Desarrollo de un Algoritmo de Corte de Mallas Tridimensionales en Ambientes Virtuales con Aplicaciones en Simuladores Quirúrgicos

Amaury Andrés Peniche Gonzalez

Escuela de Ingenierías
Departamento de Sistemas
Universidad EAFIT
21 de octubre de 2009

Desarrollo de un Algoritmo de Corte de Mallas Tridimensionales en Ambientes Virtuales con Aplicaciones en Simuladores Quirúrgicos

Amaury Andrés Peniche Gonzalez

Tesis de Pregrado para optar al título de Ingeniero de Sistemas

Asesor: Christian Andrés Diaz León

Escuela de Ingenierías
Departamento de Sistemas
Universidad EAFIT
21 de octubre de 2009

Índice general

Índice de figuras

Índice de cuadros

Agradecimientos

FALTA EDITAR!!...
A Dios quien me da la

A mi familia quienes siempre estan ahi para apoyarme en cualquiera de mis decisiones.

A mi asesor Christian Andrés Diaz León quien estubo siempre dispuesto a ayudarme con cualquiera de los inconvenientes que se iban presentando al igual que siempre estubo pendiente de que el trabajo se estubiera desarrollando en el sentido correcto.

A Helmuth Trefftz por brindarme la oportunidad de desarrollar la tesis en este proyecto.

??

Resumen

Con las ventajas que ofrecen las técnicas de cirugías mínimamente invasivas frente a las cirugías convencionales, ha surgido la necesidad de un método efectivo para el entrenamiento de los nuevos cirujanos en este tipo de técnicas. Los métodos convencionales para el entrenamiento de cirujanos en cirugías mínimamente invasivas presentan una serie de problemas en terminos de costos, disponibilidad, consideraciones éticas, entre otros. Sin embargo teniendo en cuenta la aceptación que han tenido los simuladores quirúrgicos en general y más recientemente los simuladores de cirugías mínimamente invasivas los cuales buscan solucionar esta problemática, se han abierto varios campos de investigación enfocados al mejoramiento tanto del desempeño como del realismo de los mismos.

Los procedimientos de corte son una parte fundamental de toda cirugia, y en una cirugía mínimamente invasiva, donde se requiere de mayor exactitud en los cortes, cobra mucha mayor importancia. Por tal razón es necesario que un ambiente virtual enfocado al entrenamiento de cirugías laparoscópicas haga posible un buen entrenamiento de los cirujanos en estos procedimientos de corte. En esta tesis se presenta el desarrollo de un algoritmo de corte de mallas tridimensionales en ambientes virtuales con aplicaciones en simuladores quirúrgicos, dicho algoritmo determina la trayectoria seguida por el cirujano y ajusta el patrón de corte a esta tendencia con el fin de lograr cortes más realistas, esto es logrado mantiendo la interactividad del simulador ya que es capaz de ser ejecutado en tiempo real, y minimizando el número de nuevas primitivas creadas en la malla.

Capítulo 1

Introducción

Durante los últimos años, las cirugías mínimamente invasivas han sido objeto de estudio de diferentes ramas de la ciencia debido a las grandes ventajas que ofrecen con respecto a la salud del paciente y a los costos asociados a las cirugías. Desde sus inicios con la invención de la cámara comptacta, los procedimientos mínimamente se han desarrollado y diversificado en una gran variedad de procedimientos y aplicaciones, logrando de esta manera su viabilidad e importancia clínica.

La cirugía mínimamente invasiva es aquella que utiliza las vías naturales o incisiones pequeñas para introducir las herramientas con las cuales se lleva a cabo el procedimiento, asistido generalmente por métodos de visión como la endoscopia o alguna otra técnica de imagen. Las cirugías mínimamente invasivas son un término global que agrupa una diversidad de técnicas, las cuales se pueden clasificar según el espacio anatómico donde esta es aplicada. En el ámbito de la cirugía endocavitaria están técnicas como la artroscopia, aplicada en las articulaciones, toracoscopia, aplicada en el tórax, y la laparoscopia, aplicada en la cavidad abdominal. Igualmente en la cirugía endolumnial se encuentran la cirugía digestiva y urológica entre otras.

Específicamente, la laparoscopia es una técnica de cirugía mínimamente invasiva que consiste en realizar pequeñas incisiones en la pared abdominal del paciente por las cuales se introducen los instrumentos útiles durante el procedimiento quirúrgico, tales como pinzas, endoscopio, bisturí, electrocauterizadores, etc. Como las incisiones realizadas son considerablemente mas pequeñas que en un procedimiento convencional, la cirugía laparoscópica resulta mucho menos traumática para el paciente haciendo que se reduzca el periodo de hospitalización, recuperación y molestias post-operatorias, además de la evidente ventaja estética ya que se reducen en número y tamaño las cicatrices resultantes del procedimiento, además la exposición de estructuras corporales internas al ambiente exterior es reducido considerablemente cuando se le compara con un procedimiento quirúrgico abierto, lo que conlleva a un menor riesgo infeccioso sobre el paciente.

Para que un cirujano pueda realizar una cirugía laparoscópica tiene que pasar por un proceso de entrenamiento riguroso ya que las técnicas y condiciones en

las cuales se lleva a cabo la cirugía son muy distintas a las convencionales. Para empezar, en la cirugía convencional el cirujano dispone de más espacio para trabajar, mientras que, en la cirugía laparoscópica el movimiento está restringido por el tamaño del orificio, más complicado que eso, frecuentemente el cirujano debe trabajar con ángulos invertidos que ve mediante una pantalla en la que no hay una concordancia directa entre los movimientos que realiza y lo que ve, adicional a esto hay un perdida de la percepción de profundidad dentro del ambiente quirúrgico.

El entrenamiento de los cirujanos es entonces un proceso de vital importancia y cuidado. Actualmente los cirujanos se entrenan mediante varias técnicas que pueden ser costosas o inapropiadas. En general hoy en día se requiere de una combinación de estas técnicas con el fin de que el cirujano interiorize mejor estos procedimientos. El modelo convencional de aprendizaje utilizado para la enseñanza de este tipo de procedimientos consiste en la observación e interacción del aprendiz con un cirujano experto en un ambiente quirúrgico real sobre una paciente real. A medida que el aprendiz avanza en el proceso de aprendizaje va tomando un parte más activa durante el procedimiento quirúrgico, aumentando la interacción y disminuyendo al máximo la observación. Sin embargo este modelo de aprendizaje presenta varias desventajas como los son sus altos costos y el riesgo que debe correr el paciente cuando el cirujano en entrenamiento toma mayores responsabilidades. También se pueden llevar a cabo entrenamientos con cadáveres o con animales de anatomía similar, pero los costos en los que se incurren son muy elevados y la frecuencia de los entrenamientos puede ser muy baja, sin mencionar los problemas de tipo ético. También existen simuladores mecánicos que requieren de un lento proceso de preparación antes de iniciar cada sesión de entrenamiento, y que aparte de ser bastante costosos puede tener problemas de fidelidad (visual y háptica) cuando simulan un procedimiento quirúrgico real.

Las operaciones de corte son parte esencial de cualquier cirugía, y específicamente en la cirugía laparoscópica se hace de gran importancia debido a la limitada percepción que los cirujanos poseen en este tipo de ambientes quirúrgicos. Adicionalmente, el reducido espacio de movimiento de los instrumentos require de una cuidadosa manipulación de los instrumentos de corte. Por estas razones es importante que el cirujano adquiera las habilidades necesarias para realizar un operación de corte sin cometer errores y manteniendo en todo momento la integridad del paciente.

Los simuladores ofrecen un ambiente que soluciona gran parte de la problemática que esta presente en los métodos tradicionales, alcanzando una alta similitud con los procedimientos reales y eliminando todos los peligros al paciente, los altos costos, y finalmente proporcionando un entorno seguro, con alta disponibilidad y donde se pueden incluso aplicar métricas con las cuales es posible evaluar el desempeño del cirujano. Siendo muy importante las operaciones de corte, es vital que en cualquier simulador quirúrgico se pueda reproducir este fenómeno con el fin de que el cirujano logre obtener las destrezas necesarias para enfrentarse a una situación real.

La mayoría de los algoritmos que se han propuesto y que siguen el esquema

de separación están orientados básicamente a lograr una trayectoria de corte fiel a la trayectoria de la herramienta y no se han preocupado en definir dicha trayectoria basandose en la tendencia que sigue el cirujano con su herramienta, con el fin de lograr una trayectoria suave, tal y como se presenta en los procedimientos quirúrgicos. En esta tesis se presentará un algoritmo de corte enfocado a los simuladores de cirugías mínimamente invasivas con la idea de permitir al cirujano entrenarse en este tipo de procedimientos y adquirir las habilidades requeridas para llevarlas a cabo en un entorno real. Este algoritmo detecta un patrón de corte y trata de suavizar la trayectoria de corte de modo que se logre un mayor realismo durante el procedimiento.

Capítulo 2

Trabajos Relacionados

Gracias a los avances que se han presentado en el área de simulación durante la última década, en la cual cada vez se alcanzan niveles mayores de realismo conservando la interactividad, la comunidad médica y científica ha identificado a las técnicas de simulación no solo como un complemento al entrenamiento y educación de médicos cirujanos en procedimientos quirúrgicos, sino en un futuro como una herramienta esencial en la educación médica.

Hacia este fin se han encaminado muchos esfuerzos por parte de diferentes grupos de estudio a nivel internacional en los diferentes componentes tecnólogicos que hacen parte de un simulador quirúrgico, como lo son la representación gráfica, los modelos físicos, los algoritmos de corte, los métodos de retroalimentación háptica, los algoritmos para la detección de colisiones, etc. Esto con la meta de optimizar cada uno de dichos procesos y poder así lograr los requerimientos básicos de un simulador quirúrgico que son la ejecución en tiempo real, es decir la interactividad háptica y visual, y un nivel adecuado de realismo, lo cual tiene que ver con el nivel de detalle presente en las graficas e interacciones entre objetos. Ambos objetivos son difíciles de lograr al tiempo, ya que si por ejemplo se desea un nivel de realismo muy alto, el numero de cálculos a realizar aumenta y es posible que no se logre con el objetivo de ejecución en tiempo real, por tal razón hacer un balance entre ambos factores y recurrir a métodos que optimicen cada uno de estos procesos es esencial.

En cuanto a la representación gráfica básicamente hay 3 tipos, mallas triangulares, mallas tetraédricas y *mesh-free*. Las más utilizadas para la representación de tejidos biológicos en simulación quirúrgica son las mallas triangulares y las mallas tetraédricas, ya que las mallas *mesh-free* son más utilizadas en la simulación de fluidos, aunque en los últimos años varias investigaciones han encontrado que los métodos *mesh-free* ofrecen flexibilidad en la simulación de objetos típicos de los procedimientos quirúrgicos [3]. La diferencia principal entre una malla triangular y una tetraédrica es que la malla triangular define solo la superficie mientrás que la tetraédrica define el volumen, cada una de ellas presenta ciertas ventajas y desventajas, cuando se trabaja con mallas triangulares se reduce el realismo ya que en las deformaciones estas no conservan el volumen, aunque se ha venido trabajando para mejorar este aspecto, por ejemplo en [5] proponen un métodos para conservar el volumen usando mallas triangulares. Las

mallas tetraédricas son más realistas pero más costosas computacionalmente, igualmente hay muchos trabajos al respecto en los cuales se intenta mejorar el desempeño de las aplicaciones que utilizan mallas tetraédricas como por ejemplo [10].

En realidad no es mucho lo que se puede hacer ya sobre la representación para mejorar el desempeño, en general la mayoría del esfuerzo en este sentido se encamina hacia los modelos físicos y la deformación mecánica de los tejidos y órganos, los cuales se basan en algún tipo particular de representación gráfica. Existen muchos modelos como LEM (*Long Elements Method* o Metodo de Elementos Largos), FEM (*Finite Elements Method* o Metodo de Elementos Finitos) [7], [13] y [6] y *Mass-Spring*, [1], [14] y [4] entre otros, los cuales tienen una serie de ventajas y desventajas, y dependiendo de las necesidades del ambiente de simulación, es posible utilizar uno u otro. Estos modelos están ligados a un modelo de representación gráfica, cosa que se debe tener también en consideración. Básicamente hay unos modelos que son bastante realistas como el FEM pero computacionalmente costosos, por tal razón la investigación en esta área se ha encaminado al mejoramiento del desempeño de este modelo, como en [9] y [11], por otro lado hay modelos bastante eficientes y sencillos como es el caso del *Mass-Spring*, pero con algunos problemas respecto al realismo, por lo cual se han llevado a cabo también muchas investigaciones dirigidas a mejorar este aspecto [5].

