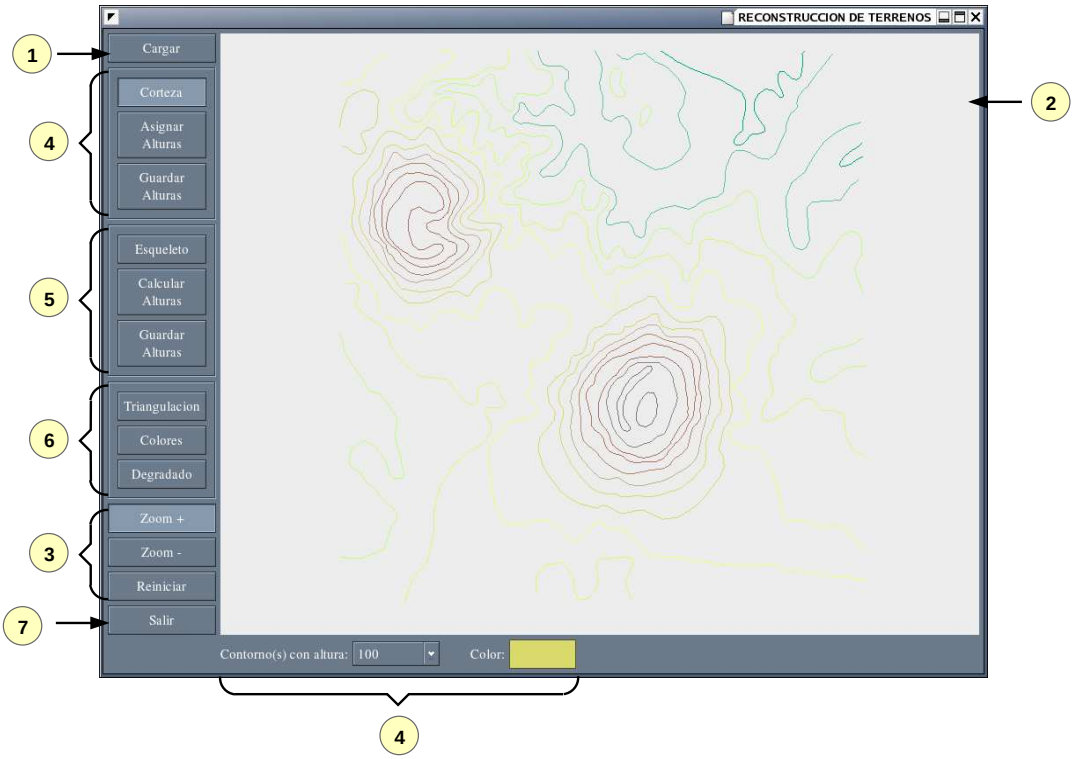


Figura 5.7: Interfaz gráfica de la aplicación para la reconstrucción de terrenos.

El primer componente permite cargar a la aplicación la información de algún terreno, la cual comprende un archivo por cada elemeto del terreno: corteza, esqueleto y triangulación. El formato de tales archivos es el que se presentó en los cuadros 3.2, 4.7 y 4.3, respectivamente. Cuando se presiona el botón *"Cargar"* se abre un cuadro de diálogo donde se debe seleccionar una carpeta que contenga los tres archivos.

El segundo componente es el área de visualización. En ella se puede apreciar la aplicación de las funciones de manipulación y transformación sobre el terreno.

Las funciones del componente 3, corresponden a las tranformaciones de acercamiento y alejamiento que pueden aplicarse sobre la escena dibujada en el área de visualización y la reinicialización del dibujado. Si el botón *"Zoom +"* se encuentra activado, entonces se realizará un acercamiento sobre la región que sea releccionada con el ratón al momento en que el usuario presione el botón izquierdo de éste. Lo mismo sucede con el botón *"Zoom -"*, excepto que se realiza un alejamiento. Sólo uno de los botones *"Zoom +"* y *"Zoom -"* puede estar activo a la vez, de modo que si se activa el botón *"Zoom -"* cuando el otro está activo, este último se desactiva y viceversa. Cuando ninguno de los dos botones se encuentran activos, la función del ratón sobre el área de dibujado consiste en realizar las transformaciones de rotado

de la escena sobre los ejes x y y. Estas funciones se activan por medio del arrastre del ratón: si se arrastra con movimiento sobre el eje x, la escena se rotará sobre el eje y y viceversa. Con el botón *"Reiniciar"* se redibuja la escena quitando todas las transformaciones aplicadas.

El componente número 4 comprende las funciones que pueden realizarse sobre la corteza o curvas de nivel. La primer función está asociada al botón *"Corteza"* y corresponde al activado y desactivado del dibujado de ésta. Con la segunda función, asociada al botón *"Asignar Alturas"*, se redibuja la escena sólo con las curvas de nivel, pintándolas todas con un sólo color (café) y quitando todas las transformaciones activas; también se cambia la función actual que tiene el ratón sobre el área de visualización por la función de selección de curvas de nivel que se describió en la sección 5.3.1. Cuando una curva haya sido seleccionada, ésta será redibujada con un color diferente (verde), para distinguirla entre las otras, y se desplegará un cuadro de diálogo para introducir el valor de elevación para la curva seleccionada. Después de que el valor de elevación se haya introducido, la curva seleccionada será redibujada nuevamente pero esta vez con un tercer color (rojo), para diferenciarla de las otras curvas a las que aún no se les haya asignado un valor de elevación. Este proceso se ilustra en la figura 5.8. Finalmente, con el botón *"Guardar Alturas"* se invoca a la función encargada del almacenamiento de los datos de elevación en el archivo de la corteza.

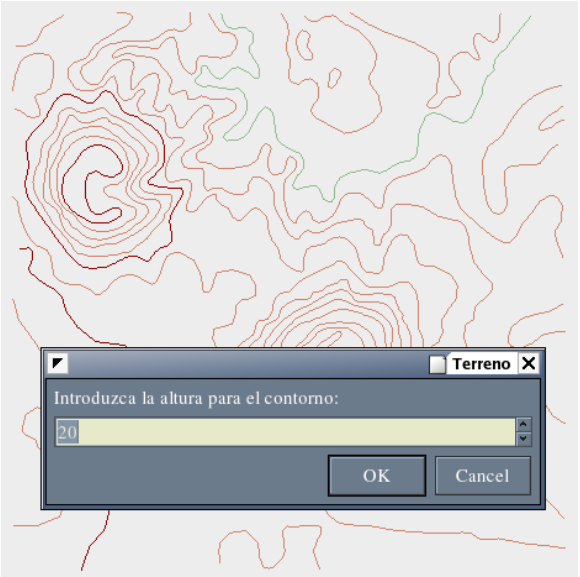


Figura 5.8: Selección de curvas de nivel y asignación de su valor de elevación. En café se muestran las curvas que no tienen valor asignado, en rojo las que ya lo tienen y en verde a la que se está asignando.

La última función del cuarto componente es la encargada del asignamiento de los colores a las curvas de nivel, el cual se realiza de acuerdo al valor de elevación que tengan las curvas. En el combo etiquetado como *"Contorno(s) con altura: "*, se cargan los diferentes valores de elevación entre las curvas de nivel cuando se carga la información del terreno, y además se van agregando los nuevos valores que sean introducidos en la selección de curvas. Para asignar algún color a las curvas que tengan un valor de elevación h, primero se debe seleccionar este valor en el combo y después se debe presionar el botón etiquetado como *"Color:"* para abrir un cuadro de diálogo en el cual se podrá seleccionar cualquier color deseado. Al cerrar el cuadro de diálogo el color quedará asociado a todas las curvas con valor de elevación h y éstas serán redibujadas en la escena con el nuevo color.

El quinto componente, comprende las funciones para el manejo del esqueleto. La primer función está asociada al botón *"Esqueleto"*, la cual simplemente activa o desactiva el dibujado de éste. La segunda función, asociada al botón *"Calcular Alturas"*, se encarga de realizar los cálculos para los valores de elevación de los puntos en el esqueleto, de acuerdo a lo planteado en la sección 4.2. La última función para el esqueleto es la asociada al botón *"Guardar Alturas"*, la cual simplemente se encarga de grabar en el archivo correspondiente los datos de elevación de los puntos del esqueleto.

Las funciones del componente 6 son las encargadas de la manipulación de la triangulación. La función asociada al botón *"Triangulación"*, al igual que en los casos anteriores, sólo activa o desactiva el dibujado de la malla de triángulos. Con la función asociada al botón *"Colores"* se activa el coloreado de la superficie del terreno empleando los colores asignados a las curvas de nivel. Finalmente, con el botón *"Degradado"* se invoca a la función que activa o desactiva el coloreado de la superficie del terreno empleando sólo dos colores.

Por último, el componente 7 sencillamente es la función que se utiliza dar término a la ejecución de la aplicación.

Capítulo 6

Conclusiones y Trabajo Futuro

6.1. Conclusiones

En este trabajo de tesis se ha tratado el problema de la reconstrucción de terrenos a partir del mapa impreso de sus contornos o curvas de nivel. La solución a este problema ha sido dividida en dos etapas principales:

1. Procesar la imagen digitalizada del mapa de contornos del terreno a reconstruir para extraer el conjunto de puntos que definan a cada contorno y
2. Realizar la interpolación entre los contornos para obtener una reconstrucción del terreno con cambios suaves en la superficie de éste.

Para la primera etapa, se emplearon las imágenes digitalizadas de los dibujos obtenidos al calcar con un pincel negro y sobre una hoja blanca, las curvas de nivel que se encuentran en los mapas topográficos. Esto se hizo con la finalidad de simplificar el proceso de extracción de los pixeles que conforman a cada contorno en el mapa.

Es importante realizar la digitalización de las imágenes con una resolución alta, pues de lo contrario, la separación entre los pixeles de las líneas de contorno en la imagen digital puede resultar muy pequeña, o incluso nula, lo cual es indeseable. La resolución empleada para la digitalización de imágenes en este trabajo fue de 150 dpi (dots per inch), para un mapa topográfico del INEGI a escala 1:50,000.

