Universidad del Bío-Bío, Chile

Facultad de Ciencias Empresariales

Departamento de Sistemas de Información

Herramienta de apoyo a la planificación del despliegue de redes de sensores inalámbricos industriales

Tesis presentada por Ignacio Muñoz Zurita
para obtener el título de Ingeniero de ejecución en computación e informática
Dirigida por Dr. Pedro Ángel Rodríguez Moreno
Codirigida por Dr. Cristian Durán Faúndez, Dr. Patricio Galdames Sepúlveda

2019

Universidad del Bío-Bío, Chile

Facultad de Ciencias Empresariales

Departamento de Sistemas de Información

Herramienta de apoyo a la planificación del despliegue de redes de sensores inalámbricos industriales

Tesis presentada por Ignacio Muñoz Zurita
para obtener el título de Ingeniero de ejecución en computación e informática
Dirigida por Dr. Pedro Ángel Rodríguez Moreno
Codirigida por Dr. Cristian Durán Faúndez, Dr. Patricio Galdames Sepúlveda

2019

Agradecimientos

A mi profesor guía Dr. Pedro Rodríguez. **(QEPD)**

Quiero agradecerle a él por cada detalle y momento dedicado para aclarar cualquier tipo de duda que me surgiera, agradecerle por la caridad y exactitud con la que enseño cada clase, por su gran motivación para la culminación de mis estudios profesionales y para la elaboración de esta tesis, por impulsar el desarrollo de mi formación profesional, por sus palabras de aliento en los momentos complejos de esta tesis, por la orientación y conocimiento que me brindó, por su apoyo que me permitieron aprender mucho más que lo estudiado en el proyecto, por confiar en mi y mis capacidades desde el primer día. Gracias profesor por haber elegido ser profesor, gracias por haberme enseñado tan bien y por haberme permitido el desarrollo de esta tesis. Gracias profesor.

A mi familia.

A mis padres quienes han sido un pilar fundamental en mi vida, en todo lo que soy, en toda mi educación, tanto académica, como de la vida, por su incondicional apoyo perfectamente mantenido a través del tiempo y darme las fuerzas para poder seguir adelante y no dacaer en aquellos momentos adversos. A mis tíos y tías por darme ánimo y preocuparse de mi. A mis hermanos, mis primos y primas y a todos aquellos que de alguna u otra manera fueron parte de este proceso.

A mis amigos.

A mis amigos que conocí en el transcurso de mi carrera universitaria, por todo el apoyo brindado, en especial a mis amigos Ricardo Muñoz y Cristian Venegas, por el apoyo, la preocupación, el ánimo, el conocimiento brindado y por esos grandes y gratos momentos compartidos.

A mis profesores.

A mis profesores Cristian Durán Faúndez y Patricio Galdames Sepúlveda, por su buena voluntad y por su motivación hacia mi para seguir adelante y cumplir con mis objetivos.

Resumen

En la Universidad del Bío-Bío se está trabajando en el problema de la optimización del despliegue de nodos de redes de sensores inalámbricas industriales. Como resultado de la investigación realizada, se han implementado aplicaciones que utilizan meta-heurísticas para la generación de soluciones semi-óptimas. Las soluciones encontradas son útiles en términos de sus resultados, pero no existe una aplicación gráfica que permita la interacción con un usuario final y eso hace que las soluciones encontradas y su información sean entenbibles solo para quién conoce las aplicaciones actualmente implementadas.

Este proyecto nace de la necesidad de poder mostrar de forma gráfica y en tres dimensiones las distintas opciones de despliegue de redes de sensores inalámbricos de visión utilizando la ayuda de un módulo optimizador externo, mostrando su lugar de ubicación y la posición de estos, además de incorporar la trayectoria del cableado que une el nodo sensor con el transmisor.

Palabras claves — Redes de sensores inalámbricas industriales, Despliegue de nodos.

Índice general

1. Introducción **1**

1.1. Aspectos generales 1
1.2. Descripción del problema 2
1.3. Solución del problema 3
1.4. Objetivos general 5
1.5. Objetivos específicos 5
1.6. Aporte del proyecto 6
1.7. Organización 7

2. Estado del Arte **8**

2.1. Redes de sensores inalámbricos 8
2.1.1. Características de los Nodos Sensores 9
2.1.2. Aplicaciones 10
2.2. Redes de sensores inalámbricas industriales 10
2.2.1. Aplicaciones IWSN 11
2.3. Despliegue de Nodos 12
2.4. Problemas del despliegue de Nodos 14
2.5. Conectividad y Cobertura en una WSN 14
2.6. Problema del Modelado de Entorno Industrial 15
2.7. Modelado 3D 16
2.8. AutoCAD para Modelado 3D 17
2.9. Otros software para Modelado 3D 18

3. Mallas geométricas de superficie **19**
3.1. Definición . . . 19
3.2. Archivos con extensión .m2d . . . 19
3.3. Elementos topológicos geométricos . . . 20
3.3.1. Vértice . . . 20
3.3.2. Arista . . . 21
3.3.3. Triángulo . . . 21
3.4. Estructura extensión .m2d . . . 22

4. Refinamiento de Mallas **25**
4.1. Bisección de Triángulos . . . 25
4.2. Refinamiento mediante la Bisección de triángulos . . . 26
4.3. Lepp: Longest edge propagation path . . . 26
4.3.1. Arista terminal . . . 27
4.4. Bisección de un triángulo por arista más larga . . . 28
4.5. Algortimo Lepp-Bisecction . . . 29

5. Ambiente de Ingeniería de Software **32**
5.1. Metodología de desarrollo . . . 32
5.2. Técnicas y notaciones . . . 33
5.3. Herramientas de apoyo al desarrollo de software . . . 33
5.3.1. Hardware utilizado . . . 33
5.3.2. Planificación inicial del proyecto . . . 34
5.4. Riesgos asociados al desarrollo del proyecto . . . 34
5.4.1. Riesgos asociados al desarrollo del proyecto . . . 35
5.4.2. Riesgos asociados al software . . . 36
5.5. Definiciones, Siglas y Abreviaciones . . . 36
5.6. Especificación de Requerimientos de Software . . . 38
5.6.1. Alcances . . . 38
5.6.2. Límites . . . 38
5.7. Descripción Global del Producto . . . 39
5.7.1. Interfaz de usuario . . . 39
5.7.2. Interfaz de hardware . . . 39

5.7.3. Interfaz de software 39
5.7.4. Interfaces de comunicación 40
5.8. Requerimientos Específicos 40
5.8.1. Requerimientos funcionales (RF) 40
5.8.2. Requerimientos no funcionales(RNF) 41
5.9. Atributos del producto 41
5.9.1. Usabilidad 41
5.9.2. Funcionalidad 41
5.9.3. Fiabilidad 42
5.9.4. Eficiencia 42
5.9.5. Portabilidad 42

6. Análisis **43**
6.1. Casos de uso 43
6.1.1. Actores 43
6.2. Diagrama casos de uso y descripción 44
6.2.1. Diagrama de casos de uso 44
6.2.2. Descripción casos de uso 45
6.2.2.1. Caso de uso <Cargar modelo> 45
6.2.2.2. Caso de uso <Visualizar modelo> 46
6.2.2.3. Caso de uso <Refinar modelo> 47
6.2.2.4. Caso de uso <Guardar modelo> 48
6.2.2.5. Caso de uso <Ingresar elementos> 49
6.2.2.6. Caso de uso <Visualizar información> 51
6.3. Diagrama de flujo 53
6.3.1. Descripción diagrama 54
6.3.1.1. <Salir> 54
6.3.1.2. <Guardar> 55
6.3.1.3. <Cargar Modelo> 57
6.3.1.4. <Interacción con modelo> 59
6.3.1.5. <Selección> 61
6.3.1.6. <Selección arista más larga> 62

6.3.1.7. <Refinar selección> 63
6.3.1.8. <Ejecutar LEPP-Bisección> 64
6.3.1.9. <Desplegar Sensores/Gateways> 65
6.3.1.10. <Interacción con Sensores/Gateways> 68
6.3.1.11. <Optimizador> 70
6.3.1.12. <Despliegue Transmisores> 72
6.3.1.13. <Interacción con Gateways, Sensores y Transmisores> . . 74

7. Diseño del software **77**
7.1. Biblioteca OpenGL 77
7.1.1. Biblioteca añadidas 77
7.2. De vértice a píxeles 78
7.3. Técnicas utilizadas en el software 80
7.4. Técnica para realizar el despliegue 82
7.5. Pintado de un elemento sobre un vértice 83
7.6. Layout de ventana 84
7.6.1. MainWindow y MeshViewer 84
7.7. MainWindow 86
7.7.1. Barra de Herramientas 86
7.7.2. Gráfica 86
7.7.3. Status bar 87
7.8. Diagrama de secuencia 88

8. Pruebas de funcionalidad del software **90**
8.1. Pruebas gráficas 90
8.2. Rotación 90
8.3. Desplazamiento 92
8.4. Acercamiento (Zoom) 93
8.5. Sobrecarga de triángulos y vértices 94
8.6. Pruebas de selección 98
8.6.1. Select by longest edge 98
8.6.2. Selección por lado menor a valor 99
8.6.3. Selección por lado mayor a valor 100

8.7. Prueba de algortimo 101
8.7.1. LEPP Bisección 101
8.8. Pruebas de interacción con el modelo 102
8.8.1. Despliegue de Sensores 102
8.9. Despliegue de Gateways 103
8.10. Despliegue de Transmisores mediante optimizador 104
8.11. Pruebas de herramientas de interacción 106
8.11.1. Info Points 107
8.11.2. Info Mesh 108
8.11.3. Info Elements 109
8.11.4. Connection Info 110
8.11.5. Paint Mode y Zoom 111
8.11.6. Clear all window 112

9. Conclusiones **113**

Referencias **116**

Índice de figuras

1.1. Arquitectura de solución del problema. 4

2.1. Arquitectura básica de un nodo sensor 9
2.2. Nodo sensor conectado con el transceptor 13

3.1. Ejemplo de Vértices 20
3.2. Ejemplo de Arista 21
3.3. Ejemplo de Triángulo 22
3.4. Triángulos formados por distintos vértices 23

4.1. Bisección Triángulo 25
4.2. Lepp: Longest edge propagation path 27
4.3. Ejemplo arista terminal 27
4.4. Bisección de triángulo por arista más larga [24]. 28
4.5. Nueva bisección triángulo 29
4.6. Refinamiento LEEP-Bisección del triángulo t_0 31

5.1. Metodología incremental 32
5.2. Arquitectura de referencia de la herramienta. 39

6.1. Diagrama de Casos de uso. 44
6.2. Diagrama de flujo. 53

7.1. Secuencia de trabajo OpenGL. 78
7.2. Método para inicializar area de visualización. 79
7.3. Método diseñado para obtención de coordenadas. 80

7.4. Método diseñado para dibujar sensor en el modelo. 81
7.5. Selección vértice y transformación coordenadas. 82
7.6. Pintado de elemento sobre el modelo. 83
7.7. Mockup de la ventana principal. 84
7.8. Diagrama de clases de MainWindow con Meshviewer. 85
7.9. Mockup MeshViewer 87
7.10. Diagrama de secuencia del software. 88

8.1. Rotación modelo en base al eje Y. 91
8.2. Rotación modelo en base al eje X. 91
8.3. Desplazamiento del modelo 92
8.4. Desplazamiento del modelo 92
8.5. Rotación modelo en base al eje X. 93
8.6. Gráfico de frames por segundo. 95
8.7. Sobrecarga de triángulos y vértices menor a 7000 y fps. 96
8.8. Sobrecarga de vértices y triángulos mayor a 7000 y fps. 97
8.9. Menú select by longest edge en barra de herramientas. 98
8.10. Selección Smaller. 99
8.11. Selección Biggest. 100
8.12. LEEP Bisección. 101
8.13. Draw Sensor 102
8.14. Draw Gateway 103
8.15. Selección optimizador 104
8.16. Selección optimizador 105
8.17. Info Points 107
8.18. Info Mesh 108
8.19. Info Mesh 109
8.20. Connection Info 110
8.21. Paint mode y Zoom menú 111
8.22. Clear all Window 112

Índice de tablas

4.1. Algoritmo Lepp-Bisection 30

5.1. Riesgo de proyecto 35
5.2. Riesgo asociados al software 36
5.3. Requerimientos funcionales herramienta visualización. 40
5.4. Requerimientos no funcionales herramienta de visualización. 41

6.1. Descripción Caso de Uso <Cargar modelo> 45
6.2. Descripción Caso de Uso <Cargar modelo> 46
6.3. Descripción Caso de Uso <Refinar modelo> 48
6.4. Descripción Caso de Uso <Guardar modelo> 49
6.5. Descripción Caso de Uso <Ingresar elementos> 51
6.6. Descripción Caso de Uso <Visualizar información> 52
6.7. <Salir> 54
6.8. <Guardar> 56
6.9. <Cargar Modelo> 58
6.10. <Interacción con Modelo> 60
6.11. <Selección> 61
6.12. <Selección arista más larga> 62
6.13. <Refinar Selección> 63
6.14. <Ejecutar LEPP-Bisección> 64
6.15. <Desplegar Sensores/Gateways> 68
6.16. <Interacción con Sensores/Gateways> 69
6.17. <Optimizador> 72

6.18. <Despliegue Transmisores> 73
6.19. <Interacción con Gateways, Sensores y Transmisores> 76

8.1. Tabla frames por segundo. 94

Capítulo 1

Introducción

1.1. Aspectos generales

En la actualidad la programación en tres dimensiones (3D) es la misma que podemos encontrar en los diferentes o mayoría de proyectos que hoy en día están desarrollando los programadores que usan la programación orientada a objetos. Existen formas para definir los diferentes objetos en tres dimensiones, utilizando coordenadas en tres variables: *X, Y, Z*, que corresponden a coordenadas cartesianas, donde X indica el ancho, Y largo y Z alto. Esta forma y técnica de programación trae muchos beneficios en la actualidad, como por ejemplo en la creación de video-juegos, robótica, programas computarizados de espacios tridimensionales como lo es AutoCAD, FreeCAD y otros modeladores en tres dimensiones que hoy en día son esenciales en la industria y cumplen con el objetivo de construir o modelar distintos escenarios, cómo por ejemplo un entorno industrial.

Otra forma de interpretar un modelo en tres dimensiones, es a través de una malla de triángulos, que es una superficie creada mediante un método tridimensional generado por sistemas de vértices posicionados en un espacio virtual con datos de coordenadas propios. Hoy en día, las mallas se encuentran en una infinidad de aplicaciones y distintas disciplinas que las usan, entre las cuales destacan el diseño asistido por computadora (CAD) y métodos de elementos finitos (MEF o FEM) y una de las utilidades de las mallas de triángulos también podría ser para sistemas de reconocimiento de objetos, la comprensión de una escena, y el modelado de un entorno industrial.

Estas tecnologías como lo son las mallas geométricas y la programación, combinadas con otros tipos de tecnologías, han de ser el futuro hacia el que avanzamos, en este caso, la programación en tres dimensiones (3D) en combinación con la automatización industrial, se convierten en una poderosa herramienta de trabajo, facilitando muchas tareas en la industria con el apoyo de herramientas de programación.

Un tipo de automatización industrial son las redes de comunicación inalámbrica que si bien data del 1880, hoy en día se ha llevado a un nivel muy superior. Un ejemplo de estas redes son las Redes de Sensores Inalámbricos (WNS, del inglés *W*ireless Sensor Network). Están compuestas por una serie de dispositivos pequeños y de bajo consumo de energía, llamados nodos que trabajan comunicándose entre sí, es decir, que es una red con numerosos dispositivos, distribuidos espacialmente, que utilizan sensores para controlar distintos puntos, entre ellas, la temperatura, el sonido, la vibración, la presión, movimiento y los contaminantes [1].

Estos sensores vinculan lo físico con el mundo digital, capturando y revelando fenómenos del mundo real y convirtiéndolos en una forma que puede ser procesada y almacenada por el computador, para actuar en consecuencia. Integrados en numerosos dispositivos, máquinas y entornos, los sensores proporcionan un gran beneficio para la sociedad. Pueden ayudar a evitar fallas de infraestructura catastróficas, conservar recursos naturales valiosos, aumentar la productividad, mejorar la seguridad y habilitar nuevas aplicaciones, como sistemas sensibles al contexto y tecnologías domésticas inteligentes.

1.2. Descripción del problema

Actualmente el proceso de despliegue de nodos de redes de sensores inalámbricos industriales desarrollado en la Universidad del Bío-Bío, es realizada solo por un módulo optimizador externo, el cual determina la posición de los nodos transmisores, de acuerdo a ciertas posiciones de los sensores y otros parámetros. Este módulo optimizador externo tiene un límite y es que su información asociada no está en forma gráfica, es decir, no existe un apoyo visual realista que permita observar cómo es que se aplican en un ambiente

industrial los datos obtenidos por este módulo.

La información asociada o datos de salida de este módulo optimizador no es completamente entendible por un usuario final.

1.3. Solución del problema

Luego de haber analizado la problemática y haber recopilado datos sobre las herramientas a utilizar, se ha diseño una solución acorde, que cumple y cubre las necesidades del problema completamente.

Teniendo en cuenta los objetivos y requerimientos, se presenta una solución a implementar, con el fin de resolver los problemas que existen en el despliegue de redes de sensores inalámbricas que se está trabajando en la Universidad del Bío-Bío. Bajo este enfoque, la solución planteada consiste en diseñar e implementar nuevas funcionalidades de una herramienta de visualización que será adaptada para representar entornos industriales en tres dimensiones (3D) que servirá de apoyo al módulo optimizador externo diseñado previamente en la Universidad del Bío-Bío. Para lograr el objetivo de esta solución, será necesario utilizar varias herramientas:

- **AutoCAD**: herramienta necesaria para el modelado o simulado de los entornos industriales en tres dimensiones (3D).

- **FreeCAD**: herramienta de visualización de mallas, también para la plataforma Linux. Necesario para realizar la conversión de formatos.

- **Algoritmo de transformación**: algoritmo programado el lenguaje C++ para lograr la conversión de un archivo exportado desde FreeCAD en formato **.off** y convertilo en un formato **.m2d**.

- **Malla de superficie:** archivo con extensión .m2d que corresponde a una malla de triángulos y vértices.

- **Visualizador 3D:** programado en lenguaje C++, con entorno gráfico implementado con la biblioteca **OpenGL** y las bibliotecas de **Qt**. Esta herramienta inicial de visualización 3D fue desarrollada y proporcionada por el profesor Pedro Rodríguez Moreno. Este visualizador 3D acepta modelos creados, usando una malla geométrica de superficie, en formato **m2d**.

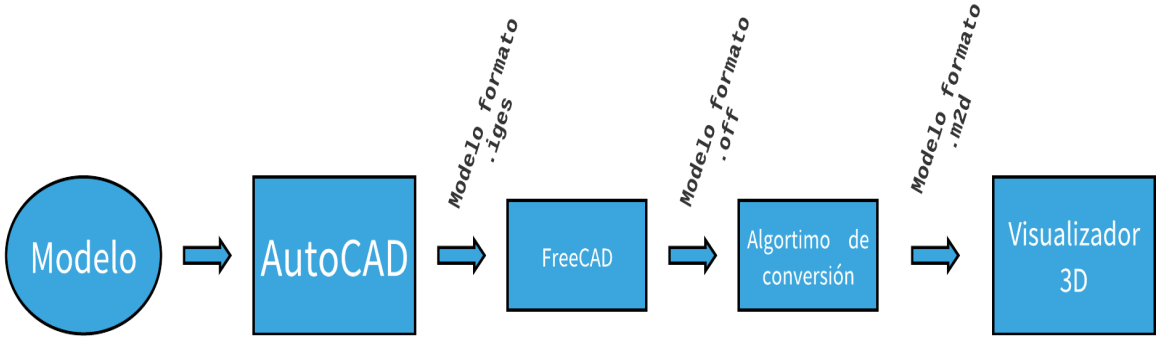


Figura 1.1: Arquitectura de solución del problema.

Para la realización del espacio en tres dimensiones, será necesario utilizar AutoCAD, ya que esta herramienta es la más idónea, completa, y que actualmente es la más utilizada por las grandes empresas. Una vez creado el entorno en la herramienta de AutoCAD, se exportará con formato **.iges** para poder realizar la importación hacia el programa de Linux, FreeCAD, dado que este formato es aceptado por estos software y son capaces de interpretar completamente el modelo previamente diseñado en AutoCAD. De esta forma, se debe realizar nuevamente una exportación desde FreeCAD, pero esta vez en formato **.off**. Una vez realizados todos estos pasos, es momento de aplicar el algoritmo de transformación diseñado en lenguaje C++, el cual tomará como entrada el modelo exportado desde FreeCAD en formato .off y su salida o archivo generado será la creación de un nuevo modelo en formato **.m2d** (Malla de superficie, ver Capítulo 3), formato de entrada aceptado que recibirá la herramienta de visualización en tres dimensiones. Para poder visualizar el entorno en la herramienta de visualización será necesario aplicar todos estos pasos y en ese orden y sin omisión de alguno de estos pasos descritos previamente.

La solución propuesta ofrece una herramienta de visualización de un modelo o entorno previamente diseñado en AutoCAD. Esta herramienta acepta o lee archivos en formato **.m2d** (malla de datos geométricos del modelo, triángulos y vértices) y lo lee/interpreta para mostrar por pantalla el modelo diseñado en primera instancia por AutoCAD y poder realizar la tarea de apoyo al módulo externo optimizador en la labor del despliegue de redes de sensores inalámbricas industriales.

Es así como se busca automatizar los procesos que hoy en día se hacen de forma manual. También proporcionar a los usuarios un acceso al sistema, ya que para usar el sistema solo se necesita un computador convencional sin grandes características de hardware.

1.4. Objetivos general

Desarrollar una herramienta de software que permita a un planificador de proyectos de implementación de redes de sensores inalámbricos, levantar modelos de ambientes industriales tridimensionales (3D), definir puntos de sensaje y, con ayuda del módulo optimizador externo, mostrar gráficamente opciones de despliegue de redes de sensores inalámbricos que permitan asegurar la comunicación de los diferentes elementos de medición.

1.5. Objetivos específicos

Los objetivos específicos que contribuirán a desarrollar el *objetivo general* del trabajo son los siguientes:

- Desarrollar una interfaz que permita la visualización e interacción tridimensional de ambientes industriales utilizando como entrada modelos diseñados previamente en AutoCAD.

- Determinar a través de esta herramienta gráfica los puntos más adecuados para la colocación de sensores de medición inalámbrica. Con lo que se obtendrían algunos beneficios como:

  - Realizar sensado en zonas donde establecer una comunicación inalámbrica es muy difícil o imposible.

- Diseñar una aplicación que se adapte a la investigación y trabajos que se están realizando en la universidad, referente a este tema.

- Realizar las pruebas necesarias para asegurar el correcto funcionamiento de la aplicación.

1.6. Aporte del proyecto

Con esta herramienta gráfica, los resultados de la investigación relacionada podrían ser mostrados a empresas, posibilitando la transferencia tecnológica, es decir la transferencia de habilidades, conocimiento y métodos de fabricación entre la universidad y empresas o incluso con otras universidades para así favorecer y explotar aún más esta tecnología, agregándole más valor, aplicaciones o servicios.

Aportes:

- Incorporación de una interfaz de usuario para el uso de esta aplicación y visualización de espacios tridimensionales, a partir de la adaptación de un visualizador de mallas 3D de superficie.
- Apoyo gráfico para un usuario final.
- Facilitar la correcta colocación de estos sensores.
- Posibilitar la transferencia tecnológica.
- Disminuir riesgo humano en la labor del despliegue de redes de sensores inalámbricos.
- Obtener mayor beneficios de sensado al utilizar una posición óptima en el despliegue de redes de sensores inalámbricos industriales.

1.7. Organización

Este informe desarrolla los siguientes capítulos:

Capítulo 1: Introducción. Se introduce el planteamiento de la investigación, los objetivos generales y específicos, ademas de los aportes del proyecto.

Capítulo 2: Estado del Arte. Se aborda la investigación sobre redes de sensores inalámbricas industriales y el problema de despliegue de ISWN.

Capítulo 3: Mallas geométricas de superficie. Se aborda el tema de las mallas geométricas de superficies, estructura de formatos .m2d y nomenclatura.

Capítulo 4: Refinamiento de Mallas. Se describe lo qué es un refinamiento de mallas, y se define en qué consiste la bisección de un triángulo mediante el algoritmo de refinamiento Lepp-bisection. También se define y explica los algoritmos utilizados para del refinamiento de mallas, Leep-bisección y Leep-Delaunay.

Capítulo 5: Ambiente de Ingeniería de Software. Se detalla la metodología de desarrollo, así como también algunas técnicas y notaciones utilizadas.

Capítulo 7: Diseño del software. Se explica el diagrama de clases del Software.

Capítulo 6: Análisis. Se hace un análisis del software mediante diagrama de casos de uso, describiendo cada una de las interacciones del usuario con el software.

Capítulo 8: Pruebas de funcionalidad del software. Se realizan distintas pruebas de funcionalidad del software.

Capítulo 9: Conclusiones. Se realizan las conclusiones del desarrollo del software.

Capítulo 2

Estado del Arte

Dentro de este capítulo se describe la solución propuesta, las bases por las que se rige, es decir, el estado del arte tanto de las redes de sensores inálambricas como de la programación en tres dimensiones (3D).

A modo de introducción se detalla lo que son las redes de sensores inálambricas y como se están utilizando. Finalmente se hace énfasis a la programación en tres dimensiones y las técnicas utilizadas para esta.

