Universidad de las Ciencias Informáticas

Facultad 5

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

MÓDULO DE DESCRIPCIÓN DE CONTORNOS EN IMÁGENES MÉDICAS DIGITALES

Autor: Anay Sosa Santos

Tutor (es): Ing. Ernesto Carrasco de la Torre

Lic. Alejandro Alonso Fuster

Cotutor: MSc. Osvaldo Pereira Barzaga

Ciudad de la Habana, Cuba

2010-2011

Declaración de Autoría

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____

_____

Anay Sosa Santos

_____ _____

Ing. Ernesto Carrasco De la Torre

Tutor

Lic. Alejandro Alonso Fuster

Tutor

_____

MSc. Osvaldo Pereira Barzaga

Co-Tutor

Datos de Contacto

Tutor: Ing. Ernesto Carrasco De la Torre.

Edad: 24.

Ciudadanía: Cubano.

Institución: Universidad de las Ciencias Informáticas (UCI).

Título(s): Ingeniero en Ciencias Informáticas.

Categoría Docente: Instructor recién graduado

E-mail: ecarrasco@uci.cu

Graduado de la UCI, con dos años de experiencia en el tema de la Gráfica Computacional y profesor de un proyecto de Realidad Virtual en la Universidad de las Ciencias Informáticas.

Co-Tutor: Lic. Alejandro Alonso Fuster.

Edad: 30.

Ciudadanía: Cubano.

Institución: Universidad de las Ciencias Informáticas (UCI).

Título(s): Licenciatura en Matemática.

Categoría Docente: Profesor asistente.

E-mail: alejadroaf@uci.cu

Co-Tutor: MSc. Osvaldo Pereira Barzaga.

Edad: 26.

Ciudadanía: Cubano.

Institución: Universidad de las Ciencias Informáticas (UCI).

Título: MSc. en Informática Aplicada.

Categoría Docente: Instructor.

E-mail: opereira@uci.cu

Graduado de la UCI, con cuatro años de experiencia en el tema de la Gráfica Computacional y líder de un proyecto de Realidad Virtual en la Universidad de las Ciencias Informáticas.

Dedicatoria

A mi Poli (mami):

Esa persona tan especial y maravillosa que ha sido mi amiga, mi hermana, mi ángel protector, que me ha guiado siempre por un buen camino dándome siempre el mejor consejo y apoyándome en todas las decisiones que he tomado en mi joven vida. Gracias por ser una fuente de inspiración para seguir adelante en los momentos más difíciles. TE AMO.

A mi Ñango (papi):

Ese hermano y amigo de toda la vida que siempre me ha dado todo su cariño y me enseñó a ser la persona que soy, el padre más lindo del mundo, el que nunca regaña el que sabiamente me dio sus consejos, el que me montaba en su bici y era para todos lados con él. Gracias por todo, TE AMO.

Agradecimientos

Quisiera agradecerle a cada una de las personas que de una forma u otra han contribuido a mi formación intelectual y han hecho posible la conclusión de este trabajo.

A mis padres:

Que han sido, son y serán lo más importante en mi vida, los amo con todo mi corazón, gracias por todo su apoyo, sus consejos y cariño, gracias por ayudar en mi formación, por guiarme por un buen camino, gracias por criarme en un ambiente de amor, respeto y confianza. Los tres estaremos siempre unidos, LOS AMO.

A mis tutores, principalmente a Ernesto, por guiarme siempre en el trabajo, por apoyarme, por darme fuerzas y aliento de que sí se podría. Él me enseñó que con esfuerzo y sacrificio es posible alcanzar las metas que nos tracemos. Eres súper especial para mí, te estaré agradecida por siempre, gracias por tu paciencia, gracias por tu amistad.

A mi tía Estrella:

Mi segunda madre, la cual me enseñó muchas de las cosas que sé, con la que desde chiquita ando y la que me ha dado mucho amor de madre.

A mi amiga y hermana Suset porque desde que entré en primer año hemos estado juntas aceptándonos como somos, formando el dúo inseparable, gracias por todos tus consejos, cariño y apoyo.

A Yoander:

Gracias por tu todo tu apoyo y cariño que durante estos 4 cursos me has brindado, gracias por demostrarme que con esfuerzo, dedicación y espera todo es posible, de todo corazón GRACIAS.

AGRADECIMIENTOS

A mi familia:

Quiénes desde el día en que nací me han apoyado y me han brindado todo su amor y cariño.

A mis profesores:

A mis profesores, que me han enseñado todo lo que sé desde que comencé a dar mis primeros pasos en los estudios desde la primaria hasta la universidad.

A mis compañeros y amigos:

A mis compañeros de laboratorio, de aula, de trabajo de la FEU, de apartamento a ellos gracias por el cariño que siempre me han brindado y porque siempre los llevaré presente, de todos aprendí algo importante. A mis amigas Lisbeth, Adis y Maybe, por todo su apoyo, cariño, paciencia en los momentos difíciles que me ha tocado vivir y su ánimo de seguir adelante. A mis amigos Juli sabes que te quiero un montón, Adrián por su amistad y sus chistes, a todos los que por cuestión de espacio no menciono, GRACIAS.

A la UCI:

A esta hermosa Universidad que me ha formado como profesional y me compromete más con este valioso proyecto que supera sus resultados cada día.

A Fidel:

A nuestro máximo líder y conductor de la Revolución Cubana, el Comandante Fidel, el principal creador de esta universidad que me ha dado la oportunidad de convertirme en un profesional lista para servir a la Revolución.

Resumen

Gracias a los avances tecnológicos de las últimas décadas los especialistas de la medicina disponen de nuevas potencialidades para la observación y el diagnóstico de enfermedades, es el caso de las Tomografías Computarizadas (CT) y las Resonancias Magnéticas (MRI). En la Universidad de las Ciencias Informáticas (UCI) se desarrolla el proyecto de visualización médica VisMedic, dedicado al procesamiento de imágenes, la visualización y reconstrucción de volúmenes, y la identificación de estructuras anatómicas relevantes. Cuenta con técnicas de segmentación como parte de su solución, pero necesita algoritmos para la descripción de contornos pues de esta manera se obtienen características cuantitativas que serían utilizadas en el reconocimiento de patrones.

Este trabajo propone la implementación de descriptores simples y de Fourier por su facilidad de cálculo, información asociada e invariancia ante las transformaciones geométricas. Lo que permitió que se obtuviera un módulo para definir características cuantitativas de contornos segmentados, así como permitir realizar transformaciones a dichos contornos constituyendo la entrada fundamental para el reconocimiento de patrones en imágenes digitales.

Palabras clave: Descriptores de Fourier, imágenes médicas digitales, invariancia, transformaciones de contornos.

Índice

Índice de Figuras

Índice de tablas

Introducción

Durante las últimas décadas el vertiginoso desarrollo de la Realidad Virtual (RV) ha permitido al hombre simular distintas realidades e interactuar con ella en función de disímiles objetivos, todo esto ha generado cambios de filosofía y grandes movimientos científicos en las áreas de conocimiento afines. Hoy en día las aplicaciones de la RV constituyen un salto evolutivo en las ciencias informáticas, debido a sus posibilidades de inmersión y tratamiento de la información.

Las ventajas que proporciona la RV son utilizadas en distintos sectores de la sociedad. Un ejemplos de ello es en el ejército con la utilización de simuladores de vuelo para la capacitación de los pilotos, en el campo de la educación son utilizados distintos sistemas de entrenamiento, el sector empresarial utiliza la RV en el área de marketing, generando escenarios tridimensionales representativos para atraer la atención del cliente. La medicina es otra esfera que se ha visto altamente beneficiada por los avances de la RV, con la creación de los simuladores quirúrgicos y los sistemas de visualización tridimensional que le han permitido al personal médico perfeccionar su capacitación sin poner en riesgo vidas humanas, además de las ventajas que propone la observación y el diagnóstico de diversas patologías sin necesidad de técnicas invasivas.

En el área de la visualización médica, cada año se perfecciona la calidad del análisis y tratamiento de las imágenes digitales en busca de un mayor realismo, característica indispensable para lograr un diagnóstico confiable. Estas aplicaciones resultan de gran utilidad dado que permiten un óptimo aprovechamiento de la información de los pacientes obtenida a través de equipos médicos.

Las imágenes obtenidas de la estructura anatómica humana son procesadas para mostrar o extraer información útil. Uno de los pasos fundamentales en el tratamiento de imágenes es identificar características de las estructuras segmentadas. Uno de los rasgos más importantes para el análisis de estas imágenes son los bordes, dado que permiten obtener información como la longitud. Conocer el contorno de una imagen es vital para definir las características de la región externa.

Este tema ha sido incluido en aplicaciones biomédicas como el tratamiento ortodóntico, en investigaciones sobre deformación craneal, estudios morfológicos del cartílago y en la

colección de datos antropométricos. El análisis de los contornos de imágenes se ha utilizado para estudiar la periferia de la próstata en tomografías realizadas, también se usa para cualificar las mandíbulas humanas y de esta forma analizar su variabilidad. El objetivo de estos trabajos ha sido obtener los rasgos de objetos biológicos.

Cuba y especialmente la UCI se ha adentrado en el mundo del software para suplir gradualmente las necesidades nacionales. La Facultad 5 lleva a cabo el proyecto VisMedic, que tiene como objetivo principal el desarrollo de un sistema para la visualización y reconstrucción tridimensional de imágenes médicas digitales para el diagnóstico de patologías y la planificación preoperatoria. El procesamiento de imágenes juega un papel fundamental para lograr altos niveles de realismo y calidad en la visualización y reconstrucción, permitiendo entre otras muchas funcionalidades el mejoramiento de las imágenes así como la identificación de estructuras anatómicas relevantes mediante el proceso de segmentación.

El resultado de la segmentación lo constituyen contornos de regiones que son expresadas en píxeles y que solamente permiten separar una región de otra. Por tanto, es necesaria la conversión de estos datos a un formato adecuado, proceso conocido como descripción de regiones, para su posterior procesamiento y uso en el análisis de las imágenes, y en particular en el reconocimiento de patrones con alto valor agregado en la visualización médica.

Partiendo de esto se genera la siguiente **situación problémica:**

Las técnicas de segmentación implementadas en el proyecto VisMedic obtienen como resultados contornos definidos solo por los píxeles que lo componen, sin conectividad entre ellos y que solo son utilizados para su visualización. El proyecto no cuenta con algoritmos para la descripción de los contornos segmentados mediante la definición de características utilizadas en el reconocimiento de patrones, que permitan realizar transformaciones a los contornos antes mencionados.

Por todo lo anterior el **problema científico** queda formulado de la siguiente manera: ¿Cómo describir los contornos segmentados en imágenes médicas digitales?

Teniendo como **objeto de estudio:** Descripción de contornos en imágenes digitales y como **campo de acción:** Descripción de contornos en imágenes médicas digitales.

Para dar solución al problema antes mencionado se propone como **objetivo general:** desarrollar un módulo que permita la descripción de contornos segmentados en imágenes

médicas digitales. El presente trabajo tiene como **idea a defender**: con el desarrollo del módulo de descripción de contornos en imágenes médicas digitales se logrará extraer características cuantitativas de los mismos contribuyendo a que el reconocimiento de patrones en dichas imágenes sea más efectivo.

Para dar cumplimiento al objetivo planteado es imprescindible realizar las siguientes **tareas investigativas**:

1. Elaboración del marco teórico a partir del estado del arte del tema a investigar.
2. Selección de las técnicas de descripción de contornos para el uso de los descriptores en estudios futuros.
3. Implementación de las técnicas de descripción seleccionadas para el desarrollo de un módulo informático.
4. Validar el correcto funcionamiento de la solución propuesta.

Métodos científicos a emplear:

Métodos teóricos

- Analítico-Sintético Método teórico utilizado con el objetivo de analizar todas las teorías y documentos relacionados con la descripción de imágenes.
- Histórico-Lógico Método teórico utilizado para poder conocer los avances tecnológicos en el tema de la descripción de imágenes teniendo en cuenta su evolución y principales tendencias.

Métodos empíricos

- Análisis de fuentes de información: Se utilizará para analizar las distintas fuentes de información como internet, libros, revistas, entre otras.
- Pruebas: Se realizarán algunas pruebas al algoritmo implementado para ver si el resultado obtenido es el esperado.

Este trabajo abarca el estudio de las técnicas de descripción de imágenes hasta la selección de alguna de ellas para conformar un modelo y su posterior implementación. Garantizando claridad y un mejor aprovechamiento de la información este trabajo se define en cuatro capítulos según la evolución gradual de la investigación y su aporte al objetivo general.

Capítulo I. Fundamentación teórica: Se hace una descripción de las técnicas más usadas en la descripción de contornos. Se realiza un estudio de la Transformada de Fourier así como de las etapas del procesamiento digital de las imágenes y de las bibliotecas usadas en el procesamiento digital de imágenes.

Capítulo II. Solución propuesta: Se propone una solución al problema científico después de haber analizado las diferentes variantes. Se detallan los métodos de descripción seleccionados para su posterior implementación. Por último, se hace un breve resumen de las metodologías y herramientas de desarrollo a utilizar.

Capítulo III. Análisis y Diseño de la Solución Propuesta: Queda defino el modelo de dominio, los requisitos funcionales y no funcionales. Se elabora el Modelo de Casos de Uso que estará formado por los actores, el Diagrama de Casos de Uso y la descripción textual de cada Caso de Uso (CU). Finalmente se muestran los diagramas de clases y secuencia del diseño.

Capítulo IV. Implementación y Validación de los Resultados: Se detallan los elementos de la implementación y se realizan las pruebas para validar los resultados de la investigación. Se modela el diagrama de componentes para la implementación y se muestran los resultados de las pruebas realizadas al módulo elaborado con el objetivo de medir el rendimiento del mismo.

Capítulo I. Fundamentación teórica

Este capítulo incluye el estado del arte sobre el tema a investigar. Propone una breve descripción sobre la definición de imágenes digitales y un análisis de los métodos de descripción existentes hasta el momento. Se incluyen además las etapas del procesamiento digital de imágenes y un estudio sobre la Transformada de Fourier.

1.1 Imagen digital. Definición

Una imagen es definida como una función bidimensional *I (m, n),* donde *m* y *n* son coordenadas espaciales y la amplitud de *I*, para cualquier par de coordenadas *(m, n),* es llamada intensidad o nivel de gris de la imagen en ese punto. Podemos hablar de imagen digital cuando los valores de *m, n* y los valores de amplitud de *I* son finitos, es decir, cantidades discretas. La función de modo discreto es representada en un computador a través de arreglos de números digitales, esto es requerido para representar las imágenes como arreglos bidimensionales de puntos [8][13].

Píxel o pel, se le llama a un punto en la matriz; los píxeles están localizados en una malla rectangular y la posición de los mismos está dada por la notación común en las matrices, donde el primer índice *m* es la fila y el segundo *n* es la columna. Si la imagen digital contiene *M x N* píxeles, entonces es representada por una matriz de *M x N*, donde el índice *m* va desde 0 a *M* – 1 y *n* va desde 0 a *N* – 1. Por tanto, *M* representa el número de filas y *N* el número de columnas. Además, cuando la intensidad de la imagen es cuantificada en *Ng* valores discretos, la secuencia de niveles de gris *I (m, n)* puede tomar cualquier valor entero 0,1,…, *Ng – 1*; el valor *Ng* es usualmente una potencia de dos y puede tomar valores grandes (64, 256, entre otros.) cuando la imagen es almacenada en la computadora. Una opción suficiente para la representación de imágenes son los valores de *Ng* = 32 o 16 ya que para el ojo humano es difícil notar diferencias detalladas en cuanto a la intensidad [8].

Las imágenes se representan en una matriz de enteros, donde cada número constituye la reflectancia de los objetos de la escena en un tiempo discreto al plano. La resolución de una imagen como se ha dicho anteriormente está dada por una matriz de elementos representada por *M x N* en el cual cada elemento es un píxel y los valores asignados correspondientes a la

luminosidad o nivel de gris. Estos niveles de gris están en el rango de 0 a 255 donde el 0 representa el negro absoluto y el 255 el blanco absoluto. Concluyendo, se puede decir que una imagen es una función de intensidad bidimensional representando cada píxel como *f (x, y)* [1].

1.2 Etapas del procesamiento digital de imágenes

Las etapas básicas del procesamiento digital de imágenes son: adquisición de la imagen, preprocesado, segmentación, descripción y reconocimiento de patrones (ver Figura 1) [1][8].

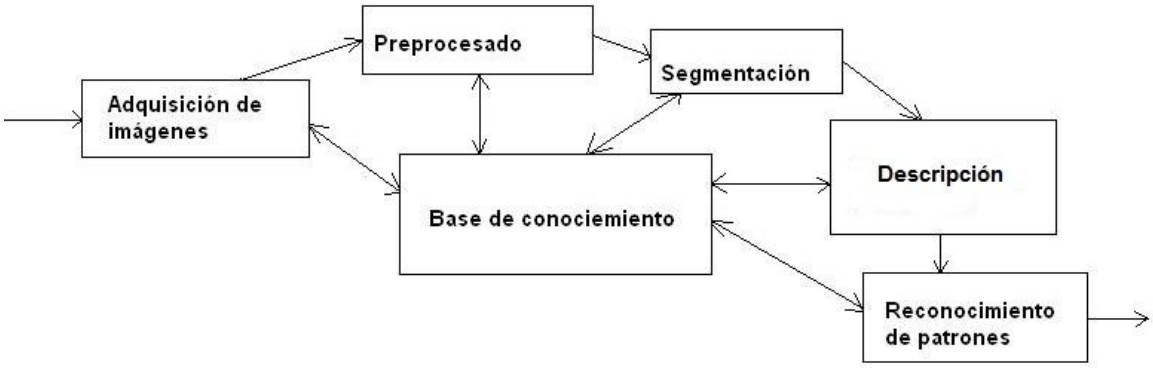

Figura 1. *Etapas del procesamiento digital de imágenes*

La adquisición de la imagen es el primer paso con el objetivo de digitalizar la imagen, la cual se realiza con la ayuda de algunos dispositivos, como los tomógrafos [1][7].

Luego de obtener la imagen, sigue el preprocesado, su función es mejorar la imagen de manera que se incremente la calidad en los siguientes procesos, generalmente trata con técnicas para realizar el contraste, disminuir el ruido y realzar los bordes [1][7].

La segmentación es una de las tareas más complicadas e importantes del procesamiento digital de imágenes. Como salida en este proceso se tiene los píxeles en bruto que constituyen el contorno de una región determinada, o bien de toda la región. En cada caso hay que convertir los datos a una forma adecuada para el proceso automático según el problema que se quiera resolver en un momento determinado [1][7].

La descripción es un proceso posterior a la segmentación, cuya salida generalmente son datos crudos de píxeles que constituyen el contorno, es decir un conjunto de píxeles que separan una región de la imagen o de otros puntos de la región. Para un adecuado tratamiento informático

primeramente se debe decidir si los datos a representar son del contorno o de la región completa. La descripción también conocida como selección de rasgos, consiste en extraer información cuantitativa de interés, fundamental para diferenciar una clase de objetos de otra [1].