Las imágenes fueron digitalizadas en color y posteriormente convertidas a imágenes en tonos de gris, con el fin de calcular el histograma de la imagen en ese dominio y obtener un valor de umbral global, el cual fue utilizado para convertir la imagen a blanco y negro (imagen binaria). A esta última, se le aplicó un algoritmo de seguimiento de puntos para extraer la secuencia de los pixeles que conforman a cada una de las curvas de nivel.

No es necesario contar con el conjunto total de pixeles que forman cada curva para tener una definición de éstas. Se realizó entonces un muestreo sobre las curvas. La densidad de muestreo varía con respecto al nivel de detalle local de cada curva, es decir, la curva se muestrea con mayor densidad en las regiones de mayor detalle y con menor densidad en las regiones de menor detalle. Esto se hizo con la finalidad de poder obtener una reconstrucción poligonal fiel de cada curva a partir del conjunto de muestreo.

El nivel de detalle en cada punto p de la curva está dado por la función *TCL(p)* (Tamaño de la Característica Local de p), la cual es el valor de la distancia euclidiana de p hacia el punto más cercano en el eje medio de la curva.

En el conjunto de muestreo se cumple que cada punto p de la curva guarda una distancia proporcional a TCL(p) con respecto a alguna muestra s en el conjunto de muestreo. El valor de la proporción está dado por r. La obtención de una reconstrucción poligonal adecuada está en función del valor utilizado para r; de modo que un valor muy grande influirá en la introducción de aristas no deseadas a la reconstrucción. El valor para r empleado en este trabajo ha sido de 0.415, con lo cual se garantiza que ninguna arista extra será introducida. Esta aseveración fue demostrada en [22] y verificada experimentalmente en este trabajo.

Para el cálculo de la función TCL, primeramente se calculó el eje medio de las curvas empleando mapas de distancias y un filtro pasa altas. El eje medio obtenido tiene la desventaja de no ser continuo y además de no ser completamente exacto, sin embargo es util para estimar la distancia de los puntos de la curva hacia él. Para obtener estas distancias, se aplicó la transformada de la distancia sobre la imagen del eje medio. La imagen resultante sirve como una tabla de consulta para el TCL de los puntos de la curva.

La segunda etapa de este trabajo, corresponde a la interpolación de los contornos. En esta etapa se siguió una metodología para la obtención del modelo que defina la reconstrucción de un terreno, la cual está basada en las construcciones geométricas de la corteza y el esqueleto del conjunto de puntos muestreados; estas construcciones están basadas a su vez dos construcciones geométricas básicas, a saber, el diagrama de Voronoi y la triangulación de Delaunay. La corteza corresponde con la reconstrucción poligonal de las curvas mencionada anteriormente.

El objetivo de calcular ambas construcciones es introducir los puntos del esqueleto al conjunto de puntos que definen a los contornos (puntos de la corteza) para realizar la triangulación del conjunto completo de los puntos y obtener el modelo de una malla de triángulos que defina la superficie del terreno presentando cambios suaves entre los contornos.

Para realizar el cálculo de la corteza y el esqueleto del conjunto de puntos muestreados, se requiere calcular el diagrama de Voronoi y la triangulación de Delaunay del conjunto de puntos. Se hace uso de la prueba estándar *DentroDelCirculo* para determinar cuáles aristas de Delaunay pertenecen a la corteza y cuáles aristas de Voronoi pertenecen al esqueleto. El esqueleto, entonces, está formado por vértices de Voronoi. El cálculo del diagrama de Voronoi y la triangulación de Delaunay fueron realizados con el programa qhull [20].

Algunos de los vértices del esqueleto obtenidos, se encuentran muy cercanos entre sí. Se realizó entonces una reducción del número de puntos en el esqueleto, eliminando aquellos pares de puntos cuya distancia entre ellos sea menor o igual a determinado valor de umbral; en su lugar se agregó el punto medio entre ellos. El valor de umbral utilizado corresponde con el menor valor de la medida de la característica local ($r\mathrm{TCL}(p)$) encontrado en la imagen.

El orden en el que se obtuvieron los puntos del esqueleto no define la trayectoria de éste. Debido a que el esqueleto está formado por varias curvas, en las cuales se presentan ramificaciones y ciclos; los puntos del esqueleto fueron reorganizados en una estructura de grafos, con ramas y troncos.

Se realizó la triangulación de los puntos de la corteza y del esqueleto en conjunto, para obtener la malla de triángulos que define la superficie del terreno.

Los valores de elevación para los puntos de la corteza (curvas de nivel), fueron tomados del mapa topográfico. Los valores de elevación para los puntos del esqueleto fueron calculados en base a las elevaciones de las curvas de nivel. Se identificaron tres casos diferentes entre los puntos del esqueleto, a saber, puntos que se encuentran entre dos contornos diferentes (*tipo 1*), puntos que se encuentran en una cima (*tipo 2*) y puntos rodeados por secciones de contornos reentrantes, es decir, puntos en una rama (*tipo 3*).

El valor de elevación para los puntos del esqueleto de tipo 1, es el promedio de los valores de elevación de los contornos entre los que se encuentran.

Para cada punto p del esqueleto de tipo 2, el valor de elevación que le corresponde, es el valor de elevación del contorno de la cima más el valor del radio del círculo que no contiene ningún punto de la corteza, que está centrado en p y que pasa por al menos tres puntos de la corteza, es decir, el radio del *círculo vacío* centrado en p.

El valor de elevación para cada punto p del esqueleto de tipo 3, está en función del valor del radio del círculo vacío centrado en p y el radio del círculo vacío centrado en el punto que inicia la ramificación.

La información de la corteza, el esqueleto y la triangulación fue organizada y almacenada en tres archivos diferentes. Tal información representa el modelo para la visualización de la reconstrucción del terreno.

Finalmente, se realizó una interfaz gráfica para visualizar la reconstrucción del terreno.

6.2. Trabajo futuro

En la siguiente lista se presenta una serie de características y posibles mejoras para nuestro sistema.

- Extraer de forma automática los pixeles que conforman las curvas de nivel, directamente de la imagen digitalizada en RGB del mapa topográfico.
- Asignar los valores de elevación a las curvas de nivel de manera automática. Esto podría lograrse detectando la anidación entre los contornos, indicando el menor valor de elevación y la separación entre los contornos.
- Trabajar con los datos de las curvas de nivel obtenidos de fuentes diferentes a los mapas topográficos impresos, por ejemplo, de los datos de elevación en continuo que proporciona el INEGI.
- Incorporar la manipulación del terreno por niveles, esto es, permitir la visualización interior del terreno por capas y mostrar información sobre éstas en caso de contar con ella.
- Integrar cantidades masivas de información, agregando porciones terrestres mucho mayores, como por ejemplo toda la República Mexicana.
- Permitir la navegación sobre la superficie terrestre reconstruida y la visualización desde diferentes ángulos, como por ejemplo, permitir un modo de vista como si se sobrevolase el terreno.

Apéndice A

Algoritmo para la extracción de grafos del esqueleto

Aquí se presenta el algoritmo para la extracción de los grafos del esqueleto según se describió en la sección 4.1.5. En el algoritmo se hace uso de tres funciones: *ordenaAristas()*, *encuentraExtremo()* y *encuentraArista()*. A continuación se describe cada una de ellas.

ordenaAristas()

Entrada: La lista de arista del esqueleto $\mathcal{LA}$.

Salida: La lista de aristas ordenada.

Descripción: En el algoritmo se requiere hacer búsquedas en la lista de aristas, es por eso que se hace un ordenamiento de ésta. Los elementos de la lista son de la forma (v_1, v_2), donde v_1 y v_2 son índices de la lista de vértices del esqueleto. En primer lugar, cada elemento de la lista es ordenado de modo que $v_1 < v_2$. Luego la lista completa es ordenada de acuerdo a v_1 y después a v_2. De modo que si se tiene inicialmente la lista:

```
 8 14
15  2
12  8
 2 20
22 20
```

después del ordenamiento quedará:

```
 2 15
 2 20
```

```
 8 12
 8 14
20 22
```

encuentraExtremo()

Entrada: La lista de arista del esqueleto $\mathcal{LA}$ y la lista de vértices del esqueleto $\mathcal{LV}$.

Salida: El primer vértice extremo que se encuentre en el esqueleto.

Descripción: Un vértice extremo es aquel que sólo tiene una arista asociada. Para determinar cuáles vértices son extremos, se crea previamente una lista, $\mathcal{LO}$, con el número de ocurrencias que tienen los vértices del esqueleto en la lista $\mathcal{LA}$. En caso de no encontrarse ningún vértice extremo, la función retorna el primer vértice que encuentre en la lista $\mathcal{LV}$. De no haber más vértices en $\mathcal{LV}$, la función retorna -1.

encuentraArista()

Entrada: El índice, v, de un vértice en la lista $\mathcal{LV}$ y lista de aristas ordenada, $\mathcal{LA}$.

Salida: El índice de la arista en la lista $\mathcal{LA}$, que contenga al vértice v.

Descripción: Se realiza una búsqueda binaria de v sobre la lista $\mathcal{LA}$ y se retorna la posición donde fue hallado. En caso de que v no haya sido encontrado, se retorna -1.

Algoritmo para la extracción de grafos

El algoritmo toma a la entrada dos listas: la lista de vértices $\mathcal{LV}$, y la lista de aristas $\mathcal{LA}$. Cada elemento de la lista de vértices está formado por dos número, (x, y). El formato de la lista de aristas es el que se explicó en la función *ordenaAristas()*.

De manera general, el algoritmo extrae de la lista de aristas todos los grafos que hay en el esqueleto. Funciona como sigue: después de haber ordenado la lista de aristas; se busca una arista, a, que contenga un nodo terminal, v, y se comienza un ciclo dentro del cual se llevará el seguimiento de cada grafo. Se agrega v a la lista $\mathcal{L}$, la

cual será la lista ordenada de los vértices del esqueleto. El otro nodo de a se mantiene en v_{sig} y se elimina la arista de $\mathcal{LA}$. En pr se va acumulando el número de vértices recorridos en la rama o tronco actual.

Se guarda en i el registro de v como nodo que inicia una rama o tronco en el grafo actual. Si el número de puntos recorridos está entre 1 y 2, se guarda en s el registro de v como segundo nodo en la rama. Nótese que es necesario mantener el registro tanto cuando el número de puntos recorridos es 1 como cuando es 2, puesto que en el caso de ramas o troncos de sólo dos nodos, la codificación de la rama es, por ejemplo,

```
1 1 0 2
```

lo cual significa que la rama o tronco inicia en el nodo 1, seguido del nodo 1, con 0 nodos intermedios y termina en el nodo 2. Es por casos como este que el primer vértice también se guarda como el siguiente.