2.1. Redes de sensores inalámbricos

Las Redes de Sensores Inalámbricos (WSN), se constituyen por un gran número de dispositivos, con limitaciones de potencia y rendimiento, conocidos como nodos sensores, donde cada nodo está conectado a uno o a varios sensores. Estos dispositivos permiten realizar tareas de sensado, cómputo y comunicación. Los datos obtenidos por los nodos sensores se envían a un dispositivo coordinador, llamado "puerta de enlace"(en inglés denominado gateway o sink node), el cual envía estos datos mediante Internet, a sistemas remotos de procesamiento, visualización, análisis y almacenamiento [8].

Las WSN(Wireless sensor networks) han tenido una amplia gama de aplicaciones y una atención significativa durante la última década, debido a su naturaleza adaptable a diversos entornos y a los rápidos avances tecnológicos, como por ejemplo el desarrollo de nuevas aplicaciones y la combinación con otras tecnologías. Estas redes pueden operar

independientemente en lugares de difícil acceso, donde la presencia humana es arriesgada o incluso imposible. Es por eso que cada vez van surgiendo nuevos desafíos, que apuntan a mejorar el rendimiento de dichas redes y así aumentar el uso con el que hoy día cuentan estas redes, por ejemplo, monitoreo ambiental, detección de productos químicos, servicios de atención de la salud, respuesta de emergencia, misiones de vigilancia, movimientos de vehículos, entre otros, y ocuparse o atender de las diferentes particularidades como el enrutamiento, la tolerancia a fallos y la eficiencia energética que es esencial para la eficacia de esta red [28].

2.1.1. Características de los Nodos Sensores

Una de las restricciones más importantes en los nodos sensores es el bajo requerimiento de consumo de energía. "Los nodos sensores se caracterizan por ser de pequeño tamaño y de bajo costo. Cada nodo sensor típicamente está conformado por una unidad de sensado, un transceptor de radio con una antena interna o una conexión a una antena externa, un microcontrolador, un circuito electrónico para la interfaz con los sensores y una fuente de energía, usualmente una batería"[23].

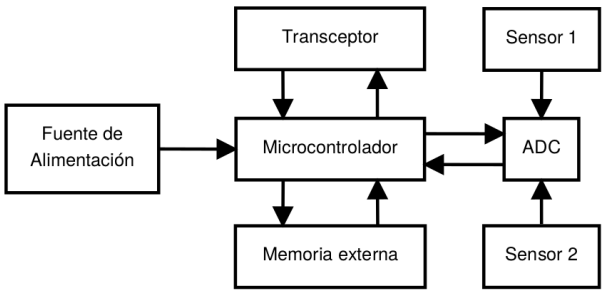


Figura 2.1: Arquitectura básica de un nodo sensor

Los nodos sensores llevan fuentes de energía limitadas generalmente irreemplazables [2].

2.1.2. Aplicaciones

Las WSN (Wireless sensor networks), pueden contar con distintos tipos de sensores, como sísmico, de muestreo magnético, térmico, visual, infrarrojo, acústico, y de radar que son capaces de monitorear una amplia variedad de condiciones ambientales, incluso se desempeñan en el ámbito militar, el hogar, la salud, entre otros. Esto es posible porque cuentan con una alta gama de aplicaciones debido a su naturaleza adaptable a diversos entornos [2].

Para este trabajo de título en particular, la aplicación de WSN, estará orientada netamente al ámbito industrial.

2.2. Redes de sensores inalámbricas industriales

"Las redes de sensores inalámbricas industriales corresponden a un conjunto de nodos autónomos que poseen la capacidad de comunicarse entre ellos para entregar datos obtenidos de sensores incorporados a ellos, a un elemento controlador central denominado recipiente"[5].

Las redes de sensores inalámbricos industriales (IWSN, del ingles *Industrial Wireless Sensor Networks*), son un derivado de las WSN (Wireless sensor networks). Las redes de sensores inlámbricos industriales (IWSN), incorporan redes de sensores inalámbricos con sistemas industriales inteligentes proporcionando muchas ventajas sobre las aplicaciones industriales existentes, tales como la comunicación inalálmbrica, bajo costo, rápida implementación, una autoorganización, control inteligente, y una mucho mejor capacidad de procesamiento.

Con la proliferación de redes de sensores inalámbricos en aplicaciones industriales, las tecnologías IWSN prometen desempeñar un papel importante en el desarrollo de sistemas industriales más confiables, eficientes, estables, flexibles y más centrados en la aplicación [15].

2.2.1. Aplicaciones IWSN

Las WSN dentro en entornos industriales son utilizadas para monitorear diferentes aspectos, y basado en las condiciones de la producción industrial, estas se pueden clasificar en 3 grupos:

1. **Sensado ambiental:** este grupo generalmente representa el campo más ancho de las aplicaciones WSN de hoy en día. Las aplicaciones de IWSN para la detección ambiental cubren el problema del aire, la contaminación del agua, el control de la contaminación del material de producción y la operación en entornos peligrosos, como la detección de incendios, inundaciones o deslizamientos de tierra.
2. **Monitoreo de condiciones:** este grupo generalmente cubre los problemas de la estructura (salud de la estructura, construcciones, puentes, rutas de suministro, etc.) y el monitoreo de la condición de las maquinas, incluyendo posible automatización industrial.
3. **Automatización de procesos:** Este último grupo de aplicaciones proporciona a los usuarios la información sobre los recursos para la producción y la prestación del servicio, incluidos el material, el stock actual y el estado de los suministros, así como la mano de obra incluida en el proceso industrial.

En general, las aplicaciones para Redes de sensores inalámbricas (IWSN) deben ser robustas, asegurando un nivel mínimo aceptable confiabilidad y calidad en la recuperación de datos, debido a que una sola falla en la red de sensores inalámbricos desplegada para la automatización industrial puede ocasionar pérdidas financieras para la industria e inclusive poner a las personas en peligro [11, 13].

Las comunicaciones en IWSN están basadas en las tecnologías de Redes inalambricas de área personal (WPAN, del ingles *Wireless Personal Area Network*) cuyas soluciones son variaciones desarrolladas por el grupo de trabajo IEEE 802.15 y otras comisiones de estandarización como ISA (International Society of Automation) e IEC (International Electrotechnical Commission). Estas tecnologías han presentado buenos resultados para las IWSN, con especial énfasis en los estándares WirelessHART e ISA100.11a, específicamente creados para este tipo de entorno. Este rico escenario ha permitido la creación

de nuevas tecnologías de sensores inalámbricos para entornos industriales, con diferentes soluciones disponibles en el mercado [3].

En la actualidad, las necesidades y posibilidades de los sensores inalámbricos (WNS) en todos los sectores industriales y sucursales están muy extendidos y el ritmo del desarrollo industrial empuja y fomenta aún más el desarrollo de redes industriales de sensores inalámbricos [13].

2.3. Despliegue de Nodos

El despliegue y la instalación de las WSN, son una de las tareas más importantes y que se complican aún más en aplicaciones industriales. Hay varios aspectos a considerar en el diseño de las IWSN, siendo unas de las más importantes el impacto que tendrá el medio ambiente sobre las redes. Hay que tener en cuenta que la propagación de las ondas de radio frecuencia tendrán un comportamiento distinto dependiendo al entorno en donde se hayan desplegado los sensores. Un ejemplo con dos escenarios de despliegue sería:

1. **Despliegue al aire libre:** la propagación de radiofrecuencia puede tener comportamiento casi omnidireccional.
2. **Despliegue en interiores:** el comportamiento de radiofrecuencia podría variar, debido al rebote en paredes, maquinaria o ser obstruida por paredes.

El despliegue de una WSN afecta a casi todas sus métricas de rendimiento, como la conectividad entre los sensores, la cobertura efectiva de la red y la duración de la red. En general, los métodos de implementación de WSN se dividen en dos categorías: despliegue planificado y despliegue aleatorio [10].

- **Despliegue planificado:** se define, mediante una decisión selectiva, la ubicación de los sensores para optimizar uno o más objetivos de la WSN. Si bien se puede lograr una cobertura óptima, puede ser difícil de realizar en algunas áreas, debido a la dificultad del acceso humano.

- **Despliegue aleatorio:** el despliegue aleatorio, en ocasiones es la única opción factible. Los sensores son por lo general dispersos. Esta opción es útil para zonas devastadas, etc. Lógicamente su rendimiento es subóptimo (No es el mejor o el más satisfactorio).

Los sensores generalmente se colocarán en lugares donde se desea algún tipo de monitoreo, de acuerdo con la naturaleza de la información a ser monitoreada y la unidades de sensores empleadas (por ejemplo, sensores de cámaras de datos de una manera diferente de sensores de temperatura). Como se espera que las WSN tradicionales sean sistemas autónomos autoorganizados, los protocolos de comunicación ajustan dinámicamente su funcionamiento para componer una topología de comunicación, lo que significa, en la práctica, que la posición para detectar funciones es más relevante que la posición para la comunicación, ya que es probable que ocurran transmisiones incluso en posiciones no óptimas de los transceptores de los sensores. De esta manera, ya que una o más unidades de detección y el hardware del transceptor generalmente están integrados en un diminuto sensor de mote (alias) único, la mayoría de los enfoques de implementación de sensores han intentado optimizar la cobertura de detección y la comunicación de transmisión por igual [9, 11].

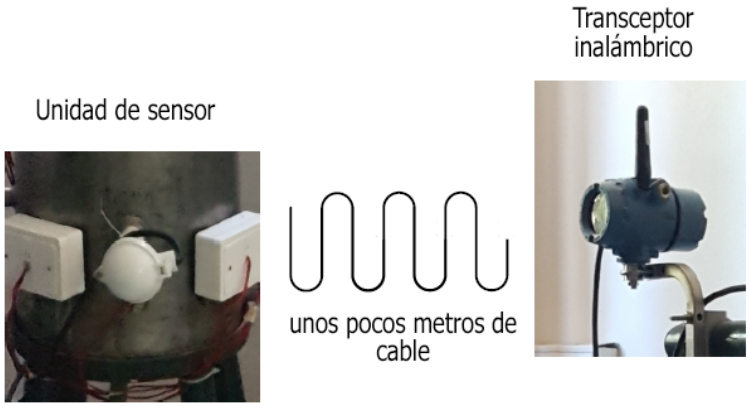


Figura 2.2: Ejemplo de un nodo de sensor con el transceptor conectado a través de un cable.[1]

[1]Nodo sensor conectado con un cable: [11]

En la Figura 2.2 se observa un ejemplo de un nodo sensor conectado a través de unos metros de cable al transceptor inalámbrico.

2.4. Problemas del despliegue de Nodos

La implementación del sensor en la IWSN, puede ser considerado como el posicionamiento de los nodos del sensor con respecto al instrumento sensor principal llamado "sensor" y una unidad transceptora, denominada como "transceptor" [12].

Al elegir la mejor ubicación para posicionar el sensor, habrá una limitación para posicionar el transceptor, que será el tamaño máximo del cable que conecta estos dos dispositivos. Es por eso que el posicionamiento del transceptor de los sensores tendrá que ver con las restricciones de comunicación, como la potencia de transmisión de comunicación y la presencia de obstáculos que puedan existir, además, la posición de los sensores correspondientes. Este escenario crea un problema de implementación específico que requiere un tratamiento adecuado, abriendo nuevos problemas de investigación para abordar, como por ejemplo el modelado de escenarios para el posicionamiento de las IWSN [11].

2.5. Conectividad y Cobertura en una WSN

En una WSN, cada sensor tiene un rango de comunicación limitado, y un rango de sensado limitado. La conectividad está relacionada con la comunicación entre los sensores, por lo que una conectividad eficaz, significa que todos los sensores de la red son capaces de transmitir sus datos evitando el aislamiento de sensores [23].

Para una IWSN, la topología y la conectividad de la WSN pueden variar, debido a fallas de enlaces y nodos de sensores. Además, los sensores también podrían estar sujetos a interferencia de RF (del ingles, *Radio frecuency*), ambientes altamente cáusticos[2] o corrosivos, altos niveles de humedad, vibraciones, suciedad como polvo u otras condiciones que retan o desafían el rendimiento de la WSN. Estas complejas condiciones ambientales producidas en la industria y la topología de la red pueden ocasionar el mal funcionamiento de una porción de nodos sensores desplegados en el entorno [12].

[2]Cáustico: Que quema y destruye los tejidos orgánicos.

El problema de cobertura es el caso más importante y básico en el despliegue de una WSN. Existen 3 tipos de cobertura [14]:

- **Cobertura de área:** en esta cobertura el problema más importante es el tipo de área de cobertura. El objetivo principal es cubrir y supervisar el medio ambiente por completo.
- **Cobertura de punto:** Apunta a cubir un punto en especifico del entorno. Se puede decir que este metodo es un subconjuto de la cobertura de área.
- **Cobertura de barrera:** el área total no esta cubierta por los sensores en la cobertura de barrera. Este modelo es adecuado para aplicaciones de inserción (por ejemplo, detección de intrusos).

2.6. Problema del Modelado de Entorno Industrial

El modelado (técnica que sirve de ayuda para visualizar el sistema a construir) del entorno industrial es un problema complejo. Se debe tener en cuenta diferentes cuestiones al momento de modelar entornos industriales para IWSN. Existen problemas a considerar que se describen en 3 niveles:

- Entorno.
- Nodo.
- Red.

Todos estos problemas se pueden tener en cuenta al diseñar métodos de optimización [11].

Primero que nada, un modelo ambiental es obligatorio. Para una IWSN el entorno es la fábrica. “Las fábricas se pueden modelar como espacios tridimensionales, donde encontramos diferentes tipos de elementos. Los elementos están relacionados con la infraestructura, maquinaria, además, la naturaleza del terreno, la geografía, la naturaleza (árboles, etc.). Diferentes elementos pueden tener diferentes propiedades y, por supuesto, son factores que pueden afectar problemas de red tales como propagación, etc. Por supuesto, para proble-

mas informáticos, un escenario tan complejo debe simplificarse y existen varios modelos de propagación que consideran la propagación en entornos con objetos (como los modelos de trazado de rayos). Otro problema a tener en cuenta es que los nodos, cables y otros instrumentos tienen diferentes tipos de lugares adecuados para ser instalados, y hay algunos lugares que son imposibles o inadecuados"[11].

En segundo lugar, se puede considerar diferentes parámetros de nodos y sensores inalámbricos. Los diferentes nodos deben tener distintos parámetros configurables, como la salida de la potencia de la señal inálambrica. Un modelo de propagación de señales inalámbricas en el entorno es un aspecto básico y útil que se debe considerar, ya que un requisito básico para IWSN es tener nodos alcanzables en la práctica, es decir, que podemos obtener cualquier nodo a través de repetidores y que la topología permite la conectividad a una o más puertas de enlace [11].

2.7. Modelado 3D

Las siguientes áreas son parte principal del campo de los gráficos por computadora [16]:

- **Modelo:** trata con la especificación de las propiedades de forma y apariencia de una manera que pueda almacenarse en la computadora.

- **Interpretación/Representación:** término heredado del arte y trata de las creaciones de imágenes sombreadas de modelos.

- **Animación:** técnica para crear una ilusión de movimiento a través de secuencias de imágenes. Aquí, se usan modelos y representaciones.

Hay muchas otras áreas que involucran gráficos de computadora, y no todos están de acuerdo con que sean gráficos centrales en cuestión. Tales áreas relacionadas incluyen las siguientes:

- **Interacción del usuario:** se ocupa de la interfaz entre los dispositivos de entrada,

como por ejemplo ratones (mouse), las aplicaciones y los comentarios al usuario sobre imágenes y otros comentarios.

- **Visualización:** intenta dar a los usuarios una idea a través de la visualización.

- **Procesamiento de imágenes:** se ocupa de la manipulación de imágenes en 2D y se usa tanto en el campo de los gráficos como en la visión.

En base a estos puntos, una de los principales consumidores de tecnología de gráficos por computadora son las herramientas CAD:

- **CAD:** es sinónimo de diseño asistido por computadora y fabricación asistida por computadora. Estos campos utilizan la tecnología informática para diseñar piezas y productos en la computadora y luego, utilizando estos diseños virtuales, para guiar el proceso de fabricación. Por ejemplo, muchas piezas mecánicas ahora se diseñan en un paquete de modelado de computadora 3D y luego se producen automáticamente en una computadora. dispositivo de fresado controlado por computadora. Al igual que el modelado de entornos gráficos, como los industriales[16].

2.8. AutoCAD para Modelado 3D

Si bien es cierto que muchos consideran que AutoCAD (software de diseño asistido por computadora utilizado para dibujo 2D y modelado 3D) es un software complejo y severo, con una geometría estricta, tiene a su favor que para los que estén desde muchos años utilizándolo, puedan encontrar en él una herramienta poderosa de diseño y modelado en *2D* y *3D*, con una geometría digital excelente que evidentemente requiere del usuario un sólido conocimiento de la lógica del software para poder interactuar con el programa y así obtener su máximo aprovechamiento.

En el campo de la enseñanza del diseño en ingeniería y arquitectura, es una herramienta fundamental [17].

El uso del modelado *3D* es muy complejo en entornos industriales. La mayoría de las técnicas existentes para el modelado industrial, es mediante complejas técnicas activas de triangulación láser. En entornos industriales, la iluminación variable, aparecen y desaparecen, y las superficies irregulares son comunes. Estos factores afectan directamente al trabajo realizado para modelar mediante herramientas computacionales[27].

2.9. Otros software para Modelado 3D

Actualmente existen distintas herramientas con las que se puede llevar a cabo la tarea de modelar espacios tridimensionales, dado que los expertos de la automatización coinciden en que la visión *3D* se encuentra al comienzo de un desarrollo vertiginoso. El futuro consistirá en un manejo sencillo y una alta flexibilidad que puede ofrecer el modelado y la visión *3D.*

Existen una varios software que permiten el modelado *3D*, pero las herramientas **CAD** -acrónimo que se refiere a *computer aided design* o diseño asistido por computadora- son unas de las más conocidas, como por ejemplo:

- **AutoCAD:** uno de los software impertérritos en el mundo del **CAD**, una de las opciones más atractivas para el mundo profesional. Poderosa herramienta, pero de paga.
- **FreeCAD:** completamente opuesto a AutoCAD, dado que es *open source* (código abierto) y gratuito, no tan poderoso como AutoCAD, pero con la ventada de ser multiplataforma.
- **LibreCAD:** otra opción **CAD** *open source* y gratuita, multiplataforma y con mucho contenido, perfecto para los primeros pasos en el mundo **CAD**.
- **Blender:** conocido por ser un software de modelado *3D* pero que también tiene parte de diseño **CAD**, Blender es gratuito y muy utilizado por profesionales.
- **SolidWorks:** ligado al sector profesional y muy asentado en las ingenierías, pero abierto al mundo educativo con las Student Edition (edición para estudiantes) gratuitas.
- **123D Design:** completamente online y enfocado a la creación de productos tridimensionales, con un toque de diseño más que técnico.

Capítulo 3

Mallas geométricas de superficie

3.1. Definición

Una malla de superficie de triángulos es una colección de triángulos y vértices que aproximan un dominio Ω, siendo este dominio algún objeto o entorno de la realidad.

Aunque el campo de aplicación de la generación automática de una malla triangular, ha sido tradicionalmente la obtención de modelos digitales de elevaciones de terrenos, su aplicación es mucho más amplia. Cualquier variable espacial relacionada con una cierta topología, es susceptible de ser modelizada como una superficie tridimensional.

3.2. Archivos con extensión .m2d

Para este proyecto, se utilizan archivos con extensión **.m2d** para la representación de triángulos en tres dimensiones, ya que estos archivos están diseñados de tal forma que contienen toda la información necesaria para poder modelar una tiangulación en tres dimensiones en el espacio euclidiano. Dentro de este archivo, se encuentra la siguiente información con su correspondiente nomenclatura:

#: representa un comentario o reseña, una línea que no es necesaria para la representación de un triángulo.

v: carácter que indica que el contenido de esa línea es un vértice.

t: carácter que indica que el contenido de esa línea es un triángulo.
n: carácter que indica que el contenido de esa línea es un triángulo vecino.

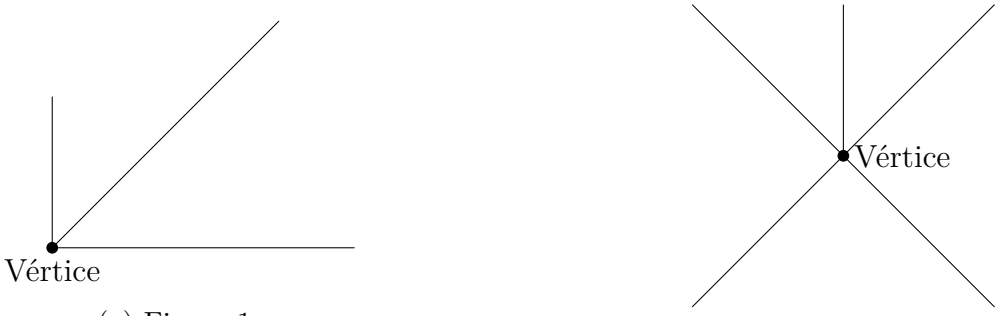
3.3. Elementos topológicos geométricos

Para poder comprender a cabalidad un archivo con extensión .m2d, primero es necesario conocer la siguiente información:

- **Vértice**
- **Arista**
- **Triángulo**

3.3.1. Vértice

La palabra vértice procede etimológicamente del latín "vertex" y su significado es "aquello que gira". En geometría, es el punto que marca la unión entre los segmentos que originan un ángulo o un punto donde dos o más líneas se encuentran.



(a) Figura 1

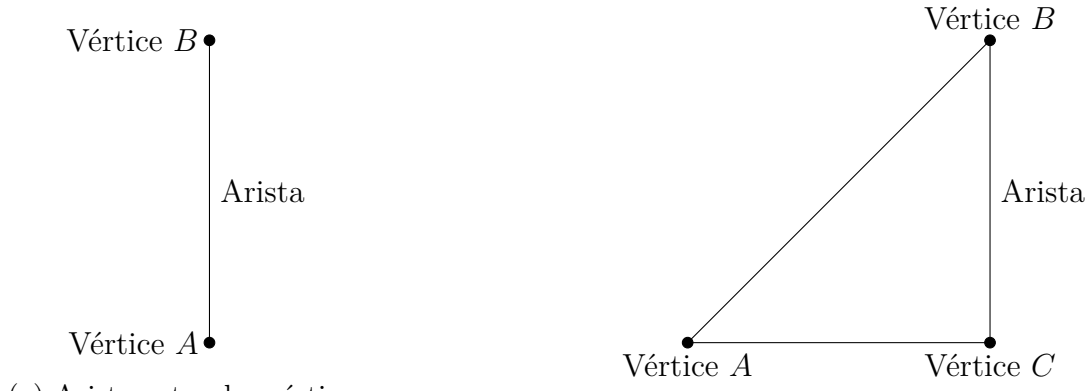
(b) Figura 2

Figura 3.1: Ejemplo de Vértices

En la Figura 3.1 se puede observar la representación de vértices (3.1a, 3.1b), en donde se marca la unión entre los segmentos que originan un ángulo.

3.3.2. Arista

Arista, del latín "arista" que en geometría es la línea resultante del cruce de dos superficies o planos. Las aristas también son los segmentos de una recta que marcan el límite de los lados de una figura plana. En la geometría sólida se le llama arista al segmento de línea donde se encuentran dos caras. Una de las cualidades de una arista, es que tienen una longitud finita y estas están delimitadas por 2 puntos o vértices (ver figura3.2a).



(a) Arista entre dos vértices

(b) Aritas de un triángulo

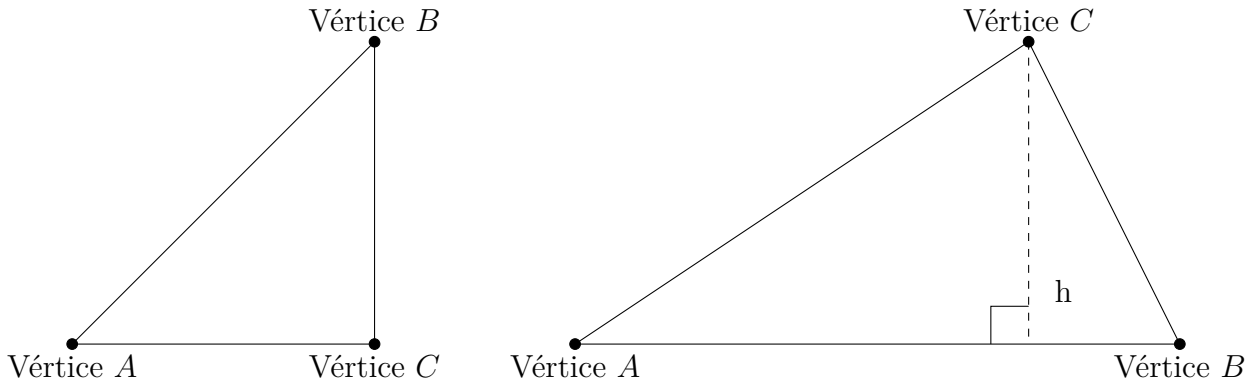
Figura 3.2: Ejemplo de Arista

En la figura 3.2, se observan dos ejemplos de aristas, la figura 3.2a representa una arista formado por 2 puntos o vértices y en la figura 3.2b, una arista de un triángulo.

3.3.3. Triángulo

Con origen en el latín "triangulus", la palabra triángulo se utiliza para identificar un polígono compuesto por 3 lados llamados aristas o por tres puntos que dan origen a tres vértices y tres ángulos internos.