Los esquemas de descripción de contornos deben tener ciertas propiedades [10]:

- Unicidad: cada objeto debe tener una única descripción.
- Invariancia frente a transformaciones geométricas, dígase: rotación, traslación, escala y cambio de posición en el primer piso.
- Sensibilidad o capacidad para diferenciar objetos casi iguales.
- Abstracción del detalle o capacidad para representar los rasgos característicos básicos de los objetos de manera abstracta.

Reconocimiento de patrones es el proceso en el cual se asigna una etiqueta a un objeto, en base a la información proporcionada por sus descriptores. La interpretación implica asignar significado a un conjunto de objetos reconocidos [1].

1.3 Transformada de Fourier en el tratamiento digital de imágenes

La técnica transformada de Fourier consiste en cambiar la representación de los datos del plano espacial al de frecuencia. Cuando el interés se enfoca en las imágenes digitales y la naturaleza de las señales es bidimensional y discreta es necesario el uso de la transformada discreta de Fourier (DFT).

A través de la DFT se representan los datos en el dominio frecuencial. El mismo realiza una representación de la variación espacial de los píxeles de la imagen. Cabe destacar que la representación de la transformada de Fourier es otra señal bidimensional, en este caso compleja y por lo tanto está compuesta de una parte real y otra imaginaria [4].

Las frecuencias espaciales altas reflejan la ocurrencia de cambios rápidos en los niveles de gris entre los píxeles cuando se recorre la imagen espacialmente, por ejemplo, los que ocurren en

los bordes, las líneas y cierto tipo de ruido en las imágenes. Por el contrario, las bajas frecuencias son producidas por cambios leves en el brillo de una imagen.

1.3.1 Transformada de Fourier Discreta

A continuación se introducirá la transformada de Fourier discreta unidimensional dado que el proceso bidimensional puede ser explicado como una extensión del caso unidimensional.

DFT unidimensional

Transformada de Fourier unidimensional

Primeramente se analiza las funciones definidas para la transformada de Fourier para luego pasar a la transformada discreta. La transformada de Fourier *F(u)* para una variable simple de función *f(x)* se define mediante [1][7]:

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-j2\pi ux} dx \qquad (1)$$

Dado *F(u)*, *f(x)* puede obtenerse por la transformada de Fourier inversa:

$$f(x) = \int_{-\infty}^{\infty} F(u) e^{j2\pi ux} du \qquad (2)$$

Transformada de Fourier bidimensional

Puede extenderse fácilmente a dos dimensiones, quedando [7]:

$$F(u,v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y) e^{-j2(ux+uy)} dxdy \qquad (3)$$

la transformación inversa:

$$f(x,y) = \int_{\infty}^{\infty} \int_{\infty}^{\infty} F(u,v) e^{j2\pi(ux+vy)} dudv \qquad (4)$$

Para una función discreta con *n* número de muestras (Δx), se tiene:

$$f(x) = f(x_0 + x\Delta x) \text{ donde } x = 0,1,2,\ldots,N-1 \qquad (5)$$

A partir de la ecuación 5 se establece la transformada de Fourier unidimensional como:

$$F(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)e^{-j2\pi ux/N} \quad \text{con } u = 0,1,2,\ldots,N-1 \qquad (6)$$

Se logra retornar al plano espacial mediante la transformada inversa de la ecuación presentada como Fourier discreta unidimensional, de la siguiente manera:

$$f(x) = \sum_{u=0}^{N-1} F(u)e^{j2\pi ux/N} \quad \text{con } x = 0,1,2,\ldots,N-1 \qquad (7)$$

En este tema casi siempre se estudiarán funciones f(x) que son reales. Sin embargo, la transformada de Fourier de una función real en general será compleja, es decir:

$$F(u) = R(u) + \mathrm{iI(u)} \qquad (8)$$

Donde *R(u)* e *I(u)* son respectivamente las partes real e imaginaria de *F(u)* con $i = \sqrt{-1}$. La ecuación 8 puede escribirse en forma exponencial [1][7]:

$$F(u) = \left|\mathrm{F(u)}\right| \exp\left[i\phi(u)\right] \qquad (9)$$

donde

$$\left|F(u)\right| = \sqrt{\mathrm{R}^2(u) + \mathrm{I}^2(u)} \qquad (10)$$

y

$$\phi(u) = \mathrm{arctg}\left[\mathrm{I(u)}/R(u)\right] \qquad (11)$$

La función de magnitud |F(u)| recibe el nombre de espectro de Fourier de *f(x)* (ver ecuación 10) y $\phi(u)$ recibe el nombre de fase (ver ecuación 11). El cuadrado del espectro suele recibir el nombre Espectro de Potencias.

La variable *u* suele recibir el nombre Frecuencia de Dominio. A través de la fórmula de Euler, puede ser calculada [1]:

$$e^{j\theta} = \cos\theta + jsen\theta \qquad (12)$$

$$\exp\left[2\pi iux\right] = \cos(2\pi ux) + \mathrm{jsen}(2\pi ux) \qquad (13)$$

y por tanto *u* representa el número de ciclos completos por unidad de *x*.

Sustituyendo la ecuación 13 en la de la transformación discreta unidimensional (ecuación 6) se obtiene [1]:

$$F(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)[\cos 2\pi ux/M - jsen2\pi ux/N] \qquad (14)$$

DFT bidimensional

En funciones discretas de dos variables, la transformada discreta se define como se muestra a continuación:

$$F(u,v) = \frac{1}{MxN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi(ux/M+vy/N)} \qquad (15)$$

Donde u= 0,1,2,…,M-1 y v =0,1,2,…,N-1

y su inversa como

$$f(x,y) = \frac{1}{MxN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} F(u,v)e^{j2\pi(ux/M+vy/N)} \qquad (16)$$

Donde x= 0,1,2,…,M-1 e y =0,1,2,…,N-1

La función F(u,v) está compuesta por dos funciones conocidas como parte real y parte imaginaria:

$$F(u,v) = R(u,v) + \mathrm{jI(u,v)} \qquad (17)$$

La ecuación anterior puede escribirse en forma exponencial [1][7]:

$$F(u,v) = \left|\mathrm{F(u,v)}\right| \exp\left[i\phi(u,v)\right] \qquad (18)$$

donde

$$\left|F(u,v)\right| = \sqrt{\mathrm{R}^2(u,v) + \mathrm{I}^2(u,v)} \qquad (19)$$

y

$$\phi(u.v) = \mathrm{arctg}\left[\mathrm{I(u,v)}/R(u,v)\right] \qquad (20)$$

La función de magnitud |*F(u,v)*| (ver ecuación 19) recibe el nombre de Espectro de Fourier de *f(x)* y $\phi(u)$ recibe el nombre de Fase (ver ecuación 20). El cuadrado del espectro suele recibir el nombre de Espectro de Potencias [1].

La variable *u* suele recibir el nombre de Frecuencia. Este nombre proviene de usar la fórmula de Euler [1]:

$$\exp\left[2\pi iux\right] = \cos(2\pi ux) + i\,\mathrm{sen}(2\pi ux) \qquad (21)$$

y por tanto *u* representa el número de ciclos completos por unidad de *x*.

1.3.2 Transformada Rápida de Fourier

La Transformada Discreta de Fourier puede ser calculada de una forma más eficiente a través de la Transformada Rápida de Fourier (FFT, del inglés *Fast Fourier Transform*). La FFT constituye uno de los mayores desarrollos en la tecnología del tratamiento de la imagen. La misma surge en 1965, su principio fundamental es la descomposición del cálculo de la transformada discreta de una secuencia de longitud *M* en transformadas discretas más pequeñas [19][20].

La transformada discreta de Fourier viene dada por la siguiente ecuación [1]:

$$F(k) = \sum_{x=0}^{M-1} \mathrm{f(x)W_M^{ux}} \qquad \text{u=0,1,…,M-1} \qquad (22)$$

Donde $W_M = \mathrm{e}^{-\mathrm{j}2\pi/M}$ (23)

M es asumida de la siguiente forma:

$$\mathrm{M=2^n} \qquad (24)$$

Siendo *n* un entero positivo, además se puede expresar *M* de esta forma también:

$$\mathrm{M=2K} \qquad (25)$$

Sustituyendo la ecuación 25 en la ecuación 22 queda [1]:

$$F(u) = \frac{1}{2K} \sum_{x=0}^{2K-1} f(x) W_{2K}^{ux} \qquad (26)$$

$$= \frac{1}{2}\left[\frac{1}{\mathrm{K}}\sum_{x=0}^{K-1} f(2x)W_{2k}^{u(2x)} + \frac{1}{K}\sum_{x=0}^{K-1} f(2x+1)W_{2K}^{u(2x+1)}\right]$$

Al quedar expresada la ecuación 23 como $W_{2K}^{2uK} = W_K^{ux}$, la ecuación 27 queda [1]:

$$F(u) = \frac{1}{2}\left[\frac{1}{K}\sum_{x=0}^{K-1} f(2x)W_{2K}^{ux} + \frac{1}{\mathrm{K}}\sum_{x=0}^{K-1} f(2x+1)W_K^{ux}W_{2K}^{u}\right] \qquad (27)$$

Definiendo [1]:

$$F_{par}(u) = \frac{1}{K}\sum_{x=0}^{K-1} f(2x)W_K^{ux} \text{ donde u = 0,1,2,….,K-1} \qquad (28)$$

$$F_{impar}(u) = \frac{1}{K}\sum_{x=0}^{K-1} f(2x+1)\mathrm{W}_{\mathrm{K}}^{\mathrm{ux}} \text{ donde u = 0,1,2,….,K-1} \qquad (29)$$

Sustituyendo las ecuaciones anteriores en la ecuación 27, queda [1]:

$$F(u) = \frac{1}{2}\left[F_{par}(u) + F_{impar}(u)W_{2K}^{u}\right] \qquad (30)$$

Dado que $W_M^{u+M} = W_M^u$ y $W_{2M}^{u+M} = -W_{2M}^u$, la ecuación 30 quedaría [1]:

$$F(u+K) = \frac{1}{2}\left[F_{par}(u) - F_{impar}(u)W_{2k}^{u}\right] \qquad (31)$$

La implementación de estas ecuaciones constituye el algoritmo FFT que se basa en divisiones sucesivas. La importancia de la transformada de Fourier radica en que permite representar funciones complicadas de forma que presenten propiedades muy útiles, que a menudo facilitan el tratamiento de la función original.

1.4 Descriptores de contorno

Los descriptores son un conjunto de información que puede ser expresada mediante números, vectores, entre otros, que se producen para describir un contorno determinado; a través de ellos se obtiene información cuantitativa que puede ser utilizada posteriormente, fundamentalmente en el reconocimiento de patrones. Para una buena descripción de contorno

se requiere descriptores mediante los cuales es posible encontrar similitudes perceptuales entre contornos que estén rotados, trasladados y/o escalados [1][24].

1.4.1 Código de cadena

El código de cadena es un tipo de estructura de datos para representar el contorno de un objeto en una imagen binaria mediante una secuencia de segmentos, conectados consecutivamente, de longitud y orientación específica, que conectan píxeles adyacentes [1]. Normalmente, esta representación se basa en 4 y 8 conectividad de segmentos. La dirección de cada segmento se codifica utilizando un esquema de numeración (ver Figura 2) [10].

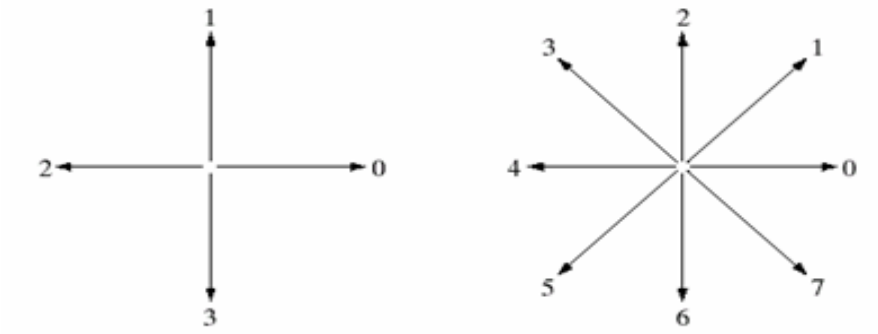

Figura 2: *Dirección de número de 4 y 8 conectividades en cadenas de código*

Normalmente las imágenes digitales son adquiridas y procesadas en un formato de cuadrícula con una separación igual en la dirección *x* e *y*, por lo que un código de cadena se podría generar siguiendo un contorno (en el sentido de las agujas del reloj) y la asignación de una dirección a los segmentos de la conexión de cada par de píxeles. Este método generalmente es inaceptable por dos razones: (1) la cadena resultante de códigos tiende a ser bastante largo, y (2) las perturbaciones pequeñas a lo largo del contorno debido al ruido, producen cambios en el código que puede no estar relacionado con el mismo [1].

Para dar solución a los inconvenientes generados, se aconseja volver a muestrear la frontera mediante la selección de un espaciado de malla más grande, por tanto cada punto del contorno es asignado a cada cuadrícula grande, dependiendo de la proximidad del contorno original. Luego puede ser representado por conectividades de 4 y 8 respectivamente, el punto de inicio es arbitrario. Al ser representado en conectividades de 4 y 8 de manera separada se obtiene

como resultado que la cadena de representación depende de la separación de la red de muestreo (ver Figura 3).

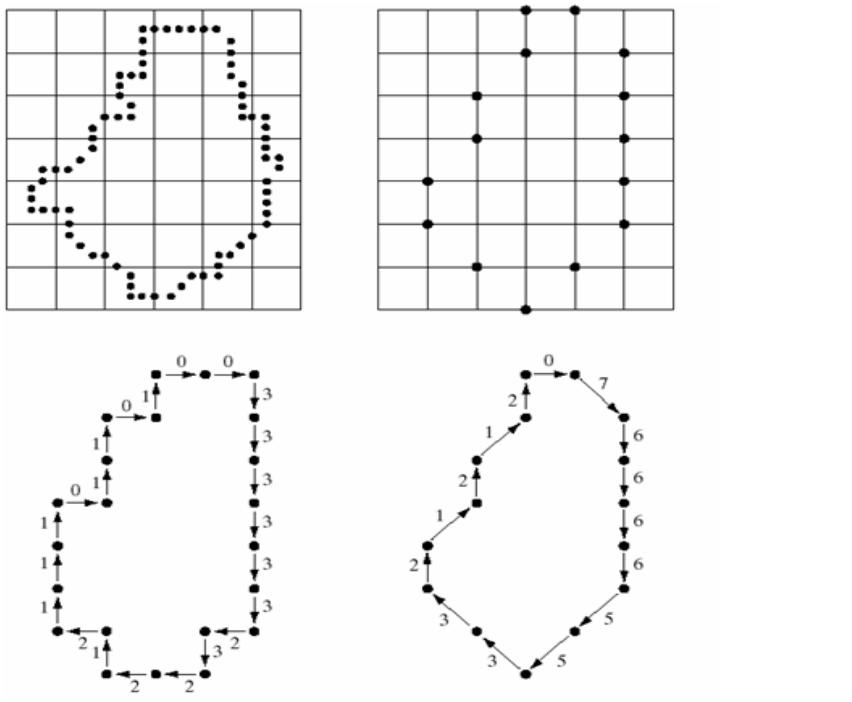

Figura 3: *Remuestreo de la frontera*

Al aplicar remuestreo a la frontera se obtiene, que la precisión de la representación del código resultante dependa del espaciado del cuadriculado de muestreo, el mismo necesita ser normalizado para la generación del código, de forma tal que los códigos con diferente punto de inicio sean iguales ya que el código cadena de un punto depende del punto de partida.

Código de cadena es invariante a la traslación, esta propiedad facilita la comparación de objetos. La invariancia de la escala se consigue al cambiar el tamaño de la reja en la cual se obtiene el mismo código como resultado, y la rotación se obtiene a través de la diferencia de la cadena de código, haciendo que las normalizaciones anteriores sean más exactas [1]. El código de cadena es de dimensiones grandes y sensible al ruido, normalmente se usa como entrada para un análisis de un nivel más alto [9].

1.4.2 Descriptores simples

Los descriptores simples con cálculos sencillos con el objetivo de obtener información cuantitativa del contorno para que pueda ser utilizada en cálculos posteriores. La longitud de contorno es uno de los descriptores simples más sencillo. La longitud es invariante a traslación y rotación, pero no al escalado de los objetos, pero la misma se puede obtener.

El valor de diámetro y la orientación de una línea que conecta los dos puntos extremos del diámetro (esta línea se denomina eje mayor del contorno), son descriptores útiles de un contorno. El diámetro se calcula como la distancia máxima entre dos puntos, el mismo es invariante frente a las transformaciones geométricas.

La curvatura es definida como el ritmo de variación de la pendiente. Es difícil de obtener medidas fiables en el contorno digital ya que tienden a ser localmente inestables [1].

1.4.3 Números de forma

El número de forma de un contorno basado en el código de 4-direcciones es definido como la primera diferencia del módulo más pequeño. Un contorno codificado en cadena depende del punto de comienzo. El orden *n* de un número de forma se define como el número de dígitos de su representación, *n* es par para un contorno cerrado, y su valor limita el número de posibles formas. La Figura 2 muestra todas las formas 4, 6 y 8, junto con sus representaciones de codificación en cadena, las primeras diferencias y sus correspondientes números de forma [1].

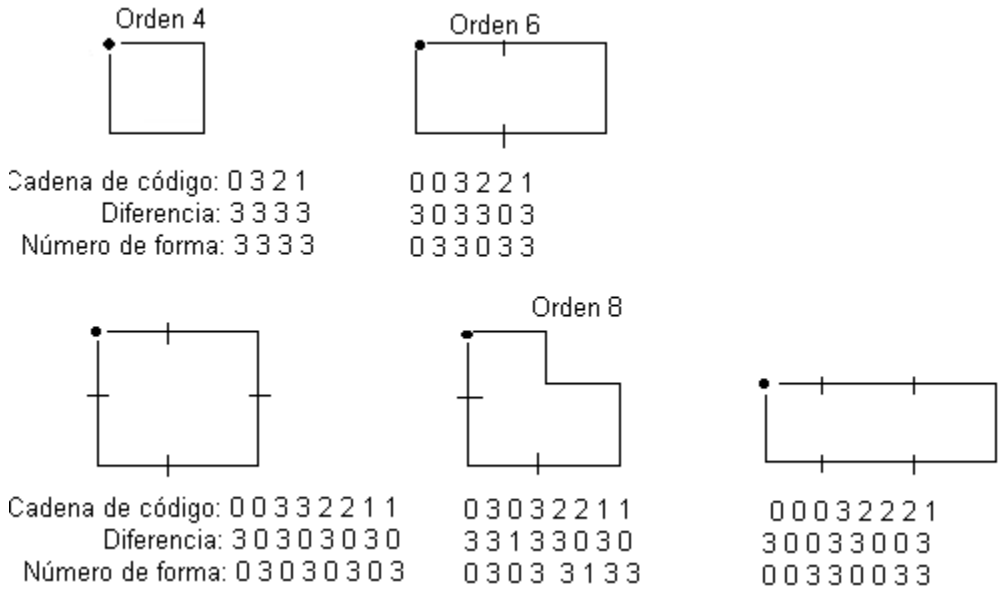

Figura 4: *Todas las formas de orden 4, 6 y 8*

El eje mayor de un contorno es el segmento de recta que une los dos puntos más separados entre sí. El eje menor es perpendicular al mayor y de tal longitud se podría formar un rectángulo que contenga exactamente al contorno. La relación entre el eje mayor y el menor se denomina excentricidad del contorno y el rectángulo descrito anteriormente se denomina rectángulo básico. En la mayor parte de los casos se obtendrá un único número de forma alineando el cuadriculado del código de cadena con los lados del rectángulo básico [1][11].