En cuanto al tema de corte, hay básicamente tres aproximaciones que se han propuesto para la simulación del proceso de corte, dichas aproximaciones son Subdivisión [12], Separación [10] y Destrucción [2]. La ultima ha sido muy poco usada ya que es poco realista debido a que genera un efecto de destrucción de la materia, La primera aunque es la más realista de las tres, es bastante costosa computacionalmente. Por último la aproximación de separación es la más usada en el ámbito de simulación quirúrgica, ya que los cálculos involucrados durante el proceso son bastante sencillos, aunque tiene un problema y es que al no cortar exactamente por la trayectoria que sigue la herramienta puede ser poco realista. Sin embargo, varias propuestas han sido realizadas con el fin de definir el patrón de corte más adecuado a la trayectoria seguida por el instrumento del cirujano. (refs).

Capítulo 3

Conceptos de Simulación Quirúrgica

3.1. Introducción

En el ámbito de la simulación quirúrgica hay diversos campos de investigación que con el paso de los años, avances en las tecnologías computacionales, en las técnicas algorítmicas y gráficas utilizadas, van cambiando de orientación paulatinamente. En un principio, las investigaciones estaban concentradas en la modelación geométrica del cuerpo humano, y básicamente se enfocaban en la representación gráfica de los órganos y tejidos. En este tipo de aplicaciones había poca o ninguna interacción con el usuario, y la meta era lograr que las formas, colores y texturas fueran lo más realistas posible. Cuando ya se tenía una base sólida en la representación gráfica básica la investigación migro hacia la simulación de las interacciones físicas de estos órganos y tejidos; se plantearon entonces varios modelos para simular la deformación ocasionada por las fuerzas resultantes de las cargas externas y la oposición propia del material. En estos modelos se presenta una diferenciación de las estructuras anatómicas a partir de las propiedades físicas y mecánicas de cada una. Actualmente se siguen presentando avances en este campo aunque el trabajo investigativo se está centrando en la modelación del sistema fisiológico, logrando un nivel de detalle más alto en donde se pueden simular fenómenos como la circulación del torrente sanguíneo, e incluso el comportamiento del sistema nervioso.

De igual manera, en la actualidad los desarrollos se han enfocado en la simulación de procedimientos básicos en una cirugía como el sangrado, el corte y la sutura. Estos procedimientos son parte fundamental de la cirugía y además se encuentran presentes en cada uno de ellas, por lo que se han realizado también muchas investigaciones en este aspecto.

Todos los modelos, algoritmos y demás métodos desarrollados para un simulador de cirugías tienen que tener unas características especiales con el fin de cumplir con el requerimiento principal de este tipo de sistemas. Dicho requerimiento se refiere a lograr un alto grado de inmersión por parte del estudiante de cirugía para que este pueda adquirir las habilidades y el conocimiento deseado

en las tareas y procedimientos que práctique. Para su cumplimiento se deben analizar los siguientes aspectos técnicos:

- Ejecución en tiempo real: esto hace referencia a ciertos parámetros de la percepción humana con los cuales podemos determinar si se ejecuta en tiempo real o no. Se conoce que el ojo humano puede percibir una serie de imágenes como un movimiento continuo cuanto estas se proyectan a mas de 24 Hz, pero se toma un criterio de 30Hz para poder decidir si un simulador de cirugías se ejecuta en tiempo real o no, por lo que se requiere que cada ciclo del programa se ejecute en un tiempo máximo de 30 milisegundos. Además si se incorpora un elemento de retroalimentación háptica (Pie de página explicando este término) se requiere un tiempo mucho menor ya que el tacto y la retroalimentación de fuerza en el ser humano trabaja a una frecuencia mucho más rápida que la vista. Esta frecuencia es alrededor de 300 a 1000 Hz.

- Realismo: una de las metas de un simulador quirúrgico es lograr en el usuario la sensación de estar manipulando órganos reales. Para esto se requiere un alto grado de detalle visual, por lo que los modelos de representación gráfica juegan un papel muy importante, también es muy importante que el modelo de deformación modele adecuadamente el comportamiento y las características físicas de los órganos, haciendo una salvedad, que en simulación quirúrgica para el entrenamiento de cirujanos prima el realismo que la precisión de la simulación. Hay que tener en cuenta que para poder lograr un alto grado de realismo se requiere realizar un gran número de cálculos, debido a que los modelos requeridos son computacionalmente costosos y las mallas utilizadas tienen un alto número de primitivas, esto hace que se ejecute en un mayor tiempo, lo que es contraproducente en el logro de la ejecución en tiempo real. Esto plantea la necesidad de un balance entre el realismo y la ejecución en tiempo real.

Un simulador quirúrgico está compuesto por varios componentes principales, por una parte está el modelo de representación grafica, que es el encargado de definir las estructuras mediante las cuales se almacenan y representan los objetos visualmente; los modelos de simulación física o modelos de deformación son los que se encargan de simular las deformaciones y fuerzas de interacción de los objetos deformables; los algoritmos de detección de colisiones son los encargados de notificar cuando se presenta una colisión y de definir cuales son los elementos de contacto entre dos objetos; los algoritmos de retroalimentación háptica se encargan de calcular las fuerzas de retroalimentación que los dispositivos hápticos deben entregarle al cirujano. Finalmente estan los algoritmos encargados de los procesos de corte, sutura y sangrado. Debido a que este documento se concentra en los procedimientos de corte, se hará mención de los componentes que interactúan directamente con este algoritmo, modelos de representación grafica, modelos de deformación, algoritmos de detección de colisiones y algoritmos de corte. Este último es el algoritmo desarrollado en esta trabajo.

3.2. Modelos de Representación Geométrica

Hay varias formas de representar visualmente un mismo objeto, en el caso de objetos rígidos no habría diferencia aparente entre ellos ya que visualmente serian iguales, y al no haber deformaciones, el efecto final sería el mismo, pero cuando queremos aplicar un modelo de deformación el cual cambia la forma del objeto dependiendo de una serie de interacciones entre fuerzas externas e internas, o cuando tenemos un corte en donde se cambia la topología, algunos modelos de representación gráfica pueden ser más convenientes que otros. Se puede decir que hay 2 tipos básicos de modelos, los modelos volumétricos y los modelos superficiales. En los modelos volumétricos se tiene una estructura en la cual los vértices y bordes u otro elemento que componen la malla no se encuentran solamente en su superficie sino también al interior, mientrás que en los modelos superficiales todos los elementos se encuentran en la superficie del objeto. Hay entonces varios modelos de representación gráfica, que pueden tomar uno de los dos paradigmas mencionados anteriormente. A continuación se hará referencia a los 3 más utilizados.

3.2.1. Mallas Triángulares

Las mallas triangulares son un tipo de representación gráfica basado en el paradigma del modelo superficial. Una malla triangular, como su nombre lo indica, está formada por una serie de triángulos unidos entre sí en su superficie. Cada triangulo está compuesto por 3 vértices y 3 bordes, los cuales comparten con sus triángulos vecinos. En la figura 3.1 se muestra la composición de un triángulo.

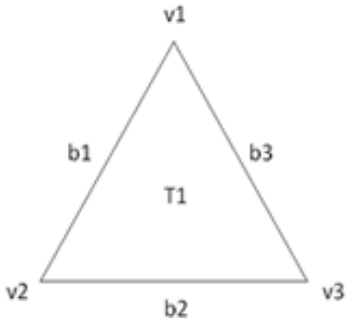

Figura 3.1: Triángulo: esta compuesto por tres vértices y tres bordes

Como características generales tenemos que este tipo de mallas utilizan un número relativamente bajo de primitivas para representar un objeto, ya que al ser superficial solo modela la superficie y no tiene primitivas en su interior, este aspecto es bastante benéfico ya que entre menos primitivas compongan la malla, menor número de cálculos deben realizarse y por consiguiente, menor tiempo de ejecución entre cada frame es necesario, esto resulta en una buena interactividad con el usuario. Por otra parte, las mallas triangulares, cuando son sometidas a deformaciones pueden no seguir el principio físico de conservación del volumen

del objeto, esto es debido a que no tiene puntos en su interior que cubran su volumen, por lo que la superficie se puede mover libremente, esto hace que sea poco realista el proceso de deformación usando una malla triangular como modelo de deformación, aunque existen técnicas que solucionan este problema. En la figura 3.2 se puede ver un ejemplo de este tipo de mallas.

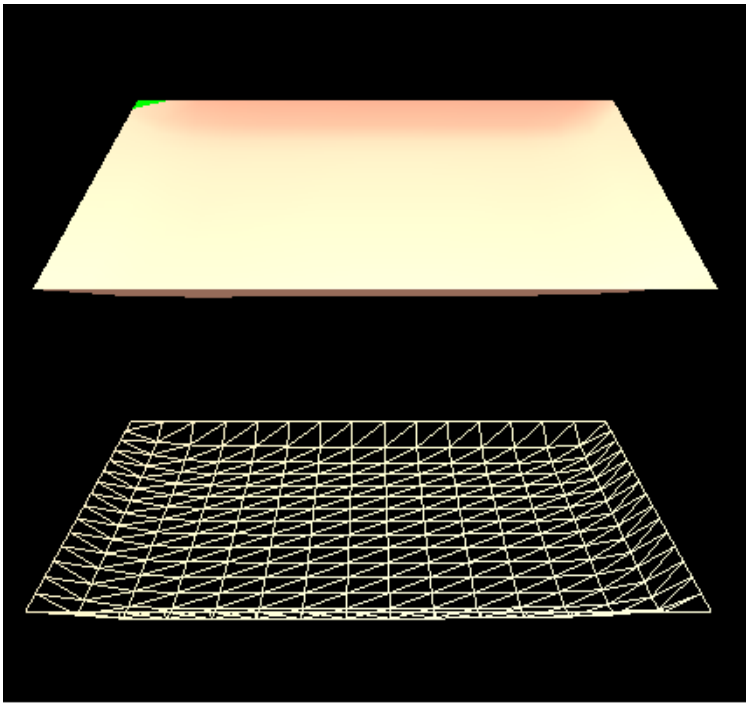

Figura 3.2: Ejemplo de una malla triangular, a la izquierda con visualización *smooth*, a la derecha con visualización *wireframe*

3.2.2. Mallas Tetraédricas

Este tipo de representación grafica está basado en el modelo volumétrico, el cual se compone de una serie de puntos tanto en la superficie como en el interior. La unidad básica de una malla tetraédrica, como su nombre lo indica, es el tetraedro (ver figura 3.3), estos tetraedros comparten sus vértices y bordes con los demás tetraedros formando un solo volumen.

Debido a que los puntos no solo se encuentran en la superficie sino también en su interior, las mallas tetraédricas tienen una mayor cantidad de primitivas cuando se le compara con una malla triangular, lo que hace que el número de cálculos que se debe hacer sobre la malla aumente también, sin embargo este tipo de mallas al ser volumétricas permiten realizar una conservación del volumen mucho más sencilla y adicionalmente permiten simular propiedades mecánicas de los objetos como su anisotropía. en la imagen 3.4 se puede observar un ejem-

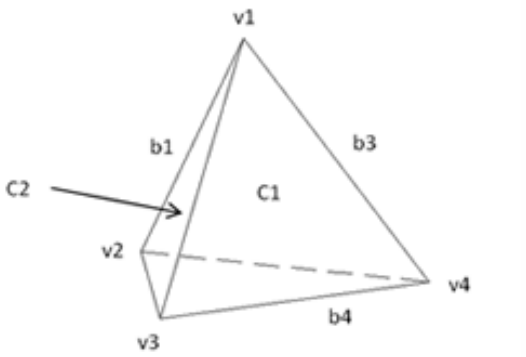

Figura 3.3: Tetraedro: esta compuesto por cuatro vértices, seis bordes y cuatro caras

plo de este tipo de malla.