De acuerdo a la longitud de sus lados o según la medida de sus ángulos un triángulo puede ser clasificado, además de esto, uno de sus lados o aristas corresponden a la base del triángulo y la altura de un triángulo es el segmento perpendicular a un lado que va desde el vértice opuesto a este lado.



(a) Triángulo con vértices (b) Triángulo con vértices y altura

Figura 3.3: Ejemplo de Triángulo

En la figura 3.3 se observa un ejemplo de triángulos, indicando sus vértices y su altura, en donde se identifica que mediante 3 puntos se forma el polígono.

3.4. Estructura extensión .m2d

Un archivo con extensión .m2d, está diseñado de tal forma que la información que contiene en su interior es la precisa y necesaria para comprender cómo se estructura la tiangulación. En su interior, se encuentran la cantidad de vértices representados por la letra **'v'**, también se encuentran todos los triángulos generados mediante la letra **'t'** y también todos los triángulos vecinos generados representados por la letra **'n'**.

Para realizar la representación gráfica de los datos contenidos en este archivo, es necesaria la información contenida en él, como los vértices, triángulos, y triángulos vecinos. Estos datos están contenidos de la siguiente manera:

Vértices

```
1 # Vertices
2 v 1 3.8929 2.9199 0
3 v 2 0.497752 2.9199 0
4 v 3 0.497752 0.278143 0
5 v 4 3.8929 0.278143 0
```

- **Vértice:** carácter **'v'** encontrado en la primera columna indica el tipo, en este caso, un vértice, seguido de un identificador único, de tipo entero para cada vértice, que indica que n° de vértice, para finalizar con las últimas tres columnas que indican las coordenadas (x,y,z) de ese vértice en el espacio euclidiano.

Triángulos

```
1 # Triangles
2 t 1 1 2 3
3 t 2 1 3 4
4 t 3 5 4 3
5 t 4 5 6 7
```

- **Triángulos:** una vez terminada la lista de vértices, se encuentra el carácter **'t'** en la primera columna que indica el tipo, en este caso, un triángulo, seguido de un identificador único de tipo entero para cada triángulo, qué indica el n° de triángulo, para finalizar con las últimas tres columnas que indican los identificadores de los puntos o vértices ubicados en el espacio euclidiano por los cuales está formado ese triángulo.

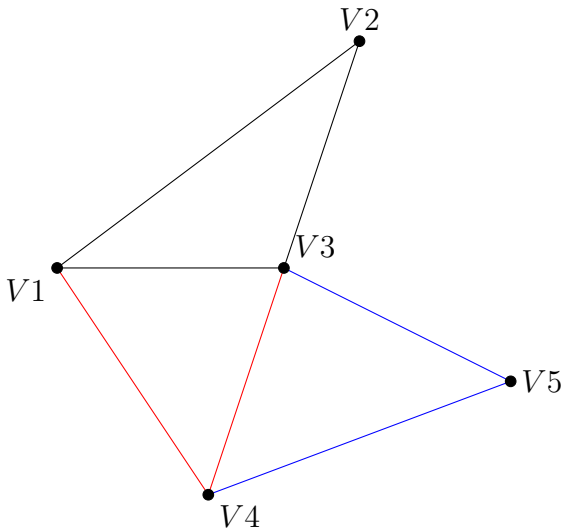


Figura 3.4: Triángulos formados por distintos vértices

Triángulos vecinos

```
1 # Neighbors
2 n 1 3788 2229 4536
3 n 2 2231 4538 3789
4 n 3 3791 0 0
5 n 4 1473 0 2777
6 n 5 3795 0 0
```

- **Triángulos vecinos:** una vez terminada la lista de triángulos, se encuentra el carácter **'n'** en la primera columna que indica el tipo, en este caso, un triángulo vecino, seguido de un identificador único de tipo entero para cada triángulo, para finalizar con las últimas tres columnas que indican identificadores de los triángulos vecinos.

Capítulo 4

Refinamiento de Mallas

Para la realización de este proyecto, será necesario aplicar un refinamiento de una malla de superficie mencionada anteriormente (ver capítulo 2). Para realizar un refinamiento de mallas mediante la bisección de un triangulo existen algoritmos como Lepp-bisection y Leep-Delaunay.

4.1. Bisección de Triángulos

En geometría, el término bisección hace referencia a la división de un elemento en dos partes iguales o congruentes, generalmente mediante una línea recta, denominada bisector.

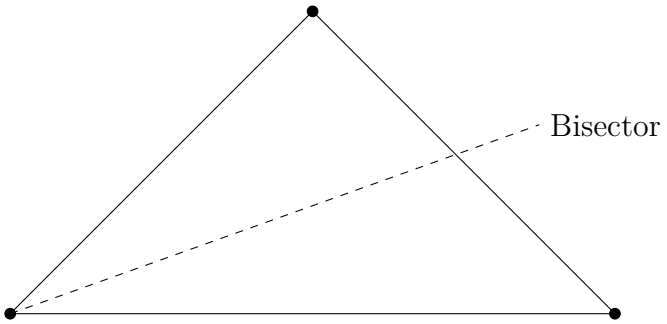


Figura 4.1: Bisección Triángulo

4.2. Refinamiento mediante la Bisección de triángulos

Esto se define como la lista ordenada (finita) de triángulos vecinos sucesivos que tienen el borde más largo mayor que el borde más largo del triángulo anterior en la ruta. Este ideal se utiliza para introducir dos tipos de algoritmos (que hacen uso de una estrategia de inserción de puntos hacia atrás de mayor longitud)[19]:

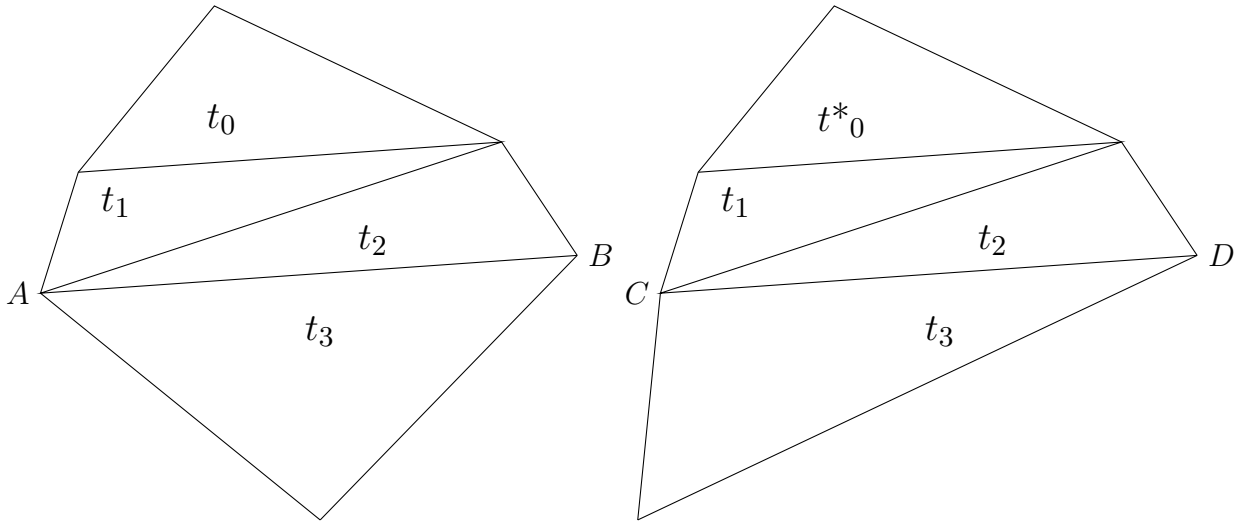
1. Un algoritmo puro de refinamiento de borde más largo hacia atrás que produce las mismas triangulaciones que los algoritmos anteriores de mayor longitud en un manera eficiente, directa y fácil de implementar [19].

2. Un nuevo algoritmo de mejora de retroceso de borde más largo hacia atrás para triangulaciones de Delaunay, adecuado para tratar (de manera confiable, robusta y efectiva) los tres aspectos relacionados importantes del problema de generación de malla (triangular): refinamiento de malla, mejora de malla y generación automática de triangulación de superficie y volumen de buena calidad de geometrías generales, incluidos pequeños detalles [19].

Para la realización de este proyecto se utilizó un algoritmo de refinamiento de triangulaciones basado en la bisección de triangulos [18], por la arista más larga, la cual conserva la calidad de la triangularización de entrada. Esto debido a la bisección de triángulos por la arista más larga y a la estrategia de propagación por la arista mas larga [18]. Es necesario aclarar que para este proyecto solo se utiliza el método de bisección por arista más larga.

4.3. Lepp: Longest edge propagation path

Definición 1. Para cualquier triángulo t_0 a refinar, el **Lepp** de t_0 (Lepp(t_0)) será la lista ordenada de todos los triángulos t_0, t_1, ... t_{n-1}, t_n, donde el triángulo t_i, para i = 1,2,...,n es adyacente al triángulo t_{i-1}, por el lado (arista) más largo de t_{i-1}. El **Lepp**(t_0) es una secuencia finita, y termina con uno o dos triángulos terminales [18–20].



(a) (b)

Figura 4.2: **(a)** AB es una arista terminal interior compartido por triángulos terminales (t_2, t_3) asociados a Lepp (t_0) = t_0, t_1, t_2, t_3; **(b)** CD es un borde de límite de terminal con un triángulo terminal único t_3 asociado a Lepp (t*0) = t * 0, t_1, t_2, t_3. [21]

4.3.1. Arista terminal

Una arista terminal o mayor, es compartida por dos triángulos, que al compartir esta arista se denominan triángulos terminales, los cuales a su ves son vecinos. Una arista terminal o mayor es una arista interior de una triangulación [24].

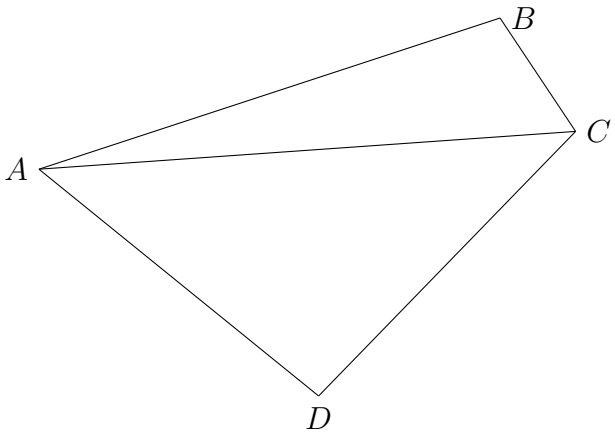


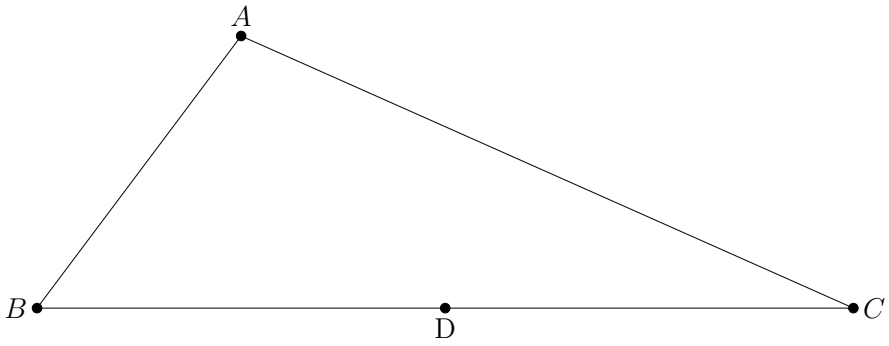
Figura 4.3: Arista terminal $\overline{AC}$ de los triángulos terminales $\triangle ABC$, $\triangle ADC$ [24, 26].

4.4. Bisección de un triángulo por arista más larga

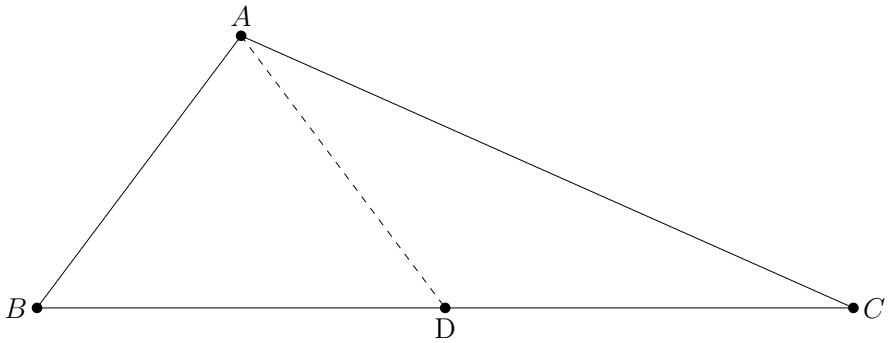
Sea $\triangle ABC$ un triángulo con vértices en A, B y C. El procedimiento de bisección de $\triangle ABC$ se define de la siguiente manera [24]:

1. Formamos dos triángulos desde $\triangle ABC$ ubicando el punto medio del lado más largo de $\triangle ABC$ y dibujando un segmento de línea recta desde este punto medio hasta el vértice de $\triangle ABC$ que está opuesto al lado más largo. (Si hay más de un lado de mayor longitud, bisectamos a cualquiera de ellos.)

Por ejemplo, si $\overline{BC}$ es el lado (arista) más larga de $\triangle ABC$, establecemos $D = \dfrac{B+C}{2}$ para formar dos nuevos triángulos $\triangle ABD$ y $\triangle ADC$.



(a) Definición punto medio D según arista más larga $\overline{BC}$.



(b) Bisección $\triangle ABC$

Figura 4.4: Bisección de triángulo por arista más larga [24].

En la figura 4.4 se observa la bisección del triángulo $\triangle ABC$ a través de su arista más larga. Se identifica la arista más larga $\overline{BC}$ y se inserta un nuevo punto D. Para la obtención del punto D se establece $D = \frac{B+C}{2}$. Esto origina una nueva arista $\overline{AD}$ formada por el punto D y el vértice opuesto A lo que genera la formación de dos nuevos triángulos, $\triangle ABD$ y $\triangle ADC$.

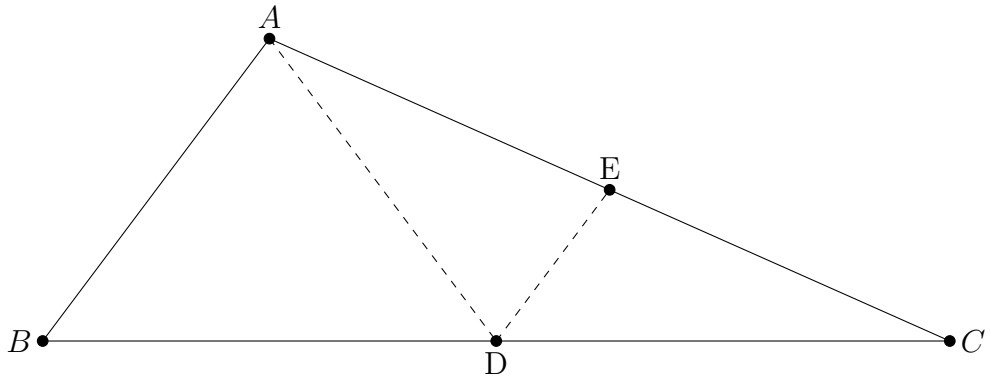


Figura 4.5: Nueva bisección a partir de bisección anterior

En la figura 4.5 se realiza una nueva bisección por la arista más larga $\overline{AC}$ formada por la bisección anterior. Se inserta un nuevo punto E que da origine a una nueva arista formada por el punto E y el vértice opuesto D y se obtienen nuevamente dos nuevos triangulos, $\triangle EDC$, $\triangle ADE$.

4.5. Algortimo Lepp-Bisecction

Leep-Bisección es un algoritmo de refinamiento de triangulaciones por la arista más larga.

El algoritmo Lepp-bisection se puede describir simplemente de la siguiente manera: cada triangulo t se refina al encontrar Lepp(t), un par de triángulos terminales t_1, t_2 y el lado (arista) terminal asociado l. Luego, la bisección de arista más larga de t_1, t_2, se realiza por el punto medio de l. El proceso se repite hasta que t se destruye (refina) en la malla. A continuación se presenta una formulación eficiente del algoritmo en el que Lepp(t_0) no se vuelve a calcular repetidamente, pero se actualiza repetidamente a partir de la parte no modificada del Lepp anterior(t_0) [20, 22].

Algoritmo 1: Lepp-Bisection(t_0)

Data: t_0: Triángulo inicial para el refinamiento.

```
1 while t_0 no esté bisectado do
2 |   t_n ← Último triángulo de Leep(t_0);
3 |   if t_n está en el borde then
4 |   |   Bisectar t_0 por arista mas larga
5 |   else
6 |   |   Bisectar por arista mas larga el último par de triángulos t_{n-1} y t_n
7 |   end
8 end
```

Tabla 4.1: Algoritmo Lepp-Bisection

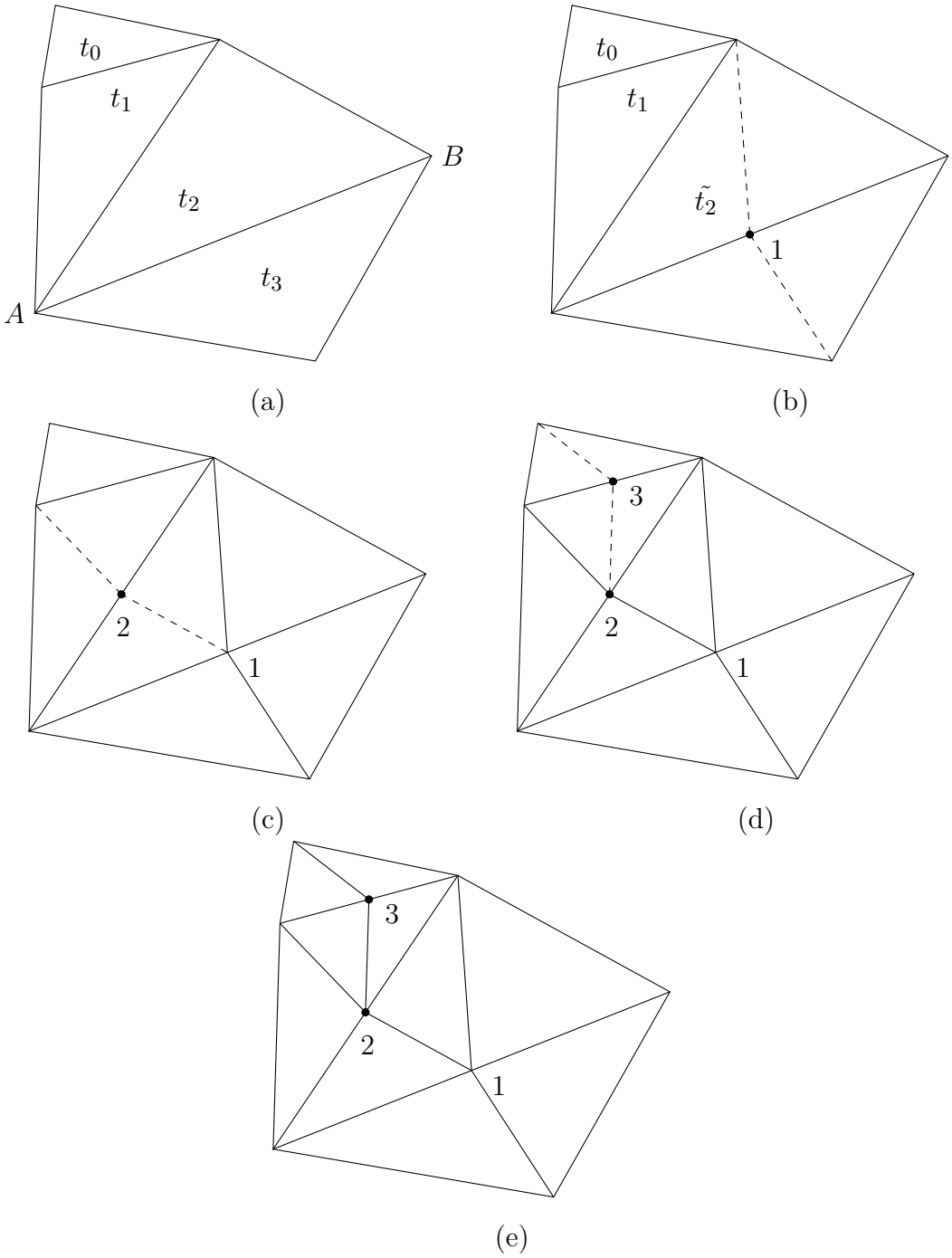


Figura 4.6: (a) Lepp(t_0) = $\{t_0, t_1, t_2, t_3\}$ y AB es arista terminal; (b) Para refinar el triángulo t, se agrega un primer vértice 1 mediante la bisección de los triángulos terminales que comparten AB; (c)(d)(e) Triangulaciones finales obtenidas para refinar t. [20–22]

Capítulo 5

Ambiente de Ingeniería de Software

5.1. Metodología de desarrollo

Para la realización de este proyecto se optó por la metodología incremental iterativa que consta de diversas etapas llamadas iteraciones. En cada una de estas iteraciones se realizará un proceso de trabajo similar para proporcionar un resultado completo sobre el producto final de manera de obtener todos los beneficios de un proyecto incremental.

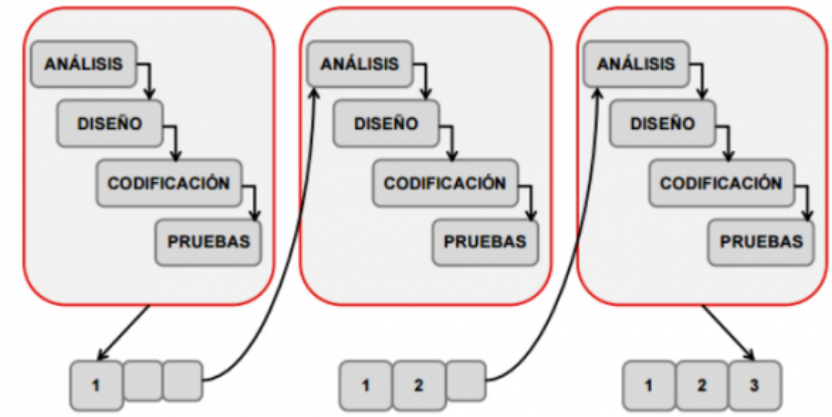


Figura 5.1: Metodología incremental[1]

[1]Metodología incremental: goo.gl/images/AWPD25

Gracias a esta metodología (Figura 5.1) de trabajo se obtienen beneficios, como la velocidad de desarrollo, la calidad y la toma de decisiones en cada iteración, pudiendo así decidir el nivel de prioridad de todos los requerimientos faltantes.

5.2. Técnicas y notaciones

- Estándares de documentación: Adaptación basada en IEEE Software requirements Specifications Std 830-1998.[2]
- ISO/IEC 9126: Tecnología de Información – Evaluación del producto de software.
- Requerimientos funcionales y no funcionales.
- Diagrama de casos de uso.

5.3. Herramientas de apoyo al desarrollo de software

- AutoCAD 2018 Versión del producto: O.49.0.0. Licencia estudiante Universidad del Bío-Bío.
- QT Creator
- Sublime Text 3
- FreeCAD
- Algoritmo de transformación (necesario para realizar la conversión de formato).
- Visor de mallas 3D.

5.3.1. Hardware utilizado

Para el desarrollo del software se utiliza un laptop o computador de escritorio con las siguientes características mínimas:

- Sistema operativo: Gnu/Linux Ubuntu 16.04 LTS x64.
- Procesador: Intel Core i3 2.3GHz 6.º generación.
- Ram: 8GB.
- Almacenamiento: HDD 250GB.

[2]Especificación de Requisitos según el estándar de IEEE 830 [en linea]: bit.ly/2HsKuHE

5.3.2. Planificación inicial del proyecto

A través de la Carta Gantt u otra herramienta de calendarización, indicando las actividades que serán llevadas a cabo en función de la metodología de desarrollo. Considerando actividades desarrolladas por el desarrollador.

5.4. Riesgos asociados al desarrollo del proyecto

Es el proceso de determinar los riesgos que pueden afectar al proyecto y documentar sus características[7].

Para poder gestionar los riesgos sobre los objetivos de un proyecto es evidente que primero hay que intentar detectarlos. Este proceso nos permitiría obtener los riesgos conocidos del proyecto que los que podemos gestionar. Los riesgos externos al proyecto no siempre son detectables [7].

Una importante tarea dentro de la gestión del proyecto es poder anticipar riesgos que puedan afectar al desarrollo de este como tal o en la calidad del software. Con esta anticipación es posible comprender los riesgos a los que se enfrenta el proceso de desarrollo del software para así poder generar requerimientos confiables y abordar dichos riesgos aplicando un plan de mitigación y contingencia dependiendo de cada caso.

5.4.1. Riesgos asociados al desarrollo del proyecto

Riesgos identificados	**Plan de mitigación y contingencia**
Subestimación o sobreestimación del tamaño del proyecto	Revaluar la estimación del proyecto en base a datos históricos del desarrollador.
Requerimientos mal definidos	Analizar el sistema en base a la problemática y replantear requerimientos.
Retraso en el diseño global y específico	Guiarse por carta gantt.
Desconocimiento de tecnologías	Buscar alguna otra opción de tecnología que aborde lo requerido o estudiar tecnología desconocida.
No realizar pruebas suficientes al software	Darle más énfasis y tiempo a la pruebas, para así lograr un software con menos probabilidad de fallos.
Falla en el equipo físico de desarrollo	Tener siempre respaldo del proyecto en alguna nube y algún otro equipo donde seguir desarrollando.
Cambios de requisitos	Trabajar con el cliente o con algún representante de la entidad codo a codo para que vea el trabajo y así analizar a tiempo si quiere algún cambio en el software.
El entorno de desarrollo no se integra con paquetes generados por AutoCAD.	Revisar tutorial del desarrollador oficial y visitar sitio web oficial de AutoCAD.