Ejemplo: Si se especifica que *n*=18 para el contorno de la Figura 5 (a). Para obtener el número de forma se siguen dos pasos, el primero es encontrar el rectángulo básico (Figura 5 b). El rectángulo más próximo de orden 18 es un rectángulo de 3x6, entonces se subdivide el rectángulo como muestra la Figura 5 (c), donde las direcciones del código de cadena se alinean con el mallado resultante. El segundo paso y final es obtener el código cadena y utilizar su primera diferencia para calcular el número de forma (Figura 5 d) [1].

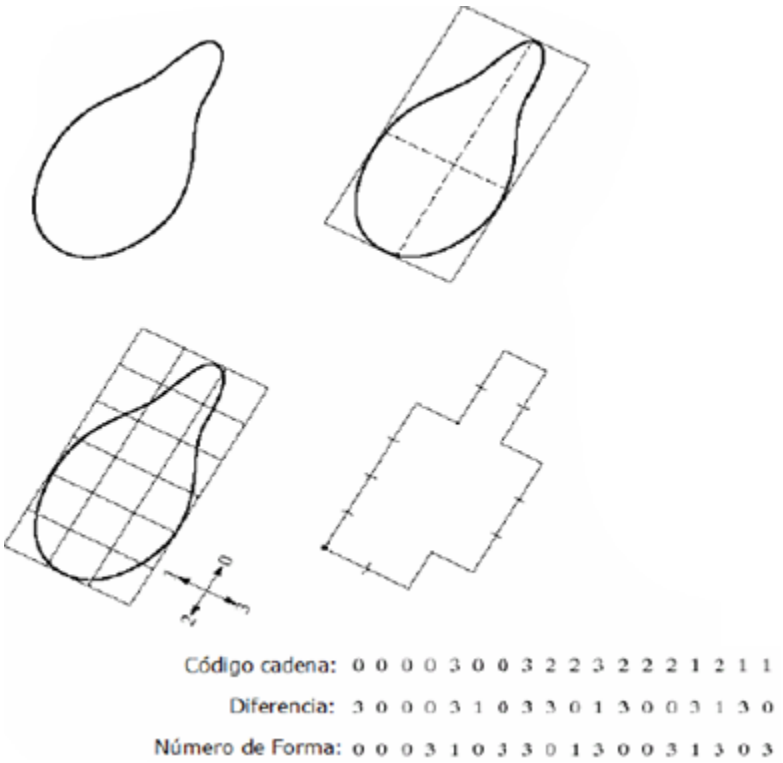

Figura 5: *Pasos para la generación de números de forma*

Esta representación está más comprimida que si utilizáramos códigos de longitud variable, sin embargo, el cálculo de descriptores como el perímetro, área y otros resulta más complejo.

1.4.4 Descriptores de Fourier

Se denomina descriptores de Fourier al conjunto de puntos en el espacio de frecuencias resultantes de transformar mediante DFT los puntos de un contorno. Los descriptores de Fourier trabajan con curvas cerradas [21].

Los descriptores de Fourier tienen como característica ser invariantes a transformaciones geométricas, como la traslación, la rotación y la escala, además son tolerantes ante el ruido [19]. Son muy utilizados en el reconocimiento de caracteres, en la clasificación de objetos, en el reconocimiento de objetos 3D y reconstrucción de imágenes 2D, muchas de estas aplicaciones han sido utilizadas con propósitos médicos.

Estos poseen ventajas como las que se mencionan a continuación:

- Fáciles de implementar.

- Cada descriptor tiene un significado físico específico.
- Simples para normalizar.
- Capturan características globales y locales.
- Invariantes a las transformaciones geométricas y tolerantes al ruido.
- Permiten lograr una buena descripción en base a pocos términos.
- Estos descriptores poseen la habilidad de discriminación de los descriptores globales, además de ser sensibles al ruido.
- Proporcionan ventajas en el análisis de situaciones físicas donde la señal contiene discontinuidades y puntos agudos.
- La mayoría de los trabajos basados en descriptores de Fourier, se enfocan a reconocimiento de caracteres y clasificación de objetos.

1.4.5 Momentos

La forma de una frontera (y su firma [1]) se puede describir cuantitativamente mediante los momentos. En la Figura 6 se muestra un contorno segmentado y su representación como una función unidimensional *g(r)* de una determinada variable *r* [7].

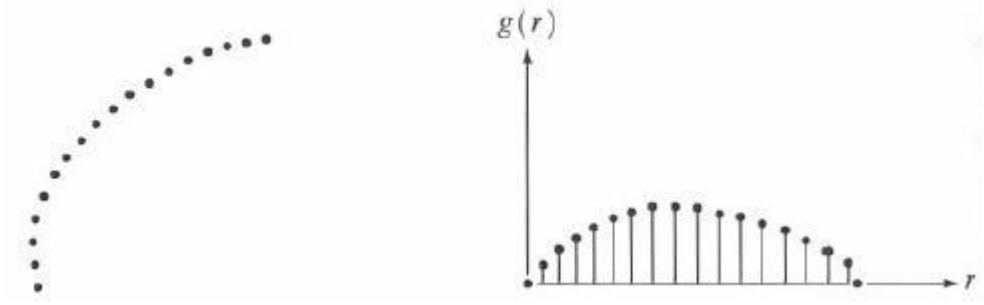

Figura 6: Contorno segmentado y su representación unidimensional

Si consideramos la amplitud de *g* como una variable aleatoria *v* y formamos un histograma $p(v_i)$, $i = 1, 2, \ldots, K$ donde *K* es el número de valores que toma la variable, entonces se define el momento como [1][7]:

$$u_n(v) = \sum_{i=0}^{K-1} (v_i - m)^n \, p(v_i) \qquad (32)$$

Donde

$$m = \sum_{i=0}^{K-1} v_i \, p(v_i) \qquad (33)$$

Esta cantidad *m* puede ser reconocida como una media y v_2 como la varianza, aunque no se esté trabajando con histogramas normalizados. Haciendo normalizaciones del histograma se tienen distribuciones de probabilidad y entonces se habla de momentos. Una alternativa para normalizar *g(r)* es que sea tratada como la probabilidad de r_i. En este caso *r* se trata como la variable aleatoria y surgen los momentos [1]:

$$u_n(r) = \sum_{i=0}^{K-1} (r_i - m)^n \, p(r_i) \qquad (34)$$

Donde

$$m = \sum_{i=0}^{K-1} r_i \, g(r_i) \qquad (35)$$

En la ecuación anterior *K* es el número de puntos que contiene el contorno y u_n está directamente relacionado con la forma de *g(r).* Los momentos reducen la tarea de descripción a funciones unidimensionales y no tienen en cuenta los píxeles del borde, sino de toda la imagen [16]. Es un método popular, sencillo, hace una buena interpretación física del contorno de la forma, pero es débil a la hora de funcionar bien con pequeñas variaciones en el borde: no es capaz de corresponder bien al objeto modelo con el objeto a detectar y no es invariante frente a la rotación [1][16].

1.5 Bibliotecas de procesamiento de imágenes

FFTW

FFTW es una librería de subrutinas C para calcular la transformada discreta de Fourier (DFT) en una o más dimensiones, de tamaño arbitrario, tanto para datos reales como complejos.

Además, por ser gratuita, es la librería FFT la elegida por la mayoría de las aplicaciones, utiliza una licencia GPL y es utilizada en aplicaciones de alto nivel científico como Matlab. Según algunas pruebas realizadas por el fabricante en varias plataformas, el rendimiento de FFTW generalmente es superior al de otras librerías o software para calcular la FFT, y es competitivo con las propietarias, dado que se adapta a la arquitectura de la computadora donde se ejecuta. Además FFTW es portable: pudiéndose ejecutar en muchas arquitecturas sin modificación. Proporciona un alto rendimiento y es de distribución gratuita [29].

CImg

Es una biblioteca de código abierto. Puede ser utilizado en aplicaciones comerciales. Es autónomo y por tanto altamente portátil. Es distribuido usando licencia es CeCILL y el mismo funciona en Linux, Windows, MacOS, entre otros. CImg define las clases y métodos para trabajar con imágenes en el código C++. Se puede utilizar para cargar/guardar varios formatos de archivo, los valores de acceso de píxeles, pantalla/transformación/filtro de imágenes, dibujar primitivas (texto, caras, curvas, objetos 3D), calcular las estadísticas de gestión de las interacciones del usuario en las imágenes, entre otras funcionalidades. Altamente portable dado que está conformada por un solo fichero .h [33].

Open Source Computer Vision Library (OpenCV)

Es una librería de funciones en C y C++ que contiene un conjunto de utilidades de procesamiento de imágenes, visión artificial, captura de vídeo y visualización de imágenes. Es de código abierto, gratuita, multiplataforma (disponible para entornos MS Windows, Mac OS y Linux), rápida, de fácil uso y en continuo desarrollo [32]. En procesamiento de imágenes, se utilizar la operación FFT muchas veces. OpenCV sólo tiene la función de DFT, por tanto se utiliza la biblioteca FFTW con el fin de realizar la operación FFT siendo una buena opción, rápido y se encuentra disponible libremente.

Visualization Toolkit (VTK)

Es un conjunto de librerías de código y distribución libres destinadas a la visualización y el procesado de imágenes, basadas en la programación orientada a objetos. Son muy amplias y complejas, pero aun así, están diseñadas para ser sencillas de emplear con cualquier lenguaje de programación orientado a objetos, como pueden ser C++, Java, Tcl, entre otras [31]. Son capaces de realizar operaciones sobre imágenes en dos y tres dimensiones y de generar

modelos en las mismas con pocas líneas de código. Debido a su gran potencia, se hacen necesarios amplios recursos de memoria en el computador para poder aprovechar en su totalidad sus funcionalidades [30].

Insight Segmentation and Registration Toolkit (ITK)

ITK es una herramienta orientado a objetos para el procesamiento, segmentación, y registración de imágenes. Aunque es grande y complejo, ITK está diseñado para ser fácil de usar una vez que se aprende sobre su elemento esencial basado en objetos y metodología de aplicación. Es una librería de código abierto, multiplataforma que ofrece a los desarrolladores un extenso conjunto de herramientas de software para el análisis de imágenes [34].

1.6 Conclusiones generales del capítulo

A lo largo de este capítulo se analizó brevemente las etapas del procesamiento de imágenes, para luego hacer un estudio de los métodos de descripción de imágenes con el objetivo de determinar la lógica de este proceso y los principales métodos que han sido desarrollados. Entre los descriptores de contornos expuestos se destaca los descriptores simples por su facilidad de cálculo e información asociada, así como los descriptores de Fourier que presentan invarianza ante la traslación, rotación, escala y cambio de posición del punto de partida; además reducen el ruido. Los números de formas y momentos si bien son utilizados en varias aplicaciones de reconocimiento de objetos, no poseen invarianza a las transformaciones geométricas. Se profundiza en la transformada de Fourier para tener vastos conocimientos de la importancia que tiene el mismo para la obtención de características cuantitativas, las cuales son utilizadas posteriormente en la comparación de objeto. Haciendo un análisis de las librerías de procesamiento de imágenes se llega a la conclusión de que para un mejor funcionamiento del módulo a implementar se necesita una librería que brinde un mejor uso de la FFT la cual ofrece características cuantitativas e importantes para el análisis de objetos en futuros trabajos, llegando a la conclusión que sería FFTW la cual es portable y posee un alto rendimiento.

Capítulo II. Solución propuesta

En este capítulo se expone detalladamente las técnicas seleccionadas para obtener una descripción adecuada en las imágenes médicas digitales. Se seleccionan las metodologías y herramientas de desarrollo que serán usadas para la implementación de la solución propuesta.

2.1 Métodos de descripción

Después de realizado el estudio sobre las principales técnicas de descripción de contorno al cual se hace referencia en el capítulo 1, se conformó la propuesta de solución, que contempla el uso de los descriptores de Fourier y descriptores simples, debido a que ambos cumplen con las necesidades, exigencias y objetivos de la investigación. Además, son invariantes a las transformaciones geométricas y reducen el ruido que puede estar presente en el contorno.

A través de los descriptores simples se obtienen características cuantitativas del contorno, la longitud es invariante a traslación y rotación, no siendo así para la escala pero esta última se puede obtener mediante la normalización. El diámetro es invariante a traslación y rotación.

Los descriptores de Fourier son fáciles de implementar, cada descriptor tiene un significado físico específico, simple para hacer normalización, obtiene características globales y locales. Los descriptores de Fourier resuelven la habilidad de discriminación de los descriptores globales, además de resolver problemas de sensibilidad al ruido. La mayoría de los trabajos basados en descriptores de Fourier, se enfocan en el reconocimiento de caracteres y clasificación de objetos. Estos descriptores son invariantes frente a la traslación, rotación, escala y cambio de posición del punto de partida.

2.1.1 Descriptores simples

Contando los píxeles del contorno se tiene una aproximación de su longitud. Si la frontera fue codificada usando código de cadena se tiene [8]:

$$L = \#\text{componente vertical} + \#\text{componente horizontal} + \text{diagonales} * \sqrt{2} \qquad (36)$$

La longitud es invariante a traslación y rotación, pero no al escalado de los objetos. Sin embargo, normalizando la longitud del contorno con respecto a su diámetro se puede obtener la invariancia al escalado.

$$l_{norm} = \frac{\mathrm{L}}{\mathrm{Diam}} \qquad (37)$$

El diámetro de un contorno se define como la distancia máxima entre dos píxeles [17]:

$$\mathrm{Diam}\ (B) = \max\ [D\ (p_i, p_j)] \qquad (38)$$

D es una medida de distancia, p_i y p_j son puntos sobre el contorno *B*.

La curvatura es definida como el ritmo de variación de la pendiente. Es difícil de obtener medidas fiables en el contorno digital ya que tienden a ser localmente inestables. Usando la diferencia entre las pendientes de los segmentos adyacentes (los que son representados como líneas rectas). Sin embargo puede ser útil emplear la diferencia entre las pendientes adyacentes del contorno como descriptor de la curvatura en el punto de intersección de los segmentos [1].

2.1.2. Descriptores de Fourier

La figura 5 muestra un contorno digital de *N* puntos en el plano *xy*. Se parte de un punto arbitrario (x_0, y_0), los pares de coordenadas (x_0, y_0), (x_1, y_1), (x_2, y_2),…, (x_{N-1}, y_{N-1}), se encuentran al recorrer el contorno, por ejemplo, en el sentido de las manecillas del reloj. Estas coordenadas también pueden ser expresadas en la forma $x(k) = x_k$ y $y(k) = y_k$ [1].

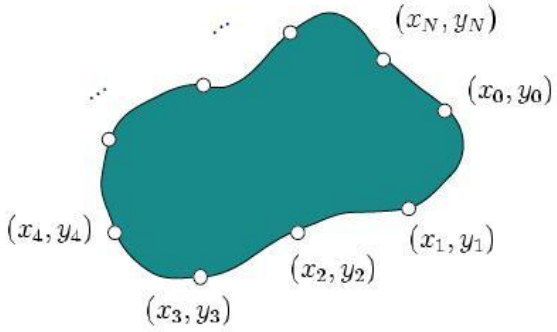

Figura 7: *Límite digital y su representación secuencial. Los puntos (x0, y0) y (x1, y1) son los primeros puntos de la secuencia.*

Denotándose de esta manera, el propio contorno se puede representar como la serie de coordenadas *s (k) = [x(k), y(k)],* para *k = 0, 1, 2, …, N-1*. Incluso cada par de coordenadas puede ser tratado como un número complejo tal que [2]:

$$s(k) = x(k) + jy(k) \qquad (39)$$

para *k = 0,1,2,…, N-1*. Tomando al eje *x* como eje real y al eje *y* como la parte imaginaria de una sucesión de números complejos. Independientemente de la interpretación de la sucesión de forma el contorno no cambia. Esta representación muestra una gran ventaja, reduce un problema bidimensional a uno unidimensional. Se tiene que la DFT de *s(k)* es [1]:

$$a(u) = \frac{1}{K}\sum_{k=0}^{K-1} s(k)e^{-i2\pi u/K} \quad \text{para u = 0, 1, 2,…, K – 1.} \qquad (40)$$

Los coeficientes complejos *a(u)* se llaman descriptores de Fourier de la frontera. La transformada inversa de Fourier de estos coeficientes devuelve *s(k)*. Es decir,

$$s(k) = \sum_{u=0}^{K-1} a(u)e^{i2\pi uk/\ K} \quad \text{para k = 0,1, 2,…, K – 1.} \qquad (41)$$

Si en lugar de tomar todos los coeficientes de Fourier, se toman sólo los primeros. Esto es equivalente a hacer *a(u) = 0* para *u > P – 1* en la ecuación 41. El resultado a esta aproximación sería [1]:

$$\hat{s}(k) = \sum_{u=0}^{P-1} a(u) e^{i 2\pi u k / K} \quad \text{para k = 0, 1, 2, …, K-1.} \qquad (42)$$

Aunque solamente se emplean *P* términos para obtener cada componente de $\hat{s}(k)$, *k* sigue variando de 0 a *K-1*. Es decir, en el contorno aproximado están presentes la misma cantidad de puntos, pero no se emplearon la misma cantidad de términos en la reconstrucción de cada punto. Luego mientras menor sea *P* se pierde mayor detalle a la hora de reconstruir el contorno. A continuación se muestra un ejemplo en el que queda demostrado [1]:

Ejemplo: Se trata de reconstruir el contorno de un cuadrado unitario, tomando un total de 64 puntos, es decir, *K = 64* y para esto se emplearán descriptores de Fourier. Los elementos de la figura 8 representan casos en los cuales se tomaron *P = 2, 4, 8, 16, 24, 32, 40, 48, 56, 61 y 62* coeficientes de Fourier para la reconstrucción respectivamente.