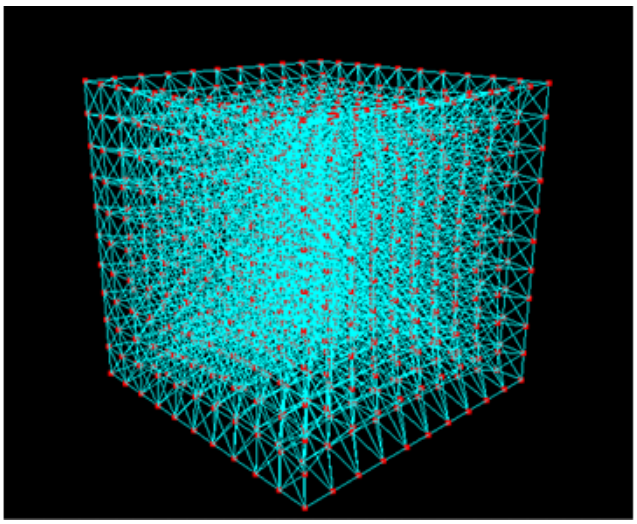

Figura 3.4: Ejemplo de una malla tetraédrica

3.2.3. *Mesh-Free*

Este tipo de representación no tiene una malla definida, es un modelo muy diferente y un poco más complicado a los dos mencionados anteriormente. Básicamente lo que define el modelo es una serie de puntos tanto internos como externos que componen al objeto, pero en este caso no hay borde que une a los puntos sino que cada punto tiene un radio de influencia, dentro del

cual ejerce una fuerza a otro punto que depende de la cercanía entre ambos. El funcionamiento es similar a la interacción entre moléculas que conforman los diferentes materiales mediante los diferentes enlaces. Esta topología es ideal para representar masas u objetos que presentan cambios bruscos en su forma, también es ideal para masas sin forma definida, muy usada en la simulación de gases y fluidos, como por ejemplo agua, sangre, humo o viento. en la figura 3.5 se puede ver un ejemplo de este tipo de malla.

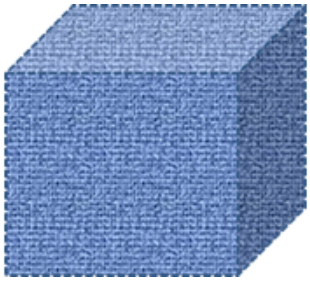

Figura 3.5: Cubo Mesh-Free

3.3. Modelos de Deformación

El modelo de deformación es aquel que nos define el comportamiento de la forma del objeto cuando es sometido a una serie de fuerzas externas. El modelo de deformación está muy ligado con el modelo de representación ya que este brinda los elementos de trabajo, vértices, bordes, triángulos o tetraedros. De igual forma que en los modelos de representación gráfica, hay varios modelos de deformación que se han propuesto en las últimas décadas, a continuación se hará mención de los dos más utilizados en simulación quirúrgica y se plateará la formulación matemática del que es usado para la modelación física de los objetos en este trabajo.

3.3.1. *Mass-Spring*

El modelo *Mass-Spring* es un modelo sencillo que ofrece grandes ventajas para la simulación de deformaciones mecánicas en general y que en el ámbito de la cirugía laparoscópica resulta muy útil y muy usado, debido a su simplicidad, ya que no requiere de un gran número de cálculos para su simulación. Por otra parte es muy fácil de implementar y permite cambios en la topología de la malla geométrica.

Los modelos *Mass-Spring* representan un objeto deformable por medio de una malla tridimensional M, dicha malla esta compuesta por nodos $N_i (i = 1, 2, ..., n)$ con propiedades asociadas como la masa m_i, conectados entre sí por conexiones L_{ij} donde i y j son los nodos conectados. Estas conexiones actuan como resortes y amortiguadores, por lo que tienen una constante de rigidez k_{ij} y un coeficiente de viscosidad C_{ij} asociados. En este modelo cada nodo de la malla es considerado

como una masa puntual y las aristas o bordes se consideran como resortes amortiguados con unas constantes definidas. El valor de dichas constantes depende de las propiedades mecánicas del objeto simulado. Las fuerzas que interactúan en cada nodo se rigen por las leyes de la física y dependen de la masa que tenga cada punto así como de la longitud y constante elástica asociada a cada resorte. El papel del amortiguador es muy importante porque evita que el sistema, luego de una acción externa, se quede inmerso en un movimiento armónico simple y, que se deforme y restablezca indefinidamente o hasta que se aplique otra fuerza. En cada iteración se calculan las fuerzas resultantes y las respectivas deformaciones. El modelo *Mass-Spring* es muy versátil ya que puede ser aplicado tanto a representaciones que utilizan mallas triangulares como a las que utilizan mallas tetraédricas, pero entre los aspectos negativos esta que aunque trata de ser una aproximación de la realidad, no simula el fenómeno físico de deformación con exactitud. Adicionalmente hay que tener control de las variables como constantes de los resortes y las fuerzas de amortiguación para evitar que el sistema pueda llegar a ser inestable.

La formulación matemática del modelo es la siguiente. La fuerza interna entre dos nodos N_i y N_j esta determinada por la ley de Hooke:

$$\vec{F_{ij}} = -k_{ij}(\Delta_{ij}\vec{u_{ij}}) \tag{3.1}$$

donde Δ_{ij} es la diferencia entre la longitud actual de la conexión menos la longitud de equilibrio, y $\vec{u_{ij}}$ es el vector unitario que apunta desde el nodo N_i hacia el nodo N_j. La rígidez k_{ij} puede ser constante o una función de Δ_{ij}. En cualquier instante de tiempo la deformación de la malla M puede ser descrita por el movimiento de cada uno de sus nodos, donde el movimiento de un nodo N_i esta dado por la siguiente ecuación:

$$m_i\vec{a_i} + c_i\vec{v_i} + \sum_{j\in\sigma(i)} \vec{F_{ij}}(\vec{x_i}, \vec{x_j}) = m_i g + \vec{F_i^{ext}} \tag{3.2}$$

En donde $\vec{x_i}$ es el vector coordenado del nodo N_i, $\vec{v_i}$ y $\vec{a_i}$ son los vectores de velocidad y aceleración del nodo i respectivamente , $m_i g$ es la fuerza de gravedad y $\vec{F_i^{ext}}$ es la fuerza total externa aplicada al nodo N_i, mientrás que $\sigma(i)$ es el conjunto de índices de los nodos adyacentes al nodo N_i en la malla M. En la figura 3.6 se puede observar una descripción gráfica de cada uno de los componentes del modelo.

3.3.2. FEM (*Finite Elements Method*) o Método de Elementos Finitos

El método de elementos finitos (FEM por sus siglas en ingles) consiste básicamente en la transformación de un cuerpo de naturaleza continua en un modelo discreto aproximado. FEM utiliza técnicas numéricas que se usan para encontrar soluciones aproximadas a ecuaciones diferenciales parciales así como a ecuaciones integrales. Para simular las deformaciones, FEM se basa en los principios de la mecánica, y analiza la energía potencial almacenada por el objeto debido a la deformación y a partir de esto se busca un punto de equilibrio en el cual

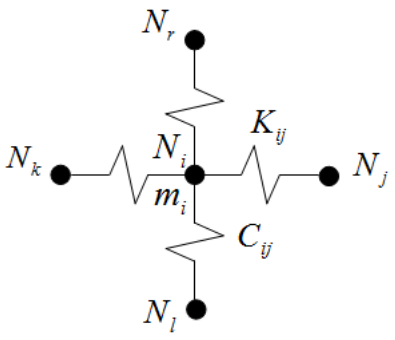

Figura 3.6: Esquema del modelo *Mass-Spring*, tomado de [8].

esta energía almacenada sea mínima.

Los elementos de la malla contienen la información de las propiedades elásticas y mecánicas del material las cuales dicen en qué forma reaccionará ante diferentes condiciones de carga. Para implementar el método de elementos finitos FEM es necesario el uso de una malla tetraédrica. El método de elementos finitos es muy realista aunque costoso computacionalmente, y ha sido la base para el desarrollo de otros modelos de deformación como LEM (Long Elements Method) (Referencia).

3.4. Corte

El proceso de corte en un simulador quirúrgico es definido como la división intencional de la malla mediante una interacción en la cual la persona que manipula los instrumentos quirúrgicos ejerce una fuerza o contacto sobre la malla ocasionando que las primitivas que la componen sean divididas.

Hay diferentes formas de llevar a cabo esta operación. Dada una trayectoria formada por una secuencia de puntos en el espacio que son generados por la colisión entre la herramienta de corte y la malla que representa el objeto, se busca una trayectoria de corte de la malla que dependiendo del método que se utilice puede variar significativamente, notemos que no es necesariamente la misma trayectoria dada por la herramienta, esto depende de cuales sean las intenciones y las necesidades de rendimiento y realismo de la simulación de corte.

En general en la literatura se han propuesto tres aproximaciones sobre las cuales se basa todo el trabajo investigativo en cuestión del proceso de simulación de corte. Estas aproximaciones básicas son destrucción, sub-división, y separación.

3.4.1. Aproximación de Destrucción

Esta aproximación lo que hace es remover los elementos que colisionan con la herramienta. Esta operación reduce el número de primitivas que contiene la malla, es decir que este algoritmo de corte no afecta el desempeño del simulador quirúrgico, ya que no crea primitivas nuevas al modelo geométrico y físico del objeto. Sin embargo, esta aproximación presenta un problema considerable, y es que a medida que va destruyendo primitivas el volumen del objeto que está siendo cortado se va reduciendo, además el modelo debe tener un número elevado de primitivas con el fin de no hacer tan visible al usuario el fenómeno de desaparición de primitivas, el cual afecta el realismo. Por otra parte, esta aproximación resulta bastante útil cuando se desea simular un proceso donde se destruya tejido o materia, como por ejemplo la electrocauterización en simulación quirúrgica. En la figura 3.7 se puede observar un ejemplo gráfico del concepto aplicado en la aproximación de destrucción.

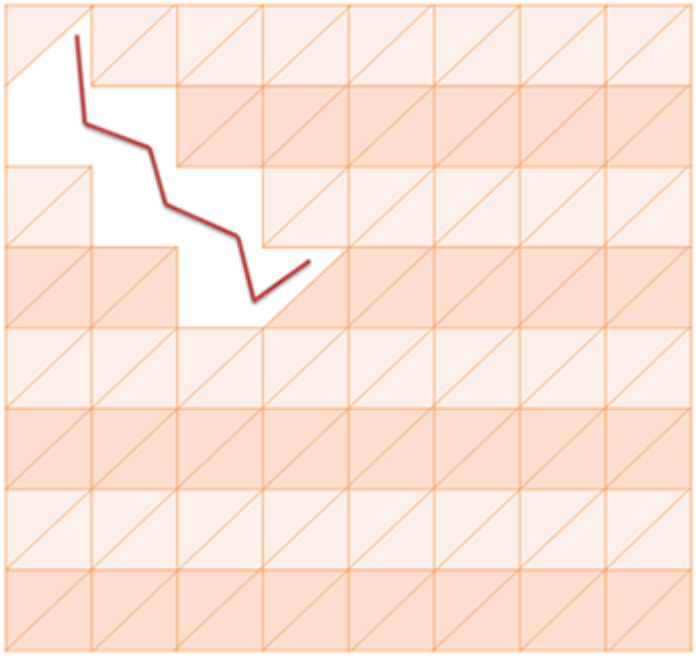

Figura 3.7: Ejemplo de un corte por destrucción, la linea roja muestra la trayectoria que describe el instrumento de corte, se puede observar que los triángulos en la trayectoria son eliminados.

3.4.2. Aproximación de Sub-División

Esta aproximación se basa en la idea de subdividir ya sean los triángulos o tetraedros que componen la malla, a partir de la trayectoria proyectada por la colisión entre la herramienta quirúrgica y el órgano o tejido. Dado que casi nunca esta trayectoria va a coincidir con una trayectoria exacta construida usando los vértices y bordes de la malla, es necesario entonces adicionar a la aproximación

algoritmos especiales encargados de seleccionar las primitivas que serán subdivididas tomando ciertos criterios y teniendo en cuenta siempre la trayectoria, además requiere de algoritmos de remallado para cerrar la primitivas que han sido divididas. Tenemos entonces con esta aproximación que con cada corte se aumenta considerablemente el número de primitivas y que entonces debido a un mayor número de primitivas el número de operaciones que se tienen que realizar es mayor afectando de esta manera el desempeño en tiempo real del simulador quirúrgico. La ventaja que tiene esta aproximación es que el corte se realiza exactamente por la trayectoria descrita por el usuario, y no hay destrucción de materia, de modo que es bastante realista, aunque si se corta demasiadas veces por la misma zona se pueden crear triángulos o tetraedros muy pequeños que pueden generar inestabilidades en el sistema cuando se calculan las fuerzas internas y externas del modelo.