Después, se verifica si el nodo actual, v, inicia alguna ramificación. Si es así, el vértice es agregado a una pila; esto también indica que v es el último nodo de un tronco. Entonces se guarda la codificación del tronco que se acaba de terminar: se almacenan en las listas $\mathcal{LI}$, $\mathcal{LS}$, $\mathcal{LN}$ y $\mathcal{LT}$, los registros de la codificación del tronco, mantenidos anteriormente en i, s y pr; el nodo final es v. En la lista $\mathcal{LI}$ se almacenarán todos los nodos iniciales de cada rama y cada tronco, en $\mathcal{LS}$ los nodos siguientes, en $\mathcal{LN}$ los números de los nodos intermedios y en $\mathcal{LT}$ los nodos finales. Recordemos que v es el índice de un vértice de la lista inicial $\mathcal{LV}$, sin embargo, en las listas que se acaban de mencionar, deben guardarse los índices de los vértices en la nueva lista $\mathcal{L}$. En pr se tiene el número total de vértices recorridos en el tronco, incluidos el nodo inicial, el siguiente y el final, pero de acuerdo a la codificación planteada en la sección 4.1.5, se debe registrar el número de vértices entre el nodo siguiente (s) y en nodo final; es por eso que el número que se almacena es $pr - 3$.

Puesto que se ha terminado de recorrer un tronco, se guarda nuevamente en i el registro de v como nodo inicial para la siguiente rama o tronco. El número de puntos recorridos, pr, se pone a 1, pues ya se ha contado el primer nodo de la siguiente rama/tronco, que es el nodo inicial v.

Para continuar con el recorrido del grafo, se busca otra arista asociada a v_{sig}, y se repite el proceso con los valores de v_{sig} y de la nueva arista. En el caso de que no se haya encontrado ninguna otra arista asociada, significa que se ha llegado al final de una rama. Entonces se guardan sus datos de codificación. Antes de guardar los datos, se verifica si el número de puntos recorridos, pr, es menor o igual que dos, por que puede presentarse el caso de que v_{sig} esté terminando una rama de sólo 2 nodos; entonces el valor del nodo siguiente, s, tiene que actualizarse antes de ser guardado. Hasta este punto, en pr se han contado todos los nodos recorridos en la rama excepto el final (v_{sig}), por eso, en los datos de codificación se guarda $pr - 2$ y no $pr - 3$ como

en el caso de los troncos.

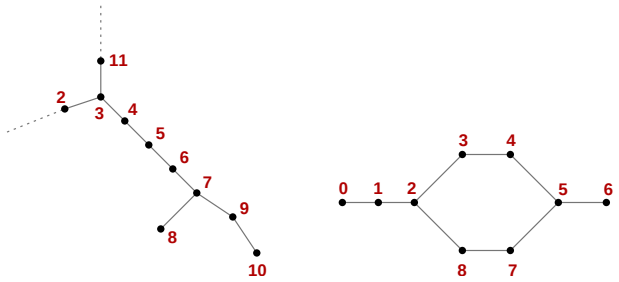
Considérese el caso de la figura A.1(a). Cuando durante el recorrido de los grafos se ha llegado al nodo 7, después de haber determinado que él termina un tronco, el valor para s es 4 y para pr es 1. Después se analiza si el vértice siguiente, v_{sig} (8 en la figura), termina una rama, lo cual resulta positivo; entonces deben guardarse los datos de codificación de la rama $(i, s, pr - 2, v_{sig})$. Si se hace el almacenamiento sin haber hecho ninguna verificación previa, los datos guardados serían

```
7 4 0 8
```

en lugar de ser

```
7 7 0 8
```

es por eso que la verificación $pr \leq 2$ es necesaria.



(a) (b)

Figura A.1: (a) Esqueleto con ramificaciones. (b) Esqueleto con un ciclo.

Después de haber guardado los datos de codificación de la rama, deben extraerse elementos de la pila hasta encontrar alguno que aún tenga aristas asociadas, pues es posible que al recorrer algún camino de el grafo se eliminen aristas que estaban asociadas a algún nodo que se introdujo previamente en la pila; entonces al sacar a dicho vértice de la pila, ya no hay camino que recorrer a partir de él. Tal caso se presenta en el esqueleto de la figura A.1(b); durante el recorrido, el nodo 2 fue introducido a la pila, pues inicia una ramificación. Lo mismo sucedió con el nodo 5. Al llegar al nodo 6, se terminó una rama, entonces se sacó de la pila al nodo 5 y se recorrió el camino (5-7-8-2). Posteriormente, se saca el vértice 9, el cual ya no tiene aristas asociadas, pues éstas se van eliminando conforme se van recorriendo.

Cuando no se encuentren más aristas por recorrer, se ha terminado de recorrer un grafo. Entonces se busca nuevamente otra arista que contenga un nodo terminal para continuar con otro grafo. Cuando no se encuetren más nodos terminales, se ha terminado la extracción de los grafos del esqueleto. El procedimiento es el que se muestra a continuación.

Entrada: La lista, $\mathcal{LV}$, de vértices del esqueleto y la lista, $\mathcal{LA}$, de aristas del esqueleto.

Salida: La lista de vértices reorganizada, $\mathcal{L}$, con p elementos; el número de grafos encontrados en el esqueleto, ng; la lista del número de ramas y troncos por grafo, $\mathcal{LR}$, con ng elementos; y las listas de codificación de los grafos, $\mathcal{LI}, \mathcal{LS}, \mathcal{LN}y\mathcal{LT}$, cada una con r elementos.

$ng \leftarrow 0$ {*Número de grafos en el esqueleto*}
$p \leftarrow 0$ {*Número total de puntos en el esqueleto*}
$r \leftarrow 0$ {*Número total de ramas y troncos en el esqueleto*}

$\mathcal{LA} \leftarrow$ ordenaAristas($\mathcal{LA}$)
$(v, a) \leftarrow$ encuentraExtremo($\mathcal{LA}, \mathcal{LV}$)

while $v \neq -1$ {*Mientras se haya encontrado un vértice extremo*} **do**
  $nr \leftarrow 0$ {*Número de ramas y troncos en el grafo*}
  $pr \leftarrow 0$ {*Número de puntos en la rama/tronco*}
  $\mathcal{L} \leftarrow \mathcal{L} + v$
  $p \leftarrow p + 1$
  $i \leftarrow v$ {*Se guarda el nodo inicial*}

  while $a \neq -1$ {*Mientras haya aristas en el grafo*} **do**
    $v\_sig \leftarrow$ el otro vértice de la arista a
    if $v\_sig \notin \mathcal{L}$ **then**
      $\mathcal{L} \leftarrow \mathcal{L} + v$
      $p \leftarrow p + 1$

    $\mathcal{LA} \leftarrow$ eliminaArista($a, \mathcal{LA}$)
    $pr \leftarrow pr + 1$

    if $pr \leq 2$ **then**
      $s \leftarrow v$ {*Se guarda el nodo siguiente*}

    if v inicia ramificación **then**
      push(v)

      if $pr > 1$ {*Se terminó un tronco*} **then**
        {*Se almacenan los datos del tronco*}
        $\mathcal{LI} \leftarrow \mathcal{LI}+$ índice del vértice i en la lista $\mathcal{L}$
        $\mathcal{LS} \leftarrow \mathcal{LS}+$ índice del vértice s en la lista $\mathcal{L}$
        if $pr > 2$ **then**
          $n \leftarrow pr - 3$ {*Se restan los puntos inicial, siguiente y final*}

else
$n \leftarrow 0$
$\mathcal{LN} \leftarrow \mathcal{LN} + n$
$\mathcal{LT} \leftarrow \mathcal{LT}+$ índice del vértice v en la lista $\mathcal{L}$

$i \leftarrow v$ {*Se guarda el nodo inicial*}
$pr \leftarrow 1$ {*Se pone a 1 pues el primer nodo de la siguiente rama/tronco (v) ya se ha contado*}
$nr \leftarrow nr + 1$ {*Se cuenta un tronco más*}
$r \leftarrow r + 1$

$a \leftarrow$ encuentraArista($v\_sig, \mathcal{LA}$)
if $a = -1$ {*Si no se hallaron más aristas, se terminó una rama*} **then**
if $pr \leq 2$ **then**
$s \leftarrow v$ {*Se guarda el nodo siguiente, para el caso de ramas de 2 nodos*}

{*Se almacenan los datos de la rama*}
$\mathcal{LI} \leftarrow \mathcal{LI}+$ índice del vértice i en la lista $\mathcal{L}$
$\mathcal{LS} \leftarrow \mathcal{LS}+$ índice del vértice s en la lista $\mathcal{L}$
if $pr > 2$ **then**
$n \leftarrow pr - 2$ {*Se restan los puntos inicial y siguiente. El nodo final (v_sig) aún no se ha contado*}
else
$n \leftarrow 0$
$\mathcal{LN} \leftarrow \mathcal{LN} + n$
$\mathcal{LT} \leftarrow \mathcal{LT}+$ índice del vértice v_sig en la lista $\mathcal{L}$

$pr = 0$
$nr \leftarrow nr + 1$
$r \leftarrow r + 1$

{*Se sacan elementos de la pila hasta encontrar alguno que aún tenga aristas asociadas*}
while $a = -1$ y la pila no esté vacía **do**
pop(v_sig)
$i \leftarrow v\_sig$
$a \leftarrow$ encuentraArista($v\_sig, \mathcal{LA}$)

$v \leftarrow v\_sig$

$(v, a) \leftarrow$ encuentraExtremo($\mathcal{LA}, \mathcal{LV}$)
$\mathcal{LR} \leftarrow \mathcal{LR} + nr$
$ng \leftarrow ng + 1$

Apéndice B

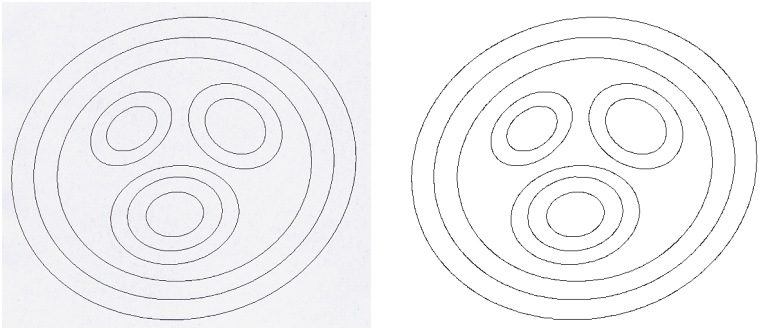
Ejemplos de Aplicación

En este apartado se presentan, como ejemplo, las imágenes tomadas con nuestra interfaz gráfica, de tres reconstrucciones:

1. Una superficie ficticia creada a partir de elipses.
2. La superficie de un par de cerros ubicados en la zona montañosa del estado de Veracruz.
3. La superficie de la Malinche.