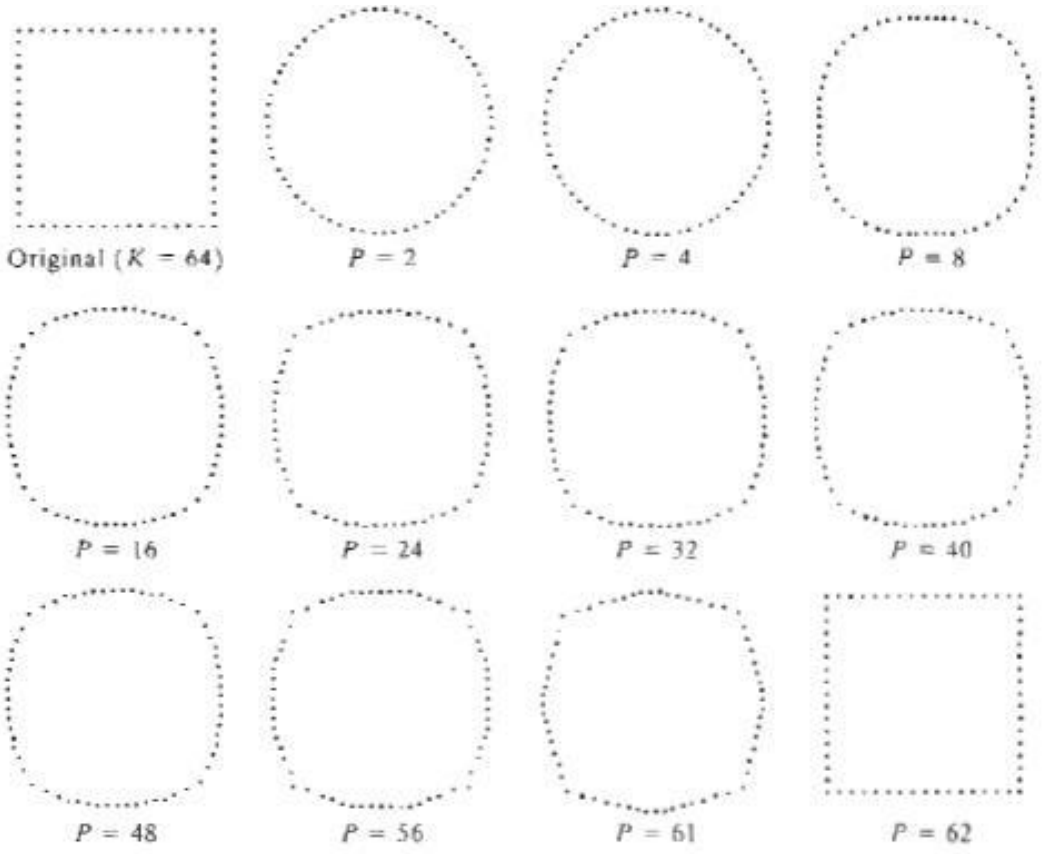

Figura 8. *Ejemplos de reconstrucción mediante descriptores de Fourier.*

En este ejemplo se demuestra que una pequeña cantidad de descriptores pueden emplearse para recoger la esencia elemental del contorno. Si se tiene en cuenta que estos coeficientes contienen información sobre forma y pueden emplearse como base para distintas formas de contorno, ésta propiedad va a resultar muy útil en el reconocimiento de objetos.

Cuando se calcula la transformada de un contorno y se obtiene sus descriptores, tal vez sea necesario realizar rotaciones, translaciones o cambios de escala. Una manera de hacerlo, sería volver al contorno original para aplicarle el cambio y calcular de nuevo la transformada, pero esto resulta muy poco práctico, por lo que es preferible realizar los cambios sobre los descriptores aplicando algunas de sus propiedades [16][24].

Por ejemplo, la rotación por un ángulo θ en un punto alrededor del origen del plano complejo es equivalente a multiplicar el punto por $e^{i\theta}$ (ver Figura 7). Haciendo esto con cada punto de toda la sucesión *s(k)* tiene como resultado la rotación de toda la sucesión alrededor del origen. La sucesión rotada resulta ser $s(k)e^{i\theta}$ que tiene como descriptor a [16]:

$$a_r(u) = \frac{1}{K}\sum_{u=0}^{K-1} a(u)e^{i\theta}e^{i2\pi uk/K} = a(u)e^{i\theta} \quad (43)$$

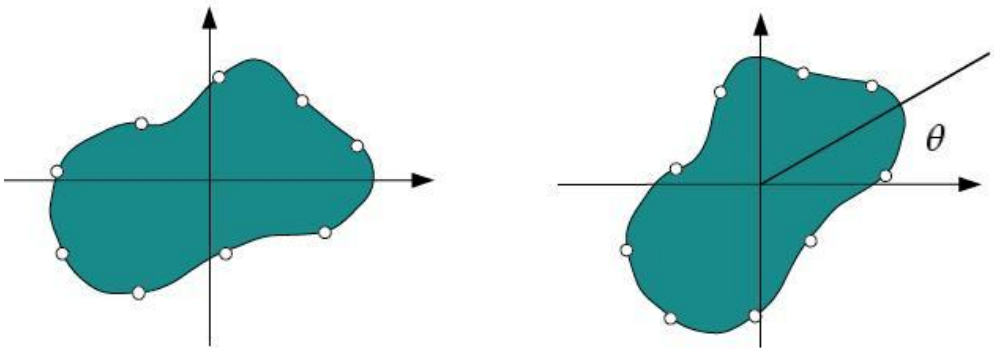

Figura 9: *Rotación de un contorno*

La traslación es definida por $s_t(k) = s(k) + \Delta_{xy}$ donde el símbolo Δ_{xy} se define como $\Delta_{xy} = \Delta x + i\Delta y$, luego la notación anterior indica la redefinición (traslación) de la sucesión como [1][16]:

$$s_t(k) = [x(k) + \Delta x] + i[y(k) + \Delta y] \quad (44)$$

La traslación solo afecta al primer descriptor *F(0)*, mientras que los demás descriptores retornan sus propios valores [16][28]. En la Figura 10 se muestra el contorno trasladado.

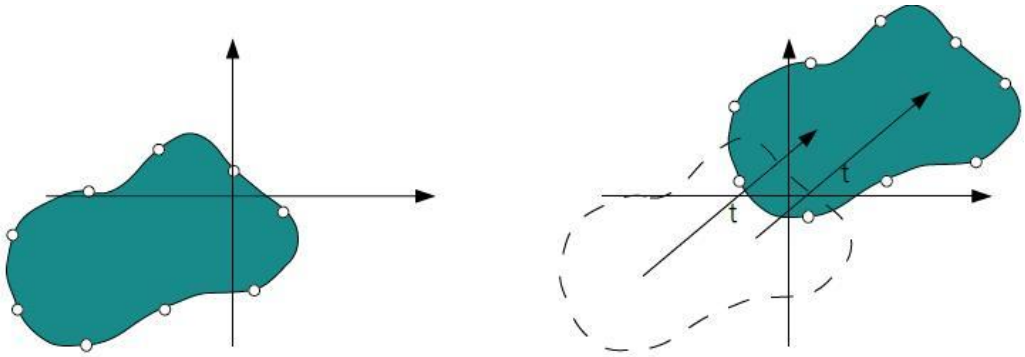

Figura 10: *Traslación de un contorno*

En la Figura 11 se observa el contorno transformado por la multiplicación de *C* y su normalización preservando así la escala. La invariancia de escalado se preserva cuando son utilizados todos los puntos del contorno [16].

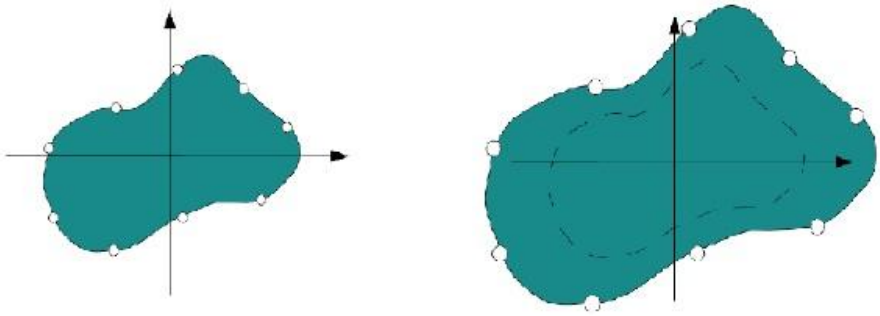

Figura 11: *Contorno escalado*

Sin embargo, en imágenes reales las herramientas de detección de bordes detectan más puntos cuando es más grande el objeto (Figura 12). De esta forma, el número de puntos cambia y el valor de los descriptores también, dificultando la normalización. Para resolver el problema sencillamente no se hace nada: si el número de descriptores es similar (un número no muy alejado entre los descriptores de la imagen escalada y la no escalada), el valor de los descriptores de Fourier no varían hasta el tercer decimal. También se ha de tener en cuenta,

que los primeros descriptores contienen la información general sobre la forma de una figura, mientras que los últimos contienen pequeños detalles [16].

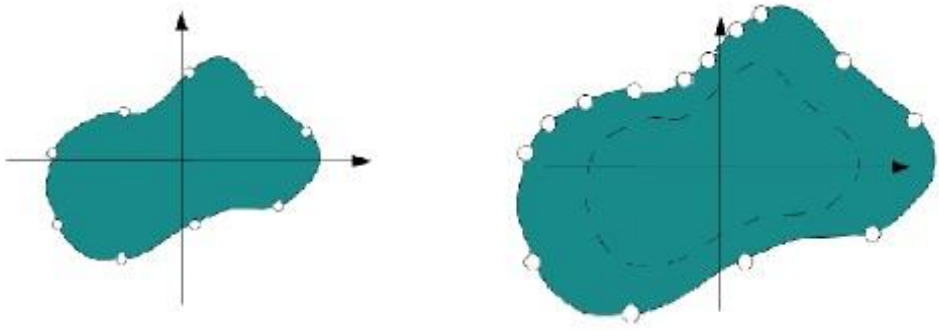

Figura 12: *Contorno escalado con más puntos*

Para la obtención de la invariancia a escalado simplemente se divide todos los descriptores por el segundo descriptor (*FD_1*) [16][24].

Cambio de posición del punto de partida de un contorno viene dado por un desplazamiento de fase lineal de $e^{-j2k\mu/K}$ en los descriptores de Fourier [1][28].

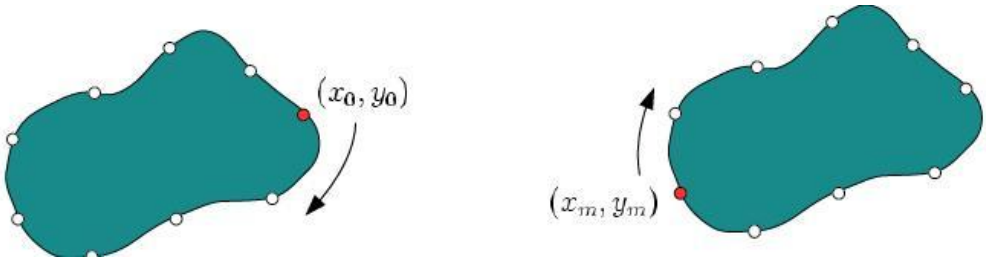

Figura 13: *Cambio de posición del punto de partida*

En la Figura 14 se muestra los valores que son adquiridos por las propiedades de los descriptores de Fourier, es decir, rotación, traslación, escala y cambio de posición del píxel de partida; siguiendo la secuencia de un contorno [1].

Transformación	Contorno	Descriptores de Fourier
Identificación	$s(k)$	$a(u)$
Rotación	$s_r(k) = s(k)e^{j\theta}$	$a_r(u) = a(u)e^{j\theta}$
Traslación	$s_t(k) = s(k) + \Delta_{xy}$	$a_t(u) = a(u) + \Delta_{xy}\delta(u)$
Escala	$s_e(k) = \alpha s(k)$	$a_e(u) = \alpha a(u)$
Punto de partida	$s_p(k) = s(k - k_0)$	$a_p(u) = a(u)e^{-j2k_0\mu/K}$

Figura 14: *Propiedades de los descriptores de Fourier*

2.2 Algoritmo de descripción de contorno mediante descriptores de Fourier

Para puntualizar el funcionamiento de la solución, se presenta un algoritmo general donde se muestra por pasos el flujo de operaciones para la descripción de contornos mediante descriptores de Fourier. Para obtener más eficiencia y rendimiento en el cálculo de los descriptores se hizo en el capítulo 1 un análisis de las principales bibliotecas que son utilizadas en el procesamiento digital de imágenes, llegando a la conclusión de que sería FFTW la biblioteca a utilizar dado la facilidad, elevado rendimiento y eficiencia en el cálculo de la DFT.

El algoritmo consta de 4 pasos fundamentales como muestra la siguiente figura:

Algoritmo1: Cálculo de descriptores de Fourier
Entrada: Contorno expresado por píxeles. ***Paso 1:*** Definir coordenadas de números complejos. ***Paso 2:*** Usando biblioteca FFTW se define *fftw_plan_dft_1d* y se calcula la transformada discreta de Fourier. ***Paso 3:*** Ejecutar plan (*fftw_ejecute (plan)*).

Paso 4:

Destruir plan (*fftw_destroy (plan)*).

Salida: Se obtiene un vector de descriptores.

Para el trabajo con la librería FFTW se comienza incluyendo el fichero fftw3.h, dado que se trabaja con la versión 3.3 de la librería (*#include <fftw3.h>)*. El siguiente paso es crear un plan, que es un objeto que contiene todos los datos que FFTW necesita para calcular la FFT, *fftw_plan* permite la realización de iteraciones tantas veces como sean posibles indicando el conjunto de datos de entrada y el conjunto de salida, a la vez que se define si la DFT a calcular es normal o inversa. El plan definido es ejecutado por la función *fftw_execute (plan)* y destruido al ser utilizado con *fftw_destroy_plan (plan)*.

A continuación se presenta un fragmento de código escrito en C++ que ilustra lo explicado anteriormente:

```
Void FourierTransform::DFT(Contour* contour,vector<FourierDescriptor>*
descriptors)
     {
         unsigned int K = contour->nPoints();
        descriptors->clear();
         fftw_complex* c, *d;
       c=reinterpret_cast<fftw_complex*>(contour->getComplexContour());
       d = (fftw_complex*) fftw_malloc(sizeof(fftw_complex) * K);
     fftw_plan plan=fftw_plan_dft_1d(K,c,d,FFTW_FORWARD,FFTW_ESTIMATE);
             fftw_execute(plan);
             for (int i = 0; i < K; i++)
             {
                   double r = d[i][0], im = d[i][1];
               descriptors->push_back(FourierDescriptor(r/K, im/K));
             }
             fftw_destroy_plan(plan);
             fftw_free(d);
 }
```

2.3 Metodologías y herramientas de desarrollo

A continuación se realiza una breve descripción de la Metodología de Desarrollo de Software, las principales herramientas y lenguaje de programación que fueron utilizados como parte de la solución propuesta en la realización de la tesis con el objetivo de tener una idea clara del desarrollo de la solución propuesta.

2.3.1 Metodología de desarrollo de software

Proceso unificado de desarrollo de software (RUP) para llevar a cabo la ingeniería de software y guiar el proceso de desarrollo. Utiliza como único lenguaje de modelado UML, este puede ser utilizado para la construcción de todos los modelos necesarios durante el desarrollo de la aplicación. Se caracteriza por estar dirigido por Casos de Uso (CU). Constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. Sus características fundamentales se basan en un desarrollo iterativo e incremental, dirigido por casos de uso y centrado en la arquitectura.

2.3.2 Herramientas de desarrollo

Visual Paradigm como herramienta de modelado ya que es una herramienta libre, muy profesional y multiplataforma que soporta todo el ciclo de vida del desarrollo de software. Además genera código en una amplia gama de lenguajes, entre ellos *C*++. Esta es una herramienta que utiliza UML como lenguaje de modelado. Es un producto de probada calidad que soporta varios idiomas y bajo licencia gratuita y comercial. La herramienta está diseñada para una amplia gama de usuarios interesados en construir sistemas de software fiables con el uso del paradigma orientado a objetos, incluyendo actividades como ingeniería de software, análisis de sistemas y análisis de negocios.

Se utilizó QT como *Framework* para la interfaz gráfica. Es de alta compatibilidad, posee características que lo hacen muy versátil y el código *C*++ que utiliza tiene un alto rendimiento. El código fuente está disponible y así como una muy buena documentación que hace de este una opción buena para los desarrolladores. Tiene una arquitectura muy flexible que permite diseñar aplicaciones sin mucho esfuerzo y con una gran calidad. Tiene un excelente apoyo técnico y sigue el principio de reutilizar el código para crear más y hacer despliegues sin importar el lugar. Posee un acceso total al código fuente para su revisión y modificación.

QTCreator como guía de desarrollo, es libre y multiplataforma. Posee una extensa librería de clases y herramientas, las cuales se encuentran bien documentadas lo que es de gran ayuda en el desarrollo de software.

2.3.3 Lenguaje de modelado

Como lenguaje seleccionado para modelar el análisis y diseño de la aplicación se escogió UML, el cual permite construir gráficos para visualizar, especificar y documentar un sistema. Es un lenguaje independiente de la plataforma y constituye un estándar mundial en el modelado. UML es un lenguaje que permite la modelación de sistemas con tecnología orientada a objetos, describiendo lo que supuestamente hará un sistema. Puede conectarse con los lenguajes de programación y posibilita la ingeniería inversa [15].

2.3.4 Lenguaje de programación

Como lenguaje de programación fue seleccionado *C*++ debido a que es el lenguaje por excelencia utilizado para las aplicaciones de RV, que hace uso eficiente del paradigma de Programación Orientada a Objetos. Permite programar a un alto nivel y tiene mecanismos muy buenos como la herencia y el polimorfismo que le brindan al desarrollador gran flexibilidad para diseñar. Al ser un lenguaje orientado a objetos permite encapsular los datos y los métodos en clases, esto posibilita obtener un código más seguro y con una mejor organización. Otra ventaja muy importante que se tuvo en cuenta es que este tipo de aplicaciones manejan grandes volúmenes de datos y necesitan interacción en tiempo real y no todos los lenguajes posibilitan esto: en el caso de *C*++ es un candidato ideal por su gran rapidez.

2.4 Conclusiones generales del capítulo

Se realizó un análisis detallado de las técnicas de representación y descripción seleccionadas, su selección se basa en una de las características fundamentales que tiene este proceso que es la invariancia a transformaciones geométricas, dígase, rotación, traslación, escala y cambio de posición del píxel inicial. Se hace una selección de las metodologías y herramientas de desarrollo a utilizar para el desarrollo e implementación del módulo, basada en las especificidades de la solución propuesta.

Capítulo III. Análisis y Diseño de la solución propuesta

Durante este capítulo se describe el sistema siguiendo lo que plantea RUP como metodología de desarrollo de software. Generándose varios artefactos se elabora el modelo de dominio a utilizar y se capturan los requisitos que serán usados posteriormente en la elaboración del Modelo de Casos de Uso y en la selección de los actores del sistema. Se modelarán diferentes diagramas de secuencia del diseño con el objetivo de proporcionar una mejor comprensión del módulo que se desea desarrollar.