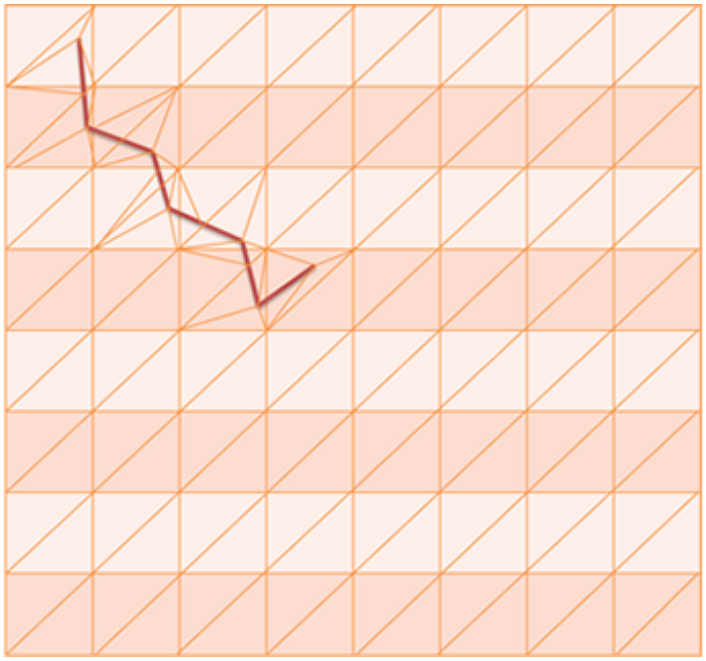

Figura 3.8: Ejemplo de un corte por Sub-división, la línea roja muestra la trayectoria que describe el instrumento de corte, el corte se produce en esta misma trayectoria y se puede observar el incremento resultante en el numero de triángulos.

3.4.3. Aproximación de Separación

Este método consiste en encontrar una trayectoria muy similar a la descrita por la herramienta de corte pero siguiendo los vértices y bordes de la malla, de modo que no se presente la necesidad de crear nuevos triángulos ni tetraedros. El problema presente en esta aproximación se refiere a la definición de esta nueva trayectoria. En este tema en específico se han llevado a cabo mu-

chos trabajos (REFS??). Dicha trayectoria se puede escoger dependiendo de las necesidades impuestas por la apliación, por ejemplo se puede elegir un criterio que siga un patrón, en donde el algoritmo es capaz de definir una tendencia en el movimiento del cirujano y ajustar la trayectoria a dicha tendencia, o uno que siga el movimiento de la herramienta, que es simplemente escoger la trayectoria mas parecida a la trayectoria descrita por el cirujano. Este método es bastante eficiente ya que no se crean gran número de primitivas (solo bordes y vértices, más no triángulos ni tetraedros) y el realismo del proceso de corte se podría considerar en un nivel intermedio ubicado entre la aproximación de destrucción y subdivisión. Además esta aproximación de corte cumple en todo momento con el principio básico de conservación de la materia, obteniendo de esta manera dos factores importantes en la simulación los cuales son la ejecución en tiempo real y el realismo.

Por otra parte, esta aproximación soluciona los problemas que se presentaban en los métodos de destrucción y subdivisión. En el primero había destrucción de la materia (destrucción de primitivas geométricas) y en el segundo se presentaba un cuantioso incremento en el numero de primitivas a medida que se lleva a cabo el procedimiento de corte.

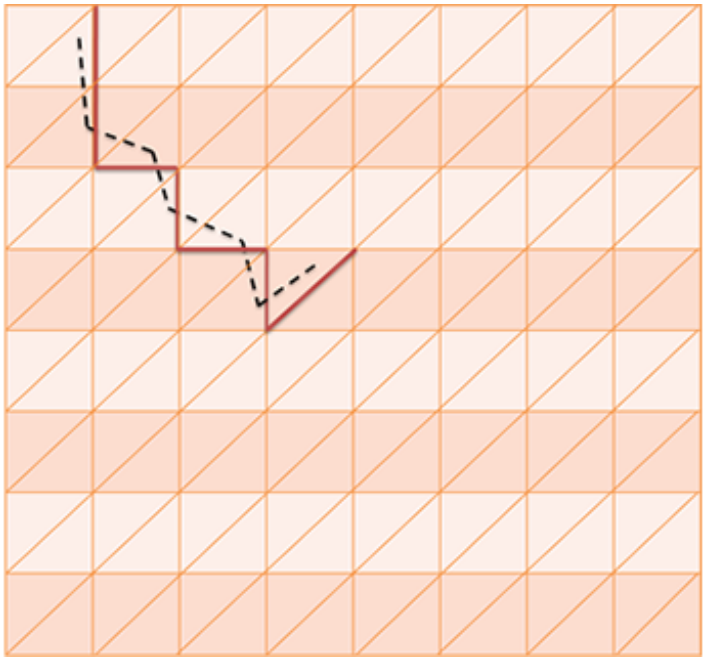

Figura 3.9: Ejemplo de un corte por Separación, la línea negra punteada muestra la trayectoria que describe el instrumento de corte y la linea roja muestra la trayectoria a ser cortada en la malla, el corte no se produce en la misma trayectoria del instrumento sino en una similar. Además se puede observar que no hay incremento en el número de triángulos.

3.5. Detección de Colisiones

El proceso de detección de colisiones es el encargado de analizar en cada momento la posición de todas las estructuras y determinar si hay o no algún tipo de contacto entre dos objetos y en caso de que así sea, establecer cuales elementos están en colisión.

Este proceso se lleva a cabo en 3 fases, fase general, fase específica y fase exacta. Esto con el fin de optimizar el funcionamiento del proceso. En la fase general se determinan los pares de objetos que podrían estar colisionando, la idea en esta fase es descartar elementos que muy probablemente no están colisionando para que en otra fase se analicen solamente los que tienen probabilidad de estar en colisión. Esto se puede realizar mediante dos aproximaciones, una es descomposición por voxeles, en donde el espacio virtual es dividido en un conjunto uniforme de celdas. La otra aproximación usa arboles de partición binaria, básicamente lo que se hace en los algoritmos que siguen esta aproximación es definir aleatoriamente un elemento como nodo raíz para posteriormente definir un plano para que en una hoja se encuentren los elementos por encima de ese plano y en otra hoja los elementos que se encuentran por debajo.

En la fase específica se analizan en detalle las posibles áreas de colisión encontradas en la fase anterior, luego de lo cual se definen las regiones específicas donde hay colisión, en caso de haber alguna. Para hacer esto hay diferentes aproximaciones, en el caso del simulador EAFIT-CES se utilizó una jerarquía de volúmenes límite, esto es, un árbol que representa datos de diferentes volúmenes limite que conforman la malla del objeto. Un volumen límite es un objeto simple tipo cubo o esfera que encierra completamente la geometría de un objeto (o en este caso de una parte del objeto) y que sirve para hallar las intersecciones entre ellos de una forma más fácil.

Ya en la fase exacta se definen las entidades particulares que están en colisión. Para esto se usa un algoritmo llamado test de intercepción, que detecta si dos primitivas de dos objetos diferentes se intersectan entre sí.

Capítulo 4

Desarrollo Modelo Vesícula y Algoritmo de Corte

4.1. Modelo de la Vesicula Biliar

4.1.1. Introducción

El componente visual, como se había mencionado anteriormente es una parte muy importante de la simulación, es el que le da al cirujano la sensación de que está llevando a cabo un procedimiento real, por esta razón es un factor clave que la modelación de las estructuras anatómicas sean, en la medida de lo posible, fieles a la realidad y así lograr que el cirujano alcance un alto nivel de inmersión en el ambiente virtual de entrenamiento.

Como parte de la simulación de un procedimiento de colecistectomía laparoscópica, el cual consiste en la remoción de la vesícula biliar la cual se encuentra situada en la cara gastrointestinal del hígado, se hace necesario la modelación tridimensional de estos órganos particularmente. A continuación se hará mención del proceso de modelación geométrica de la vesícula biliar.

4.1.2. Desarrollo de un Modelo Tridimensional de la Vesícula Biliar

Para este fin se siguieron dos metodologías ya que la implementada al principio no dio los resultados esperados por lo cual se tomo la alternativa de seguir otra metodología con la cual se obtuvieron estos resultados.

Metodología Proyecto Humano Visible

Inicialmente se decidió seguir el mismo método que se venía usando en el simulador EAFIT-CES (Referencia de mi tésis o uno de los articulos) para la modelación tridimensional de los órganos , este método consiste en la extracción de contornos a partir de una serie de imágenes que se disponían y que fueron

tomadas del proyecto Humano Visible este proyecto fue realizado por la National Library of Medicine.

Estas imágenes son fotos transversales del cuerpo humano tomadas cada 0.3mm de distancia. En cada una de dichas imágenes el usuario toma una serie de puntos, los cuales son unidos mediante la implementación de un algoritmo de interpolación spline con el fin de suavizar la forma del contorno descrito por los puntos seleccionados por el usuario. Una interpolación spline es básicamente la generación de unos polinomios simples que aproximan un grupo de puntos de una manera suave. Posteriormente, uniendo los contornos extraídos de las imágenes en las que se encontraba la vesícula, se utilizó aplicó un algoritmo de relleno de área y un algoritmo de reconstrucción tridimensional llamado marching cubes. En la figura 4.1 se puede observar el resultado de la operación anterior.

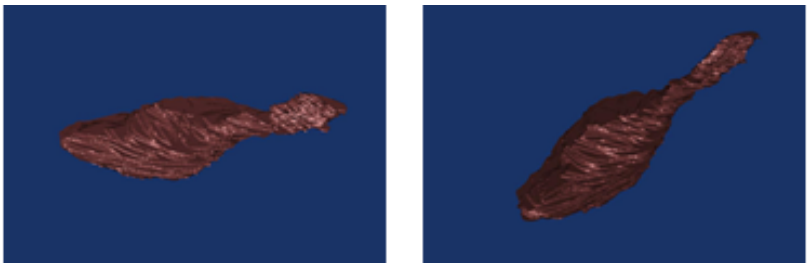

Figura 4.1: Imagen de la vesicula luego de la reconstruccion tridimensional: se pueden ver todas las irregularidades que presenta

Generalmente después de este proceso la malla que se obtiene presenta irregularidades así como una gran cantidad de primitivas, por esta razón el proceso finaliza aplicando un algoritmo de decimación y suavizado, y de esta manera obtieniendo la malla final.

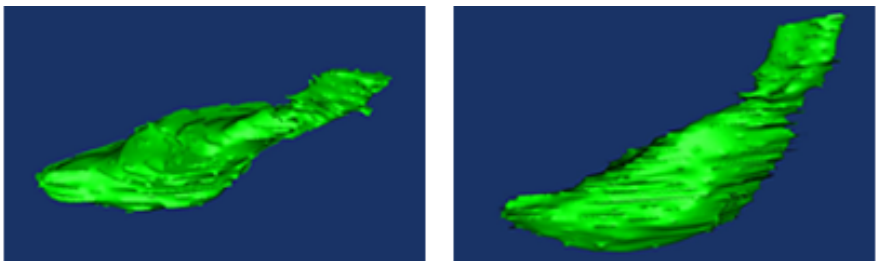

Figura 4.2: Modelo de la vesicula luego de aplicado el algoritmo de decimacion y de suavización: se puede ver que todavía presenta muchas irregularidades.

Como la figura obtenida presentaba todavía muchas irregularidades, ver 4.2, se intento reducirlas mediante un programa de modelación llamado Blender, pero aun sin obtener los resultados esperados, como se ve en 4.3.

Figura 4.3: Modelo de la Vesicula luego de aplicado el algoritmo de decimacion y de suavización, y luego de reducir las irregularidades con Blender: se observa que el numero de irregularidades es considerable y casi imposibles de remover manualmente.

En el caso de la vesícula biliar este proceso no fue exitoso debido a que la malla obtenida anteriormente presentaba demasiadas irregularidades introducidas por varios causantes de error, entre ellas la dificultad de seleccionar los puntos del contorno ya que el tamaño de la vesícula es relativamente pequeño, y también porque alrededor de la vesícula se encuentran otros órganos y tejidos muy cercanos y por lo tanto en ocasiones era difícil la diferenciación del borde.