B.1. Elipses

Para este ejemplo, el mapa de contornos se generó con la herramienta Xfig, después se imprimió y posteriomente se digitalizó para obtener una imagen semejante a las imágenes obtenidas tras dibujar las líneas de contorno de los mapas topográficos sobre una hoja blanca. En la figura B.1 se presenta la imagen digitalizada y la imagen binaria con la que se trabajó para la reconstrucción.



(a) (b)

Figura B.1: (a) Imagen digitalizada de la superficie formada con elipses. (b) Imagen binaria correspondiente a la imagen en (a).

En la figura B.2 se presenta la imagen de la corteza, el esqueleto y la triangulación de la superficie.

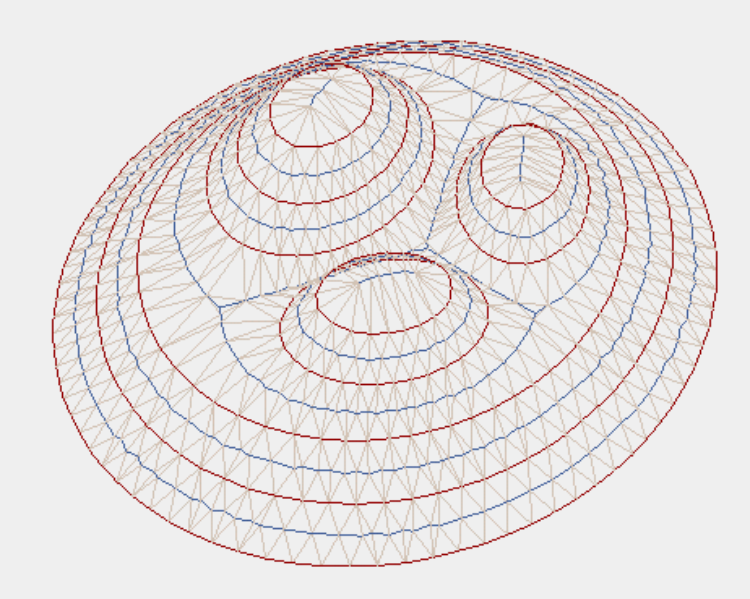


Figura B.2: Corteza, esqueleto y triangulación de la superfice de la figura B.1.

En la imagen que se presenta en la figura B.3 se ilustra la superficie coloreada usando los colores asignados de acuerdo a los valores de elevación, y en la imagen de figura B.4 la superficie coloreada utilizando sólo dos colores para la degradación.

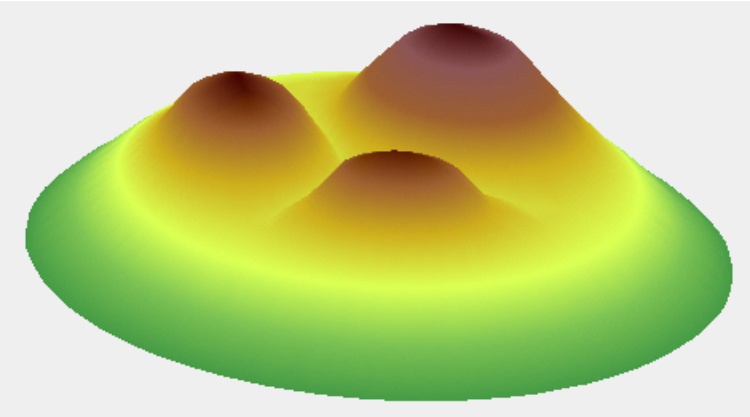


Figura B.3: Superfice de la figura B.1 coloreada usando los colores asignados de acuerdo a los valores de elevación.

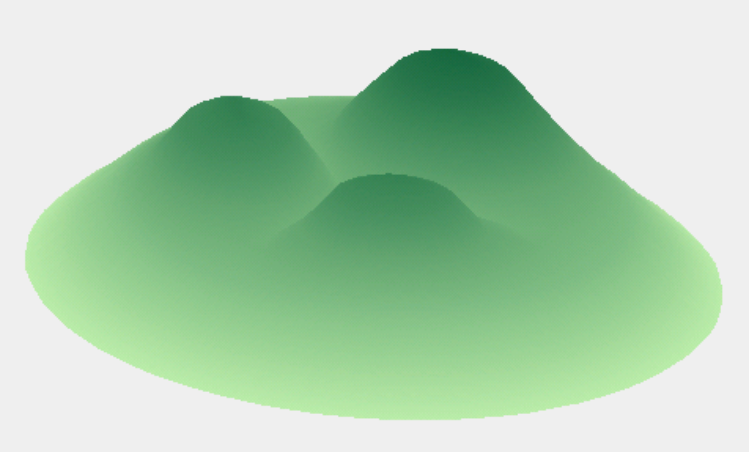
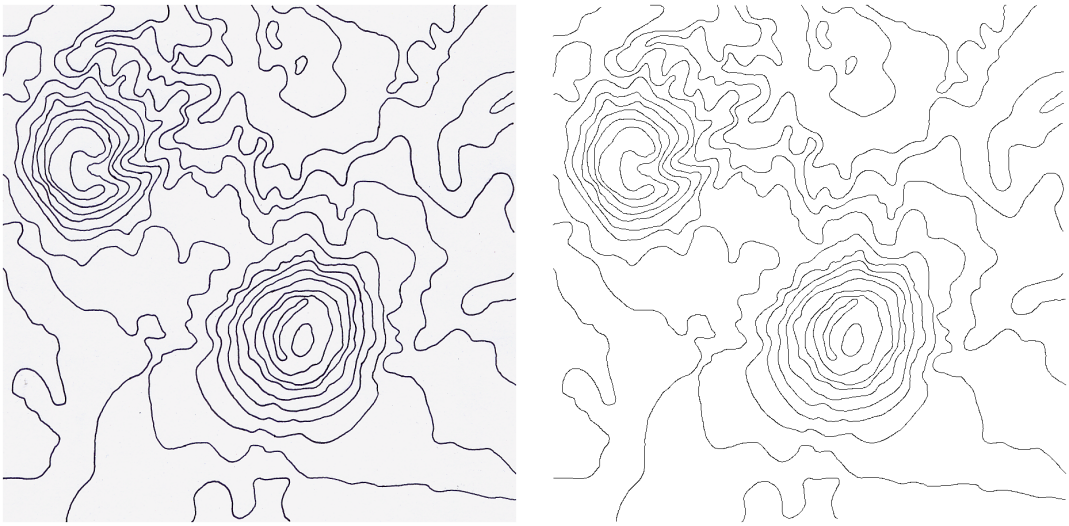


Figura B.4: Superfice de la figura B.1 coloreada usando sólo dos colores para la degradación.

B.2. Cerros

En la figura B.5 se muestran las imágenes digitalizada y binaria del mapa de contornos correspondiente al par de cerros ubicados en el estado de Veracruz, los cuales fueron empleados como un ejemplo de reconstrucción en este trabajo de tesis. La imagen se obtuvo del mapa topográfico E14B74 del INEGI.



(a) (b)

Figura B.5: Imágenes digitalizada y binaria del mapa de contornos del par de cerros en la región del estado de Puebla.

En la figura B.6 se presenta la imagen de la corteza, el esqueleto y la triangulación

de la superficie de los cerros.

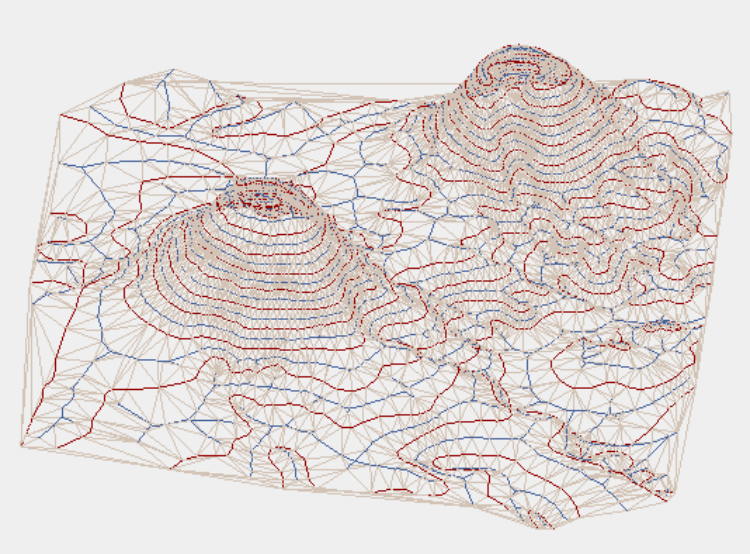


Figura B.6: Corteza, esqueleto y triangulación de la superfice de la figura B.5.

En la imagen de la figura B.7 se ilustra la superficie coloreada usando los colores asignados de acuerdo a los valores de elevación, y en la imagen de figura B.8 la superficie coloreada utilizando sólo dos colores.