Tabla 5.1: Riesgo de proyecto

5.4.2. Riesgos asociados al software

Riesgos asociados	Plan de mitigación y contigencia
Resultados poco confiables	Seguimiento del software para verificar la calidad.
Acciones frente a fallas de software	Procedimiento de mantención y correctivos agiles.
Inconsistencia de datos de entrada	Procedimiento de abandono de la operación y revisión de archivos de entrada.
Resultados poco confiables	Seguimiento del software para verificar la calidad de los datos de salida obtenidos.

Tabla 5.2: Riesgo asociados al software

5.5. Definiciones, Siglas y Abreviaciones

Algoritmo: secuencia lógica de instrucciones que representan un modelo de solución para determinado tipo de problemas.

AutoCAD: Software de diseño asistido por computadora para dibujo *2D* y modelado *3D*.[3]

IEEE: corresponde a las siglas de (*Institute of Electrical and Electronics Engineers*) en español Instituto de Ingenieros Eléctricos y Electrónicos, una asociación técnico profesional mundial dedicada a la estandarización, entre otras cosas. Es la mayor asociación internacional sin ánimo de lucro formada por profesionales de las nuevas tecnologías, como ingenieros eléctricos, ingenieros en electrónica, científicos de la computación, ingenieros en informática, ingenieros en biomédica, ingenieros en telecomunicación e ingenieros en mecatrónica.

[3]AutoCAD[en-linea]: https://www.autodesk.com/

Formato .iges: formato neutral de datos que permite el intercambio digital de información entre sistemas de diseño asistido por computadora (CAD).

Formato .off: formato de archivo de definición de geometría que contiene la descripción de los polígonos(coordenadas x,y,z) que componen el objeto 3D.

Formato .m2d: formato de archivo informático que define la geometría que contiene la descripción de los polígonos, como triángulos y vértices.

FreeCAD: aplicación libre de diseño asistido por computadora en tres dimensiones, ingeniería asistida por computadora.

Interfaz: especificación de los atributos y operaciones asociados con un componente software. La interfaz es utilizada como el medio de tener acceso a la funcionalidad del componente.

ISO/IEC 9126: estándar internacional para la evaluación de la calidad del software desde diferentes criterios asociados con adquisición, requerimientos, desarrollo, uso, evaluación, soporte, mantenimiento, aseguramiento de la calidad y auditoria de software.

Mallas 3D: objetos 3D que se conforman por vértices los cuales pueden unirse o dividirse para dar un aspecto suavizado o arrugado según lo requieras.

MeshLab: sistema de software de procesamiento de *mallas 3D* avanzado que está orientado a la gestión y el procesamiento de mallas.[4]

Nodo sensor: es un nodo en un sensor de red que es capaz de realizar algún procesamiento, reuniendo información sensible y comunicando con otros nodos conectados en la red.[5]

[4]MeshLab[en-linea]: http://www.meshlab.net/
[5]Nodo sensor[en-linea]: Wiki/Nodo_sensor

QT Creator: IDE (entorno de desarrollo integrado) multiplataforma que se ajusta a las necesidades de los desarrolladores.[6]

Software: conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora.

3D: en informática es algo simple como, el largo, ancho y alto de una imagen.

5.6. Especificación de Requerimientos de Software

5.6.1. Alcances

- La aplicación se complementará con el módulo externo optimizador.
- Posibilitará al usuario final la visualización de los datos obtenidos en el módulo externo optimizador.
- La herramienta permitirá optimizar el archivo o modelo abierto en base a los requerimientos del usuario.
- Permitirá la localización de sensores, gateways y transmisores y la visualización de las conexiones existentes entre ellos.
- La herramienta permitirá la interacción del usuario con la configuración realizada en el despliegue de redes de sensores inalámbricos industriales.

5.6.2. Límites

- Una vez creado el modelo en AutoCAD, será necesario exportar el modelo en formato .iges para luego usar el softare FreeCAD y exportar nuevamente el modelo a formato .off.
- Será necesario un espacio previamente diseñado en AutoCAD como entrada para poder hacer uso de esta herramienta. Sin esto, no será posible realizar una visualización de un entorno industrial tridimensional.

[6]QT Creator[en-linea]: https://www.qt.io/

5.7. Descripción Global del Producto

5.7.1. Interfaz de usuario

Localización con el mouse de sensores/transmisores, haciendo click sobre un vértice o en el universo de visualización.

5.7.2. Interfaz de hardware

La herramienta de visualización requiere mouse y teclado para la interacción.

5.7.3. Interfaz de software

Como se mencionó anteriormente, la herramienta de visualización estará ligado a un módulo optimizador externo que actúa como *middleware* (intercambio de información entre aplicaciones), proporcionando una mejor calidad del producto.

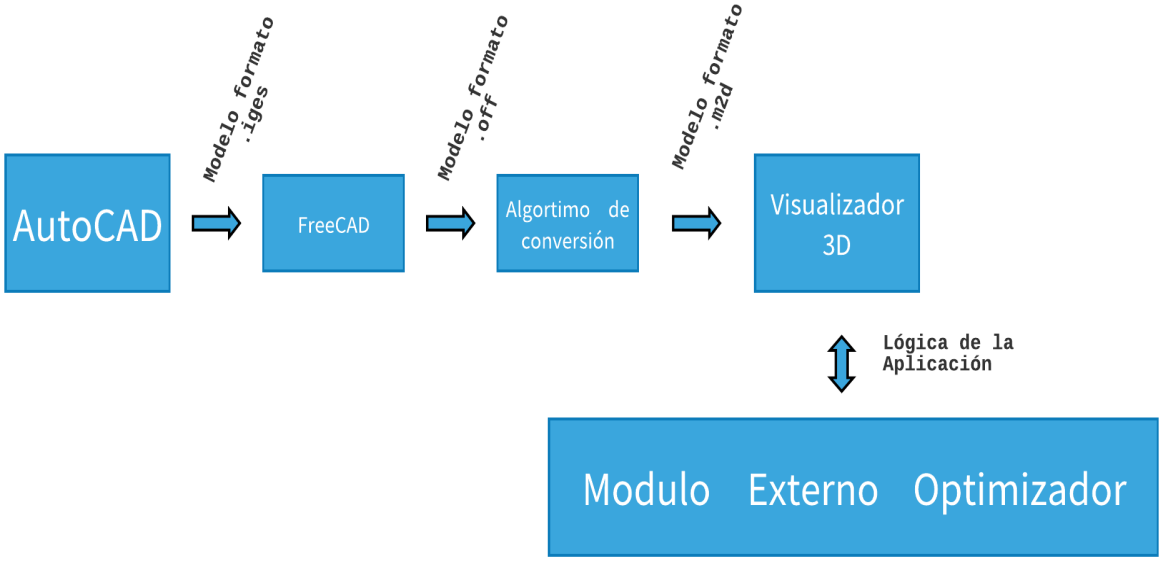


Figura 5.2: Arquitectura de referencia de la herramienta.

5.7.4. Interfaces de comunicación

No existen protocolos de comunicación para esta herramienta.

5.8. Requerimientos Específicos

5.8.1. Requerimientos funcionales (RF)

Id	Nombre	Descripción
RF_1	Importar modelo	Requerimiento que permite la importación de un modelo industrial.
RF_2	Ver/Visualizar	Requerimiento que permite la visualización en 3D del modelo importado.
RF_3	Interacción	Requerimiento que ofrece la interacción del usuario con el modelo que se esté visualizado.
RF_4	Procesa datos	Requerimiento que procesa los datos y los prepara para el módulo optimizador.
RF_5	Entrega resultados	Requerimiento que entrega el resultado final.
RF_6	Despliegue de redes	Requerimiento que ofrece la posibilidad de realizar el despliegue de sensores, gateways y transmisores.
RF_7	Salida	Requerimiento que ofrece la posibilidad de salir de la herramienta mediante previo mensaje de confirmación.
RF_8	Status Bar	Requerimiento que ofrece información relacionada con el modelo importado.
RF_9	Refinamiento	Requerimiento que ofrece la posibilidad de aplicar algoritmo de refinamiento sobre la malla.

Tabla 5.3: Requerimientos funcionales herramienta visualización.

5.8.2. Requerimientos no funcionales(RNF)

Id	Nombre	Descripción
RNF_1	Eficiencia/Rendimiento de refinamiento	El tiempo de respuesta al refinar un modelo no debe superar los 5 segundos.
RNF_2	Eficiencia/Rendimiento de despliegue	El tiempo de respuesta al realizar el despliegue de elementos, no debe superar los 12 segundos con un modelo de 35.000 vértices.
RNF_3	Desempeño	El software no presentará problemas para su manejo e interpretación.
RNF_4	Iconos	El software debe hacer uso de iconos a color en las funciones más relevantes.

Tabla 5.4: Requerimientos no funcionales herramienta de visualización.

5.9. Atributos del producto

5.9.1. Usabilidad

Facilidad de aprendizaje: el software debe ser de fácil aprendizaje para el usuario final, donde en menos de 30 minutos ya conozca el completo funcionamiento. Debe poseer una interfaz sencilla, que presente pocas dificultades o complicaciones, y acciones bien definidas con botones que indiquen explícitamente la función que cumplen.

Operabilidad: la herramienta debe mostrar información acorde al usuario para que éste pueda tener presente como está trabajando la herramienta y se pueda operar con facilidad. Los mensajes de alerta que la herramienta emita cuando el usuario realice una acción incorrecta indican claramente las posibles consecuencias que ésta pudiera tener.

5.9.2. Funcionalidad

Exactitud: la herramienta proporciona un grado de precisión requerido del 100 % para los resultados obtenidos, debido a que se complementará con un módulo externo, esta

información debe ser exacta y precisa.

Idoneidad: la herramienta debe satisfacer todos los requerimientos funcionales definidos, los cuales serán probados por el desarrollador como por los usuarios.

Seguridad: con el fin de garantizar los niveles de calidad y seguridad en el desarrollo de software, la herramienta se desarrolló bajo requisito de software seguro, garantizando la satisfacción del cliente.

5.9.3. Fiabilidad

Tolerancia a fallos: Un fallo en la herramienta puede ocasionar una gran pérdida al usuario, por eso el manejo de estos fallos es primordial. La herramienta cuenta con alertas de confirmación en caso de finalizar indebidamente la ejecución de esta.

5.9.4. Eficiencia

Comportamiento: la herramienta está diseñada para minimizar el tiempo de ejecución y la carga en memoria RAM de la máquina donde se esté ejecutando, esto para evitar la perdida de cuadros por segundo (del inglés *frames per second*, velocidad o tasa a la cual un dispositivo muestra imágenes), dado que al ser una herramienta de visualización de imágenes en tres dimensiones, es importante tener una calidad de proyección optima y fluida al momento de recorrer por el entorno.

5.9.5. Portabilidad

Adaptabilidad: la herramienta está desarrollada bajo un lenguaje de programación multiplataforma, C++ utilizando Qt Creator que es un IDE multi plataforma programado en C++. En este caso, se programó en ambiente Linux como sistema operativo.

Escalabilidad: la herramienta está programada y diseñada de manera que está habilitada para reaccionar y adaptarse o bien para estar preparada para hacerse más grande sin perder calidad en los servicios ofrecidos.

Capítulo 6

Análisis

6.1. Casos de uso

6.1.1. Actores

- **Usuario**:
  - El nivel de conocimiento requerido para utilizar la herramienta es nivel usuario y conocimiento básico de aplicaciones.
  - El usuario no cumple un rol especifico dentro de la aplicación.

Usuario, es el único actor que tiene acceso al sistema y realiza las siguientes funciones:

- Abrir modelo.
- Visualizar modelo.
- Interactuar con modelo.
- Desplegar Sensores.
- Desplegar Gateways.
- Desplegar Transmisores.
- Interactuar con Sensores/Gateways/Transmisores.
- Guardar modelo.
- Salir.

6.2. Diagrama casos de uso y descripción

Los diagramas de casos de uso se suelen utilizar en el modelado del sistema desde el punto de vista de sus usuarios para representar las acciones que realiza cada tipo de usuario [4].

Para el caso es un sólo actor involucrado que interactúa con el sistema; el usuario es quién interactúa con la herramienta.

6.2.1. Diagrama de casos de uso

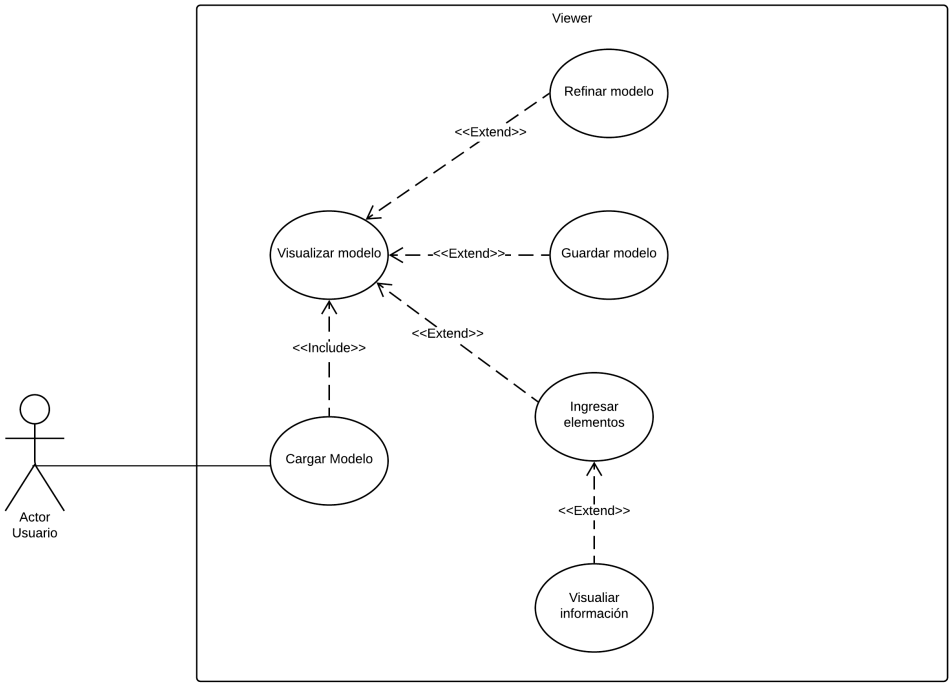


Figura 6.1: Diagrama de Casos de uso.

6.2.2. Descripción casos de uso

6.2.2.1. Caso de uso <Cargar modelo>

Nombre	Cargar modelo	**CU1**
Identificador caso de uso	001	
Actor Principal	Usuario	
Descripción: Permite cargar un modelo en formato .m2d para ser visualizado en la aplicación.		
Precondiciones: El usuario debe haber ingresado a la aplicación.		
Flujo de Eventos Básicos: 1. El actor pulsa sobre el botón File ubicado en la barra de herramientas de la aplicación. 2. La aplicación despliega el menú con las opciones. 3. El actor selecciona la opción 'Open Mesh'. 4. La aplicación despliega una ventana para realizar la búsqueda del archivo en formato .m2d a cargar. 5. El actor busca el archivo, lo selecciona y presiona el botón aceptar.		
Flujo de Eventos Alternativos: 1. La aplicación comprueba que la información contenida en el archivo tenga la estructura correcta.		
Poscondiciones: El archivo ha sido cargado en la aplicación.		

Tabla 6.1: Descripción Caso de Uso <Cargar modelo>

6.2.2.2. Caso de uso <Visualizar modelo>

Nombre	Visualizar modelo	**CU2**
Identificador caso de uso	002	
Actor Principal	Usuario	
Descripción: Visualizar el modelo cargado previamente.		
Precondiciones: El usuario debe haber ingresado a la aplicación y haber realizado la carga de un modelo.		
Flujo de Eventos Básicos: 1. La aplicación lee los datos del modelo cargado. 2. La aplicación llena las estructuras de datos correspondientes para cada objeto del modelo. 3. La aplicación pinta y hace visible el modelo. 4. La aplicación permite la interacción con el modelo cargado.		
Flujo de Eventos Alternativos: No presenta.		
Poscondiciones: El archivo ha sido cargado en la aplicación y se visualiza en la aplicación.		

Tabla 6.2: Descripción Caso de Uso <Cargar modelo>

6.2.2.3. Caso de uso <Refinar modelo>

Nombre	Refinar modelo	**CU3**
Identificador caso de uso	003	
Actor Principal	Usuario	
Descripción: Aplicar refinamiento de mallas al modelo cargado.		
Precondiciones: El usuario debe haber ingresado a la aplicación y haber realizado la carga de un modelo.		
Flujo de Eventos Básicos: 1. El actor presiona el botón 'select'alojado en la barra de herramientas. 2. La aplicación despliega un menú con tres opciones. 3. El actor selecciona la opción 'By Longest Edge'. 4. La aplicación muestra una ventana con las opciones para realizar la selección de triángulos a refinar. 5. El actor ingresa un valor para realizar la selección de triángulos en base al valor y selecciona la opción de biggest o smaller. 6. En caso de seleccionar biggest, la aplicación busca todos los triángulos cuyo lado más largo sean mayor al valor ingresado. 7. En caso de ser smaller, la aplicación busca todos los triángulos cuyo lado más largo sea menor al valor ingresado. 8. La aplicación selecciona todos los triángulos de acuerdo al criterio de selección del actor. 9. La aplicación muestra los triángulos seleccionados con un color azul. 10. El usuario presiona el botón 'Refinement Algorithms'. 11. La aplicación despliega un menú con la opción de 'execute'. 12. El actor presiona execute o bien la combinación de teclas "ctrl + e". 13. La aplicación aplica el algoritmo de refinamiento sobre los triángulos seleccionados. 14. La aplicación repinta el modelo con los nuevos triángulos creados.		

Flujo de Eventos Alternativos: No presenta.
Poscondiciones: El modelo ha sido refinado exitosamente y contiene una mayor cantidad de triángulos y vértices.

Tabla 6.3: Descripción Caso de Uso <Refinar modelo>

6.2.2.4. Caso de uso <Guardar modelo>

Nombre	Guardar modelo	**CU4**
Identificador caso de uso	004	
Actor Principal	Usuario	
Descripción: Guardar el modelo cargado previamente.		
Precondiciones: El usuario debe haber ingresado a la aplicación y haber realizado la carga de un modelo.		
Flujo de Eventos Básicos: 1. El actor presiona el botón 'File'alojado en la barra de herramientas. 2. La aplicación despliega un menú con opciones. 3. El actor selecciona la opción de 'Save Config'. 4. La aplicación abre una ventana para seleccionar el destino donde guardar el modelo. 5. El actor selecciona el destino donde guardar el modelo, le asigna un nombre al archivo y presiona aceptar. 6. La aplicación escribe los datos del modelo en un archivo con la estructura correspondiente al formado .m2d.		

Flujo de Eventos Alternativos: 1. La aplicación verifica si existen un despliegue realizado en el modelo, de ser así, guarda el modelo con los datos de los elementos desplegados.
Poscondiciones: El archivo ha sido guardado exitosamente.

Tabla 6.4: Descripción Caso de Uso <Guardar modelo>

6.2.2.5. Caso de uso <Ingresar elementos>

Nombre	Ingresar elementos	**CU5**
Identificador caso de uso	005	
Actor Principal	Usuario	
Descripción: Realizar despliegue de sensores.		
Precondiciones: El usuario debe haber ingresado a la aplicación y haber realizado la carga de un modelo.		

Flujo de Eventos Básicos:

1. El actor mediante la combinación de "ctrl + click derecho"despliega un menú contextual y selecciona la opción de Draw sensor o Draw Gateway.
2. La aplicación verifica la opción seleccionada y permite posicionar el elemento seleccionado en el modelo.
3. El actor mediante la combinación de "shift + click izquierdo"realiza el posicionamiento en el modelo del elemento seleccionado anteriormente.
4. La aplicación verifica que la posición seleccionada sea un vértice del modelo, en caso contrario busca el vértice más cercano a la posición seleccionada.
5. La aplicación pinta en el modelo un objeto con su color e identificador que representa el elemento seleccionado anteriormente.
6. El actor presiona el botón 'optimization'alojada en la barra de herramientas.
7. La aplicación despliega un menú con la opción de Execute.
8. El actor presiona Execute.
9. La aplicación verifica si existen los elementos necesarios en el modelo para continuar.
10. La aplicación asigna la posición de los transmisores en el modelo en base a la posición de los sensores.
11. La aplicación repinta el modelo con el despliegue realizado, con las conexiones realizadas y representadas por los colores asignados.
12. La aplicación mediante un mensaje de información informa al usuario que el refinamiento se realizó con éxito e indica el porcentaje de optimo de la configuración realizada.

Flujo de Eventos Alternativos: 1. La aplicación verifica si la posición seleccionada está en el suelo o ya existe otro elemento en lugar. 2. La aplicación informa al actor mediante un mensaje de información que no es posible usar esa posición.
Poscondiciones: El archivo ha sido cargado en la aplicación y se visualiza en la aplicación.

Tabla 6.5: Descripción Caso de Uso <Ingresar elementos>

6.2.2.6. Caso de uso <Visualizar información>

Nombre	Visualizar información	**CU6**
Identificador caso de uso	006	
Actor Principal	Usuario	
Descripción: Ver información asociada de los elementos desplegados en el modelo.		
Precondiciones: El usuario debe haber ingresado a la aplicación, haber realizado la carga de un modelo y un despliegue de sensores.		

Flujo de Eventos Básicos: 1. El actor presiona el botón 'Tools'alojado en la barra de herramientas. 2. La aplicación despliega un menú con distintas opciones para seleccionar. 3. El actor presiona la opción 'Info Elements'. 4. La aplicación abre una ventana con 3 secciones que contiene la información asociada a los elementos desplegados en el modelo. 5. La aplicación muestra el identificador del elemento incluido de su posición en el modelo en coordenadas (x, y, z). 6. El actor selecciona uno de los elementos. 7. La aplicación repinta en el modelo el elemento seleccionado de un color distinto a el original para tener una referencia del elemento seleccionado. 8. El actor tiene la posibilidad de eliminar el elemento seleccionado en el modelo. 9. La aplicación elimina el elemento seleccionado por el actor y repinta el modelo sin el elemento recientemente eliminado.
Flujo de Eventos Alternativos: No presenta.
Poscondiciones: La aplicación muestra la información asociada a cada elemento desplegado en el modelo.

Tabla 6.6: Descripción Caso de Uso <Visualizar información>

6.3. Diagrama de flujo

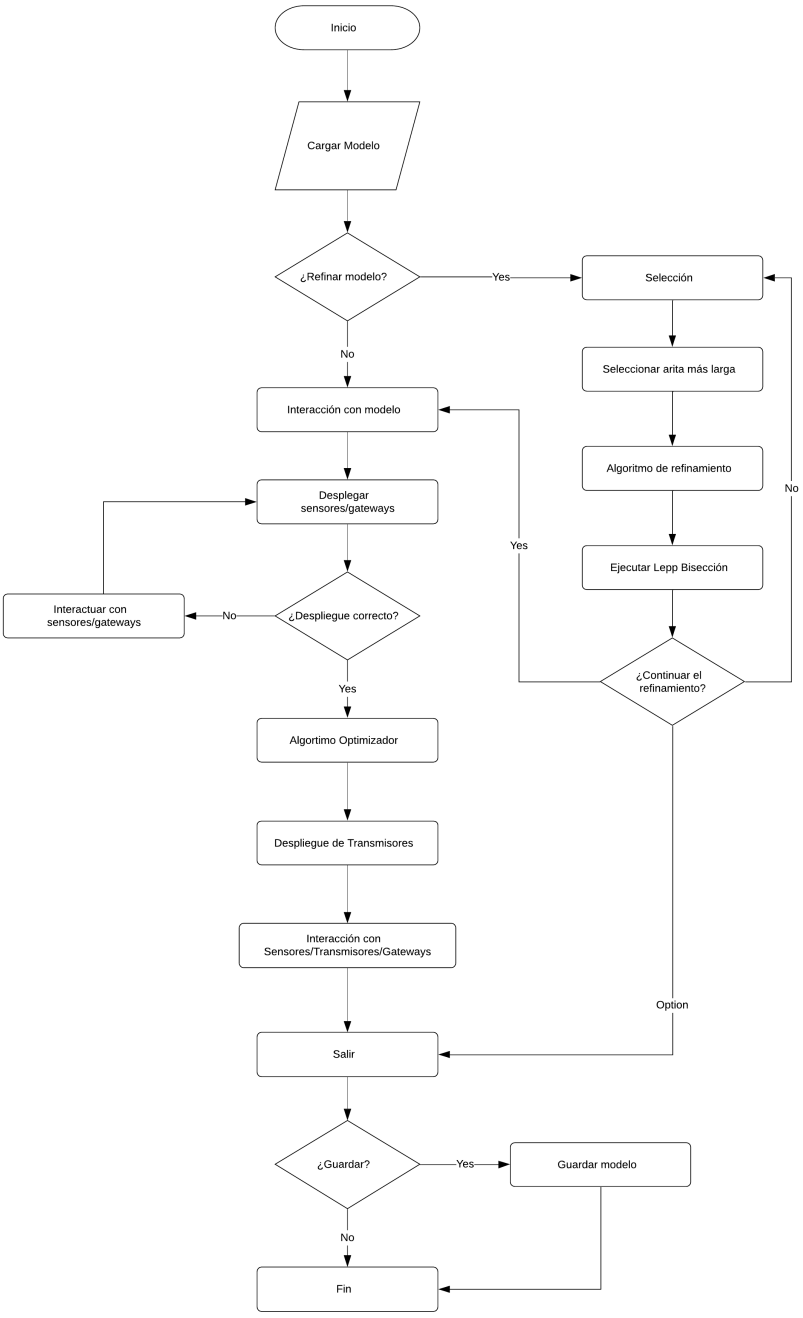


Figura 6.2: Diagrama de flujo.