Por estos motivos se decidió utilizar un método alterno para la generación del modelo 3D de la vesícula biliar. Esta metodología es descrita detalladamente en la siguiente sección. Metodología ´´Modelación 3D usando Blender".

La idea de esta metodología es la modelación tridimensional de la estructura anatómica sin tener en cuenta datos reales de su forma, es decir imágenes de las estructuras, ya sean fotografías o imágenes médicas como tomografías o resonancias magnéticas, sin embargo teniendo en cuenta los siguientes factores:

- Análisis del objeto usando varias imágenes en diferentes perspectivas.
- Análisis de la forma geométrica del órgano.

A partir de estos análisis se noto que no es una geometría demasiado compleja ya que la forma se asemeja a la de una pera, de esta manera se recurrió a un software de modelación tridimensional para crear la malla.

El software de modelación tridimensional que se escogió fue Blender, el cual es un software multiplataforma dedicado a la modelación, diseño, animación y creación de gráficos tridimensional de muy fácil acceso ya que es *open source* (software libre).

Metodología Modelación 3D usando Blender

Básicamente el procedimiento que se siguió fue crear una esfera a la cual se le elimino una sección, y posteriormente mediante extrusiones se fue creando el resto de la forma, y en cada una de ellas se aumentaba el diámetro de la circunferencia que se estaba extruyendo, para al final ir decreciendo hasta cerrar el objeto.

Posteriormente, se desplazaron los puntos con una inclinación para darle la forma curvada que le da la unión de la vesícula con los tejidos contiguos. También se utilizo un algoritmo de desplazamiento aleatorio de puntos en donde cada punto se desplazaba aleatoriamente a una nueva posición en una escala pequeña, esto con el fin de reducir un poco el aspecto esférico regular que presentaba el objeto y darle un aspecto más natural. Finalmente se utilizo un algoritmo de suavización para eliminar los bordes bruscos y puntas, luego del cual, cuando el objeto ya tenía la forma deseada se siguió con el proceso de adición de color y texturas. En la figura 4.4 se muestra el proceso de construcción de la vesicula en varias etapas.

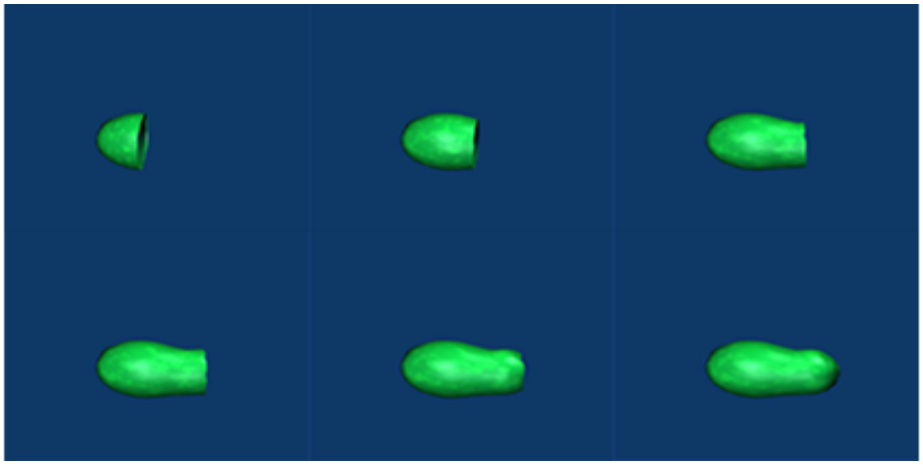

Figura 4.4: Construccion de la vesicula en Blender: en la figura se observa como se fue creando el modelo de la vesicula a partir de extrusiones con perfil circular al cual se le variaba el diámetro con el fin de obtener la forma de pera.

Igualmente se diseño el conducto biliar común o conducto colédoco, el cual es un conducto de la vía biliar originado de la fusión del conducto hepático común con el conducto cístico y que desemboca en la segunda porción del duodeno, en este conducto, la bilis puede ser dirigida a la vesícula biliar para su almacenamiento o bien puede entrar en la porción superior del conducto biliar común y continuar hasta drenar en el duodeno. Para diseñar este conducto, primero se unió la vesicula anteriormente diseñada al modelo del hígado de tal forma que estuviese posicionado en las cavidades indicadas, luego de lo cual se utilizo una serie de extrusiones con perfil circular para ir construyendo todo el conducto. En la figura 4.5 se puede observar la vesicula ya terminada junto con el conducto colédoco y el modelo del hígado.

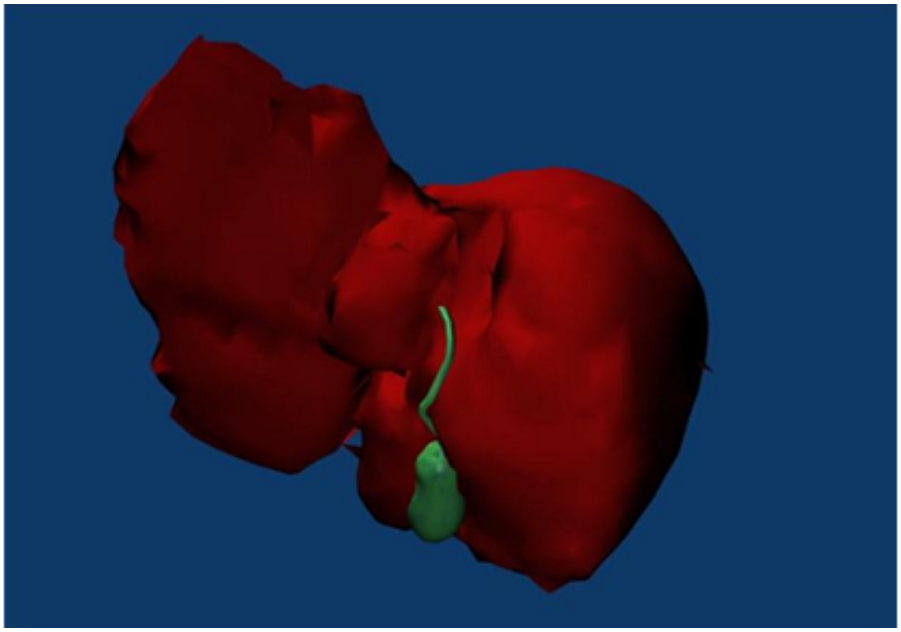

Figura 4.5: Higado y Vesicula: en esta imagen se puede observar el modelo final de la vesicula unida al higado mediante el conducto comun coledoco.

4.2. Análisis de la Estructura del Simulador

Antes de comenzar con el diseño del algoritmo de corte, es necesario conocer muy bien la estructura de datos del simulador, con el fin de adaptar el algoritmo a las necesidades. Antes de profundizar en la estructura como tal se debe saber en qué modelo de representación está basado, así como el modelo físico o de deformación que fue implementado. Actualmente el simulador quirúrgico EAFIT-CES utiliza un modelo de representación superficial basado en mallas triangulares, y como modelo físico utiliza mass-spring. Esta configuración permite una muy buena interactividad ya que ambos modelos son muy ligeros, flexibles, requieren de pocas primitivas y pocos cálculos durante la simulación, además permiten un aceptable grado de realismo.

El proceso de corte únicamente requiere interactuar con el algoritmo de detección de colisiones y con las superficies deformables asociadas a la sesión en la que se esté trabajando. El algoritmo de colisiones determina en cada instante si hay colisión o no entre dos objetos (en los ambientes de simulación quirúrgica habitualmente las colisiones se dan entre un instrumento y una estructura anatómica), en caso de que así sea, determina el triángulo en donde se presenta la colisión para ambos objetos, así como el punto exacto donde esta ocurre.

En cuanto a la superficie deformable, esta está compuesta de varias estructuras

así como de un modelo físico asociado. En general, una superficie deformable está compuesta de una serie de puntos que conectados entre sí forman bordes y triángulos y estos agrupados en uno solo forman la superficie geométrica del objeto. Un Vértice está compuesto por dos puntos, un punto que contiene la información de la posición anterior (frame anterior) y un punto que contiene la información de la posición actual (frame actual), esto es usado más que todo para calcular las deformaciones. No todos los vértices están unidos entre sí, en realidad un vértice solo está unido con algunos otros pocos vértices, mínimo dos, pero normalmente son entre tres y siete. Estos vértices están unidos por medio de un borde, y están organizados de tal forma que tres vértices unidos entre sí por tres bordes forman un triángulo, de tal manera que entre dos triángulos adyacentes comparten un borde y dos vértices. Esto se puede ver mas claramente en el grafico 4.6.

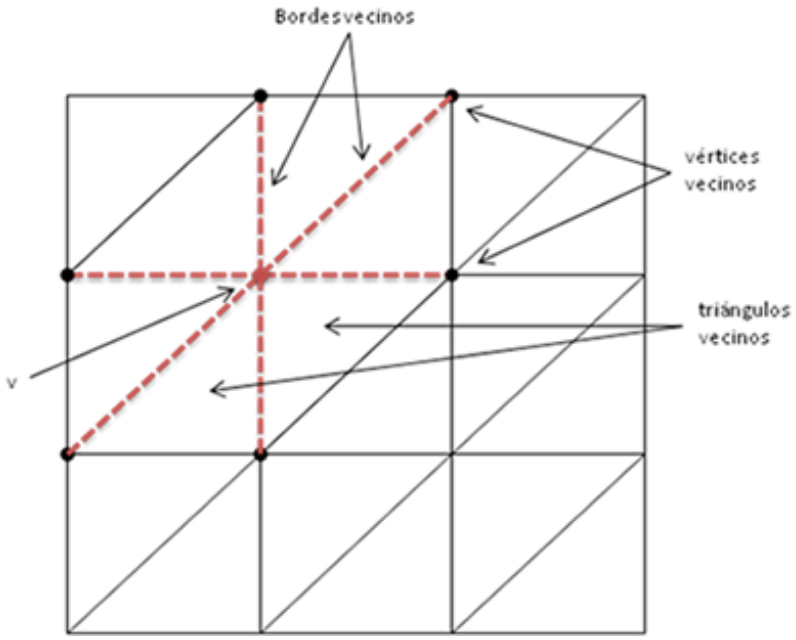

Figura 4.6: En este grafico se observa un vértice v y sus bordes vecinos (punteados en rojo), sus vértices vecinos (puntos negros) y triangulos vecinos (triangulos a los cuales pertenece)

En este orden de ideas, con el fin de representar una superficie es necesario no solo conocer todos los vértices, bordes y triángulos que componen dicha superficie, sino también conocer los vecinos de cada uno, un vértice debe saber con qué vértices está conectado, mediante que bordes, y a cuales triángulos pertenece. Cada primitiva debe tener esta información almacenada debido a los requerimientos que presentan los algoritmos de modelación física y visualización ya implementados en el simulador quirúrgico. En la figura 4.7 se puede observar la estructura de un objeto deformable.

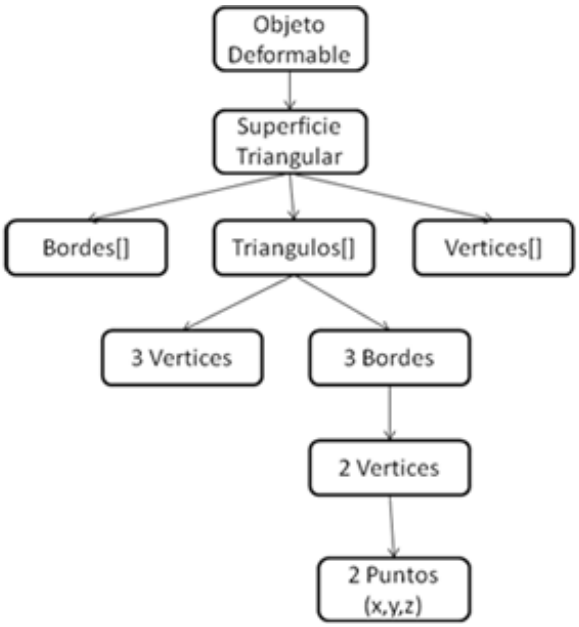

Figura 4.7: Estructura de un objeto deformable, compuesta por una superficie triangular que a su vez esta compuesta de un arreglo de bordes, un arreglo de triangulos y uno de vértices.