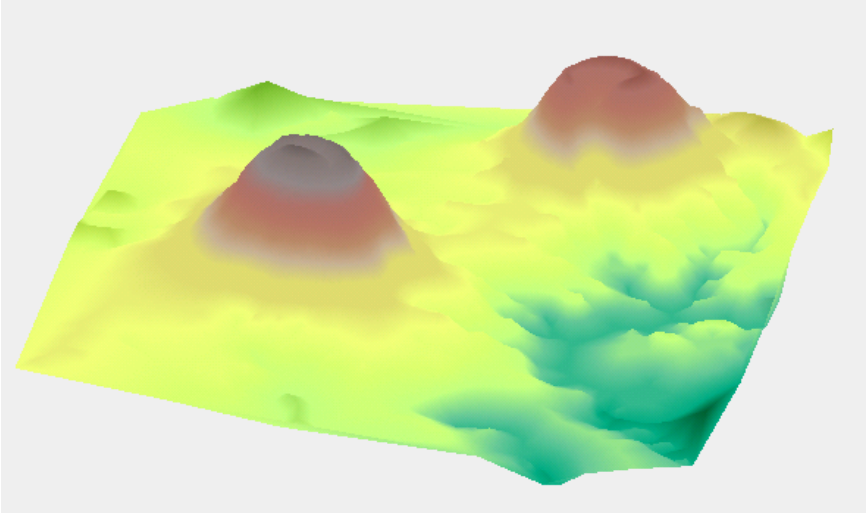


Figura B.7: Superfice de la figura B.5 coloreada usando los colores asignados de acuerdo a los valores de elevación.

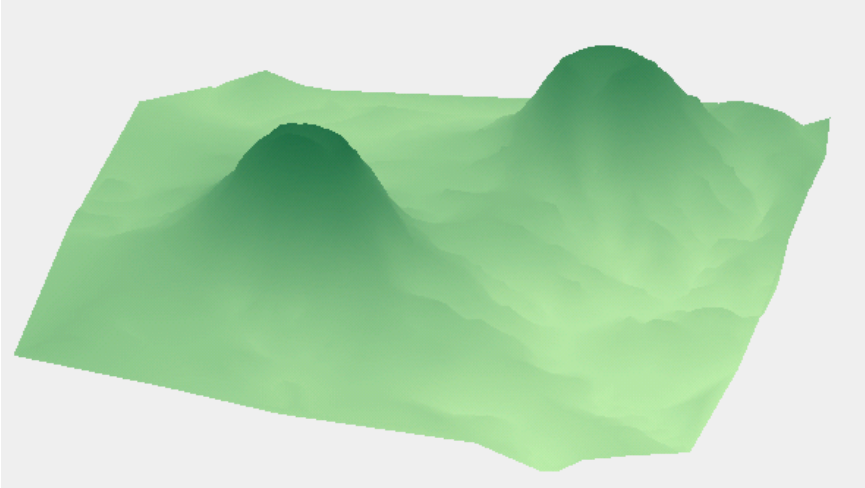
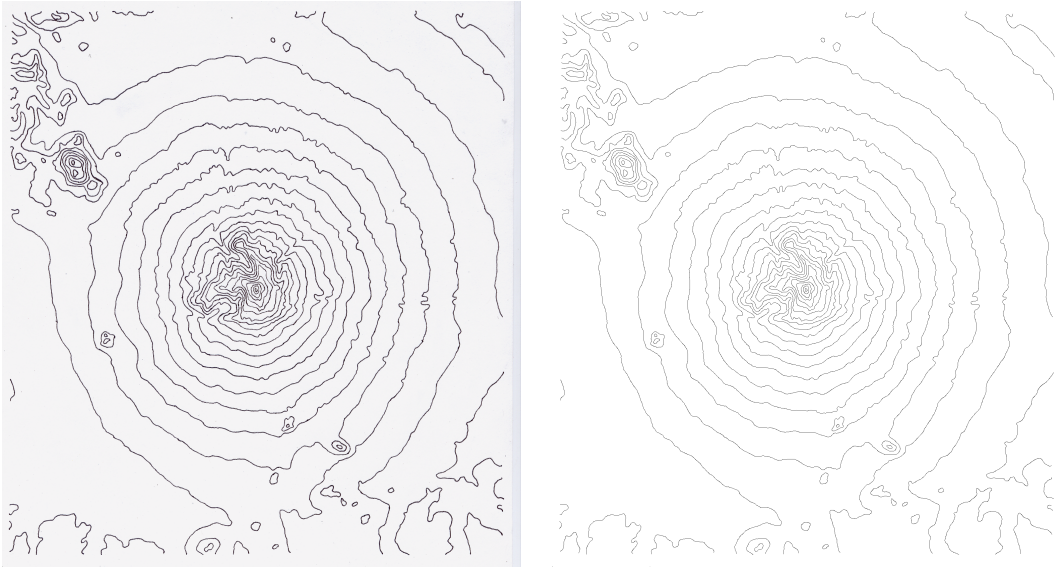


Figura B.8: Superfice de la figura B.5 coloreada usando sólo dos colores para la degradación.

B.3. Malinche

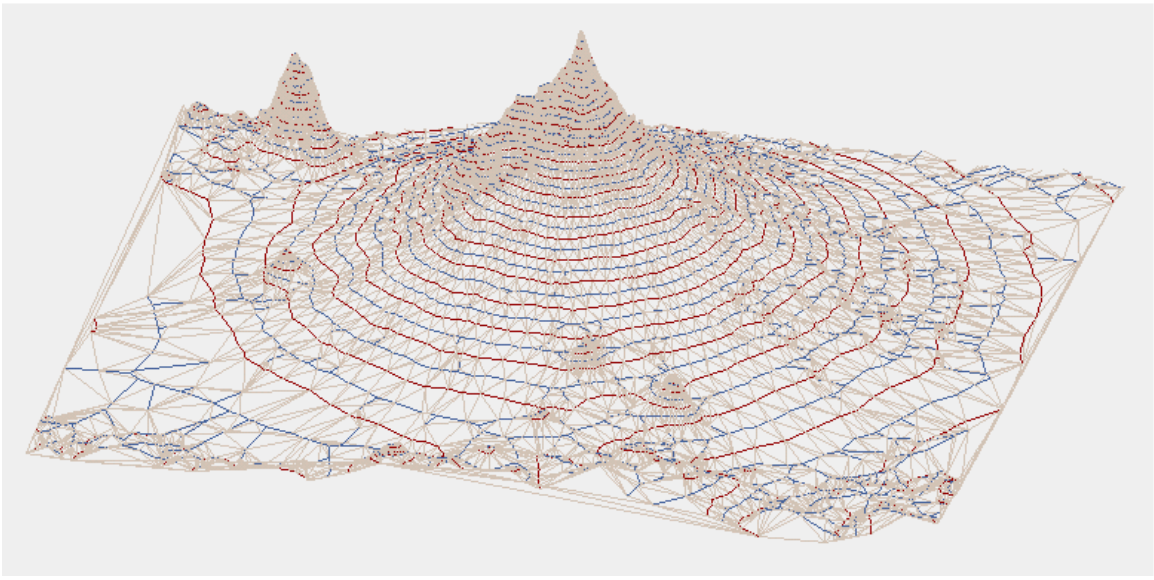
En la figura B.9 se muestran las imágenes digitalizada y binaria del mapa de contornos de la superficie del último ejemplo que aquí se presenta: la Malinche. El mapa de contornos fue tomado del mapa virtual de la República Mexicana que presenta el INEGI en su portal de Intenet [2], el cual está a una escala de 1:250,000.



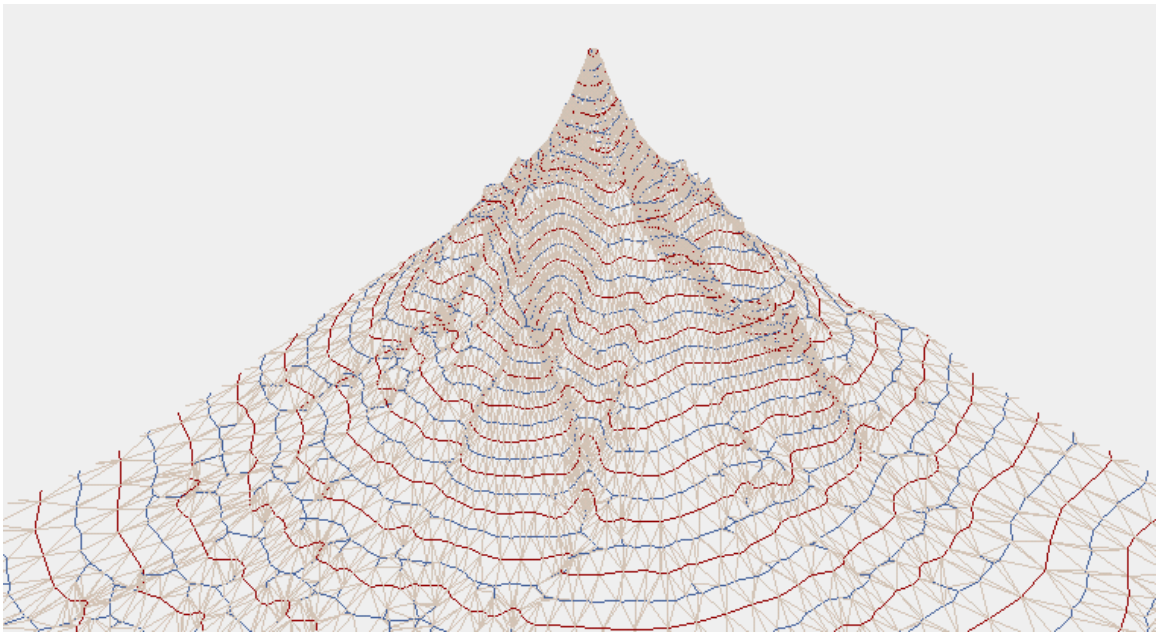
(a) (b)

Figura B.9: Imágenes digitalizada y binaria del mapa de contornos la Malinche.

En la figura B.10(a) se presenta la imagen de la corteza, el esqueleto y la triangulación de la superficie de la Malinche, y en la figura B.10(b) se presenta un acercamiento al pico de la superficie.