6.3.1. Descripción diagrama

6.3.1.1. <Salir>

Nombre	Salir
Identificador	001
Prioridad	Baja
Descripción	Describe la funcionalidad del botón salir ubicado en la barra de herramientas o el icono X ubicado en la esquina superior izquierda de la ventana del software, el cual cumple con la función de salir y cerrar la aplicación.
Precondiciones	El usuario debe haber ingresado a la aplicación.
Flujo de Eventos Básicos	
Descripción	El usuario presiona el botón salir o presionando la 'X'situada en la esquina superior derecha y el sistema termina de hacer funcionar la aplicación.
Al actor	***El sistema***
1. Presiona el botón salir.	**2.** Ejecuta una linea de comando para cerrar la aplicación.
Flujo de Eventos Alternativo	
Descripción	Guardar cambios
Al actor	***El sistema***
Guardar cambios	Si existe modelo cargado, muestra mensaje de información si desea guardar cambios.
Poscondiciones	No presenta
Nivel	Bajo

Tabla 6.7: <Salir>

6.3.1.2. <Guardar>

Nombre	Guardar
Identificador	002
Prioridad	Baja
Descripción	Describe la funcionalidad del guardado de la tiangulación, en un archivo de modelado en tres dimensiones con extensión **.m2d**.
Precondiciones	El usuario debe cargar a la aplicación un archivo de modelado en tres dimensiones, que previamente haya sido convertido a formato **.m2d**, para que ésta cargue los datos contenidos en él.
Flujo de Eventos Básicos	
Descripción	El usuario presiona el botón guardar alojado en la barra de herramientas, la aplicación abre una ventana para escoger la ubicación o dirección (path) de donde se desea guardar el archivo. Una vez escogida la ubicación, se le asignará un nombre a gusto o seguir utilizando el mismo nombre con el cual fue abierto el archivo a guardar. La aplicación asigna la extensión **.m2d** al termino del nombre del archivo. Se presiona el botón aceptar para guardar y la aplicación se encargará de crear el archivo en la ubicación, escribiendo en el los datos relevantes de manera estructurada. Si se guardó, se despliega un mensaje de guardado exitoso.
Al actor	***El sistema***
1. Presiona el botón Save Mesh.	**2.** Abre una ventana para escoger un archivo.

3. Selecciona una dirección (path) donde se desea guardar el archivo y presiona el botón aceptar.	**4.** Comprueba si se ha presionado el botón aceptar, después se obtiene la dirección o ubicación (path) de donde se desea guardar el archivo. Se escribe en el archivo los datos que se encuentran en el modelo (aristas, vértices, triángulos) de forma estructurada. Se asigna la extensión de **.m2d** y despliega un mensaje de guardado exitoso.
Flujo de Eventos Alternativo	
Descripción	El usuario presiona el botón Save Mesh y si no se ha cargado un archivo previamente en memoria temporal, como respuesta la aplicación notifica que no se puede guardar, puesto que no se encuentra un archivo cargado.
Al actor	***El sistema***
1. Presiona el botón Save Mesh.	**2.** Notifica que no se puede guardar, puesto que no se encuentra ningún archivo cargado.
3. Cancelar.	**4.** No existe la ruta donde el usuario desea guardar.
Poscondiciones	No presenta
Nivel	Bajo

Tabla 6.8: <Guardar>

6.3.1.3. <Cargar Modelo>

Nombre	Cargar Modelo
Identificador	003
Prioridad	Alta
Descripción	Describe la funcionalidad del botón Open Mesh, el cual se encarga de cargar los archivos con formato **.m2d**, archivos de modelado en tres dimensiones.
Precondiciones	El usuario debe haber ingresado a la aplicación.
Flujo de Eventos Básicos	
Descripción	El usuario presiona el botón Open Mesh alojado en la barra de herramientas de la aplicación o con la combinación de teclas *CTRL + O*, y la aplicación pregunta si existe un archivo ya cargado. Si está cargado, la aplicación se encarga de borrar esta información, luego de esto instancia las estructuras de datos correspondientes al modelo de datos y procede a cargar los datos en memoria temporal. Primero, despliega una ventana de dialogo donde el usuario se encarga de buscar, a través de la navegación en la ventana de dialogo, la ubicación del archivo de modelado en tres dimensiones con formato **.m2d**. La aplicación verifica si se presionó el botón de aceptar, si es así, procede a cargar en memoria el archivo seleccionado por el usuario. Para cargar este archivo, primero se lee el archivo línea por línea e ingresa los datos en sus respectivas estructuras correspondientes. Una vez cargado el modelo, se informa al usuario que el modelo está listo para ser visualizado mediante la aplicación de visualización.
Al actor	***El sistema***

1. Presiona el botón Open Mesh.	**2.** Pregunta si ya está cargado el modelo de datos, si no está cargado con información la aplicación despliega una ventana de dialogo para buscar el archivo de modelado en tres dimensiones.
3. Selecciona una dirección (path) donde se encuentra el archivo, busca el modelo **.m2d** y presiona el botón aceptar.	**4.** Comprueba si se ha presionado el botón aceptar, después se obtiene la dirección o ubicación (path) de donde se encuentra el archivo. La aplicación se encarga de leer cada línea del archivo seleccionado e ingresa los datos obtenidos a sus estructuras de datos correspondientes (vértices, aristas, triángulos) y se informa a través de la vista que el modelo está listo para ser visualizado.
Flujo de Eventos Alternativo	
Descripción	No presenta.
Al actor	***El sistema***
1. Presion el botón Open Mesh.	**2.** Se obtiene la dirección o ubicación (path) del archivo a abrir, se obtiene y se guarda en memoria temporal. La aplicación pregunta si ya existe un modelo cargado, de ser así es borrado y carga el nuevo modelo seleccionado.
Poscondiciones	No presenta
Nivel	Bajo

Tabla 6.9: <Cargar Modelo>

6.3.1.4. <Interacción con modelo>

Nombre	Interacción con modelo
Identificador	004
Prioridad	Baja
Descripción	Describe la funcionalidad entre la interacción que tiene el usuario y el visualizador del modelo. En este caso se puede realizar *Zoom in* (acercarse) y *Zoom out* (alejarse), rotar en base a los ejes X e Y del universo Euclidiano y mover el modelo hacia la izquierda, derecha, arriba o abajo. También acceder a distintas vistas del modelo predeterminadas.
Precondiciones	El usuario debe haber ingresado a la aplicación y haber cargado un modelo.
Flujo de Eventos Básicos	
Descripción	El usuario, mediante el mouse realiza un click con el botón izquierdo, derecho o ambos del mouse y mantiene presionado para rotar el modelo, o mediante las flechas del teclado mover la posición del modelo. A través del scroll del mouse (rueda) realiza zoom y mediante teclado número superior de 1 a 9 incluido 0, acceder a las distintas vistas predeterminadas.
Al actor	***El sistema***
1. Deja presionado el click izquierdo, derecho o ambos en el visualizador.	**2.** Se consulta si donde se hizo click está dentro del visualizador
3. Puede mover el cursor o mouse hacia izquierda, derecha, arriba, abajo, o en las direcciones que el usuario desee para rotar el modelo.	**4.** Se obtienen las coordenadas X e Y y del evento del mouse, después actualiza el origen del modelo con las nuevas coordenadas capturadas por el puntero. Al modificar el origen se actualiza la vista y se repinta a medida que se van obteniendo nuevas coordenadas, actualizando el modelo en el visualizador.

5. Puede presionar las teclas de dirección del teclado para mover la posición del modelo.	**6.** Se obtienen las coordenadas X e Y y del evento del teclado, después actualiza el origen del modelo con las nuevas coordenadas capturadas por el teclado. Al modificar el origen se actualiza la vista y se repinta a medida que se van obteniendo nuevas coordenadas, actualizando el modelo en el visualizador.
7. Puede presionar el teclado numérico del teclado para rotar el modelo a posiciones predeterminadas.	**8.** Se presiona un botón del teclado numérico del 1 al 9 incluido 0, se obtiene el número presionado y se cambia a la vista predeterminada, actualizando el origen, y las coordenadas que se obtienen dependiendo el número presionado. Luego, se actualiza el visualizador.
Flujo de Eventos Alternativo	
Descripción	El usuario posiciona el puntero en la parte gráfica del visualizador y mediante el scroll (rueda del mouse) realiza zoom in (acercamiento) o zoom out (alejamiento) en el modelo.
Al actor	***El sistema***
1. Posiciona el puntero en la parte gráfica del visualizador y mueve la rueda de desplazamiento hacia arriba o hacia abajo.	**2.** Verifica si el mouse se encuentra en la parte gráfica del visualizador. Luego verifica si el evento del puntero es del tipo "QWheelEvent"y genera una escala con un valor delta multiplicado por la cantidad de rotaciones de la rueda del mouse. A medida que se realiza el movimiento del scroll hacia arriba o abajo, se va multiplicando este valor escalado y a la vez actualizando el modelo de forma inmediata, repintando constantemente la nueva vista del modelo.
Poscondiciones	No presenta
Nivel	Alto

Tabla 6.10: <Interacción con Modelo>

6.3.1.5. <Selección>

Nombre	Selección
Identificador	005
Prioridad	Baja
Descripción	Describe la funcionalidad del botón 'select by longest edge'(selección arista más larga) ubicado en la sección select de la barra de herramientas, el cual despliega el menú de selección.
Precondiciones	El usuario debe haber ingresado a la aplicación y haber cargado un modelo.
Flujo de Eventos Básicos	
Descripción	El usuario presiona el botón 'select by longest edge', el cual muestra una ventana de dialogo con las opciones para realizar la selección con la que cuenta la aplicación.
Al actor	***El sistema***
1. Presiona el botón 'select by longest edge'	**2.** Muestra la ventana con las opciones para realizar la selección.
Flujo de Eventos Alternativo	
Descripción	No presenta
Al actor	***El sistema***
1. No presenta	**2.** No presenta
Poscondiciones	No presenta
Nivel	Bajo

Tabla 6.11: <Selección>

6.3.1.6. <Selección arista más larga>

Nombre	Selección arista más larga
Identificador	006
Prioridad	Alta
Descripción	Describe la funcionalidad de la ventana desplegada por el botón 'select by longest edge'(selección arista más larga)
Precondiciones	El usuario debe haber ingresado a la aplicación, haber cargado un modelo y haber presionado el botón 'select by longest edge'ubicado en la barra herramientas de la aplicación.
Flujo de Eventos Básicos	
Descripción	El usuario ingresa un valor flotante acorde a sus criterios de selección.
Al actor	***El sistema***
1. Presiona el botón 'select by longest edge'	**2.** Muestra la ventana con las opciones para realizar la selección.
3. Ingresa un valor flotante mayor a 0 de acuerdo a sus criterios de selección y presiona aceptar.	**4.** Busca todos los lados en base a su medida acorde a el valor ingresado por el usuario. Una vez encontrados, los toma y los pinta de color azul, indicándole al usuario mediante el visualiazor todos los sectores que seleccionados de acuerdo a sus criterios de selección. El visualizador actualiza el modelo con los lados seleccionados en color azul.
Flujo de Eventos Alternativo	
Descripción	No presenta
Al actor	***El sistema***
1. No presenta	**2.** No presenta
Poscondiciones	Se guardan los lados seleccionados para su posterior refinamiento.
Nivel	Alto

Tabla 6.12: <Selección arista más larga>

6.3.1.7. <Refinar selección>

Nombre	Refinar selección
Identificador	007
Prioridad	Baja
Descripción	Describe la funcionalidad del botón Execute, ubicado en la sección 'Refinement Algorithms'en la barra de herramientas.
Precondiciones	El usuario debe haber ingresado a la aplicación, haber cargado un modelo y haber realizado una selección mediante la herramienta de 'select by longest edge'.
Flujo de Eventos Básicos	
Descripción	El usuario presiona el botón 'Execute'el cual ejecuta el refinamiento de LEEP-Bisección.
Al actor	***El sistema***
1. Presiona el botón 'Execute'	**2.** Ejecuta algoritmo de LEEP-Bisección
Flujo de Eventos Alternativo	
Descripción	No presenta
Al actor	***El sistema***
1. No presenta	**2.** No presenta
Poscondiciones	No presenta
Nivel	Bajo

Tabla 6.13: <Refinar Selección>

6.3.1.8. <Ejecutar LEPP-Bisección>

Nombre	Refinar selección
Identificador	008
Prioridad	Alta
Descripción	Describe la funcionalidad del botón Execute ubicado en la sección 'Refinement Algorithms'en la barra de herramientas.
Precondiciones	El usuario debe haber ingresado a la aplicación, haber cargado un modelo y haber realizado una selección mediante la herramienta de 'select by longest edge'.
Flujo de Eventos Básicos	
Descripción	El usuario presiona el botón 'Execute'el cual ejecuta el refinamiento de LEEP-Bisección.
Al actor	***El sistema***
1. Presiona el botón 'Execute'	**2.** Ejecuta algoritmo de LEEP-Bisección, refina el modelo en base a los lados seleccionados y muestra en la parte inferior los nuevos detalles del modelo.
Flujo de Eventos Alternativo	
Descripción	El usuario presiona el botón Execute y no existen lados seleccionados para que el algoritmo funcione.
Al actor	***El sistema***
1. Presiona el botón 'Execute'	**2.** No se realiza ningún cambio en el modelo porque no existen lados seleccionados previamente por el usuario y no es posible refinar.
Poscondiciones	Devuelve el modelo refinado, el cual es cargado y repintado al terminar de ejecutar el refinamiento.
Nivel	Alto

Tabla 6.14: <Ejecutar LEPP-Bisección>

6.3.1.9. <Desplegar Sensores/Gateways>

Nombre	Desplegar Sensores/Gateways
Identificador	009
Prioridad	Alta
Descripción	Describe la funcionalidad del despliegue de Sensores y Gateways en el modelo.
Precondiciones	El usuario debe haber ingresado a la aplicación, haber cargado un modelo y haber activado mediante la opción de 'Draw Sensor'
Flujo de Eventos Básicos	
Descripción	El usuario a través del puntero selecciona el vértice del modelo en el cual desea desplegar un Sensor/Gateways
Al actor	***El sistema***

1. Mediante la combinación de la tecla CTRL + Click derecho despliega un menú y activa la opción 'Draw Sensor'	**2.** A través de una variable boolean, el sistema comprueba que se ha marcado la opción de 'Draw Sensor'y se desbloquea el método encargado de crear los objetos Sensor mediante la combinación de la tecla SHIFT + Click izquierdo en la posición del modelo donde se desea desplegar el Sensor. Al hacer click, se obtienen las coordenadas X e Y del puntero, las cuales el sistema mediante un método de conversión realiza la obtención de una variable faltante, en este caso Z. Luego de obtener la variable Z, el sistema retorna la posición del puntero en tres coordenadas (X, Y, Z). Estas tres coordenadas son utilizadas por la función de crear Sensor, que se encarga de obtener las coordenadas de todos los vértices existentes en el modelo y buscar el vértice más próximo o cercano a las coordenadas obtenidas por el puntero. Luego de obtener la coordenada del vértice más próximo a la posición del puntero, la aplicación toma esa coordenada de vértice y crea un nuevo objeto en la estructura de datos de Sensor, la cual consta de un identificador único, y tres coordenadas para identificar su posición el modelo. Luego de eso se pinta un objeto de color cyan (calipso) de forma redonda en las coordenadas del vértice tomado, con su respectivo identificador indicando a que Sensor corresponde.

3. Mediante la combinación de la tecla CTRL + Click derecho despliega un menú y activa la opción 'Draw Gateway'	**4.** A través de una variable boolean, el sistema comprueba que se ha marcado la opción de 'Draw Gateway'y se desbloquea el método encargado de crear los objetos Gateway mediante la combinación de la tecla SHIFT + Click izquierdo en la posición del modelo donde se desea desplegar el Gateway. Al hacer click, se obtienen las coordenadas X e Y del puntero, las cuales el sistema mediante un método de conversión realiza la obtención de una variable faltante, en este caso Z. Luego de obtener la variable Z, el sistema retorna la posición del puntero en tres coordenadas (X, Y, Z). Estas tres coordenadas son utilizadas por la función de crear Gateway, que se encarga de obtener las coordenadas de todos los vértices existentes en el modelo y buscar el vértice más próximo o cercano a las coordenadas obtenidas por el puntero. Luego de obtener la coordenada del vértice más próximo a la posición del puntero, la aplicación toma esa coordenada de vértice y crea un nuevo objeto en la estructura de datos de Gateway, la cual consta de un identificador único, y tres coordenadas para identificar su posición el modelo. Luego de eso se pinta un objeto de color lila de forma cuadrada en las coordenadas del vértice tomado, con su respectivo identificador indicando a que Gateway corresponde.
Flujo de Eventos Alternativo	
Descripción	El usuario presiona la combinación de SHIFT + Click izquierdo y no existe ninguna opción marcada previamente de pintado para que el algoritmo funcione.
Al actor	***El sistema***

1. presiona la combinación de SHIFT + Click	**2.** No se realiza ningún cambio en el modelo porque no existe ninguna opción marcada previamente en el menú desplegable por la combinación de CTRL + Click derecho
Poscondiciones	No presenta
Nivel	Alto

Tabla 6.15: <Desplegar Sensores/Gateways>

6.3.1.10. <Interacción con Sensores/Gateways>

Nombre	Interacción con Sensores/Gateways
Identificador	010
Prioridad	Baja
Descripción	Describe la funcionalidad de la interacción con los Sensores/Gateways desplegados en el modelo.
Precondiciones	El usuario debe haber ingresado a la aplicación, haber cargado un modelo y haber desplegado Sensores y/o Gateways.
Flujo de Eventos Básicos	
Descripción	El usuario presiona el botón 'Info Elements'ubicada en la barra de herramientas en la sección 'Tools', el cual despliega una ventana de dialogo para realizar la interacción con los Sensores y o Gateways.
Al actor	***El sistema***
1. Presiona el botón 'Info Elements'	**2.** Muestra una ventana de dialogo, con cuadros de información referente a cada objeto en el modelo.

3. Selecciona algún Sensor/Gateway en la ventana de información.	**4.** Obtiene las coordenadas del Sensor/Gateway seleccionado en la ventana de información, y busca esas coordenadas en la estructura de datos correspondiente a cada objeto. Una vez encontrada, la herramienta de visualización retorna el modelo con el Sensor/Gateway seleccionado de un color distinto al de origen para demostrar que ese es el elemento que está seleccionado y tener una referencia e indetificar la ubicación más fácil entre todos los Sensores/Gateways existentes en el modelo.
5. Selecciona algún Sensor/Gateway en la ventana de información y presiona eliminar.	**6.** Obtiene las coordenadas del Sensor/Gateway seleccionado en la ventana de información, y busca esas coordenadas en la estructura de datos correspondiente a cada objeto. Una vez encontrada, la herramienta de visualización quita ese elemento de la estructura de datos y el visualizador retorna el modelo con el Sensos/Gateway removido.
7. Selecciona algún Sensor/Gateway en la ventana de información y presiona eliminar todo.	**8.** Despliega un mensaje alertando que se eliminarán todos los elementos de esa estructura de datos. Luego comprueba si se presionó el botón aceptar, de ser así, la estructura de datos es vaciada (eliminada) y el visualizador retorna el modelo actualizado sin los objetos eliminados.
Flujo de Eventos Alternativo	
Descripción	El usuario presiona el botón 'Info Elements'y no existen sensores/gateways
Al actor	***El sistema***
1. Presiona el botón 'Info Elements'	**2.** No es posible una interacción con los objetos ya que no existe ninguno en el modelo.
Poscondiciones	No presenta
Nivel	Bajo

Tabla 6.16: <Interacción con Sensores/Gateways>

6.3.1.11. <Optimizador>

Nombre	Optimizador
Identificador	011
Prioridad	Alta
Descripción	Describe la funcionalidad de la función optimizar para realizar el despliegue de Transmisores.
Precondiciones	El usuario debe haber ingresado a la aplicación, haber cargado un modelo y haber desplegado Sensores y Gateways.
Flujo de Eventos Básicos	
Descripción	El usuario presiona el botón 'Execute'ubicada en la barra de herramientas en la sección 'Optimizer'.
Al actor	***El sistema***

1. Presiona el botón 'Execute'	**2.** Comprueba si es que en el modelo existen al menos un Sensor y Gateway. Si existe, obtiene las coordenadas de los vértices del modelo en los cuales se encuentran Sensores y Gateways. Luego de eso, envía esas coordenadas una por una a un método que recibe estas coordenadas y las utiliza para calcular mediante un algoritmo la posición más optima y cercana en donde iría un objeto Transmisor, que se comunicará mediante un conexión cableada con el Sensor. Lo mismo para el Gateway, una vez obtenidas sus coordenadas de los vértices en donde se encuentre algún objeto de este tipo, envía esas coordenadas para crear una conexión que será del tipo inalámbrica entre el Gateway y el Transmisor. A medida que se envía una coordenada de un Sensor, este optimizador mediante el algoritmo calcula inmediatamente la posición de donde iría el Transmisor correspondiente y se crean los objetos en la estructura de datos de Transmisor, con un identificador único y tres coordenadas que representan su posición en el modelo. Una vez analizad todo el modelo, asignando un Transmisor a cada Sensor, un algoritmo calcula el% de optimo (fittest) de la configuración realizada, desplegando un mensaje por pantalla con el valor obtenido.
Flujo de Eventos Alternativo	
Descripción	El usuario presiona el botón 'Execute'y no existen sensores/gateways
Al actor	***El sistema***
1. Presiona el botón 'Execute'	**2.** No es posible realizar la optimización porque no existen elementos Sensores.
3. Presiona el botón 'Execute'	**4.** No es posible realizar la optimización porque no existen elementos Gateways.

5. Presiona el botón 'Execute'	**6.** No es posible realizar la optimización porque no existen elementos Sensores ni Gateways.
Poscondiciones	No presenta
Nivel	Bajo

Tabla 6.17: <Optimizador>

6.3.1.12. <Despliegue Transmisores>

Nombre	Despliegue Transmisores
Identificador	012
Prioridad	Alta
Descripción	Describe la funcionalidad del despliegue de Transmisores en el modelo.
Precondiciones	El usuario debe haber ingresado a la aplicación, haber cargado un modelo, desplegado Sensores y Gateways y utilizado el método optimizer.
Flujo de Eventos Básicos	
Descripción	El usuario presiona el botón 'Execute'ubicada en la barra de herramientas en la sección 'optimizer'.
Al actor	***El sistema***

1. Presiona el botón 'Execute'	**2.** Verifica que se haya realizado con éxito el método de 'optimizer'y se hayan llenado las estructuras de datos de Sensor, Gateways y Transmisores. Luego obtiene todos los valores contenidos en la estructura de datos de Transmisor que se crearon al momento de utilizar el método 'optimizer', recorre esa estructura de datos para ir obteniendo las coordenadas que utilizarán los Transmisores en el modelo, cuyas coordenadas están contenidas en cada objeto del tipo Transmisor. A medida que recorre la estructura de datos y obtiene las coordenadas de cada Transmisor, el modelo repinta un objeto del tipo Transmisor y es actualizado visualmente. Este Transmisor es conectado directamente con el Sensor que le corresponda mediante una conexión alámbrica que es representada en el visualizador por una línea de color verde que indica la conexión entre el Sensor y el Transmisor. Así también lo hace con el Gateways. Una vez obtenida las coordenadas de la posición del Gateways, es pintada la conexión inalámbrica, representada por una línea puntuada de color azul. Esto indica que el Transmisor está conectado de forma inalámbrica al Gateway correspondiente.
Flujo de Eventos Alternativo	
Descripción	El usuario presiona el botón 'Execute'y no existen sensores/gateways
Al actor	***El sistema***
1. Presiona el botón 'Execute'	**2.** No es posible realizar el despliegue porque no existen elementos en las estructuras de datos correspondientes.
Poscondiciones	No presenta
Nivel	Bajo

Tabla 6.18: <Despliegue Transmisores>

6.3.1.13. <Interacción con Gateways, Sensores y Transmisores>

Nombre	Interacción Gateway/Sensor/Transmisor
Identificador	013
Prioridad	Baja
Descripción	Describe la funcionalidad de la interacción con los Sensores, Transmisores y Gateways.
Precondiciones	El usuario debe haber ingresado a la aplicación. Haber cargado un modelo. Haber realizado el despliegue de Sensores y Gateways y utilizado la función optimizer.
Flujo de Eventos Básicos	
Descripción	El usuario presiona el botón 'Info Elements'ubicada en la barra de herramientas en la sección 'Tools'. para realizar la interacción con los tres elementos mencionados.
Al actor	***El sistema***
1. Presiona el botón 'Info Elements'	**2.** Muestra una ventana de dialogo con cuadros de información referente a cada objeto en el modelo.
1. Presiona el botón 'Info Elements'	**2.** Muestra una ventana de dialogo con cuadros de información referente a cada objeto en el modelo.
3. Selecciona algún Sensor/Gateway/Transmisor en la ventana de información.	**4.** Obtiene las coordenadas del Sensor/Gateway/Transmisor seleccionado en la ventana de información, y busca esas coordenadas en la estructura de datos correspondiente a cada objeto. Una vez encontrada, la herramienta de visualización retorna el modelo con el Sensor/Gateway/Transmisor seleccionado de un color distinto al de origen para demostrar que ese es el elemento que está seleccionado y tener una referencia e identificar la ubicación más fácil entre todos los Sensores/Gateways/Transmisores existentes en el modelo.