4.3. Corte

4.3.1. Introduccion

El corte es un procedimiento importante en cirugía, y específicamente en cirugía laparoscópica toma un papel fundamental por dos razones. La primera es el nivel de dificultad presente en este tipo de procedimientos, el cual es mucho mayor cuando se le compara con un procedimiento de cirugía abierta. La segunda se refiere a su necesidad en el ámbito quirúrgico, ya que es una tarea que se lleva a cabo en todas las cirugías, por tal razón es necesario que el cirujano adquiera las habilidades y destrezas requeridas para realizar esta tarea.

Teniendo en cuenta esto, se decidió diseñar un algoritmo que simule las operaciones de corte y de esta manera permita al cirujano entrenar este tipo de tareas en el ambiente quirúrgico virtual. Primero que todo se analizaron las necesidades del simulador con respecto al algoritmo, por lo cual se diseño un algoritmo basado en los siguientes requerimientos básicos:

- Rapidez: se requiere que el algoritmo se ejecute de una manera rápida con el fin de que el corte se pueda llevar a cabo en tiempo real de manera que el simulador mantenga su alto grado de interactividad.
- Realismo: es fundamental que el cirujano mientras está realizando un

corte, sienta que los tejidos se rasgan en la misma forma que lo haría un tejido real, que sienta también que todo lo que sucede es reacción directa a sus movimientos y que se mantengan las propiedades físicas luego de un corte.

4.3.2. Algoritmo de Corte

Para diseñar el algoritmo, lo primero que se definió fue el esquema general base a partir del cual surgiría la implementación, es decir si se iban a subdividir, destruir o separar los triángulos que forman parte del objeto. Analizando los dos requerimientos principales, que el algoritmo sea rápido y realista, y teniendo en cuenta las características de cada aproximación se decidió que el algoritmo debía seguir el esquema de separación. Aunque el esquema de subdivisión es bastante realista, genera nuevas primitivas que hacen que el algoritmo sea más costoso computacionalmente mientras más cortes se ejecuten, y el esquema de destrucción, aunque es bastante rápido y el numero de primitivas decrece con cada corte, este es bastante irrealista ya que genera un efecto de desaparición de la materia que no concuerda con las leyes básicas de la física de un procedimiento de corte con bisturí.

En la aproximación de corte por separación lo más importante es definir los criterios mediante los cuales el algoritmo decidirá cuál es la trayectoria que más se aproxima a la trayectoria proyectada por la herramienta de corte manejada por el cirujano. Para esto hay varias opciones, se puede seleccionar el borde más cercano a cada punto de interacción y de esa forma ir armando la trayectoria, también es posible seleccionar los vértices más cercanos, o los bordes con pendiente similar a la pendiente de la trayectoria seguida por el instrumento quirúrgico.

Para seleccionar la trayectoria más ajustada se comparan los vectores directores de cada recta que pasa por cada pareja de puntos que pertenece a la trayectoria descrita por la herramienta con los vectores directores de las rectas que pasan por cada uno de los bordes, a partir de esta comparación se escoge el borde cuyo vector director se asemeje mas a la trayectoria de corte, la cual es la trayectoria del instrumento quirúrgico manipulada por el cirujano.

Descripcion del Algoritmo

Cuando se detecta la primera colisión, lo primero que se hace es hallar el vértice mas cercano a este y esperar un ciclo o hasta que el punto de colisión de la herramienta cambie. Este vértice es llamado vértice pivote, y va a cambiar a lo largo de toda la trayectoria. Luego de que el punto cambia, se halla un vector con la misma dirección que la recta que pasa por estos dos puntos y se calcula su vector director, asimismo, se calculan los vectores directores de las rectas que pasan por los bordes que están unidos al vértice pivote, y se calcula el ángulo que existe entre este vector y el vector de la trayectoria, para escoger el borde con menor ángulo. Luego de esto se debe recalcular el vértice pivote el cual es el otro vértice perteneciente al borde seleccionado. Este proceso se repite mientras

haya alguna interacción. En la figura 4.8 se describe el proceso general de corte.

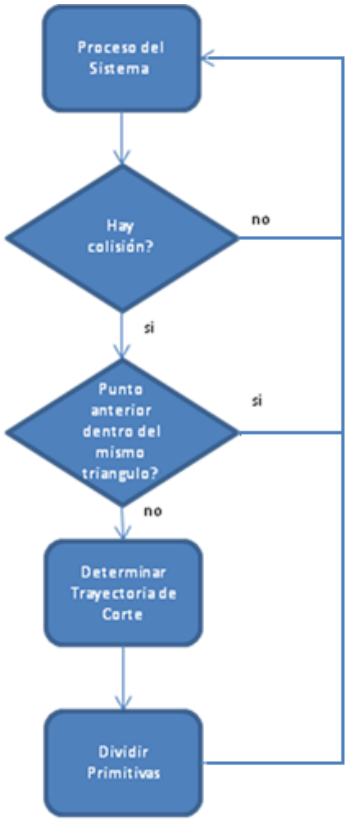

Figura 4.8: Proceso general de corte, cuando se detecta una colision se ejecuta el algoritmo de corte, el cual define la trayectoria de corte y ejecuta el algoritmo de división de primitivas.

El algoritmo puede ser descrito de la siguiente manera. Sea A el primer punto de interacción de la herramienta y B el segundo punto:

$$A = (X_a, Y_a, Z_a) \tag{4.1}$$

$$B = (X_b, Y_b, Z_b) \tag{4.2}$$

Y sean VA y VB los vectores que parten del origen hasta el punto A y B respectivamente, lo que es igual a la resta de cada punto con el origen.

$$V_a = (X_a - 0, Y_a - 0, Z_a - 0) \tag{4.3}$$

$$V_a = (X_a, Y_a, Z_a) \tag{4.4}$$

$$V_b = (X_b - 0, Y_b - 0, Z_b - 0) \tag{4.5}$$

$$V_b = (X_b, Y_b, Z_b) \tag{4.6}$$

Notese que ambos vectores tienen componentes iguales que los puntos, por lo que para hallar un vector que valla desde A hasta B (AB) se deben restar los vectores VA y VB (VB - VA), o lo que es igual, B - A.

$$V_{ab} = (X_a - X_b, Y_a - Y_b, Z_a - Z_b) \tag{4.7}$$

$$V_{ab} = (X_{ab}, Y_{ab}, Z_{ab}) \tag{4.8}$$

Luego de que tenemos el vector con la dirección del segmento de recta que une los puntos A y B, y con la dirección A-¿B procedemos a hallar el vector director de esa recta, que no es más que el mismo vector anterior pero normalizado. Para normalizar un vector se debe hallar su magnitud y dividir cada componente sobre la misma, de modo que el nuevo vector tendrá magnitud igual a 1.

$$|AB| = \sqrt{(X_{ab}^2 + Y_{ab}^2 + Z_{ab}^2} \tag{4.9}$$

$$ABN = \frac{AB}{|AB|} \tag{4.10}$$

Luego de haber hallado el vector director del segmento de recta de la trayectoria descrita por la herramienta de corte, se continua el mismo procedimiento con cada uno de los bordes unidos al vértice pivote, y luego de haber hallado cada uno, se calcula el ángulo a entre los 2 vectores. El cálculo del vector es realizado de la siguiente manera.

Sea V el vector director asociado con la trayectoria y AB el vector director asociado con cada uno de los bordes conectados al vértice pivote.

$$\alpha = cos^{-1}\left(\frac{(X * X_{ab}) + (Y * Y_{ab}) + (Z * Z_{ab})}{|V| * |AB|}\right) \tag{4.11}$$

Con el ángulo calculado, el siguiente paso es compararlo con los demás ángulos para luego escoger el borde que menor ángulo tenga. Este borde es el borde que va a ser cortado, pero como en realidad un corte no se puede llevar a cabo mediante la separación de bordes sino mediante la separación de vértices, es necesario esperar que se seleccione otro borde para luego si, poder separar el vértice que los une. El próximo paso es seleccionar el nuevo vértice pivote, a partir del cual se hallarán los bordes conectados a él y se seleccionará el nuevo más cercano, este vértice pivote será el otro vértice que se encuentre en el borde seleccionado actualmente y que no sea el mismo vértice pivote anterior. Esto con el fin de garantizar una trayectoria continúa durante toda la trayectoria

del instrumento. En la figura 4.9 se muestra como el algoritmo selecciona la trayectoria.

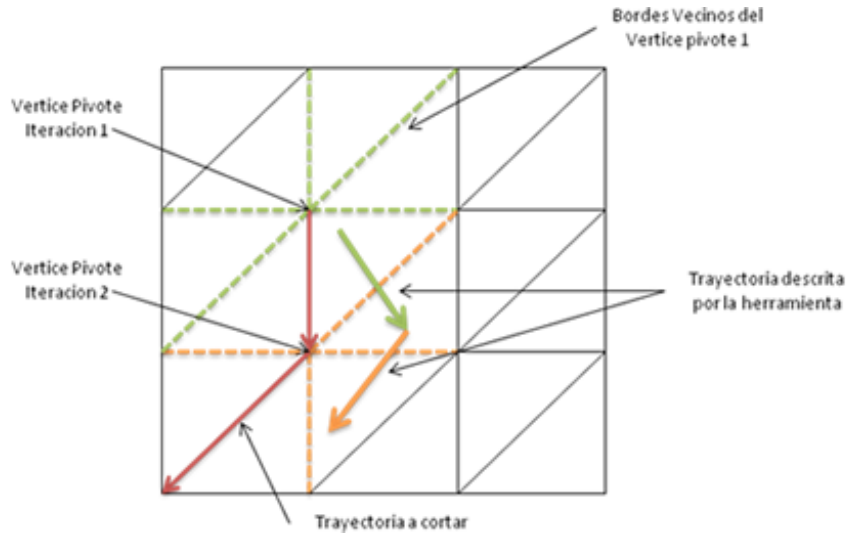

Figura 4.9: Selección de la trayectoria: en esta grafica se puede observar la trayectoria descrita por la herramienta, igualmente se puede ver los vértices pivotes en cada una de las iteraciones, con sus respectivos bordes vecinos y la trayectoria elegida de corte.

Algoritmo:

Sea:
p_{ant} : punto anterior
p_{act} : punto actual
p: : punto de colision
t_{ant} : triangulo anterior
t_{act} : triangulo actual
t : triangulo en colision
vp : vértice pivote
v : vértice a cortar, es el vértice que se le va a pasar al algoritmo de división de primitivas
b_{act} : borde actual
b_{ant} : borde anterior
pv : primera vez, bandera que indica si es la primera interaccion o no
pb : primer borde, bandera que indica si es el primer borde seleccionado o no
dir : dirección, vector que indica la dirección de la trayectoria
ang : angulo
am : angulo menor
b_{am} : borde de angulo menor

```
CORTE(p, t)
    SI(pv)
        p_ant = p
        t_ant = t
    SINO
        p_act = p
        dir = p_act − p_ant
    FIN SI
    SI(pb)
        vp = vertice mas cercano
        PARA CADA(borde vecino de vp)
            ang = hallar el ángulo entre el borde y dir
            SI(ang < am)
                b_am = borde
            FIN SI
        FIN
        b_act = b_am
        vp = otro vertice de bordeActual
    SINO
        PARA CADA(borde vecino de vp)
            ang = hallar el ángulo entre el borde y dir
            SI(ang < am)
                b_am = borde
            FIN SI
        FIN
        b_act = b_am
        v = vertice pivote
        Separar primitivas
        vp = otro vertice de bordeActual
    FIN SI
    t_ant = t_act
    b_ant = b_act
    p_ant = p_act
FIN CORTE
```

Este algoritmo funciona muy bien, pero analizando los tipos de corte que se realizan en las cirugías, se ve que los cortes son generalmente rectos, o en su defecto, trayectorias con una muy leve curvatura, la cual es única y suave, esto en parte se debe a la fuerza que ejercen las paredes de los tejidos a las paredes del bisturí impidiéndole un movimiento brusco de rotación, de tal manera que si un cirujano desease hacer un corte en V debe hacer dos cortes rectos. Siguiendo esta idea, se decidió agregar una mejora al algoritmo de tal forma que fuera capaz de reconocer una tendencia de corte especifica e ir ajustando el corte a la misma. En la figura 4.10 se puede observar un corte en forma de Zigzag.