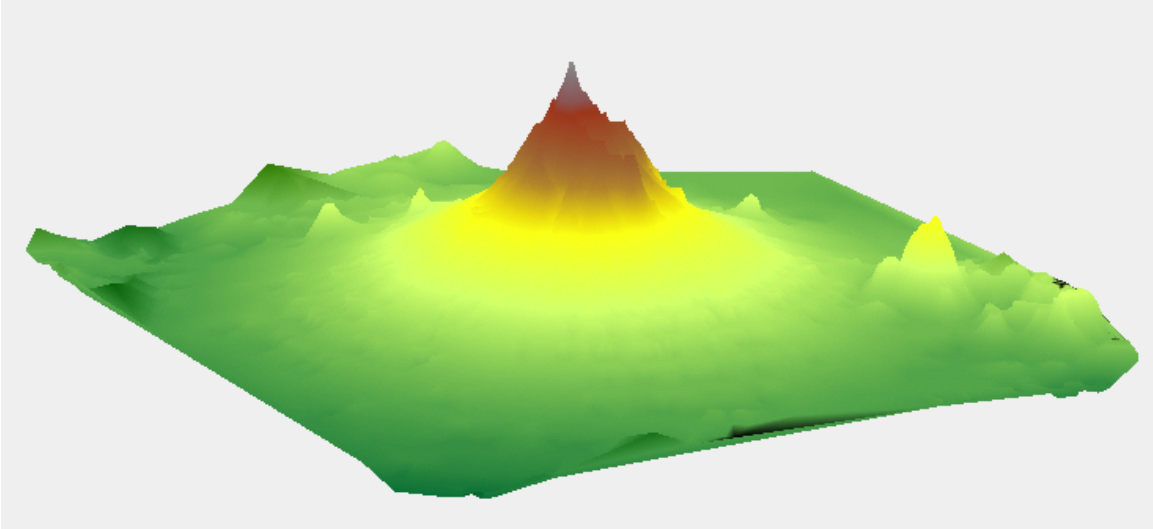
(a)



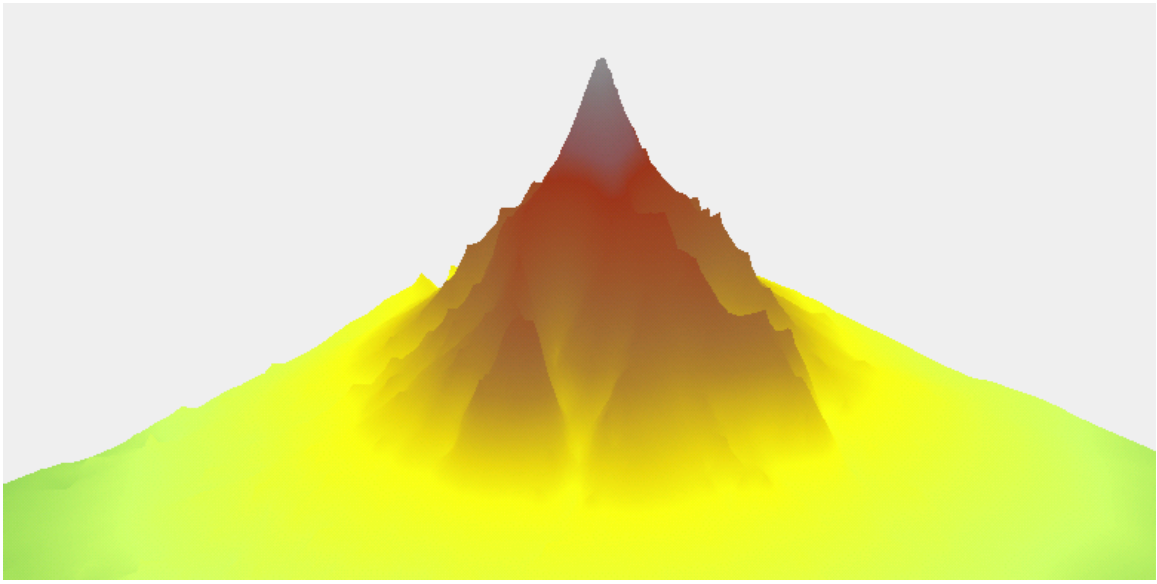
(b)

Figura B.10: (a) Corteza, esqueleto y triangulación de la superfice de la figura B.9. (b) Acercamiento al pico de la superficie.

La superficie coloreada empleando los colores asignados con respecto a los valores de elevación, se muestra en la figura B.11(a), y la superficie coloreada usando la interpolación con sólo dos colores, en la figura B.12(a). En las partes (b) de ambas figuras se ilustra un acercamiento al pico de la Malinche.

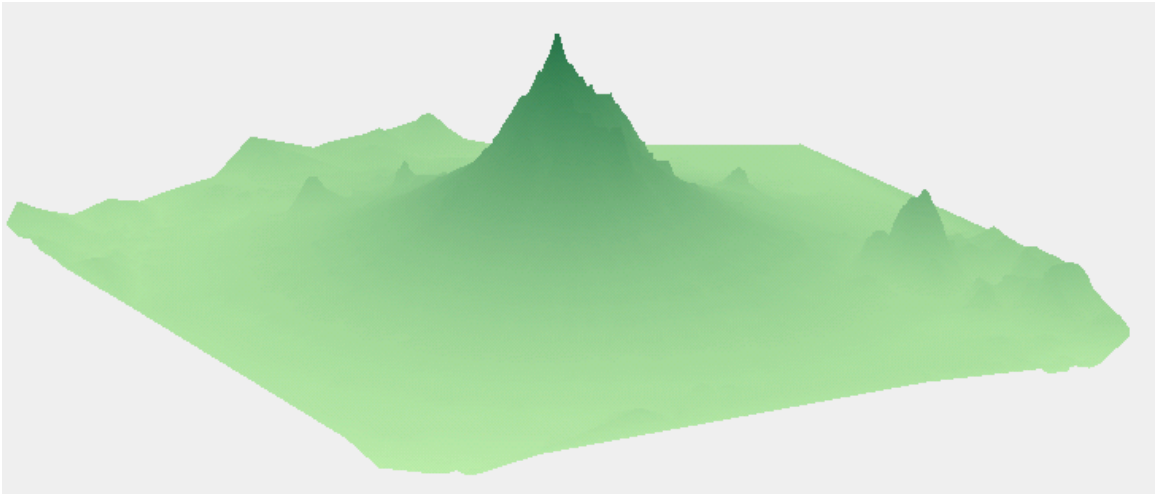


(a)

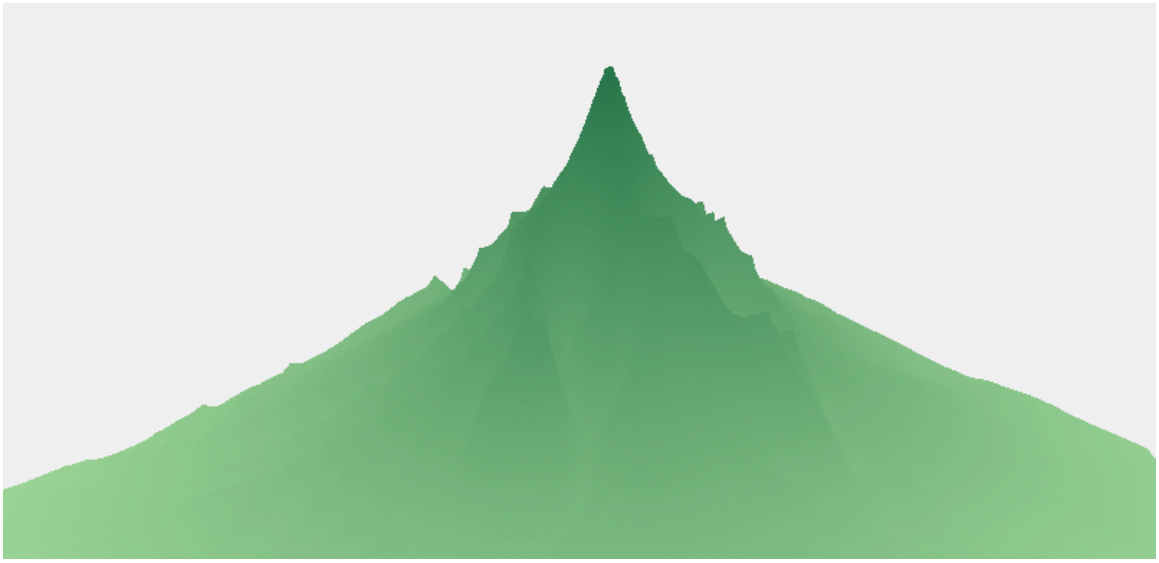


(b)

Figura B.11: (a) Superfice de la figura B.9 coloreada usando los colores asignados de acuerdo a los valores de elevación. (b) Acercamiento al pico de la superficie.



(a)



(b)

Figura B.12: (a) Superfice de la figura B.9 coloreada usando sólo dos colores para la degradación. (b) Acercamiento al pico de la superficie.

Apéndice C

Densidad de Muestreo

En este apartado, se presentan algunas de las imágenes correspondientes a muestreo de las curvas de nivel del mapa del par de cerros, presentado en el apéndice B, utilizando densidades de muestreo diferentes.

La densidad de muestreo varía de acuerdo al valor que se asigne a r en el algoritmo de muestreo. De acuerdo a [22] si se emplea un valor para $r < 0.42$ se garantiza una densidad de muestreo adecuada, de modo que en el proceso de la construcción de la corteza del conjunto de muestreo se conecten aristas sólo entre cada par de puntos adyacentes.

Hemos empleado diferentes valores de r para ilustrar lo que sucede con las construcciones de la corteza y del esqueleto del conjunto de muestreo de las curvas de nivel.

C.1. Valor de $r = 0.415$

En la imagen de la figura C.1 se presenta el resultado del muestreo utilizando un valor para $r = 0.415$.

En la figura C.2 se muestra la gráfica de la corteza y del esqueleto para este conjunto de muestreo.

Puede observarse que la corteza corresponde a la reconstrucción poligonal de las curvas, es decir, no se introdujeron aristas extras.