5. Selecciona algún Sensor/Gateway/Transmisor en la ventana de información y presiona eliminar.	**6.** Obtiene las coordenadas del Sensor/Gateway/Transmisor seleccionado en la ventana de información, y busca esas coordenadas en la estructura de datos correspondiente a cada objeto. Una vez encontrada, la herramienta de visualización quita ese elemento de la estructura de datos y el visualizador retorna el modelo con el Sensos/Gateway/Transmisor removido. Si se borra algún Sensor/Transmisor de la lista, automáticamente se borrará su par asociado, es decir el objeto con cual se encontraba conectado, esto para evitar que existan elementos sin usar y el% de fittest (optimo) siempre sea el mejor.
7. Selecciona algún Sensor/Gateway/Transmisor en la ventana de información y presiona eliminar todo.	**8.** Despliega un mensaje alertando que se eliminarán todos los elementos de esa estructura de datos. Luego comprueba si se presionó el botón aceptar, de ser así, la estructura de datos es vaciada (eliminada) y el visualizador retorna el modelo actualizado sin los objetos eliminados. Al momento de eliminar todo los elementos de la lista de Transmisor o Sensor, automáticamente se borrará completamente su par.
Flujo de Eventos Alternativo	
Descripción	El usuario presiona el botón 'Tools'alojado en la barra de herramientas del visualizador.
Al actor	***El sistema***
1. Presiona el botón 'Info Mesh'	**2.** Despliega una ventana de información correspondiente al modelo cargado. Esta información contiene las coordenadas el modelo, medidas a escala y también la simbología de los elementos que se trabajan en el visualizador.

3. Presiona el botón 'Connection Info'	**4.** Despliega una ventana de información correspondiente al modelo cargado. Esta ventana contiene una tabla que indica los pares conectados y los metros de cableado utilizado entre las conexiones existentes en el modelo.
5. Presiona el botón 'Paint Mode'	**6.** Esta opción bloquea el uso del mouse para realizar la rotación del modelo y permitir al usuario trabajar de mejor forma.
7. Presiona el botón 'Clear All Window'	**8.** Esta opción elimina todo el contenido de las estructuras de datos Sensor/Transmisor/-Gateways y actualiza el modelo limpio y sin objetos.
Poscondiciones	No presenta
Nivel	Bajo

Tabla 6.19: <Interacción con Gateways, Sensores y Transmisores>

Capítulo 7

Diseño del software

7.1. Biblioteca OpenGL

Para la realización de este software se utilizó la biblioteca OpenGL (Open Graphics Library) de C++ para Qt.

OpenGL, es una especificación estándar que define una API[1]multilenguaje y multiplataforma para escribir aplicaciones que produzcan gráficos 2D y 3D. La interfaz consiste en más de 500 funciones diferentes que pueden usarse para dibujar escenas tridimensionales complejas a partir de primitivas geométricas simples, tales como puntos, líneas y triángulos. Se usa ampliamente en CAD, realidad virtual, representación científica, visualización de información y simulación de vuelo. También se usa en desarrollo de videojuegos, donde compite con Direct3D en plataformas Microsoft Windows[25].

La ventaja de esta biblioteca es que nos aísla del hardware disponible; por tanto, si disponemos de una tarjeta aceleradora (tarjeta gráfica), no hará falta que cambiemos nuestro programa para aprovechar la potencia de nuestra tarjeta[6].

7.1.1. Biblioteca añadidas

En el desarrollo de este software, no sólo se trabajó con OpenGL. También se utilizó la biblioteca GLU y GLUT. La blbioteca GLU contiene funciones gráficas de más alto nivel

[1] Application Programming Interface. Conjunto de funciones y procedimientos que cumplen una o muchas funciones con el fin de ser utilizadas por otro software.

que OpenGL, que permiten realizar operaciones más complejas. En cambio, la biblioteca GLUT es un paquete auxiliar para construir aplicaciones de ventanas, además de incluir algunas primitivas geométricas auxiliares. La gran ventaja de este paquete es que el mismo código servirá en Windows™ y Linux™ además de simplificar mucho el código fuente del programa[6, 25].

7.2. De vértice a píxeles

Desde que le damos a OpenGL unas coordenadas 3D hasta que aparecen en la pantalla, estas coordenadas son modificadas de la siguiente manera:

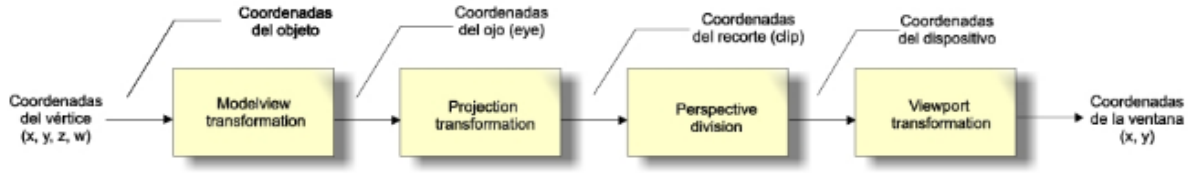


Figura 7.1: Secuencia de trabajo OpenGL.

Resumiendo, para obtener una vista de una escena en nuestra pantalla tenemos que definir:

- La matriz del Modelview, que define la colocación de los objetos en la escena y el punto de vista.
- La matriz de Proyección, que define el tipo de proyección.
- La posición del viewport en coordenadas de pantalla.

```
void
MeshViewer::initializeGL()
{
    GLfloat LightAmbient[4]= { 0.7f, 0.7f, 0.7f, 1.0f };    // Ambient Light Values ( NEW )
    GLfloat LightDiffuse[4]= { 1.0f, 1.0f, 1.0f, 1.0f };    // Diffuse Light Values ( NEW )
    GLfloat LightPosition[4]= { 0.0f, 0.0f, 2.0f, 1.0f };   // Light Position ( NEW )
    glViewport(0, 0, (GLint)width(), (GLint)height());
    qglClearColor(Qt::black);
    glClearDepth(1.0);
    glShadeModel(GL_SMOOTH);
    glShadeModel(GL_FLAT);
    glEnable(GL_DEPTH_TEST);
    glDepthFunc(GL_LEQUAL);
    glHint(GL_PERSPECTIVE_CORRECTION_HINT, GL_NICEST);
    glEnable(GL_CULL_FACE);
    glEnable(GL_LIGHTING);
    glEnable(GL_LIGHT0);
    glEnable(GL_MULTISAMPLE);
    glHint(GL_PERSPECTIVE_CORRECTION_HINT, GL_NICEST);
    glLightfv(GL_LIGHT0, GL_AMBIENT, LightAmbient);
    glLightfv(GL_LIGHT0, GL_DIFFUSE, LightDiffuse);
    glLightfv(GL_LIGHT0, GL_POSITION,LightPosition);
    glEnable(GL_TEXTURE_2D);
    glEnable(GL_COLOR_MATERIAL);
    glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
    glEnable(GL_BLEND);
    currentTime.start();
    lastTime.start();
}
```

Figura 7.2: Método para inicializar area de visualización.

En el método observado en la figura 7.2 se crea el área donde se visualizará el modelo, se configura la posición de la cámara y la posición de la luz (claridad con la que se verá el modelo).

- **LightAmbient:** incluso cuando está oscuro, por lo general todavía hay algo de luz en algún lugar del mundo (la luna, una luz distante), por lo que los objetos casi nunca están completamente oscuros. Para simular esto, utilizamos una constante de iluminación ambiental que siempre le da algo de color al objeto.
- **LightDiffuse:** simula el impacto direccional que un objeto de luz tiene sobre un objeto. Este es el componente visualmente más significativo del modelo de iluminación. Cuanto más se enfrenta una parte de un objeto a la fuente de luz, más brillante se vuelve.

7.3. Técnicas utilizadas en el software

Una de las limitaciones de trabajar con OpenGL, es la conversión de las coordenadas de pantalla (x, y) en coordenadas de tres dimensiones (x, y, z), dado que no existe un método en especifico para realizar esto. Este paso es esencial en el software, ya que permite la interacción con el modelo para realizar el despliegue de sensores, transmisores y gateways sobre el.

Para realizar esta conversión, se diseño un método que a partir de primitivas[1] de OpenGL y Qt, fue posible la obtención de coordenadas de ventana (x, y) a coordenadas en tres dimensiones (x, y, z).

```
/* Vector de 3dimensiones, Obtiene posición x, y, z de MeshViewer */
QVector3D MeshViewer::getPixelLocation(int x, int y)
{
    QString xx,yy,zz;

    GLint viewport[4];      /* [0]=x, [1]=y, [2]=width, [3]=height */
    GLdouble modelView[16]; /* Donde se guardarán los 16 dobles de la matriz Modelview */
    GLdouble projection[16];/* Donde se guardarán los 16 dobles de la matriz de proyección */
    GLdouble wx, wy, wz; /* Mantener los valores finales */

    glGetIntegerv(GL_VIEWPORT, viewport); /* Recupera los valores de la ventana gráfica (X, Y, ancho, alto) */
    glGetDoublev(GL_MODELVIEW_MATRIX, modelView); /* Recupera los valores de matriz de modelo */
    glGetDoublev(GL_PROJECTION_MATRIX, projection); /* Recupera los valores de matriz de proyección */

    /* Determinar la posición Z de la ventana desde glReadPixels() */
    GLfloat z = 0.0f;
    glReadPixels(x, height() - 1 - y, 1, 1,GL_DEPTH_COMPONENT, GL_FLOAT, &z);
    glPolygonMode(GL_FRONT_AND_BACK,GL_POINTS);

    /*Obtener coordenadas del mundo en base a la posición del mouse*/
    gluUnProject(x + 0.5, viewport[3] - 0.5 - y, z,
            modelView, projection, viewport, &wx,&wy,&wz);

    xx = QString::number(wx);
    yy = QString::number(wy);
    zz = QString::number(wz);

    return QVector3D(wx, wy, wz);
}
```

Figura 7.3: Método diseñado para obtención de coordenadas.

Debido a que cuando trabajamos con el puntero sobre ambientes tridimensionales, éste solo es capaz de obtener su posición en la ventana de trabajo en dos dimensiones, pero gracias a este método, ya es posible realizar una interacción entre el puntero y el modelo en el mismo espacio tridimensional y hacer uso de todas la funciones del software en cuanto

[1]Tipos de datos originales de un lenguaje de programación.

a interacción con el (Ver cap. 8).

Método para realizar el pintado de un sensor:

```
void
MeshViewer::drawSensor()
{
    glPointSize(10.0);
    glEnable(GL_PROGRAM_POINT_SIZE);
    /*=*/
    glEnable(GL_POINT_SMOOTH);
    glEnable(GL_BLEND);
    glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
    /*=*/
    glBegin(GL_POINTS);
    if(!_sVector.isEmpty() && vectorPos.z() >0)
    {
        QVectorIterator<Sensor*> is(_sVector);
        while(is.hasNext())
        {
            Sensor *s = is.next();
            glColor3f(0.0f, 1.0f, 1.0f);  // Color cyan
            glVertex3d(s->getXS(),s->getYS(),s->getZS());
        }
    }
    else /* Repinta si no está sobre el terreno*/
    {
        QVectorIterator<Sensor*> is(_sVector);
        while(is.hasNext())
        {
            Sensor *s = is.next();
            glColor3f(0.0f, 1.0f, 1.0f);  // Color cyan
            glVertex3d(s->getXS(),s->getYS(),s->getZS());
        }
    }
    glEnd();
    glDisable(GL_POINT_SMOOTH);
    glDisable(GL_BLEND);
}
```

Figura 7.4: Método diseñado para dibujar sensor en el modelo.

Este método es parte importante de la aplicación. Gracias a este método es posible realizar el pintado de los sensores sobre el modelo. Se inicia indicándole el tamaño para el elemento que se pintará, en este caso será un punto redondo de tamaño 10. Una vez que se inicializan las variables de la biblioteca de OpenGL, se crea un iterador que será el encargado de recorrer la estructura de datos creada para almacenar los objetos de tipo Sensor. A medida que este iterador va recorriendo la estructura de datos, obtiene las posiciones de cada elemento contenido en él en coordenadas (x, y, z) que serán necesarias para indicarle a la función encargada de realizar el pintado, las posiciones donde debe hacerlo. Al igual que este método, existe otro similar que está encargado de realizar el recorrido de la estructura de datos creada para los objetos de tipo Gateway y pintar.

7.4. Técnica para realizar el despliegue

Como se describía en la figura 7.3, este método es uno de los más importantes, porque gracias a él es posible realizar un despliegue sobre el modelo. Cabe destacar que para realizar el despliegue de redes de sensores inalámbricos es necesario obtener las posiciones de los vértices del modelo sobre el cual se está trabajando, vértices que son representados por un punto de color verde, así como se muestra en la siguiente imagen:

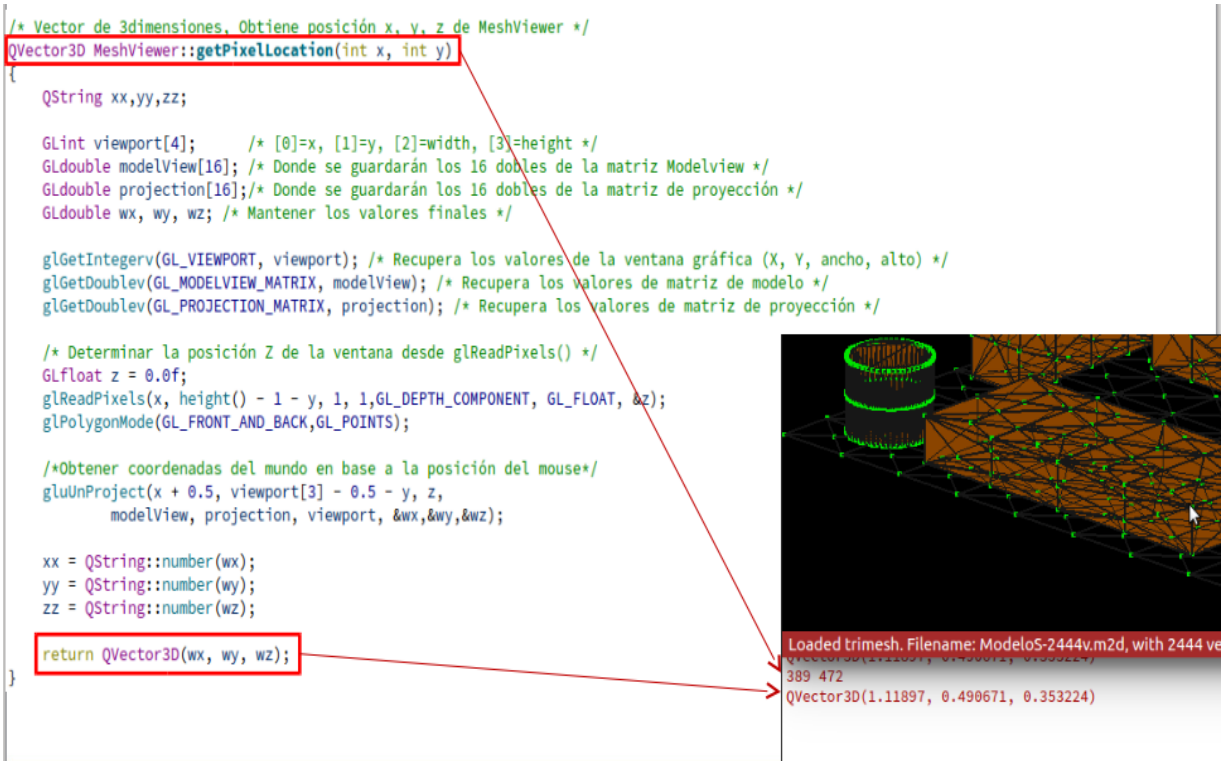


Figura 7.5: Selección vértice y transformación coordenadas.

Como se observa en la figura 7.5, una vez posicionado el mouse sobre el vértice donde se desea desplegar un elemento, se realiza un click, el cual enviará al método 7.3 como parámetro de entrada su posición en coordenadas en dos dimensiones (x, y) y le retornará la posición del mundo en el cual se encuentra este vértice.

7.5. Pintado de un elemento sobre un vértice

Realizado todos los pasos anteriores, solo queda realizar el pintado del elemento que se desea sobre el vértice seleccionado, de la siguiente manera:

```
void
MeshViewer::drawSensor()
{
    glPointSize(10.0);
    glEnable(GL_PROGRAM_POINT_SIZE);
    /*=*/
    glEnable(GL_POINT_SMOOTH);
    glEnable(GL_BLEND);
    glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
    /*=*/
    glBegin(GL_POINTS);
    if(!_sVector.isEmpty() && vectorPos.z() >0)
    {
        QVectorIterator<Sensor*> is(_sVector);
        while(is.hasNext())
        {
            Sensor *s = is.next();
            glColor3f(0.0f, 1.0f, 1.0f);  // Color cyan
            glVertex3d(s->getXS(),s->getYS(),s->getZS());
        }
    }
    else /* Repinta si no está sobre el terreno*/
    {
        QVectorIterator<Sensor*> is(_sVector);
        while(is.hasNext())
        {
            Sensor *s = is.next();
            glColor3f(0.0f, 1.0f, 1.0f);  // Color cyan
            glVertex3d(s->getXS(),s->getYS(),s->getZS());
        }
    }
    glEnd();
    glDisable(GL_POINT_SMOOTH);
    glDisable(GL_BLEND);
}
```

Declaración inicial para definir tipo de figura, tamaño e inicialización de pintado para esa figura

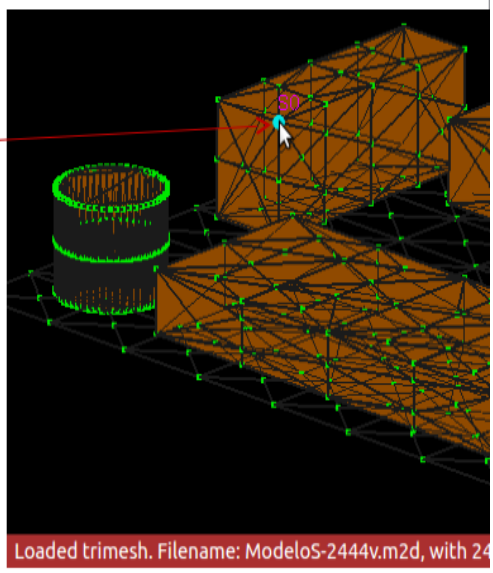


Figura 7.6: Pintado de elemento sobre el modelo.

De esta forma, es posible realizar el pintado de un elemento, ya sea del tipo Sensor como también del tipo Gateway.

7.6. Layout de ventana

A continuación se explica cómo se compone cada elemento del diseño para así poder entender la estructura detallada del sistema.

7.6.1. MainWindow y MeshViewer

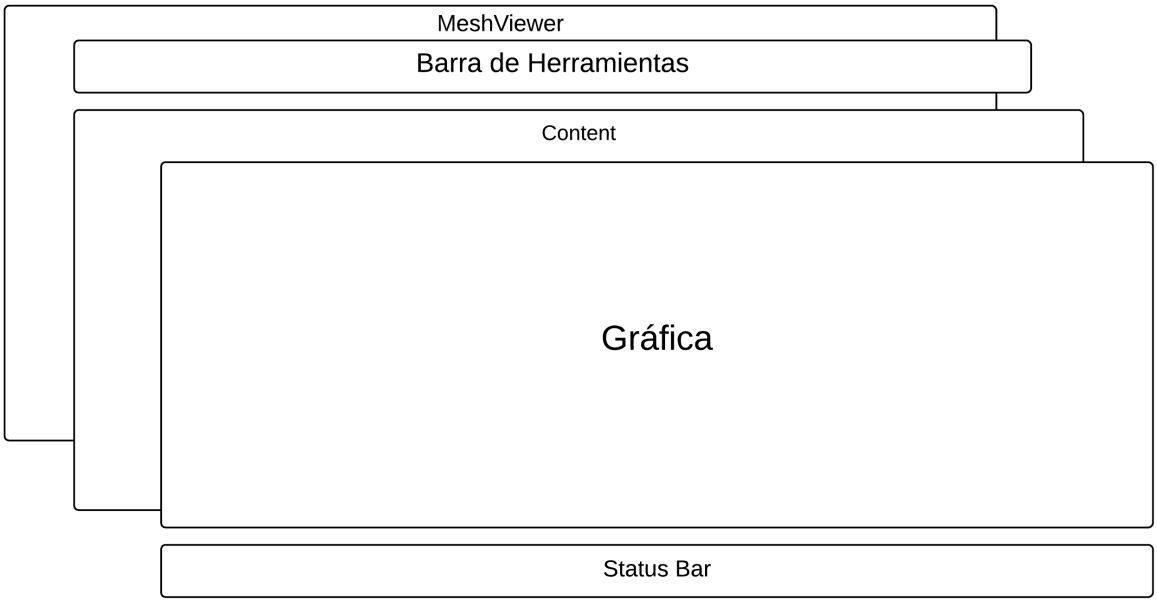


Figura 7.7: Mockup de la ventana principal.

La clase MainWindow es una clase contenedora de MeshViewer y los paneles a los cuales el usuario tiene acceso. A continuación se explica cada uno de los elementos que componen la clase MainWindow.

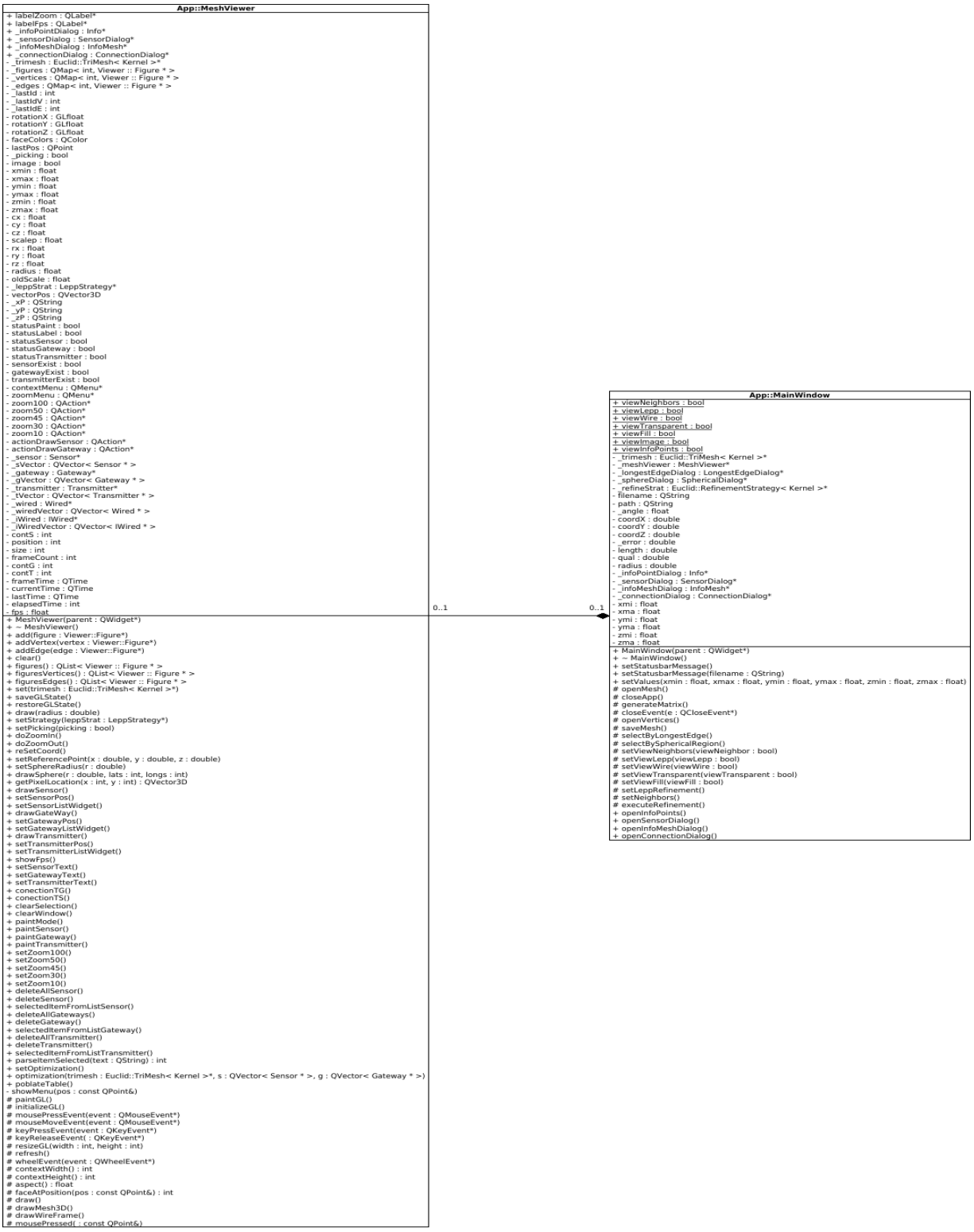


Figura 7.8: Diagrama de clases de MainWindow con Meshviewer.

7.7. MainWindow

Esta clases proporciona todo lo que necesita para una ventana principal de aplicación moderna típica, como la ventana principal, el menú y las barras de herramientas, una barra de estado, etc. Esta clase, es la encargada de contener el visualizador(MeshViewer) y todos los elementos que lo componen. A través de la clase MainWindow, es posible interactuar con los métodos alojados en la clase MeshViewer, que es la encargada de mostrar por pantalla todas las interacciones que el usuario tenga con el modelo.

A continuación se describen los elementos de MainWindow(Ver figura 7.7).