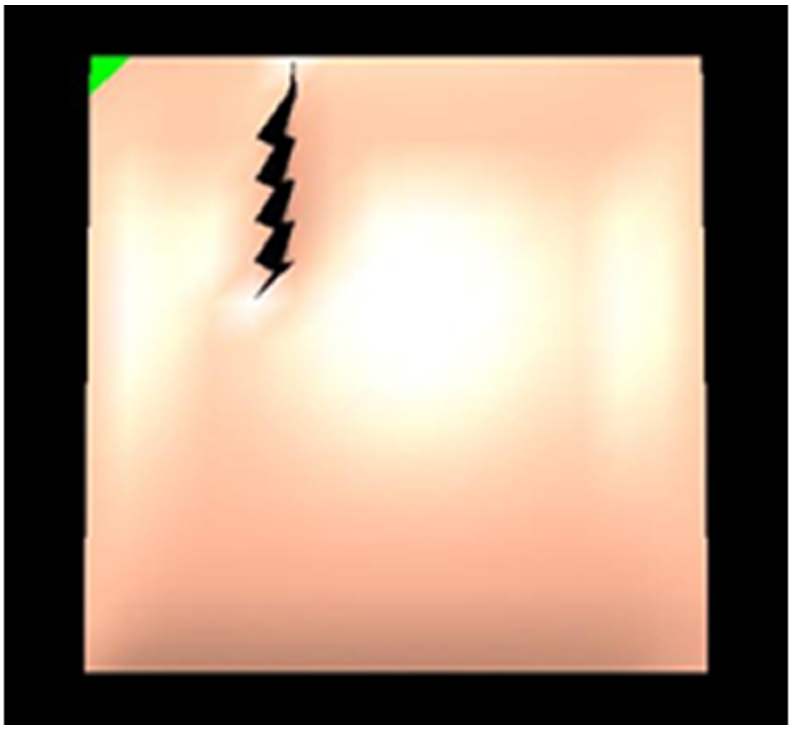

Figura 4.10: Corte en Zigzag: el algoritmo no reconoce la tendencia hacia abajo y simplemente corta según la trayectoria, que en este caso es de Zigzag, cosa que nunca se da en una cirugía real.

Ajuste de Tendencia

La corrección de la dirección mediante un ajuste de tendencia es un concepto sencillo. La idea básica es que en lugar de comparar cada vector asociado a cada borde con el vector asociado al segmento de la trayectoria, se compara con un vector que contiene la dirección de la tendencia. Este vector dirección de tendencia es igual a la suma de todos los vectores directores de los segmentos de recta de la trayectoria analizada hasta el momento, lo que nos ayuda a suavizar la trayectoria ya que aunque se presente un cambio brusco en la trayectoria, la suma de todos los vectores se ve afectada en una pequeña proporción, y va cambiando paulatinamente en caso de que continúe la nueva tendencia. Esto se puede ver en la Figura 4.11

El ajuste de tendencia simplemente es un cambio en el vector dirección, de modo que ya este no va a ser la dirección del segmento de la trayectoria que se está analizando sino una suma de todas las direcciones llamado tendencia. En la imagen 4.12 se puede ver la diferencia de un corte en ZigZag con y sin ajuste

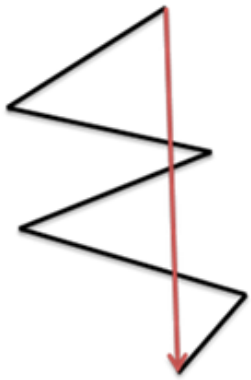

Figura 4.11: Vector dirección de la tendencia (en rojo): este vector es igual a la suma de todos los vectores que están en la trayectoria (en negro), de modo que un segmento con cambio de dirección brusco no afecta de manera significativa toda la trayectoria.

de tendencia. A continuación se describe el algoritmo.

Algoritmo:

Sea:
dir_{ten}: dirección de la tendencia

CORTE CON AJUSTE DE TENDENCIA(p, t)
SI(pv)
$p_{ant} = p$
$t_{ant} = t$
SINO
$p_{act} = p$
$dir = p_{act} - p_{ant}$
$dir_{ten} = dir_{ten} + dir$
FIN SI
FIN SI
SI(pb)
vp = vertice mas cercano
PARA CADA(borde vecino de vp)
ang = hallar el ángulo entre el borde y dir_{ten}
SI($ang < am$)
b_{am} = borde
FIN SI
FIN
$b_{act} = b_{am}$
vp = otro vertice de bordeActual

SINO
PARA CADA(borde vecino de vp)
ang = hallar el ángulo entre el borde y dir_{ten}
SI($ang < am$)
b_{am} = borde
FIN SI
FIN
$b_{act} = b_{am}$
v = vertice pivote
Separar primitivas
vp = otro vertice de bordeActual
FIN SI
$t_{ant} = t_{act}$
$b_{ant} = b_{act}$
$p_{ant} = p_{act}$
FIN CORTE CON AJUSTE DE TENDENCIA

4.3.3. Separacion de Primitivas

La separación de primitivas es un algoritmo muy importante dentro del proceso de corte. El proceso de corte esta básicamente compuesto por dos algoritmos, el primero que es llamado algoritmo de corte se encarga de la parte lógica del proceso, analiza los movimientos de la trayectoria del instrumento y decide que trayectoria cortar, el segundo que es llamado algoritmo de separación de primitivas es el que físicamente separa las estructuras de modo que se pueda visualizar el corte en el simulador quirúrgico.

Para hacer esto, se le debe indicar al algoritmo de separación de primitivas el vértice a cortar y los dos bordes que pertenecen a la trayectoria de corte y que tienen como vértice en común el vértice a cortar. Dependiendo del caso va a ser necesario crear dos bordes y un vértice o solo un borde y un vértice. En la figura 4.13, se puede observar la descripción gráfica del proceso de separación de primitivas que será definido a continuación. Si dentro del proceso se va a realizar el primer corte, es necesario que se creen ambos bordes y el vértice, a partir de b1, b2 y v. Además, si llamamos b1', b2', y v' a las nuevas primitivas creadas, tenemos que al borde b1' se le asignan las mismas propiedades de b1, a b2' se le asignan las mismas propiedades de b2, e igualmente al vértice v' se le asignan las propiedades del vértice v. Una vez creadas las primitivas y asignadas la propiedades básicas de estas a los bordes b1' y b2' se les asigna como vértice que pertenece a los bordes, el nuevo vértice creado (v'). Igualmente, se calculan los nuevos vecinos de v' y v. Los vecinos de un vértice dentro de nuestra estructura pueden ser de tres tipos: vértices, bordes y triángulos. Para definirlos como vecinos tiene que existir una conexión física y geométrica entre cada una de las primitivas relacionadas. En caso de no ser el primer corte del proceso se sigue el mismo procedimiento, solo se diferencia en que no se crea el borde b1'.

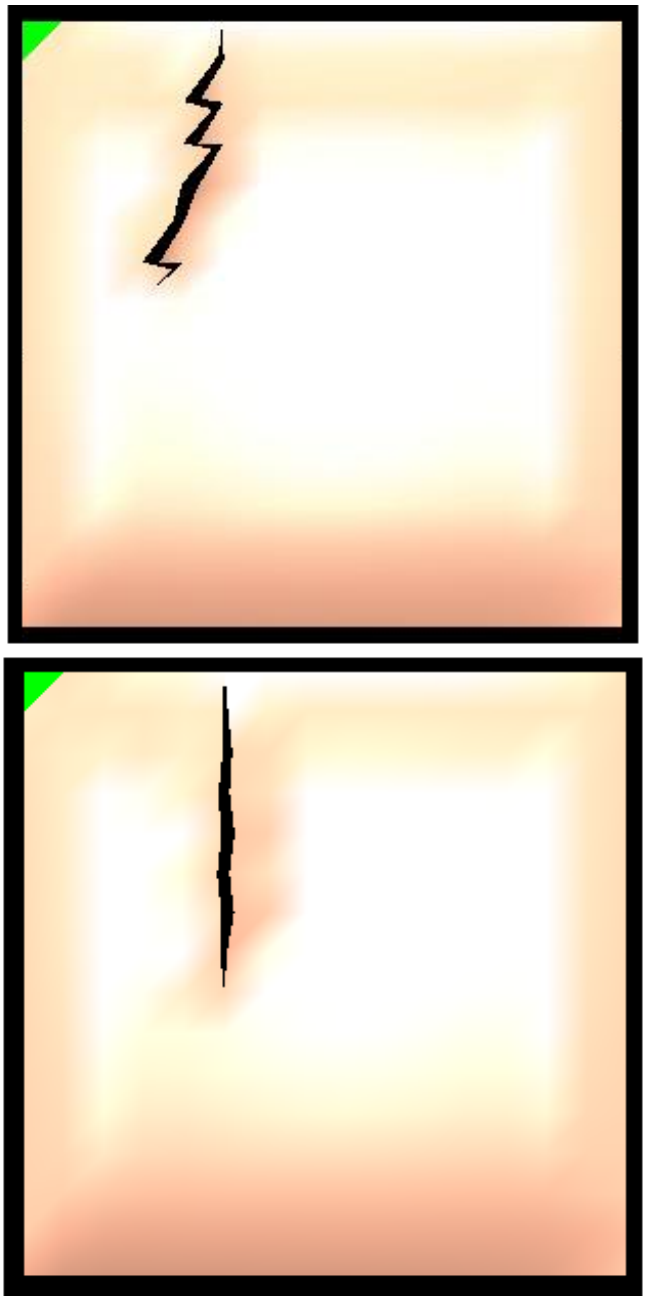

Figura 4.12: Corte en ZigZag: la imagen de arriba muestra el resultado de un corte en ZigZag sin ajuste de tendencia y la imagen de abajo muestra el mismo corte con el ajuste de tendencia.

Descripcion del Algoritmo

El algoritmo de división de primitiva recibe los dos bordes a ser cortados al igual que el vértice que une a estos dos bordes. Esta información es suficiente

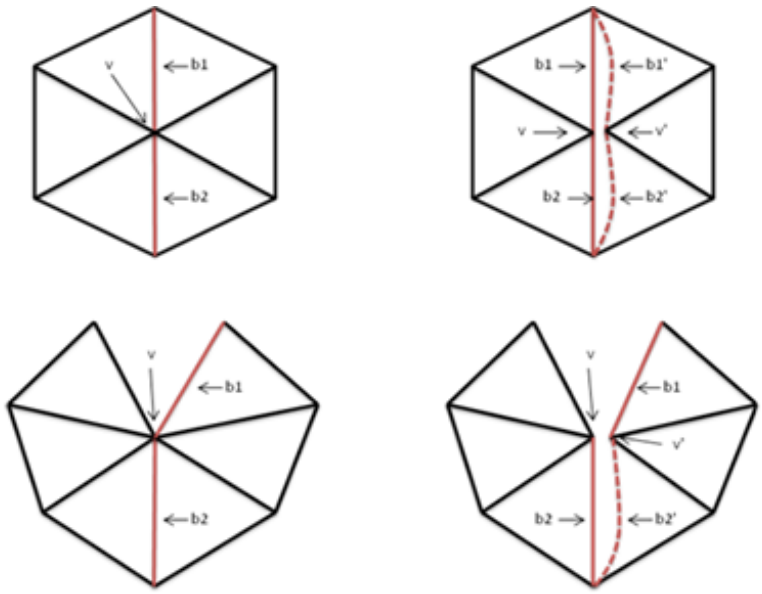

Figura 4.13: Separacion de primitivas: Caso 1 (arriba): se da cuando es el primer corte, se debe crear un nuevo vértice y dos nuevos bordes como se muestra en la figura, Caso 2 (abajo): solo de debe crear un nuevo vértice y un nuevo borde, este caso se da el resto de las veces.

para poder crear las primitivas nuevas y hacer las modificaciones pertinentes en la malla para poder ver reflejado el corte.

DIVISION DE PRIMITIVAS($v, b1, b2$)
  SI(pv)
    Crear borde $b1' = b1$
  FIN SI
  Crear vértice $v' = v$
  Crear borde $b2' = b2$
  PARA CADA(borde vecino de v desde $b1$ hasta $b2$)
    Asignar v' al borde al igual que al triangulo
    Actualizar vector de vecinos de v' y v
  FIN
FIN DIVISION DE PRIMITIVAS

Capítulo 5

Configuración Experimental y Resultados

Las pruebas son una parte muy importante ya que permiten evaluar el desempeño del algoritmo. Hay que recordar que un aspecto muy importante del proceso de corte es que se ejecute en tiempo real, cosa que hay que verificar, y con este fin se diseñaron varios análisis y pruebas.