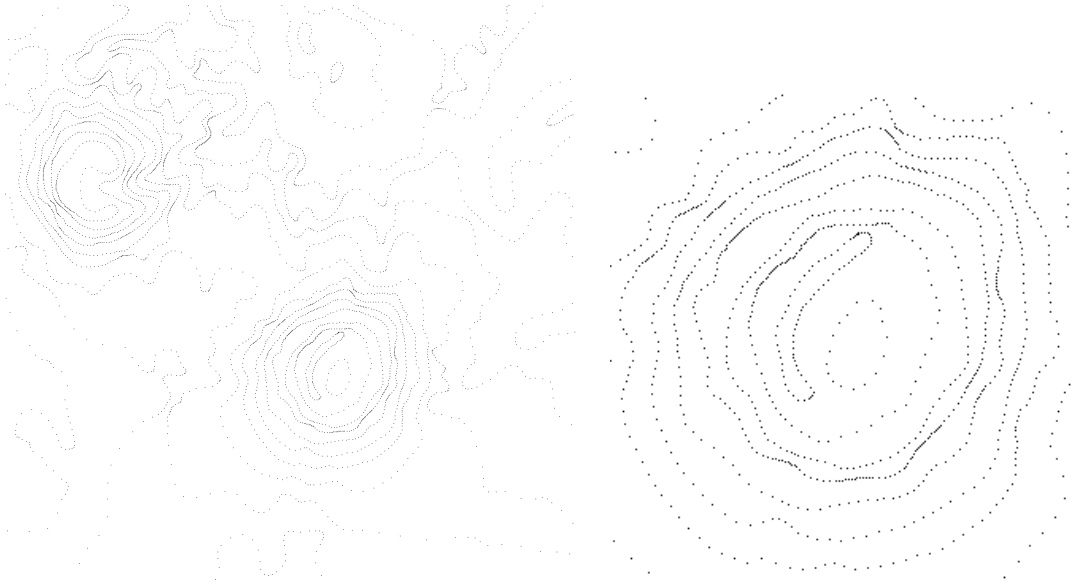


Figura C.1: Resultado del muestreo empleando un valor de $r = 0.415$.

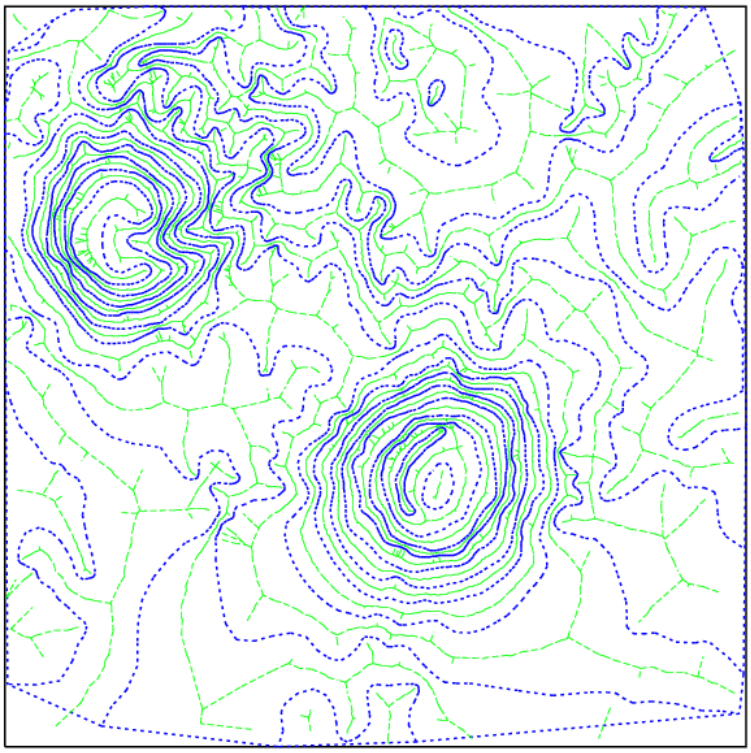


Figura C.2: Corteza (líneas gruesas) y esqueleto (líneas delgadas) del conjunto de puntos muestreados con $r = 0.415$.

C.2. Valor de $r = 0.6$

En la imagen de la figura C.3 se presenta el resultado del muestreo utilizando un valor para $r = 0.6$, para el cual se obtiene una distancia mayor entre las muestras con respecto al conjunto de muestreo anterior.

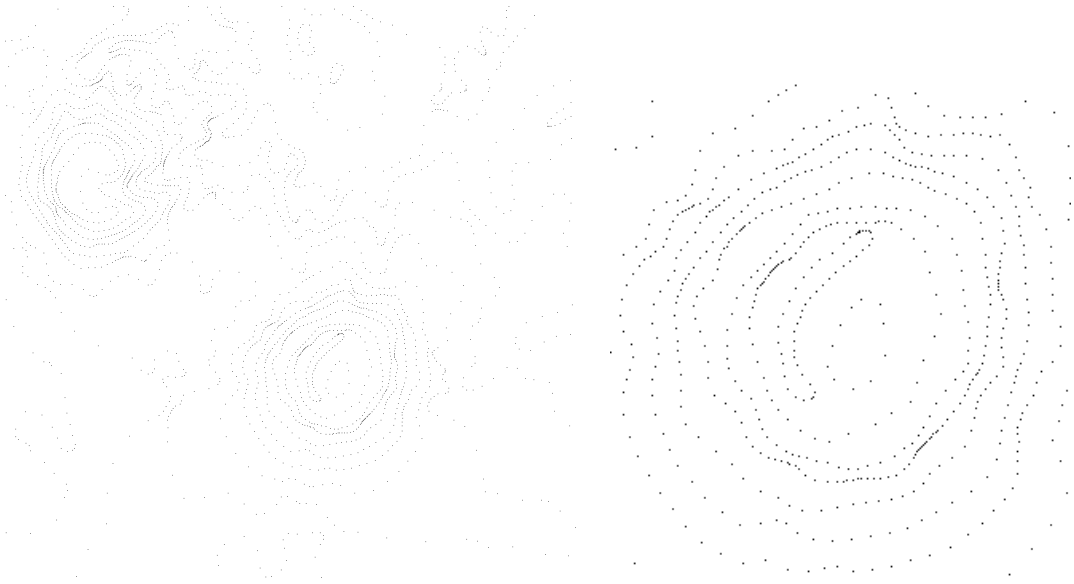


Figura C.3: Resultado del muestreo empleando un valor de $r = 0.6$.

En la figura C.4 se muestra la gráfica de la corteza y del esqueleto para este conjunto de muestreo.

Pueden observarse cambios tanto en la corteza como en el esqueleto con respecto a los que se contruyeron en la sección anterior. En algunas zonas, los cambios no son sustanciales, es decir, permanece la aproximación al esqueleto y a las curvas reales. Sin embargo en otras zonas, los cambios producidos afectan la estructura básica de ambas construcciones como puede apreciarse en el acercamiento presentado en la figura C.5.

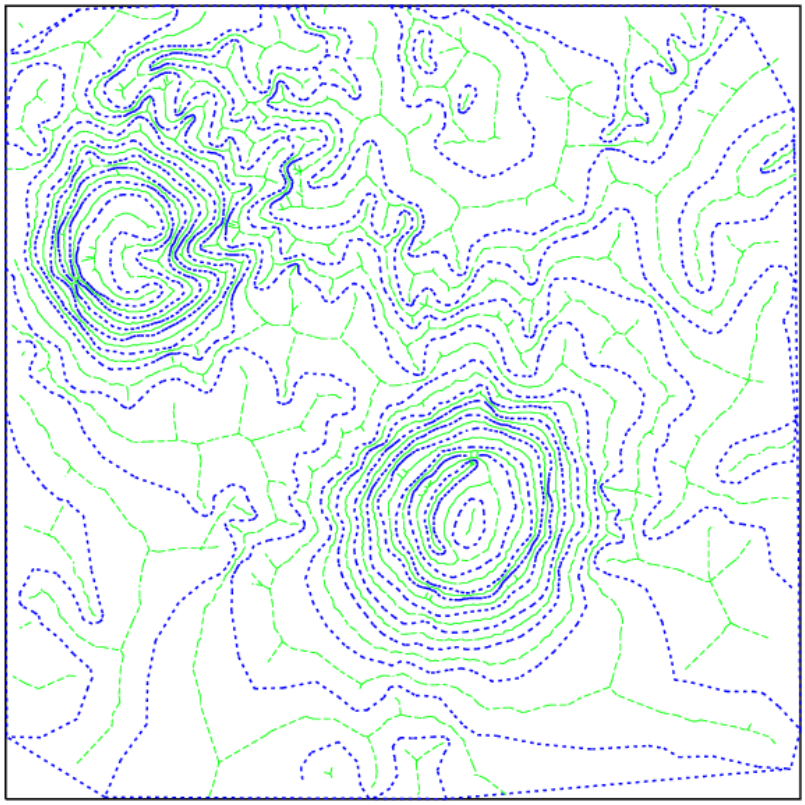


Figura C.4: Corteza (líneas gruesas) y esqueleto (líneas delgadas) del conjunto de puntos muestreados con $r = 0.6$.

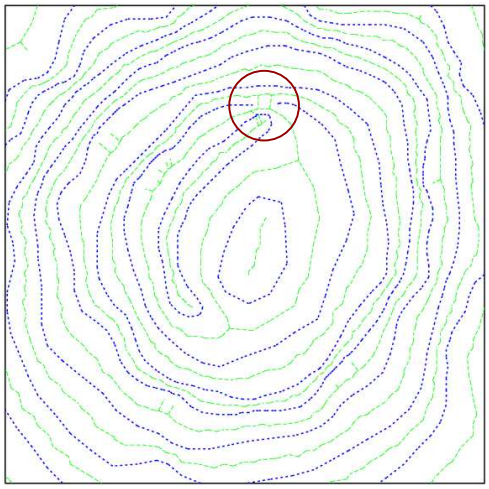


Figura C.5: Acercamiento a una región de la gráfica C.4. En la región marcada con el círculo se presenta una deformación tanto en la reconstrucción poligonal como en el esqueleto.

C.3. Valor de $r = 0.8$

En el muestreo realizado empleando un valor de $r = 0.8$ las deformaciones en el esqueleto y la corteza son mayores en ciertas regiones. En la imagen de la figura C.6 se presenta el resultado de este muestreo.