3.1 Modelo del dominio

Un modelo del dominio captura los tipos más importantes de objetos en el contexto del sistema. Los objetos del dominio representan las “cosas” que existen o los eventos que suceden en el entorno en que trabaja el sistema. En este epígrafe se muestra el modelo del dominio (ver Figura 15).

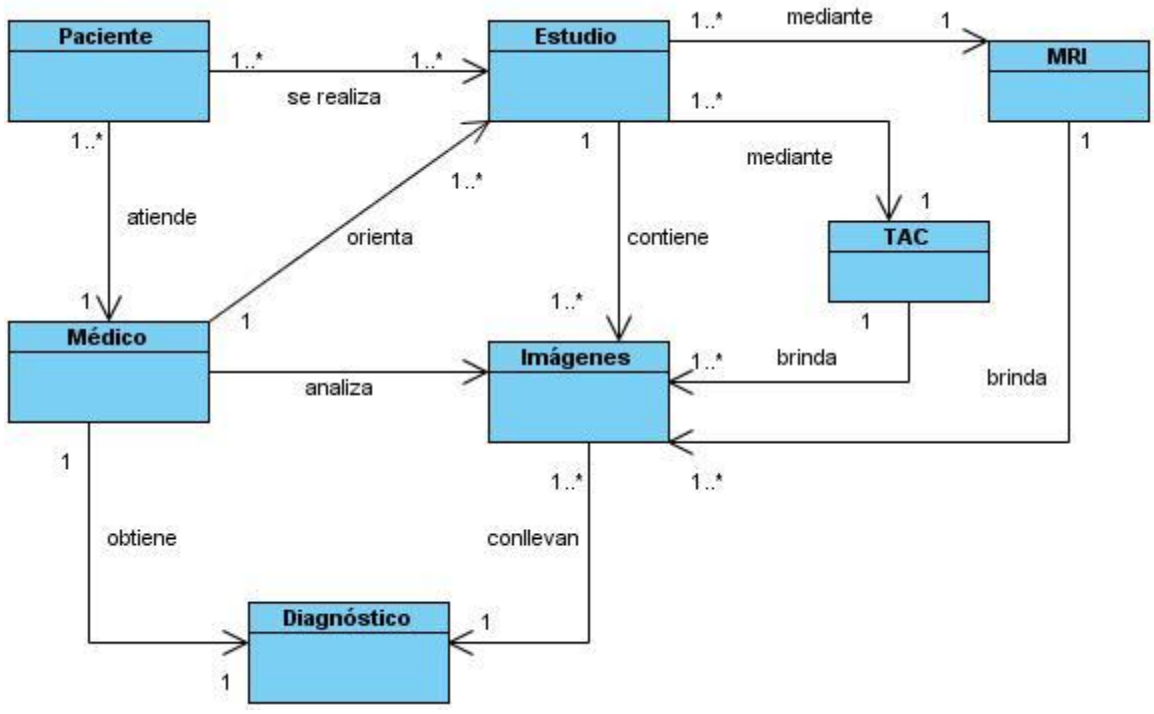

Figura 15: *Modelo de dominio*

3.1.1 Descripción de las clases del dominio

Al paciente se le realiza una serie de estudios indicados por el médico, los cuales pueden ser a través de equipos de adquisición de imágenes, tales como, tomografías computarizadas (TAC) o resonancias magnéticas (MRI). Este estudio contiene varias imágenes las cuales son analizadas por el médico a través de un software de visualización para obtener un diagnóstico adecuado y acertado del paciente.

El **médico especialista** es aquella persona que está altamente capacitada en el manejo y conocimiento de los métodos de imagenología diagnóstica.

Tomografía Axial Computarizada (CT) y **Resonancia Magnética (MRI)** son los conocidos equipos de adquisición de imágenes.

Un **Software** es un programa informático diseñado para facilitar al usuario la realización de un determinado tipo de trabajo. Resulta una solución para la automatización de ciertas tareas complicadas como puede ser el diagnóstico a través de imágenes.

Las **imágenes** constituyen el resultado del estudio orientado por el médico al paciente a través del MRI y CT.

El **diagnóstico** es el resultado que da el médico luego de realizar un estudio a un paciente.

El **paciente** es aquel que recibe los servicios de un médico u otro profesional de la salud.

3.2 Captura de requisitos

La captura de requisitos es importante en todo sistema, en la misma se definen las condiciones o capacidades que el sistema debe cumplir. En esta disciplina se capturan los requisitos funcionales y los no funcionales, los requisitos funcionales tienen como objetivo especificar lo que el sistema debe hacer y cómo debe hacerlo y los requisitos no funcionales que especifican las características que desea el cliente que tenga el sistema [15]. En los siguientes epígrafes se hace la captura de requisitos utilizados en la solución propuesta.

3.2.1 Requisitos Funcionales

Los requisitos funcionales constituyen capacidades o condiciones que el sistema debe cumplir [15]. Estos facilitan el entendimiento entre usuarios y desarrolladores del sistema a elaborar.

Los siguientes requisitos responden a las funcionalidades que el sistema debe tener una vez concluida la implementación.

RF1. Generar descriptores Fourier

- Permite el cálculo de los descriptores de Fourier de las coordenadas del contorno.

RF2. Transformar contorno

- Permite que se hagan las transformaciones geométricas al contorno.

RF2.1. Trasladar contorno

RF2.2. Rotar contorno

RF2.3. Escalar contorno

RF2.4 Cambiar origen

RF3. Generar descriptores simples

- Permite el cálculo de los descriptores simples: longitud, diámetro; brindando información cuantitativa.

RF4. Reconstruir contorno

- Permite reconstruir el contorno a través de la inversa, con menor y mayor cantidad de descriptores.

3.2.2 Requisitos no funcionales

Los requisitos no funcionales son aquellos que debe tener el sistema, pero que no son una funcionalidad específica [15]. Los requisitos no funcionales que debe tener el módulo se mencionan a continuación:

RnF1.Software

El sistema operativo sobre el cual debe ejecutarse la aplicación será Windows XP, Windows 7 o Ubuntu.

RnF2.Hardware

Microprocesador: Intel Pentium IV a 3.0 GHz o superior.

Memoria RAM al menos 1GB.

RnF3.Seguridad

Fiabilidad: Los modelos visualizados deben tener calidad para permitir un análisis lo más exacto posible para los especialistas.

Integridad: No debe haber pérdidas de información ni de calidad en las imágenes obtenidas.

RnF4.Soporte

Se brindará soporte para los sistemas operativos Windows XP, Windows 7 y Linux (Ubuntu).

RnF5.Restricciones en el diseño e implementación

Se empleará como lenguaje de programación C++ y el *framework* Qt para el diseño de las interfaces gráficas.

3.3 Modelo de casos de uso del sistema

En este epígrafe se identificarán los actores del sistema así como los CU del mismo. Además se realizará una descripción textual de cada CU del sistema para así tener una mejor comprensión del funcionamiento del módulo.

3.3.1 Actores del sistema

En este caso la Aplicación final es la encargada de llevar a cabo la ejecución de las funcionalidades del módulo.

Actores	Justificación
Aplicación final	Interactúa con el módulo ejecutando directamente las funcionalidades en el módulo: Generar Descriptores de Fourier, Transformar Contorno, Reconstruir

	Contorno, Generar Descriptores Simples.

Tabla 1. *Actor del Sistema*

3.3.2 Diagrama de casos de uso del sistema

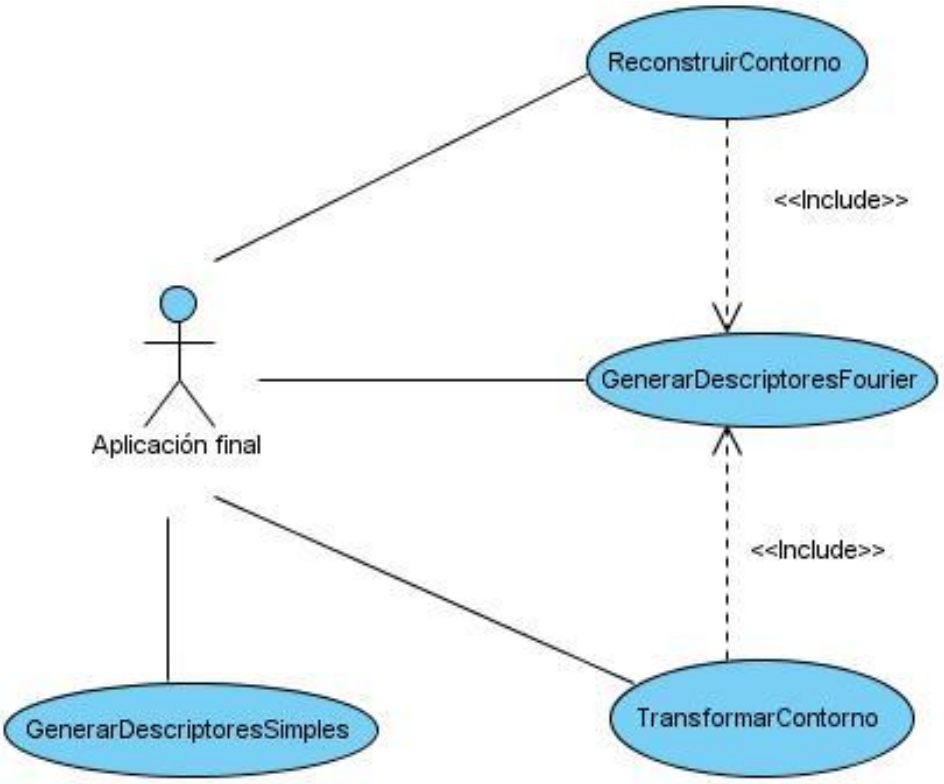

Figura 16*. Diagrama CU del Sistema*

3.3.3 Descripción de Casos de Uso del Sistema

Cada CU tiene una descripción de las funcionalidades que ejecutará el sistema propuesto como respuesta a las acciones del usuario. A continuación se muestran las tablas correspondientes a las descripciones de los CU del módulo:

Caso de Uso:	Generar Descriptores de Fourier
Actor:	Aplicación final (inicia)
Propósito:	Hacer el cálculo de los descriptores de Fourier.
Resumen:	El CU se inicia cuando a la aplicación llega el contorno segmentado para realizar el cálculo de los descriptores de Fourier. Finaliza cuando la aplicación guarda en memoria el resultado del cálculo realizado.
Referencia:	RF1
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El contorno segmentado llega al sistema y este solicita su cálculo.	1.1 Calcula descriptores de Fourier. 1.2 Guarda en memoria resultado del cálculo de descriptores de Fourier.
Post-condiciones:	Ofrece los valores frecuenciales de cada píxel y el contorno queda modificado.
Prioridad:	Crítico.

Tabla 2. *Descripción de CU Generar Descriptores de Fourier*

Caso de Uso:	Transformar Contorno
Actor:	Aplicación final (inicia)
Propósito:	Hacer la transformación del contorno.
Resumen:	El CU se inicia cuando el sistema necesita hacer algunas de las transformaciones geométricas (rotación, escala, traslación y cambio

	de posición del píxel inicial) al contorno. El sistema ofrece un contorno transformado según la categoría seleccionada.
Referencia:	RF1,RF2,RF2.1,RF2.2,RF2.3,RF2.4
Flujo Normal de Eventos: Sección Rotar.	
Acción del Actor	**Respuesta del Sistema**
1. El contorno necesitar ser rotado. El sistema solicita la rotación.	1.1 Hace los cálculos pertinentes para obtener el contorno rotado. 1.2 Ofrece contorno rotado.
Flujo Alterno de Eventos: Sección trasladar.	
Acción del Actor	**Respuesta del Sistema**
1. El contorno necesitar ser trasladado. El sistema solicita la traslación.	1.1 Realiza los cálculos pertinentes para obtener el contorno trasladado. 1.2 Ofrece contorno trasladado.
Flujo Alterno de Eventos. Sección Escalar	
Acción del Actor	**Respuesta del Sistema**
1. El contorno necesita ser escalado. El sistema solicita la escala.	1.1 Realiza los cálculos pertinentes para obtener el contorno trasladado. 1.2 Ofrece contorno escalado.
Flujo Alterno de Eventos: Sección Cambio de posición.	
Acción del Actor	**Respuesta del Sistema**

1. El contorno necesita que el píxel de inicio sea alterado. El sistema solicita el cambio de posición del píxel inicial.	1.1 Realiza los cálculos pertinentes para obtener el cambio de posición píxel inicial. 1.2 Ofrece contorno con cambio de posición.
Post-condiciones:	El contorno queda trasladado, rotado, escalado o el píxel inicial cambia de posición.
Prioridad:	Secundario.

Tabla 3. *Descripción de CU Transformar Contorno*

Caso de Uso:	Reconstruir Contorno
Actor:	Aplicación final (inicia)
Propósito:	Su propósito es hacer la reconstrucción del contorno a través de los descriptores de Fourier.
Resumen:	El CU se inicia cuando el sistema necesita la reconstrucción del contorno para obtener nuevos valores. El sistema guarda en memoria el contorno reconstruido.
Referencia:	RF4,RF1
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El sistema necesita la reconstrucción del contorno.	1.1 Ejecuta CU Calcular Descriptores de Fourier. 1.2 Reconstruye el contorno.

Post-condiciones:	Contorno reconstruido.
Prioridad:	Secundario.

Tabla 4. *Descripción de CU Reconstruir Contorno*

Caso de Uso:	Generar Descriptores Simples
Actor:	Aplicación final (inicia)
Propósito:	Su propósito es hacer el cálculo de los descriptores simples.
Resumen:	El CU se inicia cuando el sistema necesita hacer algunos cálculos al contorno para de esta manera obtener características cuantitativas de la misma, los cuales están dados por el cómputo del diámetro, longitud y curvatura. El sistema ofrece características cuantitativas del contorno.
Referencia:	RF3
Flujo Normal de Eventos: Calcular Longitud.	
Acción del Actor	**Respuesta del Sistema**
1. El sistema necesita los valores del contorno en cuanto a la longitud. El sistema solicita el cálculo.	1.1 Hace los cálculos pertinentes para obtener la longitud del contorno. 1.2 Guarda en memoria los valores obtenidos por el cálculo.
Flujo Alterno de Eventos: Calcular diámetro.	
Acción del Actor	**Respuesta del Sistema**

1. El sistema necesita los valores del contorno en cuanto al diámetro. El sistema solicita el cálculo.	1.1 Hace los cálculos pertinentes para obtener el diámetro del contorno. 1.2 Guarda en memoria los valores obtenidos por el cálculo.
Flujo Alterno de Eventos. Calcular curvatura	
Acción del Actor	**Respuesta del Sistema**
1. El sistema necesita los valores del contorno en cuanto a la curvatura. El sistema solicita el cálculo.	1.1 Hace los cálculos pertinentes para obtener la curvatura del contorno. 1.2 Guarda en memoria los valores obtenidos por el cálculo.
Post-condiciones:	Se obtiene las características cuantitativas del contorno por el cálculo del diámetro, longitud y curvatura, los mismos son guardados en memoria.
Prioridad:	Secundario.

Tabla 5: *Descripción de CU Generar Descriptores Simples*

3.4 Clases del diseño del sistema

El diagrama de clases del diseño fue concebido por paquetes con el objetivo de lograr una mayor organización e independencia en el módulo en el que se está trabajando para la elaboración de la aplicación final. Para ello se desarrollaron dos paquetes: *Segmentation* y *ContournDescription*. En el paquete de *Segmentation* se encuentra implementada la lógica de segmentación que se utiliza como entrada al proceso planteado en la propuesta de solución; mientras que en el paquete *ContournDescription* está centrada la solución al problema planteado, permitiendo el cálculo de contorno a través de las distintas técnicas de descripción analizadas en el capítulo 2: descriptores simples y descriptores de Fourier.

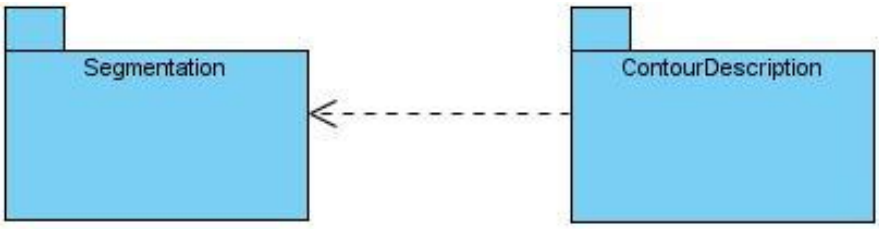

Figura 17. *Diagrama de paquetes del Sistema*

3.4.1 Diagrama de clases del diseño

El módulo fue desarrollado como muestra el diagrama de clases del diseño (ver Figura 18), donde se aplica el patrón fachada mediante la clase *ContourController* que contiene todas las funcionalidades del módulo y redirige la ejecución entre las clases definidas. El patrón fachada se utiliza para proporcionar una interfaz de alto nivel para un conjunto de clases en un subsistema, haciéndolo más fácil de usar. Simplifica el acceso a dicho conjunto de clases, ya que el cliente sólo se comunica con ellas a través de una única interfaz [15].