5.1. Analisis de Complejidad del Algoritmo

Todo el proceso de corte está compuesto por el algoritmo que selecciona la trayectoria y el algoritmo de separación de primitivas. El algoritmo que selecciona la trayectoria lo que hace es comparar el ángulo que existe entre la trayectoria y cada borde para luego escoger el menor. Esto significa que por cada iteración, el algoritmo hace n comparaciones siendo n el número de vecinos que tiene el vértice.

De igual manera, el algoritmo de división de primitivas crea las nuevas primitivas y asigna a los vértices vecinos las nuevas primitivas según sea el caso y actualiza sus listas de vecinos, por lo cual, en cada iteración, el algoritmo hace las asignaciones n veces siendo n el número de vecinos que tiene el vértice.

Esto quiere decir que el proceso de corte tiene una complejidad de orden 2n, dependiente del número de vecinos que tiene el vértice, mas no depende del número de triángulos de la malla, y generalmente un vértice tiene entre 3 y 6 vértices vecinos, y este número nunca va a ser muy elevado, por lo que prácticamente este número va a ser constante, lo que reduce la complejidad del algoritmo a orden 1.

5.2. Numero de Primitivas Creadas

Cada vez que se llama a la función de separación de primitivas se crea un nuevo borde y un nuevo vértice, a excepción de la primera vez, que se crean dos bordes y un vértice. Comparando esto con el numero de triangulos colisionados

en la trayectoria de corte, se tiene que en la primera colision no se realiza corte alguno, al igual que en la segunda, ya que estos dos puntos forman el primer segmento de recta cuya dirección va a ser comparada con las direcciones de los bordes vecinos, es entonces a partir del tercer triangulo que se empiezan a crear primitivas. El numero de primitivas creadas por triangulos colisionados se puede expresar mediante la siguiente ecuación:

$$NP = (2 * N) - 3 \tag{5.1}$$

Donde N es el numero de vecinos que tiene el vertice. Se puede ver que el numero de primitivas se mantiene siempre bajo, ya que solo se crean las que son necesarias para mostrar el corte.

5.3. Pruebas

Las pruebas que se diseñaron están enfocadas a comprobar la ejecución en tiempo real e interactividad que ofrece el algoritmo. Para esto se pueden tener varias métricas con las cuales determinar si el algoritmo cumple con estos requisitos o no, una de estas métricas es el tiempo de ejecución, que en este caso fue demasiado corto como para notar diferencias significativas en las pruebas, por lo que se decidió utilizar como métrica los FPS (Frames per Second o Frames por Segundo) que muestra el simulador, recordando los limites que se mencionaron anteriormente en el documento para que un ser humano pudiera percibir un movimiento fluido, el cual es de aproximadamente 30 FPS. El simulador Eafit-Ces, cuando no esta realizando operaciones de corte muestra 60 FPS, y tomando este valor como referencia, se hicieron pruebas en las cuales se variaba el numero de triangulos colisionados y otras en las cuales se variaba el tipo de trayectoria.

Estas pruebas se llevaron a cabo en una malla en forma de cuadrado de 15 filas y 15 columnas en donde cada cuadrado esta conformado por dos triangulos. esta malla tiene un total de 450 triangulos y 256 vertices. La maquina con la cual se llevaron a cabo estas pruebas tiene las siguientes caracteristicas:

- Modelo: Dell XPS - XPS710
- Procesador: Intel Core 2 Quad 2.66Ghz
- Memoria Ram: 2 GB
- Tarjeta de Video: Nvidia GeForce 7900GS 256MB

En la tabla 5.3 se puede observar que el numero de FPS del simulador no se ve afectado por el numero de triangulos que son cortados. Ahora, en la tabla 5.2 se encuentran los resultados de cortes con 12 triangulos con diferentes direcciones.

Triángulos	FPS
3	60
4	60
5	60
6	59
7	59
8	59
9	60
10	59
11	59
12	59
13	59
14	59
15	59

Cuadro 5.1: Numero de Frames Por Segundo mostrados por el simulador cuando realiza un corte con un numero n de triangulos en la trayectoria

Dirección	FPS
Vertical	60
Horizontal	60
Diagonal	59
En S	60
En ZigZag	59

Cuadro 5.2: Numero de Frames Por Segundo mostrados por el simulador cuando realiza un corte con 12 en la trayectoria en diferentes direcciones

Igualmente se obtuvieron los resultados esperados, el algoritmo no depende de la dirección de corte. Ahora se mostraran los resultados de las primitivas creadas dependientes del numero de triangulos.

Y gráficamente se puede ver en 5.1.

Triángulos	Primitivas
3	3
4	5
5	7
6	9
7	11
8	13
9	15
10	17
11	19
12	21
13	23
14	25
15	27

Cuadro 5.3: Numero de Frames Por Segundo mostrados por el simulador cuando realiza un corte con un numero n de triangulos en la trayectoria

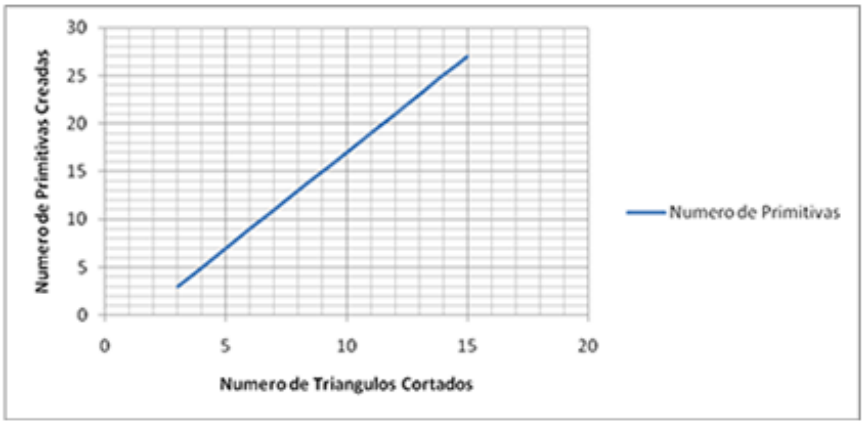

Figura 5.1: Numero de Primitivas creadas dependiendo del numero de triangulos cortados

Capítulo 6

Conclusiones

A partir de los objetivos propuestos al inicio del desarrollo y observando los resultados obtenidos podemos concluir:

- La metodología propuesta inicialmente para el modelado en 3D de la vesicular biliar no presento los resultados esperados debido a que la vesicular es un órgano muy pequeño y esta rodeado de tejidos de color similar por lo que el algoritmo de detección de bordes no era capaz de definir correctamente el borde de la vesicular y se debió hacer esto manualmente, también porque el error cometido por el algoritmo de interpolación spline se aumento en gran medida debido al tamaño de la vesícula. Todo esto condujo a que el modelo resultante por esta metodología no fuera el deseado. La metodología de modelación mediante Blender produjo un modelo de vesicular que aunque no es exactamente fiel a la realidad si es una buena representación de la vesicular y además cuenta con un numero bajo de triángulos, lo que ayuda a mantener el nivel de interactividad del simulador.
- El algoritmo propuesto inicialmente es una buena aproximación pero es muy susceptible a cualquier tipo de cambio en la trayectoria de modo que los cortes resultantes tratan de ajustarse fielmente a esta trayectoria produciendo un corte, en algunos casos irregular, que podía no representar un corte en una cirugía real. La corrección de la trayectoria mediante un ajuste de tendencia presento unos buenos resultados ya que se suaviza la trayectoria y solo permite cambios suaves en la misma.
- Las pruebas de interactividad y desempeño basadas en el número de FPS a los cuales se desempeña el simulador mostraron unos muy buenos resultados ya que este número se mantenía en los 60 FPS, comprobando la eficiencia del algoritmo.
- Las pruebas de interactividad y desempeño basadas en el numero de primitivas creadas por cada triangulo colisionado muestran que el numero de primitivas creadas se mantiene en un nivel bajo, y que nunca se crean triángulos nuevos, de manera que es poca la carga adicional que se le impone al modelo físico para que este calcule las fuerzas y deformaciones en cada nodo.

Capítulo 7

Trabajo Futuro

A medida que se desarrollaba el algoritmo se fueron observando una serie de aspectos que podrían ser mejorados con el fin de aumentar el desempeño y el realismo. Estos aspectos son:

- Con el fin de aumentar el realismo del corte es necesario hacer unas modificaciones al modelo fisico (que es el encargado de retraer el tejido cuando se produce un corte) de modo que se eliminen ciertas irregularidades en el borde debido a que las fuerzas que retraen el tejido son diferentes en cada borde y por lo tanto unos bordes lo hacen mas que otros.
- Para ampliar las funcionalidades del algoritmo de corte, se puede implementar también con mallas volumétricas, teniendo en cuenta las modificaciones pertinentes y los nuevos ajustes necesarios para lograr esto. Es importante porque no todos los simuladores trabajan con mallas superficiales, también para poder utilizar los principios del algoritmos de corte en otras aplicaciones.
- Siendo las fuerzas de retroalimentación una parte muy importante en la cirugía debido a que mediante estas el cirujano regula la fuerza que realiza en los diferentes cortes con el fin de no cortar tejido que no se debía cortar, es necesario simular también esto, para asi agregarle un grado más alto de realismo, de modo que no se realice el corte cuando hay una colisión, sino cuando la fuerza ejercida por la herramienta sobre los tejidos supera la resistencia de estos y luego se lleva a cabo el corte, también es posible simular la fuerza ejercida por las paredes abdominales a la herramienta, y utilizando un dispositivo de retroalimentación háptica hacer que el cirujano perciba todas estas fuerzas.

Bibliografía

[1] Cagatay Basdogan, Suvranu De, Jung Kim, Manivannan Muniyandi, Hyun Kim, and Mandayam A. Srinivasan. Haptics in minimally invasive surgical simulation and training. *IEEE Comput. Graph. Appl.*, 24(2):56–64, 2004.

[2] Grigore C. Burdea. *Force and touch feedback for virtual reality.* John Wiley & Sons, Inc., New York, NY, USA, 1996.

[3] Suvranu De, Yi-Je Lim, Muniyandi Manivannan, and Mandayam A. Srinivasan. Physically realistic virtual surgery using the point-associated finite field (paff) approach. *Presence: Teleoper. Virtual Environ.*, 15(3):294–308, 2006.

[4] B.L. Gray and R.S. Fearing. A surface micromachined microtactile sensor array. In *Proceedings. of IEEE International Conference on Robotics and Automation*, pages 1–6, 1996.

[5] Min Hong, Sunhwa Jung, Min-Hyung Choi, and Samuel W. J. Welch. Fast volume preservation for a mass-spring system. *IEEE Comput. Graph. Appl.*, 26(5):83–91, 2006.

[6] James T. Klosowski, Martin Held, Joseph S. B. Mitchell, Henry Sowizral, and Karel Zikan. Efficient collision detection using bounding volume hierarchies of k-dops. *IEEE Transactions on Visualization and Computer Graphics*, 4(1):21–36, 1998.

[7] Christian Laugier, Cesar Mendoza, and Kenneth Sudaraj. *Towards a Realistic Medical Simulator using Virtual Environments and Haptic Interaction*, volume 6, pages 289–306. Springer - Berlin - Heidelberg, Germany, 2003.

[8] Christian Andrés Diaz León. Diseño y construcción de un ambiente virtual enfocado a la simulación de cirugía laparoscópica. Technical report, Universidad EAFIT, 2009.

[9] Anderson Maciel, Ronan Boulic, and Daniel Thalmann. Deformable tissue parameterized by properties of real biological tissue. *Lectures Notes in Computer Science*, pages 74–87, 1999.

[10] U. Meier, O. Lopez, C. Moserrat, M. Juan, and J. Alcañiz. Real time deformable models for surgery simulation: A survey. *Computer Methods and Programs in Biomedicine*, 77(3):183–197, 2005.

[11] Jesper Mosegaard. Lr-spring mass model for cardiac surgical simulation. *Studies in Health Technology and Informatics*, 98(null):256–258, 2004.

[12] Diego C. Ruspini, Krasimir Kolarov, and Oussama Khatib. The haptic display of complex graphical environments. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 345–352, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[13] Clemens Wagner, Markus Schill, and Reinhard Männer. Intraocular surgery on a virtual eye. *Commun. ACM*, 45(7):45–49, 2002.

[14] C. B. Zilles and J. K. Salisbury. A constraint-based god-object method for haptic display. In *IROS '95: Proceedings of the International Conference on Intelligent Robots and Systems-Volume 3*, page 3146, Washington, DC, USA, 1995. IEEE Computer Society.