7.7.1. Barra de Herramientas

En el software, la barra de herramientas está contenida en la clase de MainWindow. Se encarga de tener el acceso al cargado y el guardado de datos, además de cerrar el software, seleccionar figuras a refinar, aplicar algoritmo de refinamiento y algoritmo de optimización de sensores y transmisores. Así como también tiene acceso a un apartado llamado Tools, donde se encuentran todas las funciones con respecto a la interacción con el modelo, como por ejemplo, mostrar ubicación de los objetos en el modelo, la simbología utilizada, las conexiones existentes y la información del modelo cargado.

7.7.2. Gráfica

El el software, la gráfica es proporcionada por la clase MeshViewer que está contenida por la clase MainWindow. MeshViewer es una clase de pintado y actualización, que se encarga de mostrar por pantalla de forma gráfica el modelo cargado, así como también los objetos que el usuario pueda crear al interactuar con el software.

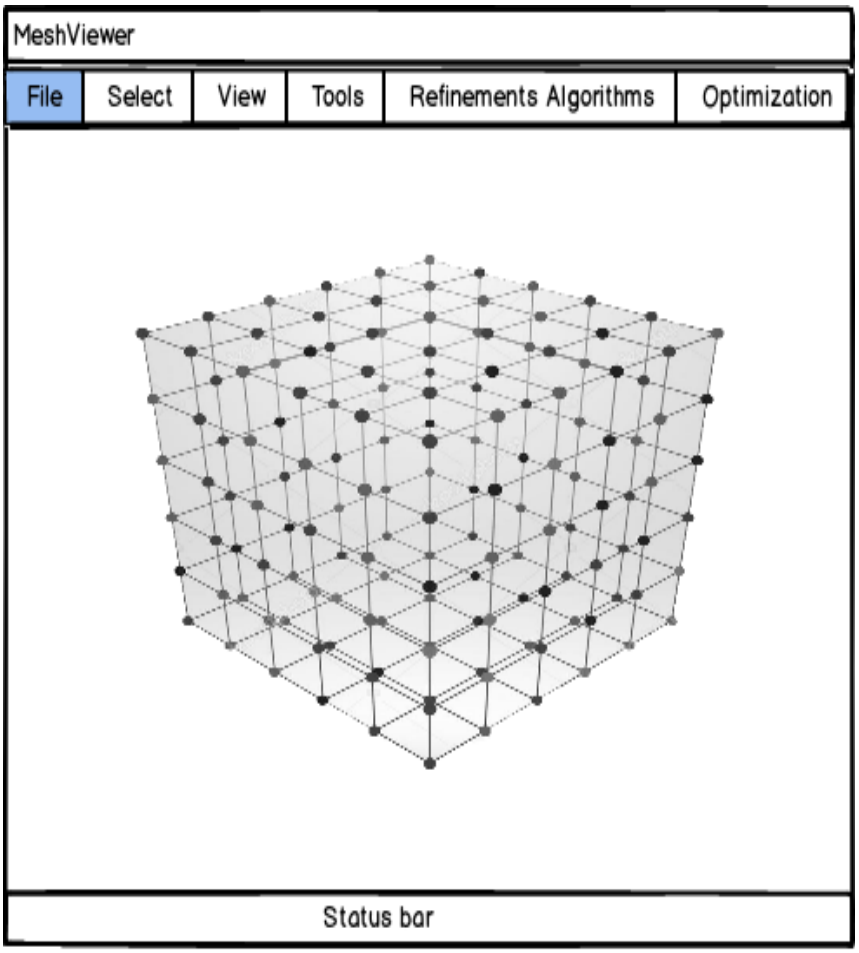


Figura 7.9: Mockup MeshViewer

7.7.3. Status bar

La clase QStatusBar proporciona una barra horizontal adecuada para presentar información de estado. Esta información en el software, indica el nombre del modelo cargado, la cantidad de vértices y triángulos que posee y la cantidad de memoria utilizada en kb[1].

[1]Kilobyte. Unidad de almacenamiento de información y equivale a 10^3 (mil) bytes.

7.8. Diagrama de secuencia

El diagrama de secuencia es un tipo de diagrama usado para modelar interacción entre objetos en un sistema según UML[1]. Los diagramas de interacción se encargan de describir el comportamiento dinámico del sistema de información mediante el paso de mensajes entre los objetos del mismo.

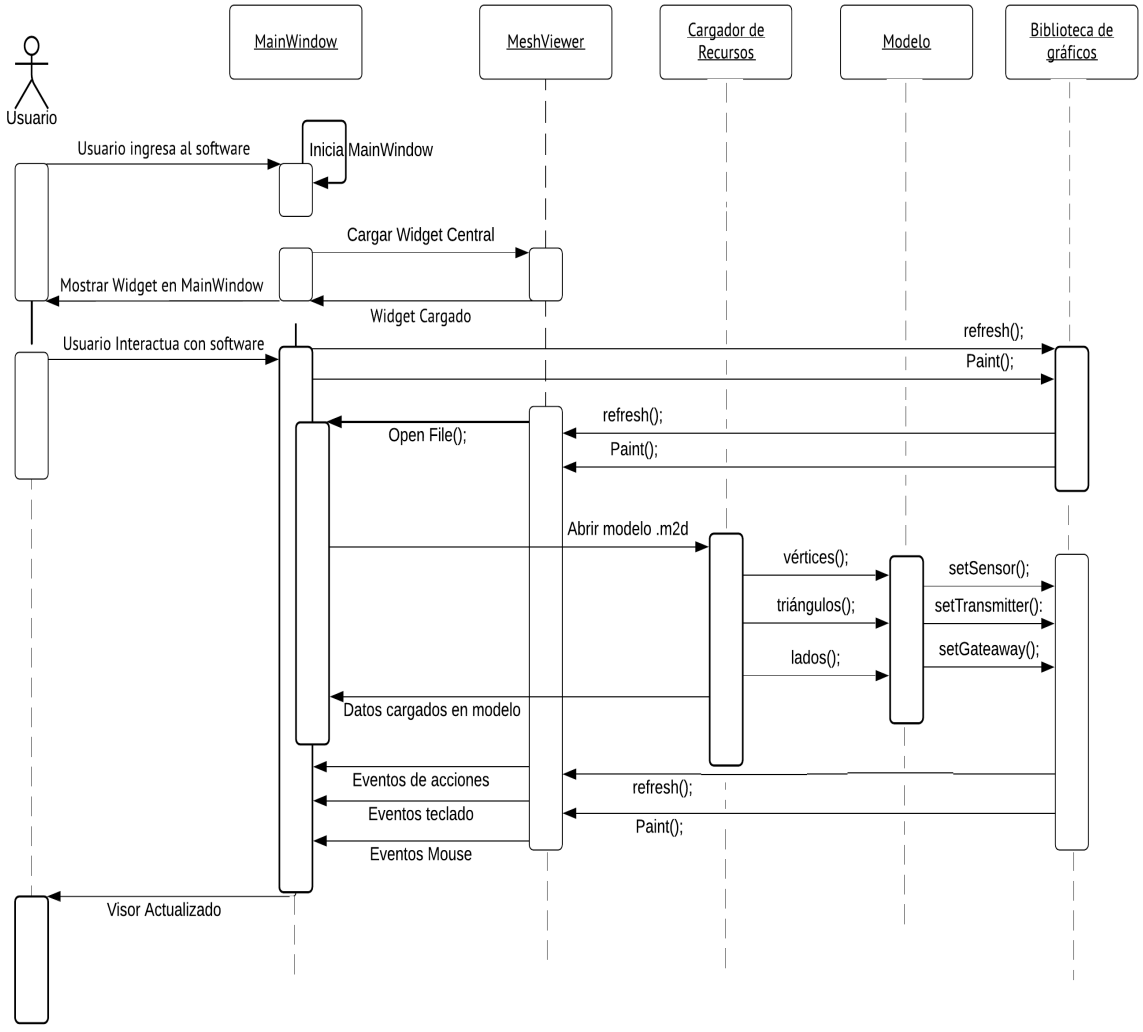


Figura 7.10: Diagrama de secuencia del software.

[1]Lenguaje unificado de modelado (**UML**, por sus siglas en inglés, Unified Modeling Language).

En el diagrama que se observa en la figura 7.10 el usuario final hace ingreso al software donde se incializa la aplicación mediante la clase principal MainWindow. Esta clase instancia todos los objetos referenciados en ella así como también inicializa el visor gráfico llamado Widget central de la clase de MeshViewer.

Al inicializar la clase principal MainWindow se detonan los siguientes métodos:

- **Cargar widget central:** se encarga de configurar el área de visualización de los elementos gráficos, configurar las dimensiones como alto y ancho (x, y), además de dar inicio a los contenidos incluidos en él con sus respectivos eventos de interacción como teclado y mouse.

- **Paint y refresh:** encargados de realizar la actualización del área de visualización cada vez que se interactué con él o cuando se agregue un nuevo elemento a la escena. Se considera interacción con él desde un simple click, hasta la presión de cualquier tecla.

Capítulo 8

Pruebas de funcionalidad del software

8.1. Pruebas gráficas

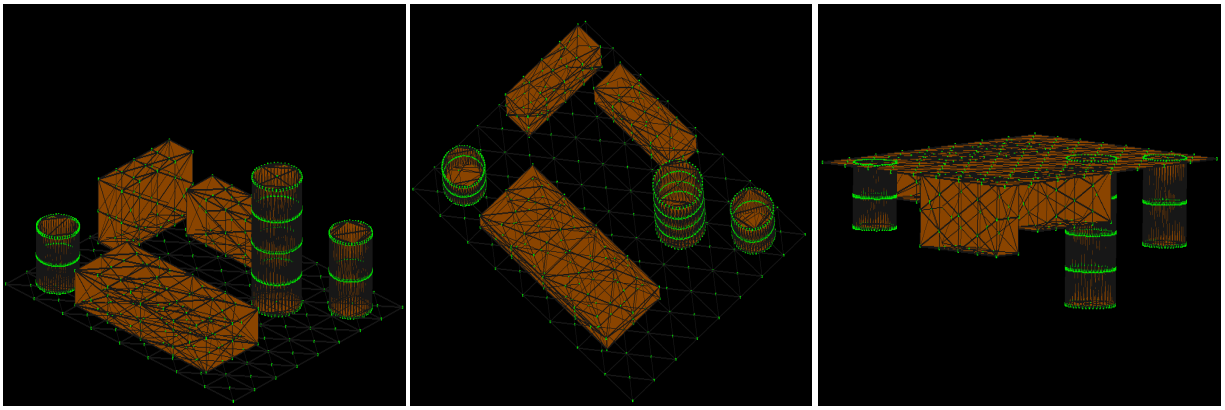
En esta sección está enfocada en las pruebas en torno a la funcionalidad del software, así como también su comportamiento con el puntero, teclado y cómo se comporta la aplicación en distintos casos de sobrecarga de figuras en el modelo. Todo esto, a través del Wiget central (QGL Widget)[1]. Este widget trabaja en base a matriz de proyecciones que son inicializadas al momento de ejecutar la aplicación.

8.2. Rotación

Referente a la interacción entre el puntero y el Widget central (QGLWidget) de visualización, en el cual se obtienen las coordenadas x e y asociadas al alto y ancho del widget central. Para poder lograr la rotación es necesario hacer uso del método *glRotate*, que produce una rotación de grados de ángulo alrededor del vector x, y, z declarado previamente para realizar la proyección de elementos gráficos a través del widget central. La matriz actual, se multiplica por una matriz de rotación con el producto reemplazando la matriz actual. Al hacer uso del puntero sobre el modelo que se está visualizado en el widget cen-

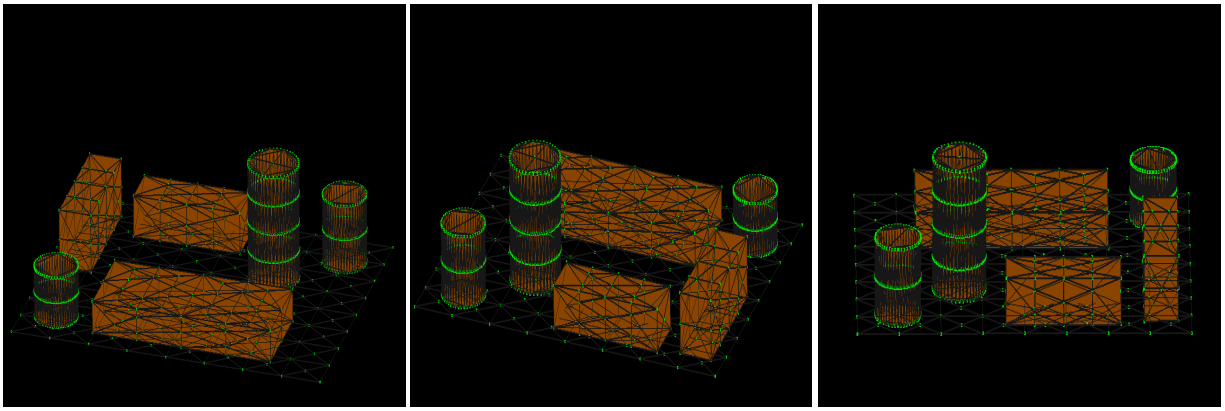
[1]Proporciona funcionalidad para mostrar gráficos OpenGL integrados en una aplicación Qt.

tral, este captura las coordenadas del puntero y las utiliza para realizar la rotación de la matriz que nos dará como resultado la rotación del modelo en base al arrastre del mouse.



(a) (b) (c)

Figura 8.1: En la figura **(a)** se presiona el puntero con click izquierdo sobre el modelo y se mantiene presionado. En la figura **(b)** y **(c)** se desplaza el mouse hacia arriba, logrando una rotación de la matriz de proyección en base al eje Y del modelo.

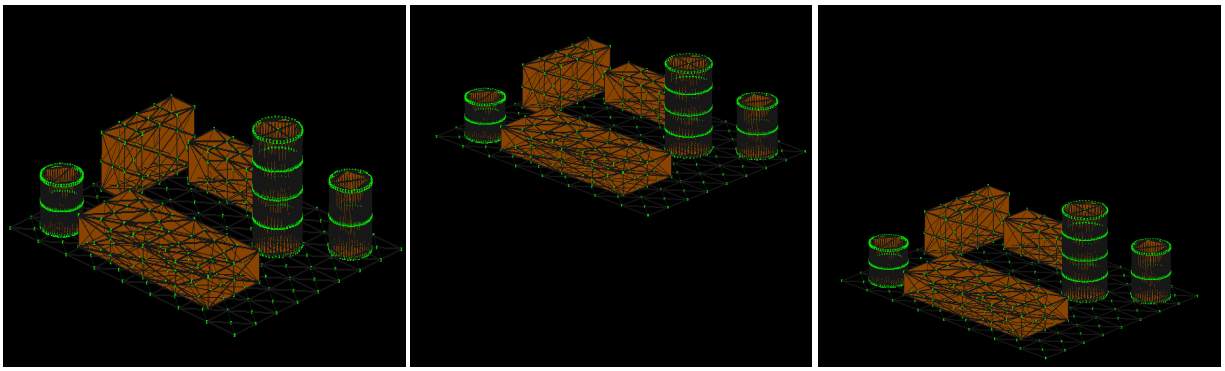


(a) (b) (c)

Figura 8.2: En la figura **(a)** se presiona el puntero con click derecho sobre el modelo y se mantiene presionado. En la figura **(b)** y **(c)** se desplaza el mouse hacia la izquierda, logrando una rotación de la matriz de proyección en base al eje X del modelo.

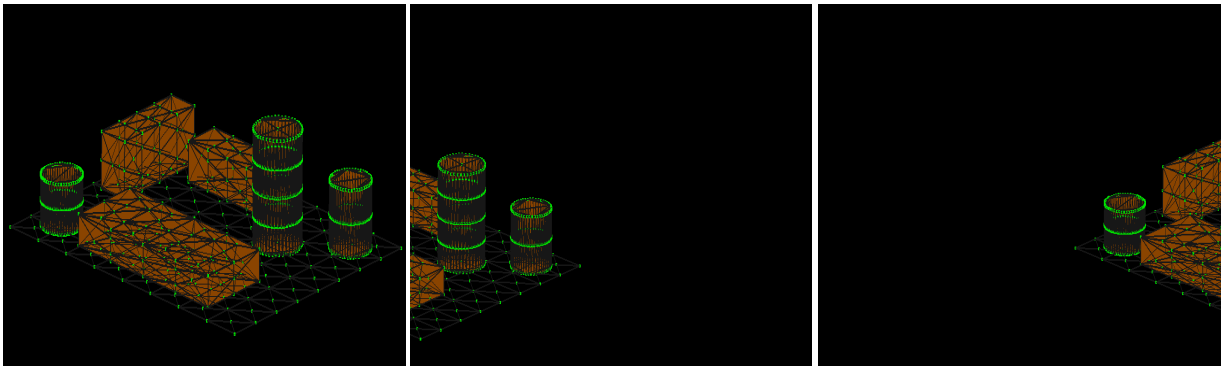
8.3. Desplazamiento

Referente a la interacción entre el teclado con sus flechas direccionales y el Widget central (QGLWidget) de visualización, en el cual se obtienen las coordenadas x e y asociadas a los movimientos de arriba/abajo o abajo/arriba con respecto al eje y y derecha/izquierda o izquierda/derecha con respecto al eje X. Para eso es necesario utilizar el metodo *glTranslate*, que produce un traslado por x y z de la matriz de proyección con lo cual al presionar sobre alguna flecha direccional del teclado, el modelo es actualizado con el traslado realizado.



(a) (b) (c)

Figura 8.3: En la figura **(a)** no existe desplazamiento. En la figura **(b)** se ha realizado un desplazamiento hacia arriba y en la figura **(c)** un desplazamiento hacia abajo.

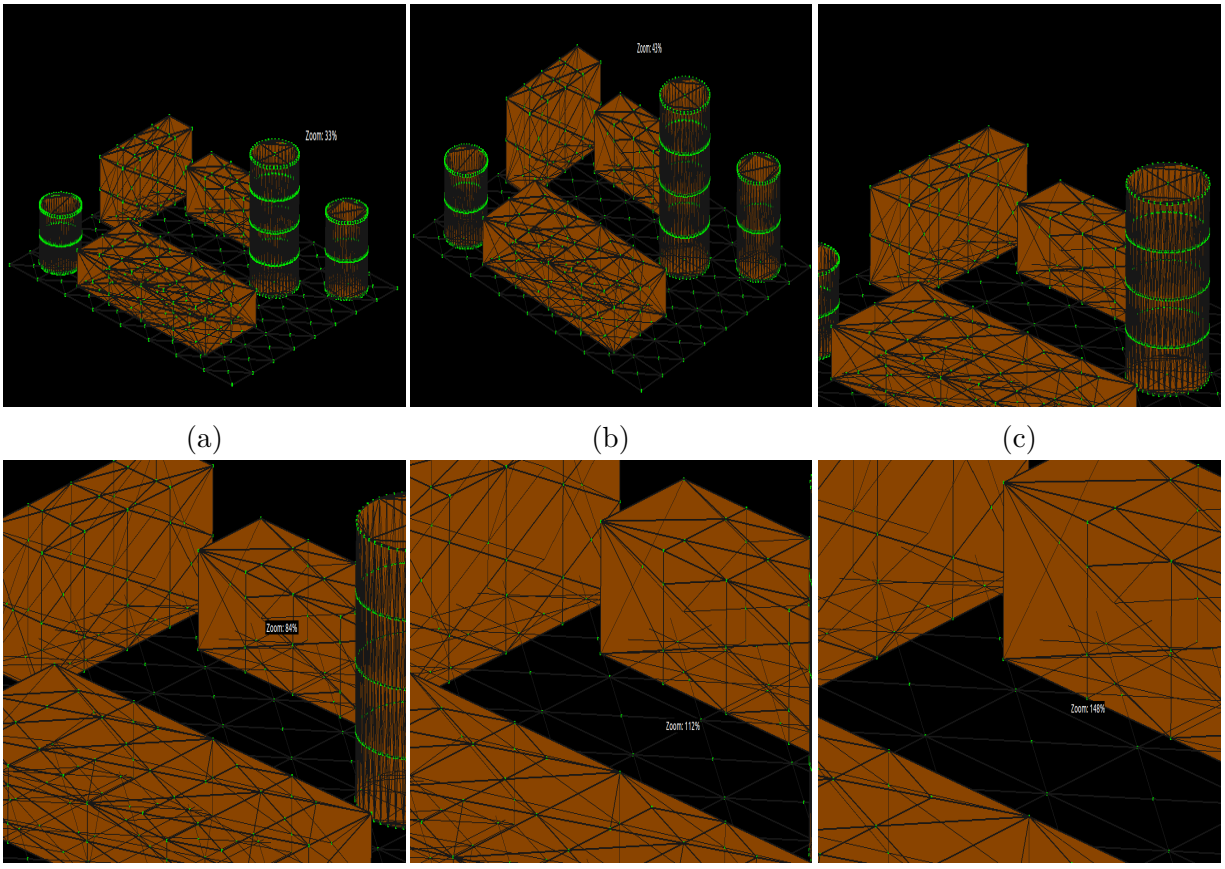


(a) (b) (c)

Figura 8.4: En la figura **(a)** no existe desplazamiento. En la figura **(b)** se ha realizado un desplazamiento hacia la izquierda y en la figura **(c)** un desplazamiento hacia la derecha.

8.4. Acercamiento (Zoom)

Referente a la interacción entre la rueda (scroll) del mouse y el Widget central (QGLWidget) de visualización, en el cual se obtiene el factor de escala inicial del modelo cargado. Luego de esto se utiliza el método *glScale*, que produce una escala no uniforme a lo largo de los ejes x, y y z. Los tres parámetros indican el factor de escala deseado a lo largo de cada uno de los tres ejes. Cuando realizamos scroll hacia arriba, el factor de escala aumenta y es multiplicado por los valores de los ejes x, y y z, y cuando se realiza scroll hacia abajo el factor de escala disminuye y es multiplicado por los valores x, y y z de los ejes.



(a) (b) (c)

(d) (e) (f)

Figura 8.5: En la figura **(a)** existe un acercamiento del 33% sobre el modelo. En la figura **(b)** un 44%. En la figura **(c)** un 77%. En la figura **(d)** un 84%. En la figura **(e)** un 112% y en la figura **(f)** un 148%.

8.5. Sobrecarga de triángulos y vértices

Corresponde a la funcionalidad del status bar[1] ubicado en la parte inferior del software el cual indica la cantidad de vértices y triángulos del modelo cargado. Además un método de medición de los fotogramas por segundo[2] (FPS, del inglés frames per second). Al sobrecargar con triángulos en el visualizador, la cantidad de fps obtenidos es cada vez más baja. Esto debido a la gran cantidad de elementos dibujados. Una vez cargado un modelo, se puede observar el comportamiento de estos elementos.

- En la figura 8.7(a) se carga un modelo con 2444 vértices y 3176 triángulos. La cantidad de fotogramas por segundo es ≈ 150.
- En la figura 8.7(b) se carga un modelo con 4586 vértices y 6412 triángulos. La cantidad de fotogramas por segundo es ≈ 80.
- En la figura 8.8(a) se carga un modelo con 7034 vértices y 9508 triángulos. La cantidad de fotogramas por segundo es ≈ 55.
- En la figura 8.8(b) se carga un modelo con 15242 vértices y 23846 triángulos. La cantidad de fotogramas por segundo es ≈ 35.

#	**Vértices**	**Fps**
1	470	200
2	1090	170
3	2500	150
4	4700	90
5	7035	55
6	15300	28
7	20500	20
8	47000	8

Tabla 8.1: Tabla frames por segundo.

[1]La clase QStatusBar proporciona una barra horizontal adecuada para presentar información de estado.
[2]Velocidad o tasa a la cual un dispositivo muestra imágenes llamadas cuadros o fotogramas.