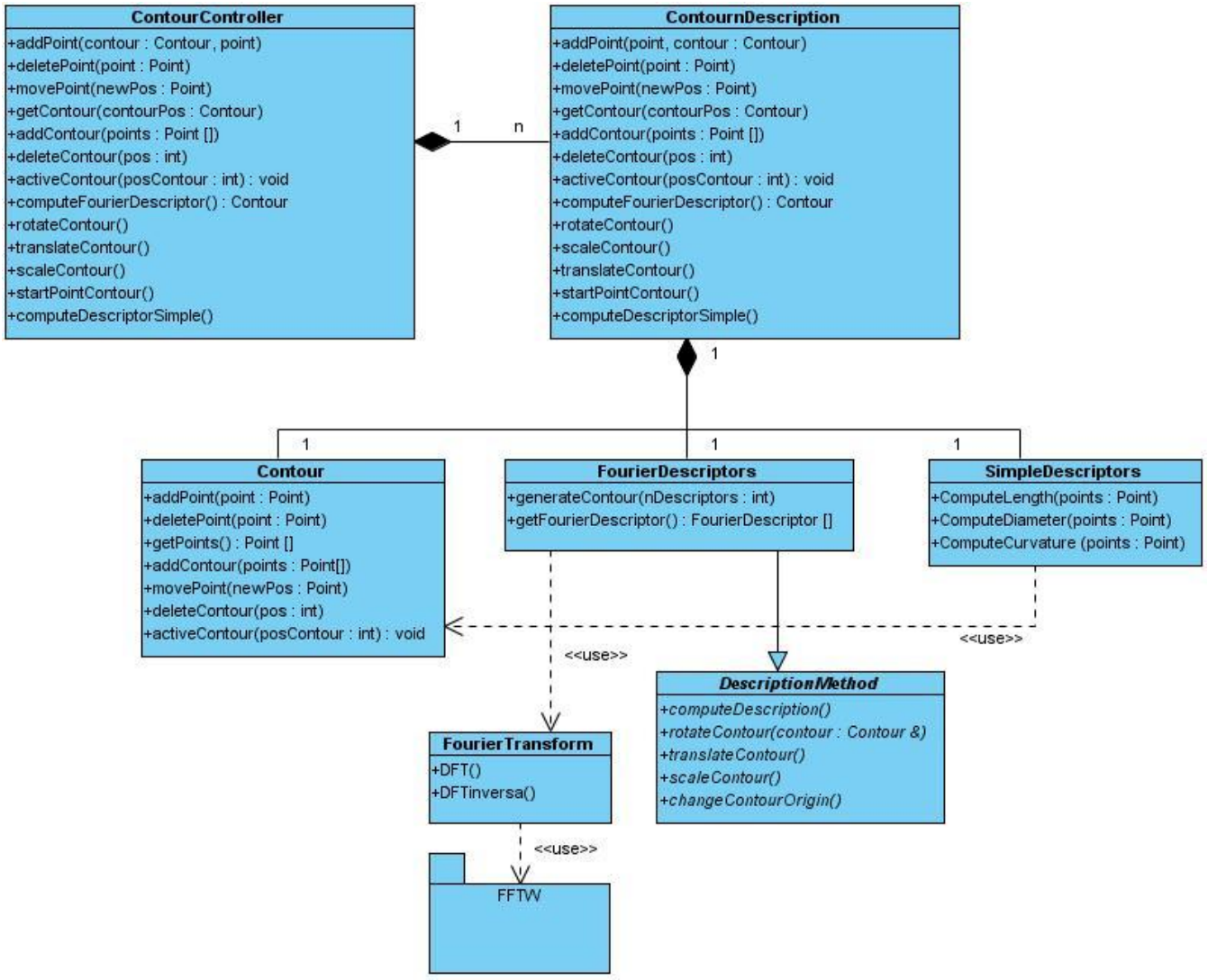

***Figura 18**. Diagrama Clases del Diseño*

3.5 Diagrama de secuencia de las clases del diseño

A continuación se representarán los diagramas de secuencia del diseño para tener una idea más general sobre el flujo que se realiza entre las clases del diseño y que posibilita comprender mejor el módulo elaborado en términos de implementación.