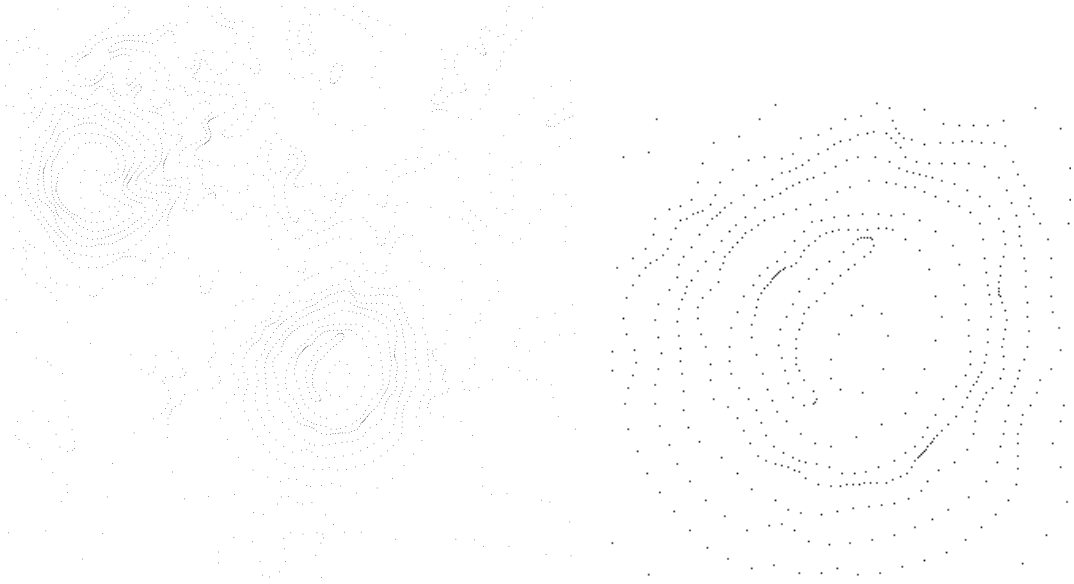


Figura C.6: Resultado del muestreo empleando un valor de $r = 0.8$.

En la figura C.7 se muestra la gráfica de la corteza y del esqueleto para este conjunto de muestreo.

En el acercamiento mostrado en la figura C.8 puede apreciarse la aparición de deformaciones en un mayor número de regiones tanto en la corteza como en el esqueleto.

En resumen, si en el muestreo se emplean valores para r mayores al propuesto en [22], la construcción de la corteza no define la reconstrucción poligonal en todas las regiones de la curva muestreada. La construcción del esqueleto es una aproximación al esqueleto real, la cual presenta ciertas variaciones en función del conjunto de muestreo; según [22], no se garantiza que estas variaciones a lo largo de toda la construcción sean aceptables cuando en el muestreo se usan valores de $r > 0.42$. Esto lo hemos comprobado experimentalmente en este apartado.

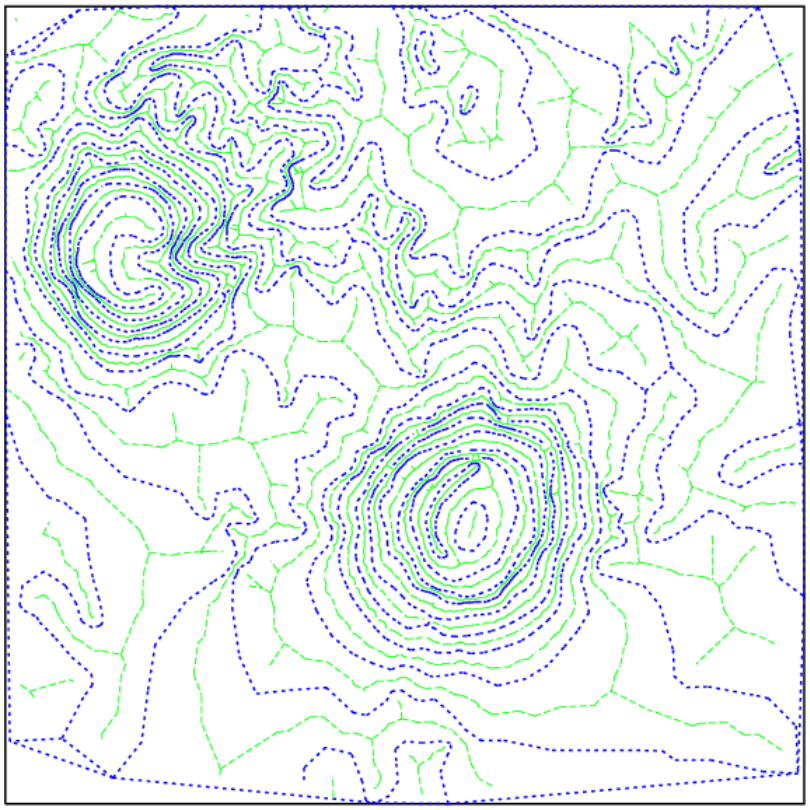


Figura C.7: Corteza (líneas gruesas) y esqueleto (líneas delgadas) del conjunto de puntos muestreados con $r = 0.8$.

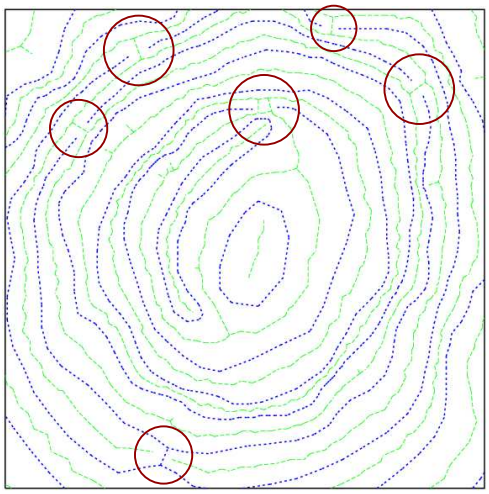


Figura C.8: Acercamiento a una región de la gráfica C.7. Las regiones marcadas con un círculo represetan las zonas con deformación.

Bibliografía

[1] A.M. Felicísimo. *Modelos Digitales del Terreno: Introducción y Aplicaciones en las Ciencias Ambientales*. Pentalfa Ediciones, Oviedo, 1994.

[2] Instituto Nacional de Estadística, Geografía e Informática (INEGI). http://www.inegi.gob.mx.

[3] D. Thibault and C. M. Gold. Terrain reconstruction from contours by skeleton construction. *GeoInformatica*, 4(4):349–373, 2000.

[4] M. B. Gousie and W. R. Franklin. Converting elevation contours to a grid. *Eighth International Symposium on Spatial Data Handling*, pages 647–656, 1998.

[5] D. E. Hamilton T. A. Jones and C. R. Johnson. *Contouring Geologic Surfaces with the Computer*. Van Norstand Reinhold, October 1986.

[6] G. W. Heine. A controlled study of some two-dimensional interpolation methods. *Computer Oriented Geological Society Computer Contributions*, 2(2):60–72, 1986.

[7] C. M. Gold and T. Ross. Surface modelling with guaranteed consistency - an object-based approach. In Ascona, editor, *Proceedings of International Workshop on Advanced Research in Geographic Information Systems IGIS'94*, pages 70–87, 1994.

[8] A. Garcia. A contour line based triangulating algorithm. In E. Corwin & D. Cowen P. Bresnahan, editor, *Proceedings of the Fifth International Symposium on Spatial Data Handling*, volume 2, pages 411–423, Charleston SC USA, 1992.

[9] H. Yamada K. Yamamoto and S. Muraki. Symbol recognition and surface reconstruction from topographic map by parallel method. In *Proceedings of the Second International Conference on Document Analysis and Recognition*, pages 914–917, Tsukuba Science City, Oct 1993.

[10] F. Dupont, M.P. Deseilligny, and M. Gondran. DMT extraction from topographic maps. In *Proceedings of Fifth International Conference on Document Analysis and Recognition. ICDAR'99*, pages 475–478, 1999.

[11] Patrice Arrighi and Pierre Soille. From scanned topographic maps to digital elevation models. In *Geovision '99, International Symposium on Imaging Applications in Geology (ICA)*, University of Liege, Belgium, May 1999.

[12] Salvatore Spinello and Pascal Guitton. Contour line recognition from scanned topographic maps. In *WSCG (Winter School of Computer Graphics)*, Feb 2004.

[13] R. C. Gonzalez and R. E. Woods. *Digital Image Processing.* Adisson-Wesley, 1992.

[14] G. Borgefors. Distance tranformations in digital images. *Comp. V. Graph. and Image Process.*, 34:344–371, 1986.

[15] Kenneth R. Castleman. *Digital Image Processing.* Prentice Hall, 1996.

[16] N. Amenta, M. Bern, and D. Eppstein. The crust and the β-skeleton: combinatorial curve reconstruction. *Graphical Models and Image Processing*, 60(2):125–135, 1998.

[17] D. Attali and A. Montanvert. Computing and simplifying 2D and 3D continuous skeletons. *Computer Vision and Image Understanding*, 67(3):261–273, Sep 1997.

[18] Franco P. Preparata and Michael I. Shamos. *Computational geometry: an introduction.* Springer-Verlag New York, Inc., New York, NY, USA, 1985.

[19] G. A. Brook K. Wang, C-P. Lo and H. R. Arabnia. Comparison of existing triangulation methods for regularly and irregularly spaced height fields. *International Journal of Geographical Information Science*, 15(8):743–762(20), Ded 2001.

[20] C.B. Barber, D.P. Dobkin, and H. Huhdanpaa. The quickhull algorithm for convex hull. *ACM Transactions on Mathematical Software*, 22(4):469–483, 1996.

[21] J. Cornejo Herrera, A. Lara López, R. Landa Becerra, and L.G. de la Fraga. Una librería para procesamiento de imagen: scimagen. In *VIII Conferencia de Ingeniería Eléctrica*, 4, 5 y 6 de septiembre, 2002. Cinvestav.

[22] C. M. Gold and J. Snoeyink. A one-step crust and skeleton extraction algorithm. *Algorithmica*, pages 144–163, 2001.

[23] C.M. Gold and M. Dakowicz. Visualizing terrain models from contours – plausible ridge, valley and slope estimation. In *Proceedings of the International Workshop on Visualization and Animation of Landscape.* Kumming, China, 2002.

[24] L. Guibas and J. Stolf. Primitives for the manipulation of general subdivisions and the computation of voronoi diagrams. *ACM Transactions on Graphics*, 4:74–123, 1985.

[25] The Mesa 3D Graphics Library. http://www.mesa3d.org.

[26] Troll Tech - Qt overview. http://www.trolltech.com/products/qt/.