3.5.1 Diagrama de secuencia del caso de uso Calcular Descriptores de Fourier

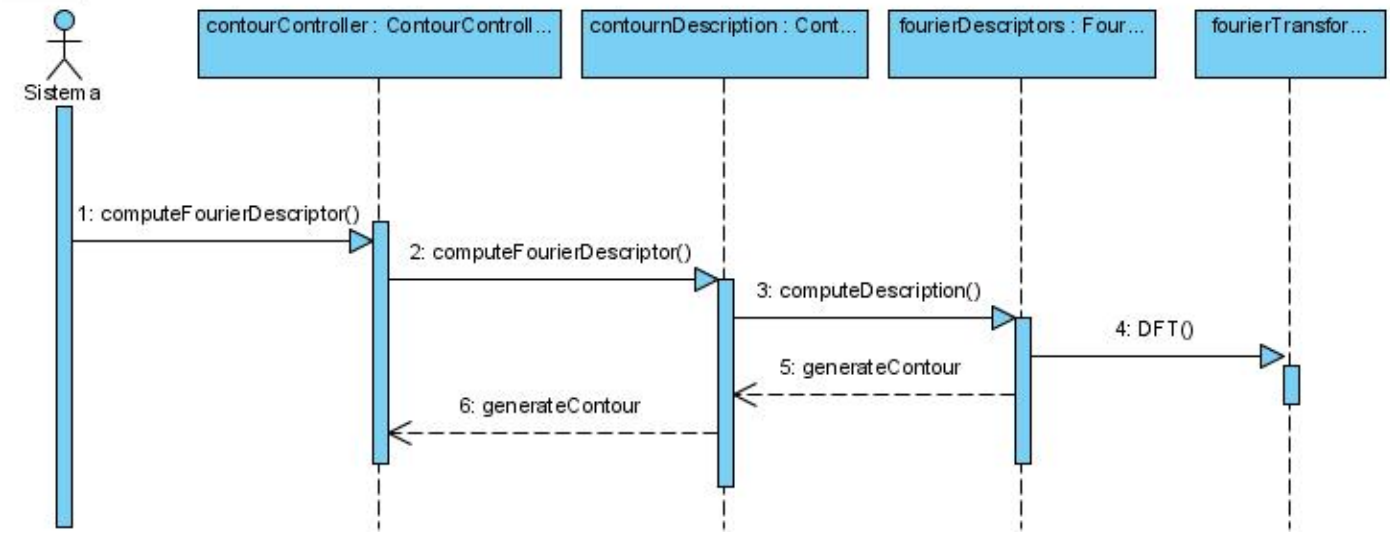

Figura 19: *Diagrama de Secuencia del cálculo de DFT*

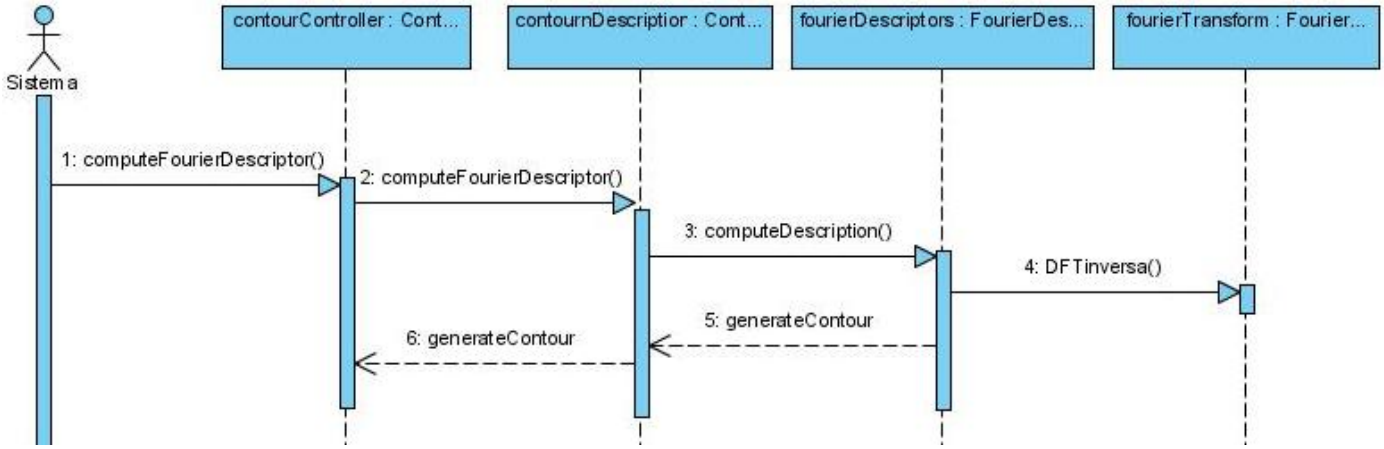

Figura 20: *Diagrama de Secuencia del cálculo de DFTinversa*

3.5.2 Diagrama de secuencia del caso de Uso Transformar Contorno

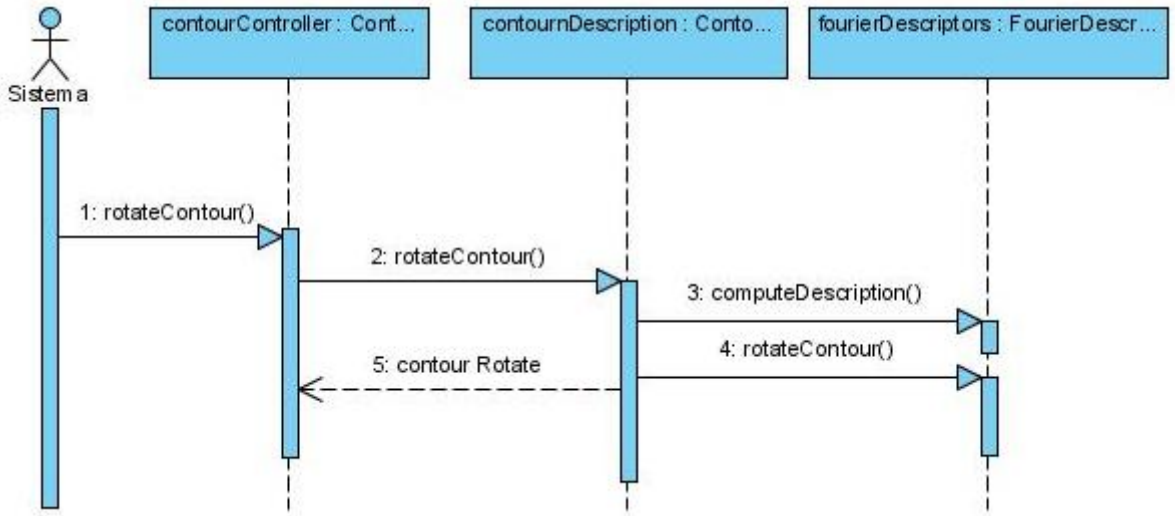

Figura 21: *Diagrama de Secuencia de Rotar Contorno*

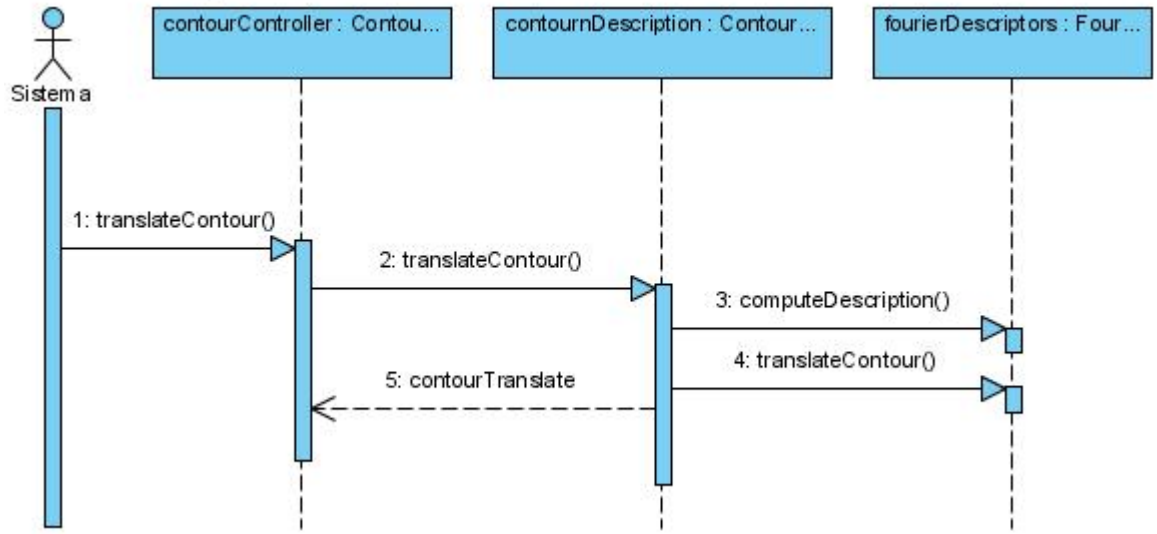

Figura 22: *Diagrama de Secuencia de Trasladar Contorno*

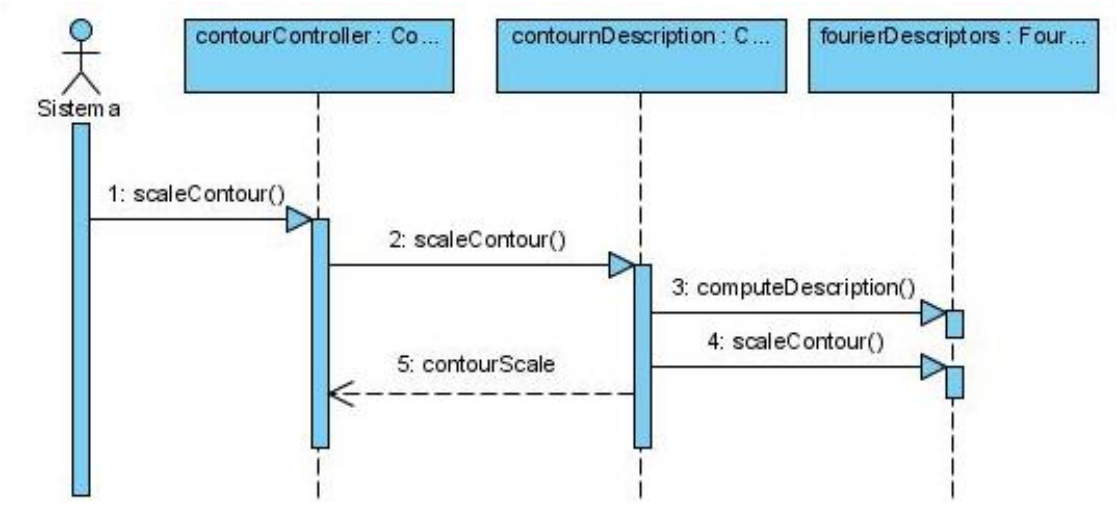

Figura 23: *Diagrama de Secuencia de Escalar Contorno*

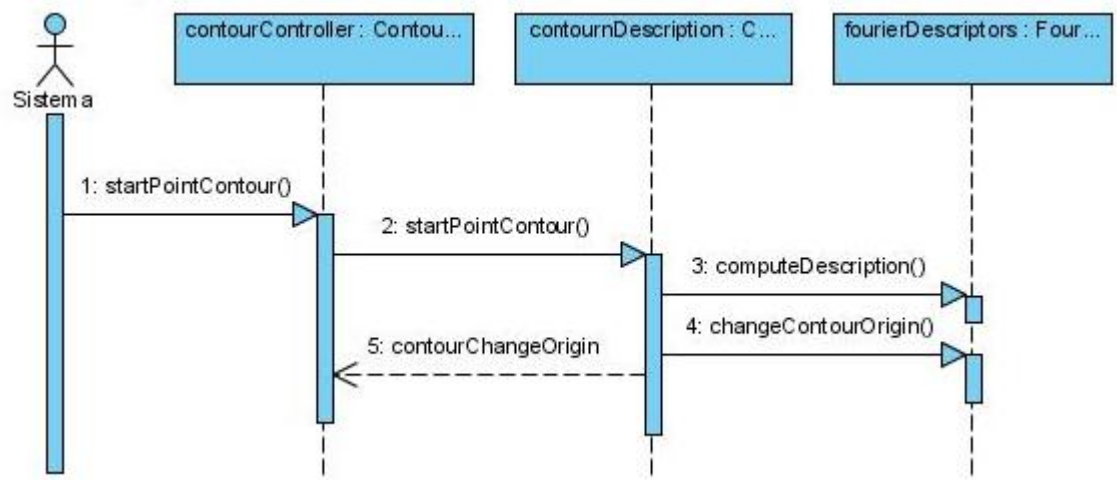

Figura 24: *Diagrama de Secuencia de Cambio de Posición del Contorno*

3.5.3 Diagrama de secuencia del caso de uso Reconstruir Contorno

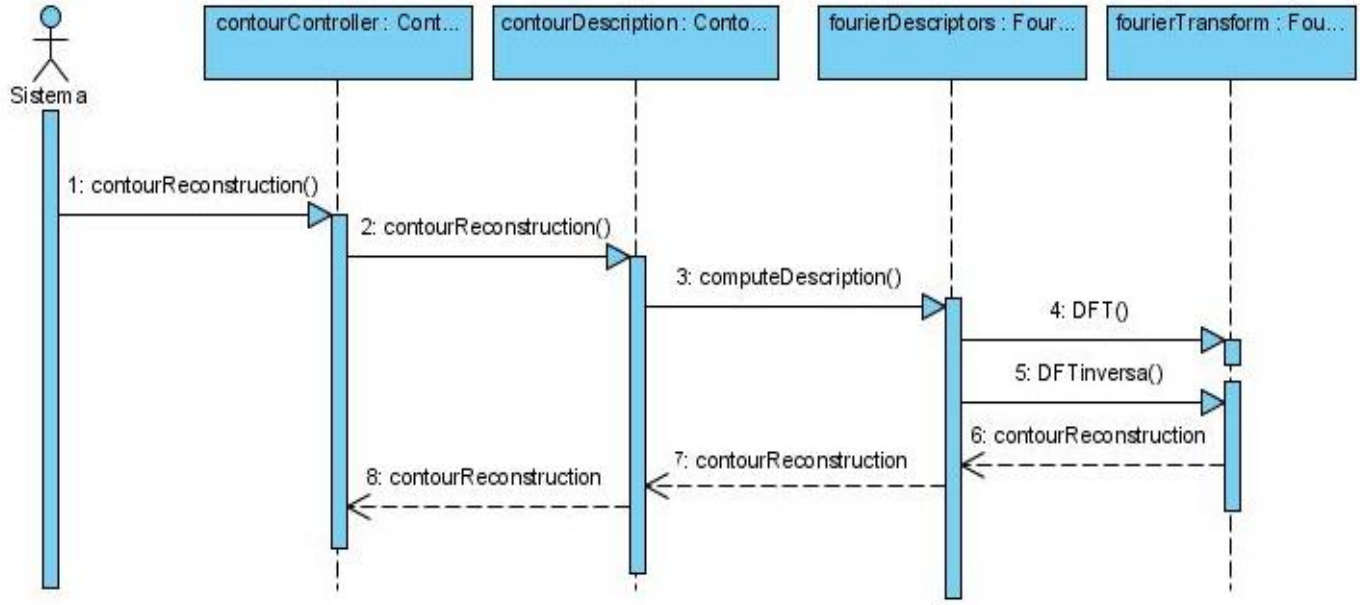

Figura 25: *Diagrama de Secuencia Reconstruir Contorno*

3.5.4 Diagrama de secuencia del caso de uso Calcular Descriptores Simples

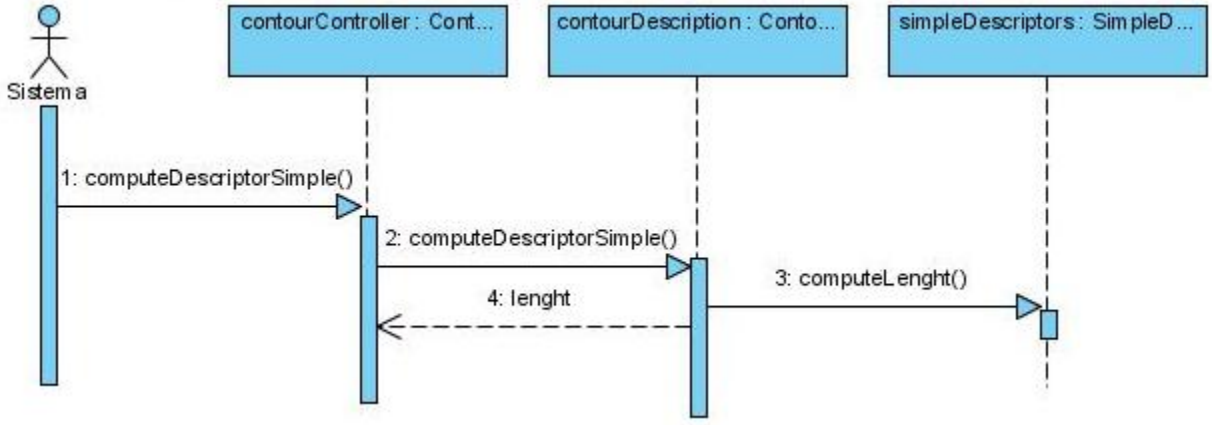

Figura 26: *Diagrama de Secuencia de Cálculo de longitud*

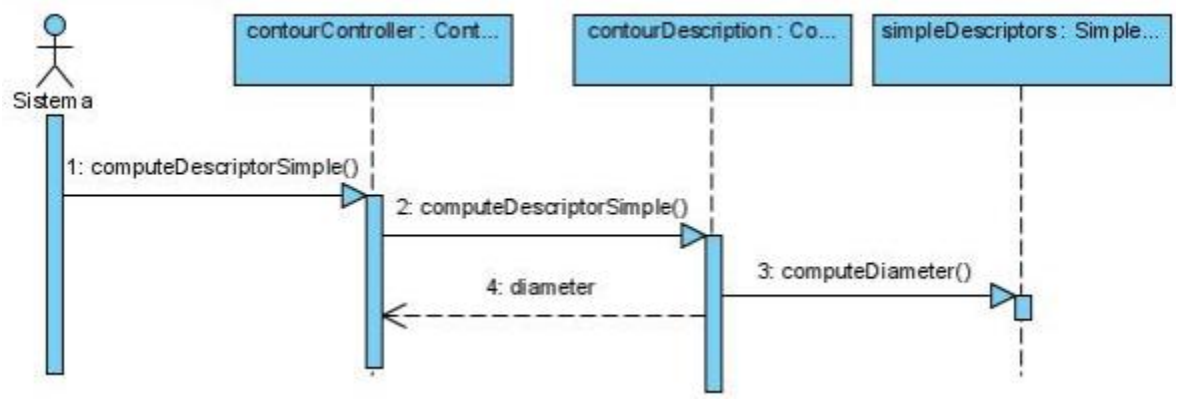

Figura 27 *Diagrama de Secuencia de Cálculo del diámetro*

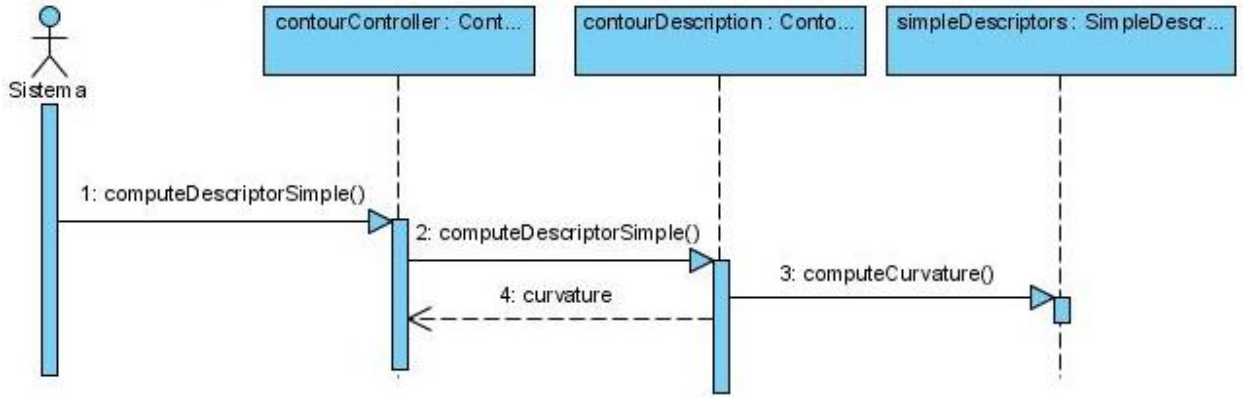

Figura 28: *Diagrama de Secuencia de Cálculo de la curvatura*

3.6 Conclusiones generales del capítulo

Quedó definido el diseño del sistema que se desarrolla, haciendo referencia a los requisitos funcionales y no funcionales, actores del sistema y diagrama de CU. De cada CU se hace una descripción detallada. También quedaron realizados los diagramas de secuencia por cada uno de los escenarios definidos a partir de los principales requisitos funcionales.

Capítulo IV. Implementación y Validación de los resultados

En este capítulo se aborda los temas de implementación del módulo utilizando el trabajo realizado en los capítulos anteriores. También se muestra el diagrama de componentes del sistema desarrollado. Se realizarán pruebas para validar los resultados y medir aspectos como el rendimiento de las técnicas implementadas en el módulo.

4.1 Implementación

En esta etapa se realiza la implementación de las clases y objetos en ficheros fuente, binarios y ejecutables. Como resultado se obtiene un sistema ejecutable que incluye todas las funcionalidades propuestas en la captura de requisitos funcionales. La estructura de todos estos modelos forma el modelo de implementación.

4.1.1 Diagrama de componentes

Las clases que se obtienen en el diseño se hacen físicas mediante componentes. Los componentes representan módulos de software (código fuente, código binario, ejecutables, archivos con extensión dll) con una interfaz bien definida. Para organizar mejor las dependencias entre los componentes que se elaboran se encuentra el diagrama de componentes, el mismo es usado para estructurar el modelo de implementación en términos de subsistemas de implementación y mostrar las relaciones entre sus elementos. A continuación se muestra el diagrama correspondiente al sistema desarrollado:

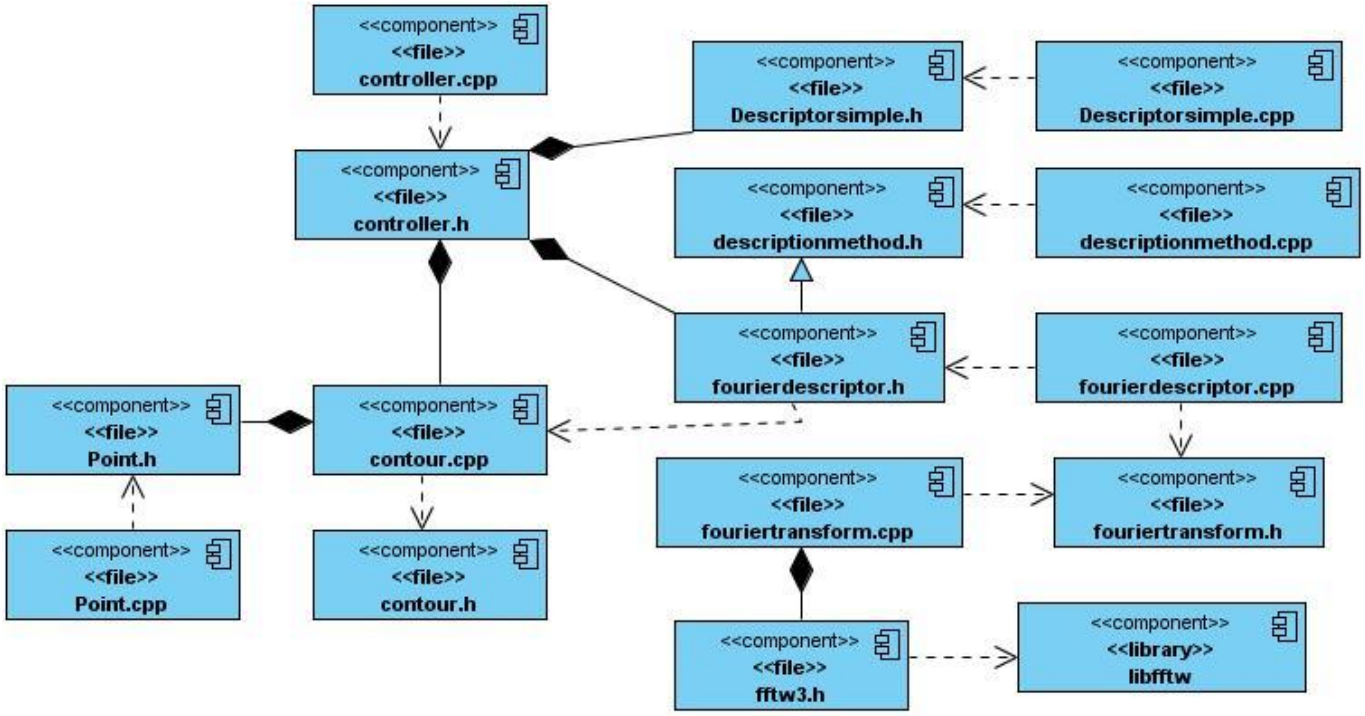

Figura29: *Diagrama de Componentes*

4.2 Validación

Para la validación de la herramienta se realizaron comparaciones funcionales, que consisten en probar el módulo contra los requerimientos capturados. Para realizar las comparaciones funcionales se siguió una estructura de la siguiente forma: Entrada, Resultados y Descripción. Las validaciones se realizarán en orden lógico o sea, primero validaremos el funcionamiento de Generar Descriptores Fourier, luego las Transformaciones de Contorno, Reconstrucción de Contorno y por último Generar Descriptores Simples.

4.2.1 Caso 1: Generar Descriptores Fourier

Entrada: Contorno segmentado para realizar el cálculo de los descriptores de Fourier.

Resultado: Se calcula los descriptores de Fourier de cada píxel del contorno.

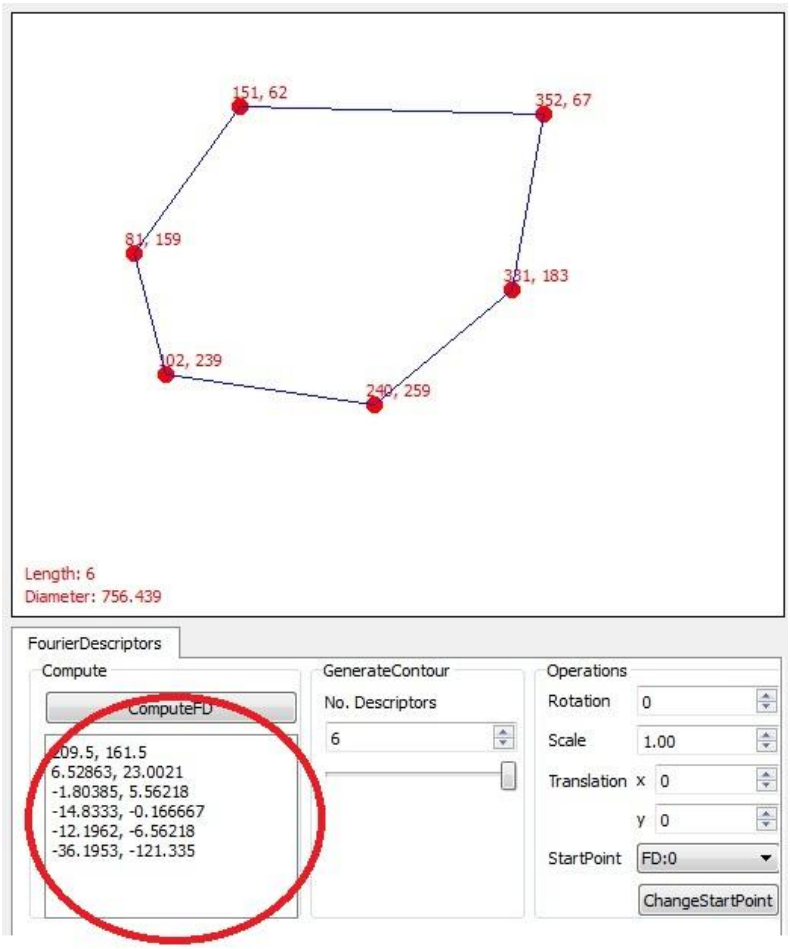

Figura 30: *Generar Descriptores de Fourier*

Descripción: Se verifica que los descriptores de Fourier son cargados y mostrados para su análisis.

4.2.2 Caso 2: Transformar Contorno

Entrada: Contorno previamente calculado por los descriptores de Fourier se ejecuta su transformación geométrica (traslación, rotación, escala, cambio de posición del punto de partida) de acuerdo a los datos pasados.

Resultado: Se muestra el contorno con su cambio según la transformación geométrica predefinida.

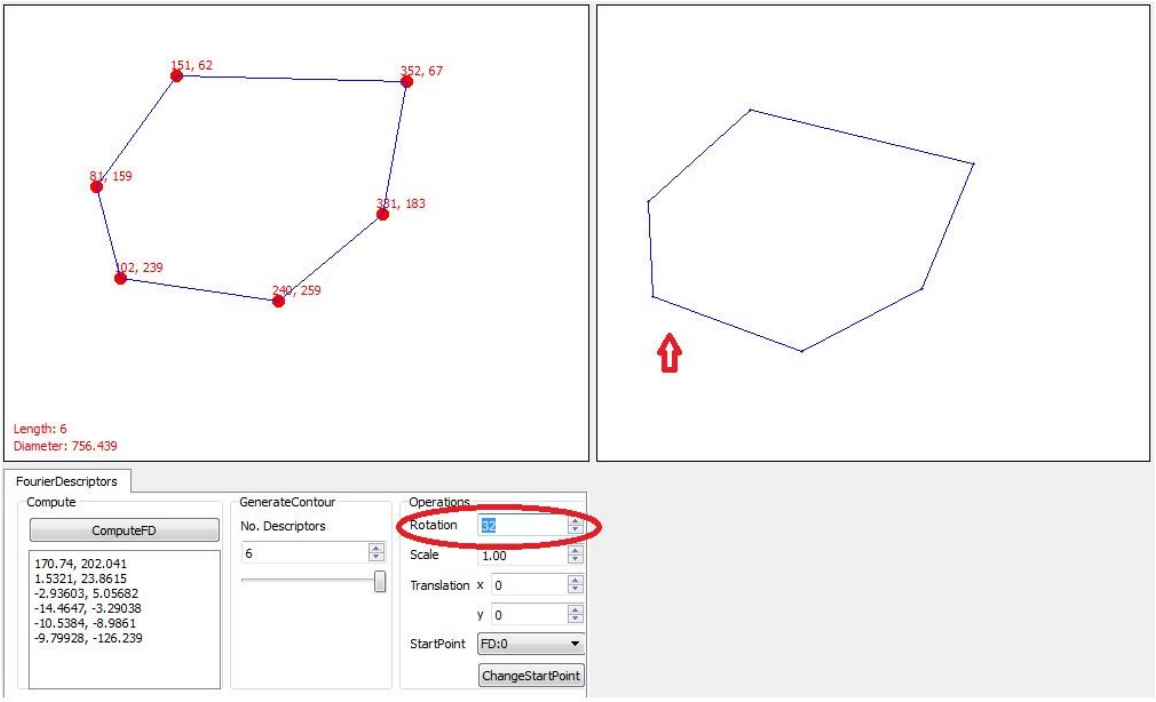

Figura 31: *Transformación geométrica: Rotación*

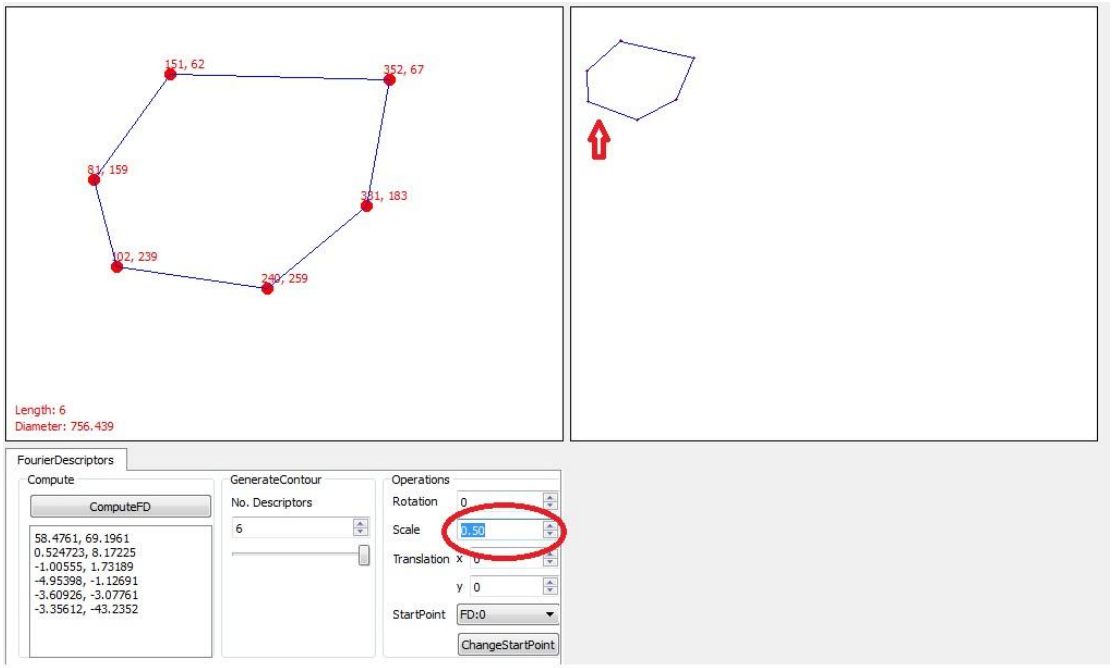

Figura 32: *Transformación geométrica: Escala*

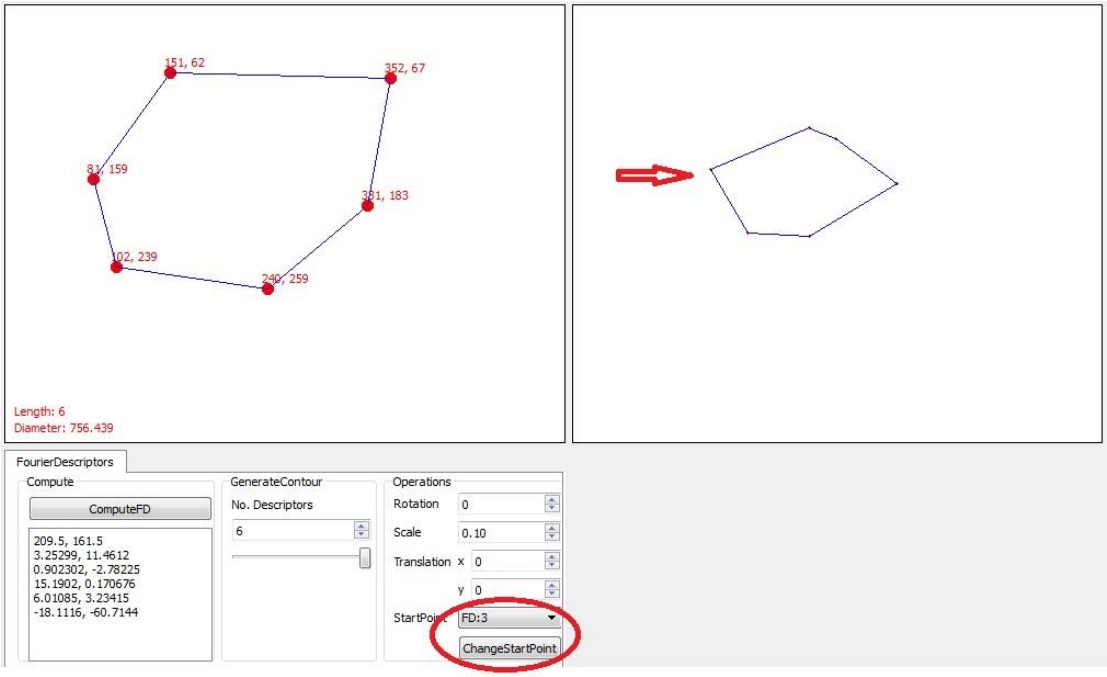

Figura 33: *Transformación geométrica: Cambio de posición del primer píxel*

Descripción: Se verifica que las transformaciones requeridas se le realicen al contorno.

4.2.3 Caso 3: Reconstruir Contorno

Entrada: Contorno de píxeles el cual tiene que ser reconstruido con la menor cantidad de descriptores o la mayor.

Resultado: Se muestra el contorno con cambios realizados de la reconstrucción del mismo.

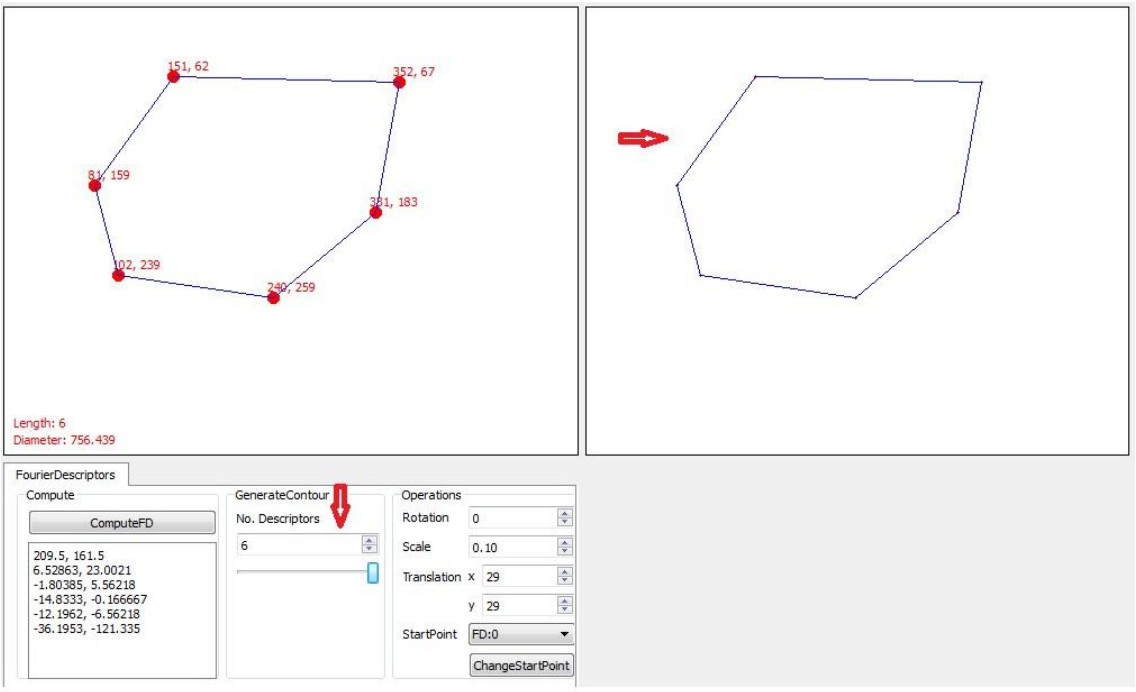

Figura 34: *Reconstruir contorno*

Descripción: Se verifica que el contorno pueda reconstruirse con el menor número de descriptores, llegando a la conclusión de que con mayor cantidad de descriptores se obtiene un contorno más definido.

4.2.4 Caso 4: Generar Descriptores simples

Entrada: Contorno expresado por píxeles.

Resultado: Se muestra los cálculos realizados al contorno a través de los descriptores simples.

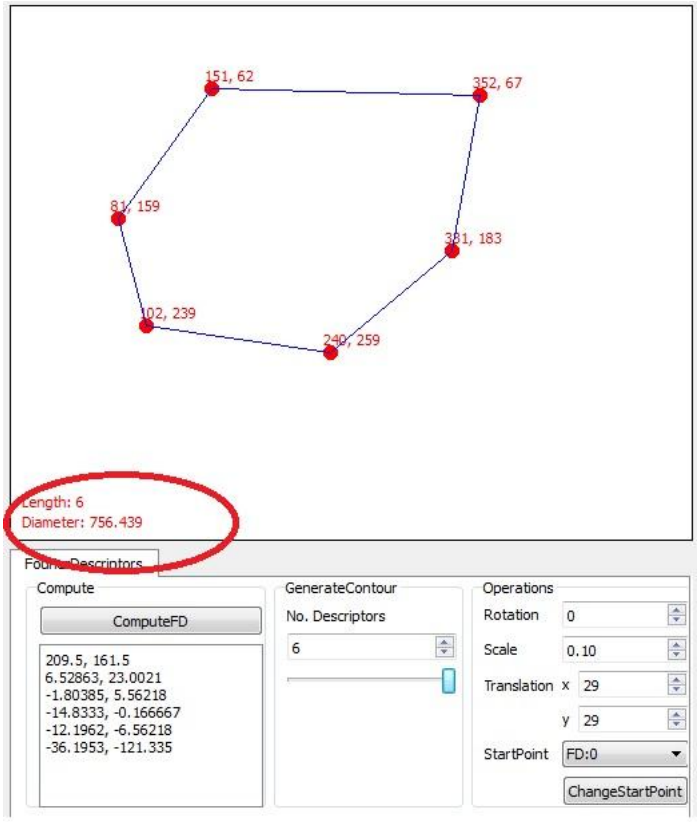

Figura 35: *Generar descriptores simples*

Descripción: Se verifica los resultados mostrados por el cálculo de los descriptores simples.

4.3 Conclusiones generales del capítulo

Quedó definido el diagrama de componentes para obtener un mejor diseño de los componentes físicos del sistema. Después de haber realizado las pruebas se comprobó que se cumplían con los RF del módulo comprobándose la funcionalidad del mismo, arrojando resultados satisfactorios.

Conclusiones

Con la realización de este trabajo se concluye que:

- Se desarrolló un módulo de descripción de contornos en imágenes médicas digitales para ser utilizado en el proyecto VisMedic, el cual a través de los algoritmos y métodos de descripción implementados permite obtener características cuantitativas para ser utilizado en el reconocimiento de patrones.
- La aplicación obtenida permite hacer transformaciones al contorno y obtener los descriptores necesarios para estudios de similitud de contornos, elemento fundamental dentro del reconocimiento de patrones.

Recomendaciones

Para este trabajo se recomienda:

- Implementar e incorporar otras técnicas de descripción de contornos y regiones para extender las funcionalidades y la aplicabilidad del módulo desarrollado.
- Extender los métodos de descripción desarrollados a 3D para poder ser utilizados en el reconocimiento de patrones en tres dimensiones.
- Integrar el módulo desarrollado al sistema de visualización médica VisMedic.

Referencia bibliográfica

1. **Woods, Richard E.** *Digital Image Processing.* New Jersey : Prentice Hall, 2002.

2. **Merry, Domingo.** *Extracción de características.* Santiago de Chile : s.n., 2006.

3. **Sarfraz, M.** *Object Recognition using Fourier Descriptors: Some Experiments and Observations.* Arabia Saudita : s.n., 2006. 1510.

4. **Rajput, G.** *Marathi Handwritten Numeral Recognition using Fourier Descriptors and Normalized Chain Code.* India : s.n., 2010. 585106.

5. **P. Lionel Evina Ekombo, Noureddine Ennahnahi.** *Application of affine invariant Fourier descriptor to shape based image retrieval.* s.l. : Vol 9 . No 7, 2009.

6. *Transformada de Fourier. Parte I.*

7. **Molina, R.** *Introducción al Procesamiento y Análisis de Imágenes Digitales.* Granada : s.n., 1998. 18071.

8. **Fraga, Luis Gerardo de la.** Introduccion Curso:Procesamiento Digital de Imágenes. *Procesamiento Digital de Imágenes.* 2001.

9. **Morse, Bryan.** Lecture 7: Shape Description (Contourn). 2000.

10. Capítulo 8: Representación de formas y descripción.

11. **Rodríguez, Jose Luis Gil.** *Estado Actual de la Representación y Análisis de textura de imágenes.* Ciudad de la Habana : s.n., 2008.

12. **Malpartida, Eddie Angel Sobrado.** *Sistema de Visión Artificial para el Reconocimiento y manipulación de objetos utilizando un brazo robot.* Lima : s.n., 2003.

13. **Freddy Edgar Carranza, Laura Elizabeth Florian Cruz.** *Extracción de características para la recuperación de imágenes médicas por contenido utilizando las wavelets de Gabor.* Trujillo : s.n., 2008.

14. Capítulo 6. Transformada Rápida de Fourier(FFT).

15. **Ivar Jacobson, Grady Booch, James Rumbaugh.** *El Proceso Unificado de Desarrollo de Software.* Madrid : FARESO.S.A, 2000.

16. **Martín, David García.** *Desarrollo de aplicación para visión artificial.* Sevilla : s.n., 2008.

17. **Merry, Domingo.** *Extracción de características.* Santiago de Chile : s.n., 2006.

18. **Karrels, Tyler.** *Fourier Descriptors: Properties and Utility in Leaf Classification.* 2006.

19. **Ismail A Ismail, Mohamed Ramadan, Talaat El-Danaf.** *An Efficient Off-line Signature Identification Method Based On Fourier Descriptor and Chain Codes.* Egypt : s.n., 2010.

20. **Ismail A Ismail, Mohammed Ramadan, Talaaf El danaf.** *Signature Recognition using Multi Scale Fourier Descriptor And Wavelet Transform.* Egypt : s.n., 2010.

21. **Emilce Moler, Juan Pastore.** *Técnicas de procesamiento digital de imágenes: una aplicación para identificación de personas a través de los senos frontales.* Argentina : s.n., 2010.

22. **Dengsheng Zhang, Guojun Lu.** *A Comparative Study on Shape Retrieval Using Fourier Descriptors with different shape signatures.* Australia : s.n.

23. **P. Lionel Evina Ekombo, Noureddine Ennahnahi, Mohammed Oumsis.** *Application of affine invariant Fourier descriptor to shape based image retrieval.* Morrocco : s.n., 2009.

24. **Jaume, Universitat.** *Aplicción de descriptores de Fourier y redes neuronales artificiales para el reconocimiento de formas.* Jaume : s.n., 2006.

25. **Babu M. Mehtre, Mohan S. Kankanhalli.** *Shape measures for content based image retrieval: a comparison.* Singapure : s.n., 1996. SO306-4573(96)00069-6.

26. **Fabian Timm, Thomas Martinetz.** *Statistical Fourier Descriptors for Defect Image Classification.* Germany : s.n., 2010. 1051-4651/10.

27. **Pratt, William K.** *Digital Image Processing.* California : WILEY, Fourth Edition.

28. **Andre Folkers, Hanan Samet.** *Content-based Image Retrieval Using Fourier Descriptors on a Logo Database.* Canada : s.n., 2002.

29. **Matteo Frigo, Steven G. Johnson.** FFTW. *FFTW.* [Online] 2009. http://www.fftw.org/.

30. VTK. *VTK.* [Online] Mayo 2010. http://www.vtk.org/.

31. **Moreno, Ignacio Berzal.** *Desarrollo de algoritmos de procesamiento de imágenes con VTK.* Madrid : s.n.

32. **Bradsky, Gary.** OpenCVWiki: Welcome. *OpenCVWiki: Welcome.* [Online] Mayo 2011. http://opencv.willowgarage.com/wiki/.

33. **Tschumperlé, David.** The CImg Library. *The CImg Library.* [Online] Octubre 2004. http://cimg.sourceforge.net/.

34. ITK Insight Toolkit. *ITK Insight Toolkit.* [Online] http://www.itk.org/.

35. **Matteo Frigo, Steven G. Johnson.** *FFTW.* 2009. Versión 3.2.2.

Glosario de términos

A

Algoritmo: Es una lista que, dado un estado inicial y una entrada, propone pasos sucesivos para arribar a un estado final obteniendo una solución.

C

Conectividad: Es usada para establecer las fronteras de objetos y las regiones componentes de una imagen. Para establecer si dos píxeles están conectados hemos de establecer si son adyacentes en algún sentido (por ejemplo si son 4-vecinos y si sus niveles de gris cumplen algún criterio de similaridad, por ejemplo ser iguales). Así o en una imagen binaria de 0 y 1 dos píxeles pueden ser 4- vecinos y no estar conectados salvo que tengan el mismo valor.

Existen tres tipos de conectividad:

- 4-conectividad. Dos píxeles *p* y *q* con valores en *V* se dicen 4-conectados si *q* pertenece a $N_4(p)$.
- 8-conectividad. Dos píxeles *p* y *q* con valores en *V* se dicen 8-conectados si *q* pertenece a $N_8(p)$.
- m-conectividad. Dos píxeles *p* y *q* con valores en *V* se dicen m-conectados si:

  a) $q \in N_4(p)$ o

  b) $q \in N_D(p)$ y $N_4(p) \cap N_4(q)$ es vacío. (Este es el conjunto de 4-vecinos de *p* y *q* con valores en *V*).

Es importante señalar que la m-conectividad se introduce para eliminar la ambigüedad en los posibles caminos que unen dos píxeles.

Contorno: Conjunto de las líneas que limitan una figura o composición.

Cuadrícula: Conjunto de líneas horizontales y verticales uniformemente espaciadas. Se utilizan cuadrículas para ayudar a dibujar gráficas o localizar puntos en una gráfica.

D

Diagnóstico: Etimológicamente el concepto diagnóstico proviene del griego, tiene dos raíces, día- que es a través de, por. Y gignoskein que es conocer, así etimológicamente diagnóstico significa conocer a través de. El concepto de este significado (imagen que representamos en la mente) es la identificación de la naturaleza o esencia de una situación o problema y de la causa posible o probable del mismo, es el análisis de la naturaleza de algo.

Digital: Quiere decir que utiliza o que contiene información convertida al código binario, el lenguaje de números (ceros y unos) que emplean los ordenadores para almacenar y manipular los datos.

F

Framework: En el desarrollo de software, un *framework* es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un *framework* puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Frecuencia: En el procesado de imagen se utiliza normalmente para referirse a frecuencia espacial, y aunque la palabra frecuencia se suele asociar a variaciones en tiempo, es importante tener claro que se refiere a la frecuencia con la que una señal (la imagen) varía como una función de las coordenadas espaciales.

Frecuencia espacial se define como la distribución espacial de iluminaciones, que sigue una ley sinusoidal caracterizada por su amplitud, frecuencia y fase.

Forma: Figura exterior (o geometría) de un cuerpo o objeto.

H

Hardware: Componentes físicos de una computadora o de una red (a diferencia de los programas o elementos lógicos que los hacen funcionar).

I

Imagen: Figura, representación, semejanza y apariencia de algo. En computación es formada por la unión de MxN píxeles (imagen 2D) o vóxeles (imagen 3D).

Imagenología: Comprende la realización de todo tipo de exámenes diagnósticos y terapéuticos en los cuales se utilizan equipos que reproducen imágenes del organismo. Los siete servicios de Imagenología son Ecotomografía, Imagenología Mamaria, Medicina Nuclear, Radiología, Rayos Infantil, Resonancia Magnética y Tomografía Computada o Scanner.

Interfaz gráfica de usuario: También conocida como GUI (del inglés *graphical user interface*). Es un programa informático que utilizando un conjunto de imágenes y objetos gráficos representa la información y acciones disponibles en la interfaz. Su principal uso consiste en proporcionar un entorno visual sencillo para permitir la comunicación con el usuario.

M

Metodología: Conjunto de métodos que se siguen en una investigación científica o en una exposición doctrinal.

P

Píxel: Abreviatura de "*picture element*". Es la mínima unidad de información dentro de una imagen bidimensional.

R

Realidad Virtual: Simulación generada por computadora de imágenes o ambientes tridimensionales interactivos con cierto grado de realismo físico o visual.

Resolución: Es el número de píxeles que se muestran en una pantalla. Al ser ésta una matriz de filas y columnas de píxeles, primero se nombra la cantidad de columnas (resolución horizontal) y luego la cantidad de filas (resolución vertical).

Resonancia Magnética (MRI): Modalidad de diagnóstico radiológico que utiliza tecnología de resonancia magnética nuclear en la que los núcleos magnéticos (especialmente los protones) del paciente se alinean en un campo magnético potente y uniforme, absorben energía de

impulsos afinados de radiofrecuencia y emiten señales de radiofrecuencia a medida que decae su excitación. Estas señales, que varían en intensidad según la abundancia nuclear y el entorno químico molecular, se convierten en imágenes tomográficas (cortes seleccionados) mediante el uso de gradientes de campo en el campo magnético, lo que permite la localización tridimensional de las fuentes de las señales.

S

Segmentación: Se utiliza en el Procesamiento de Imágenes para el reconocimiento de objetos o estructuras de interés en la imagen.

T

Tomografía Axial Computarizada (TAC): Es un examen médico no invasivo ni doloroso que ayuda al médico a diagnosticar y tratar enfermedades. Las imágenes por TAC utilizan un equipo de rayos X especial para producir múltiples imágenes o visualizaciones del interior del cuerpo, a la vez que utiliza conjuntamente una computadora que permite obtener imágenes transversales del área en estudio. Luego, las imágenes pueden imprimirse o examinarse en un monitor de computadora.

V

Vecinos de un píxel: Un píxel de *p* coordenadas *(x,y)* tiene cuatro vecinos horizontales y verticales cuyas coordenadas son $(x+1, y), (x-1, y), (x, y+1),)(x, y-1)$ este conjunto de píxeles, que recibe el nombre de 4-vecinos de *p*, se denota $N_4(p)$. Cada píxel esta a distancia unitaria de *(x,y).*

Los píxeles diagonales vecinos de *p* tienen coordenadas: $(x+1, y+1), (x-1, y-1), (x-1, y+1),)(x+1, y-1)$ y se denota $N_D(p)$. Estos puntos, junto con los cuatro vecinos, se llaman 8-vecinos de *p* y se denota $N_8(p)$.