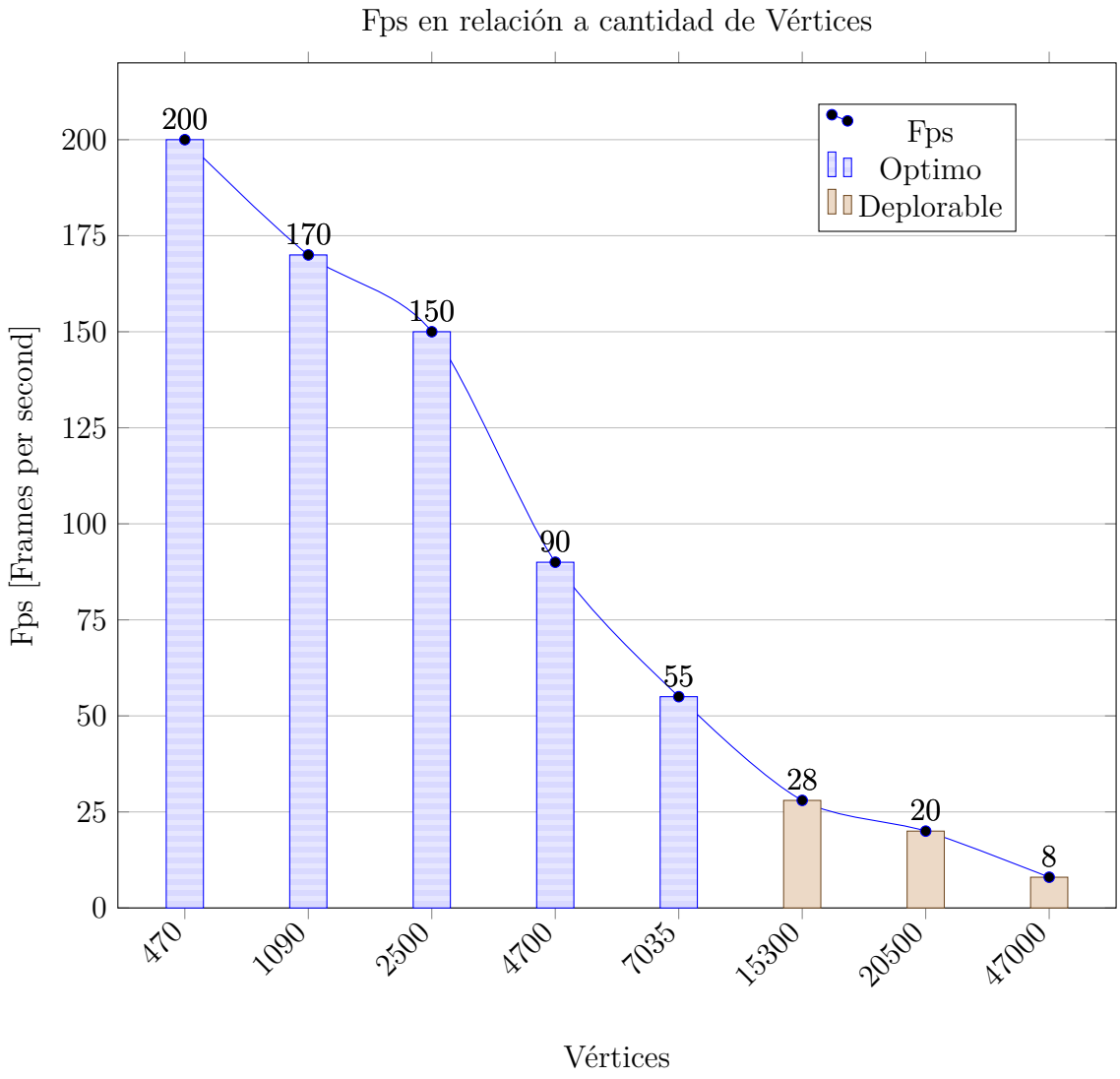
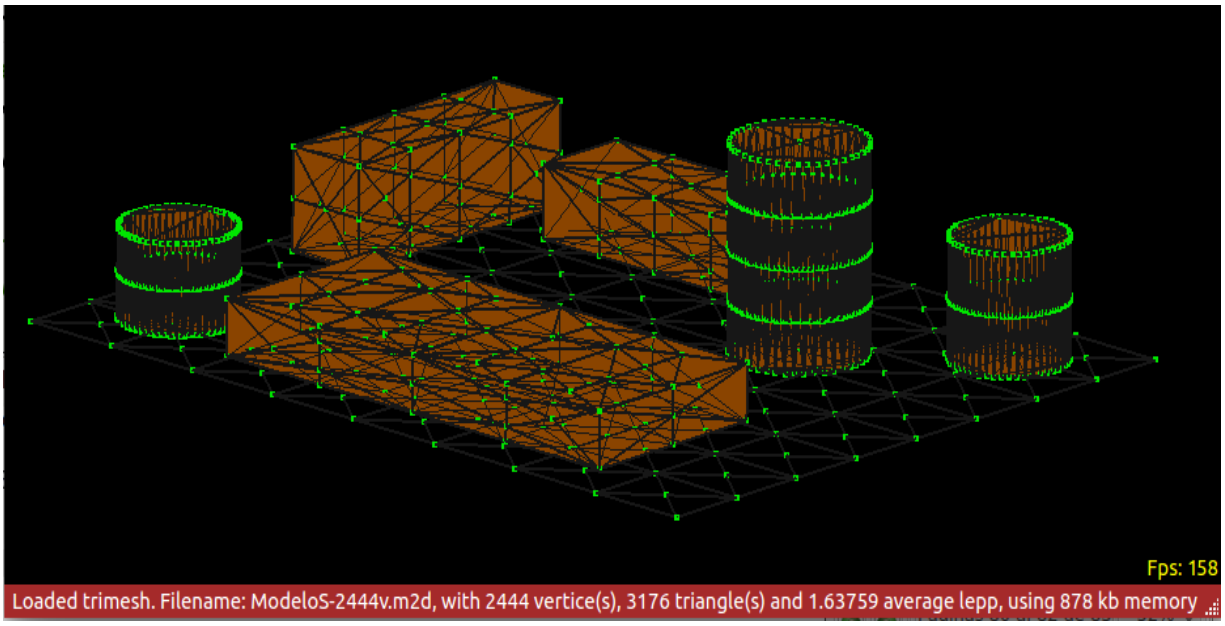


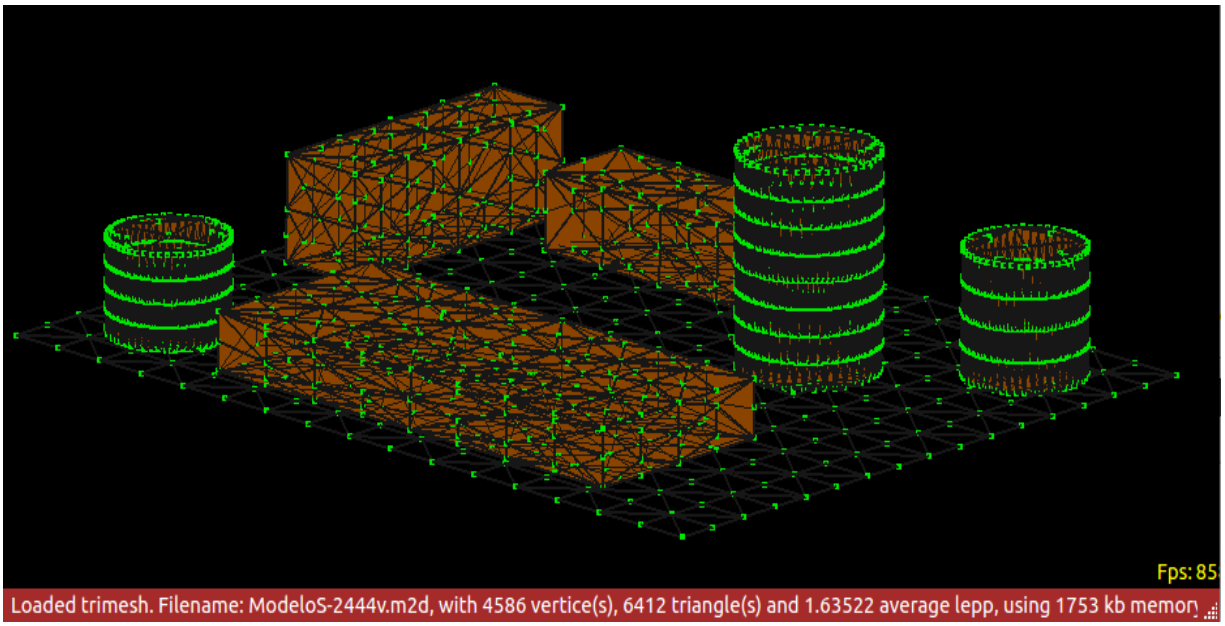
Figura 8.6: Gráfico de frames por segundo.

A medida que aumenta la cantidad de vértices y triángulos, los frames por segundo que es posible obtener es cada vez menor. Esto, debido a que el software consume una mayor cantidad de memoria a medida que aumentan las figuras en el modelo.

Una vez sobrepasado los 100.000 vértices y 180.000 triángulos o más, la cantidad de fps disminuye de tal forma que podría llegar a 0, causando el consumo total de la memoria del equipo y el congelamiento total del software imposibilitando la interacción con el modelo.

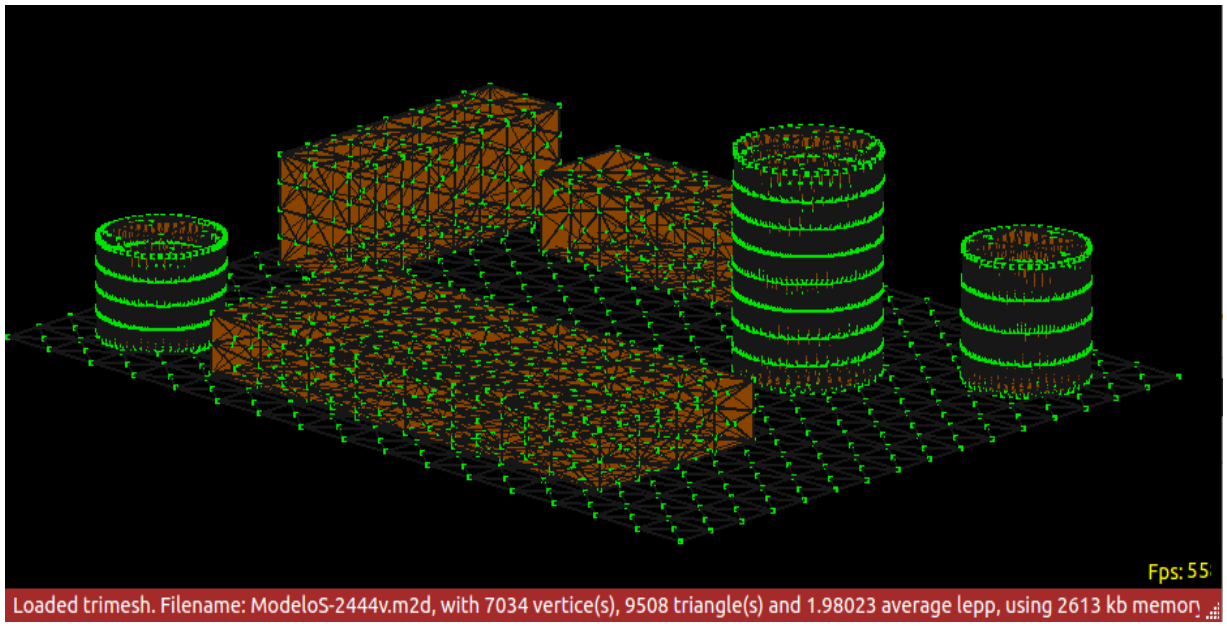


(a)

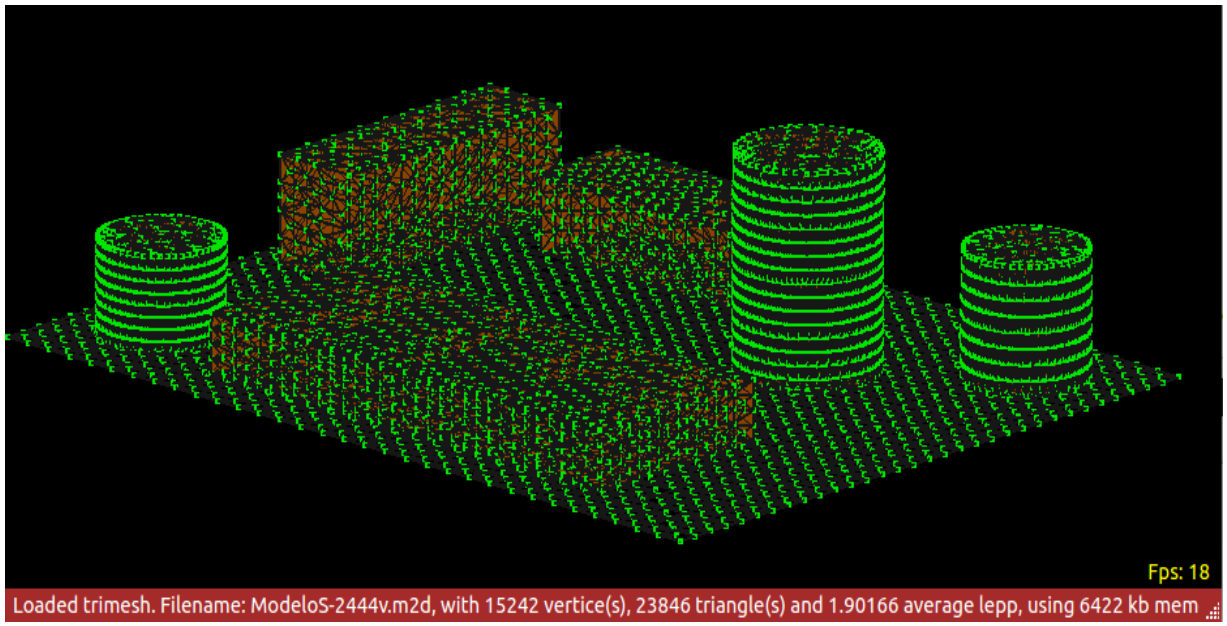


(b)

Figura 8.7: Sobrecarga de triángulos y vértices menor a 7000 y fps.



(a)



(b)

Figura 8.8: Sobrecarga de vértices y triángulos mayor a 7000 y fps.

8.6. Pruebas de selección

Esta sección muestra la funcionalidad de la herramienta de selección sobre el modelo (malla de superficie).

8.6.1. Select by longest edge

Selección por el borde más largo. Una vez cargado el modelo, es necesario ir a la barra de herramientas del visualizador y presionar **Select >> By longest edge** donde se mostrará una ventana con las opciones de selección:

- Smaller, indica que la selección se hará por los valores menores al ingresado.
- Biggest indica que la selección se hará por los valores mayores al ingresado.
- Campo para ingresar valor de selección.

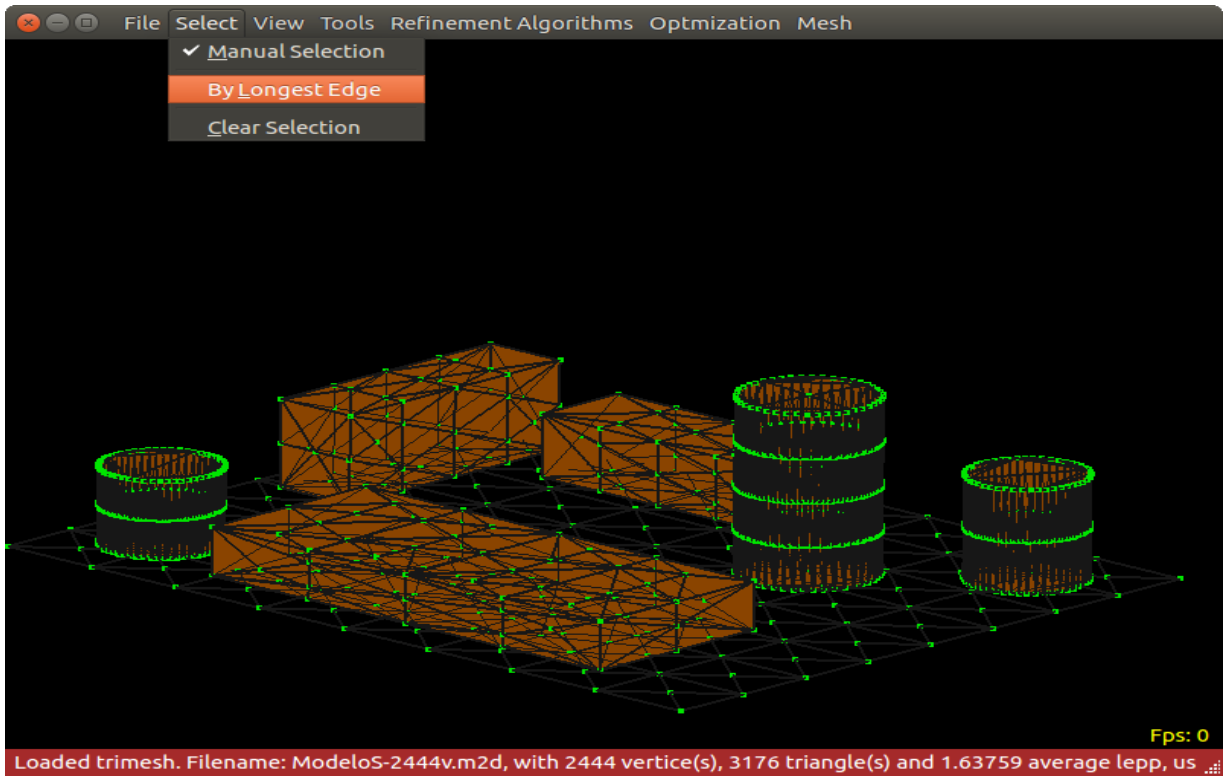
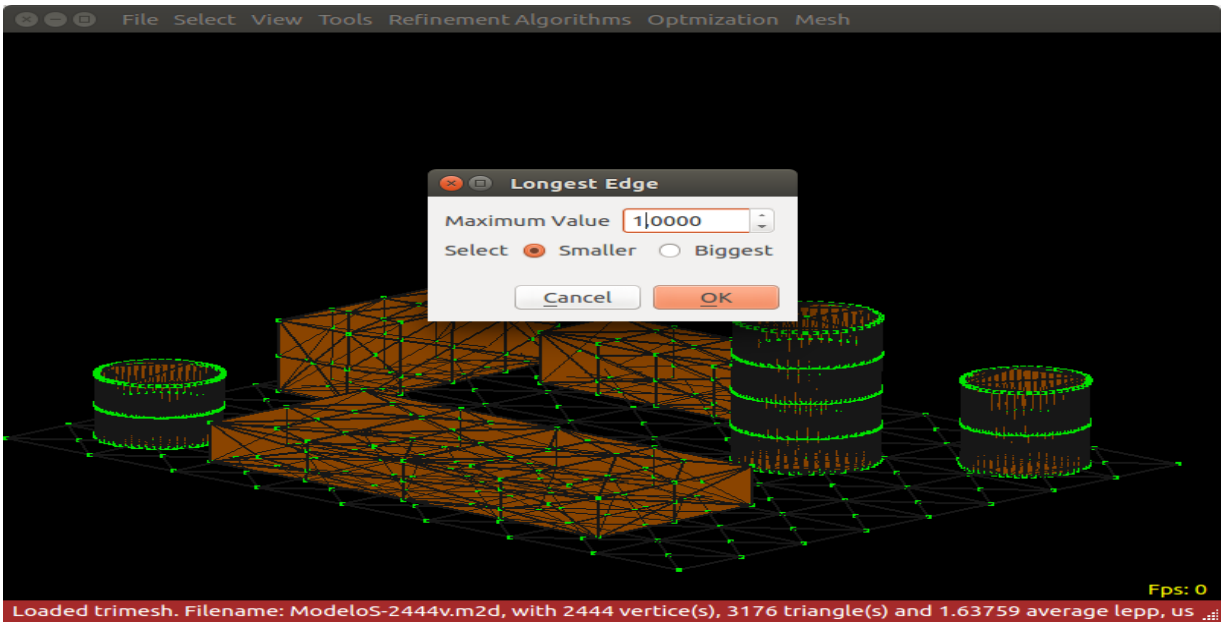


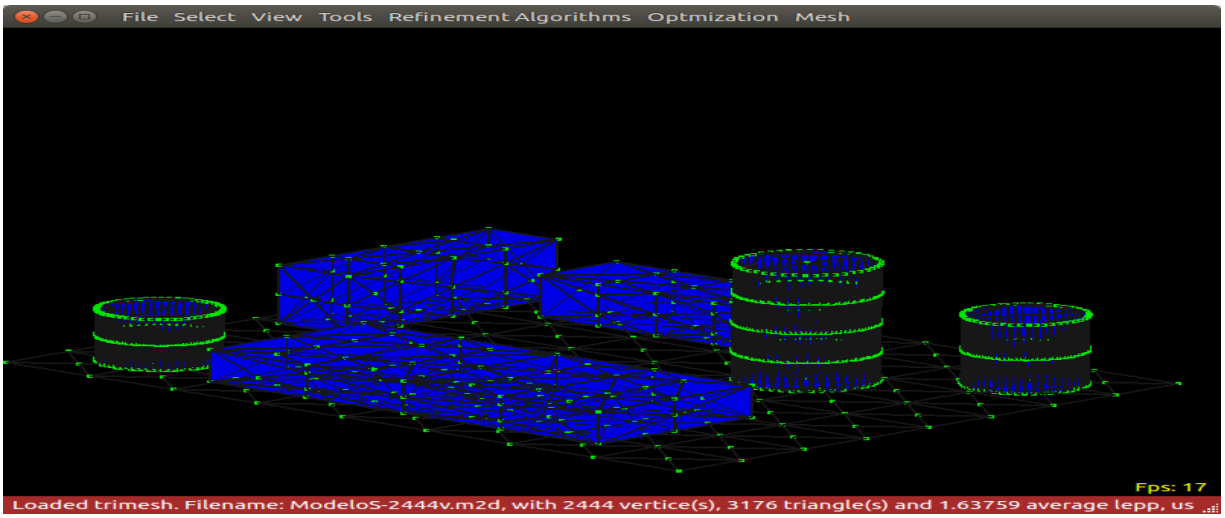
Figura 8.9: Menú select by longest edge en barra de herramientas.

8.6.2. Selección por lado menor a valor

Se seleccionan todos los triángulos cuyas aristas cumplan con la condición, que la arista más larga sea menor al valor ingresado.



(a)

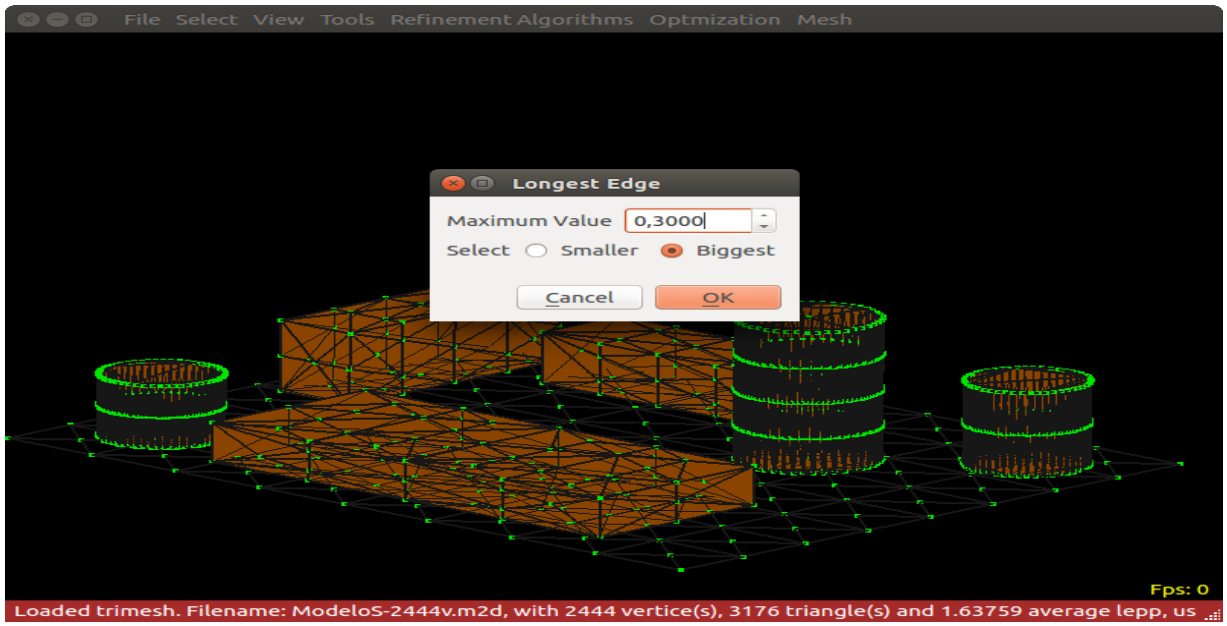


(b)

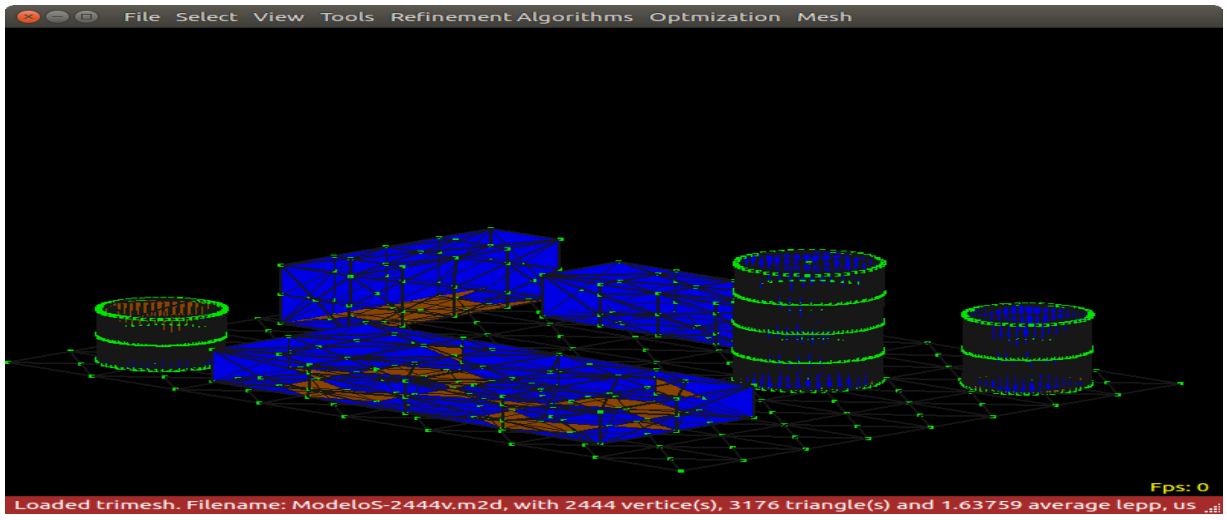
Figura 8.10: En la figura **(a)** se ingresa el valor **1,0** y se selecciona la opción **Smaller**. En la figura **(b)** se pintan todas las figuras correspondientes.

8.6.3. Selección por lado mayor a valor

Se seleccionan todos los triángulos cuyas aristas cumplan con la condición, que la arista más larga sea mayor al valor ingresado.



(a)



(b)

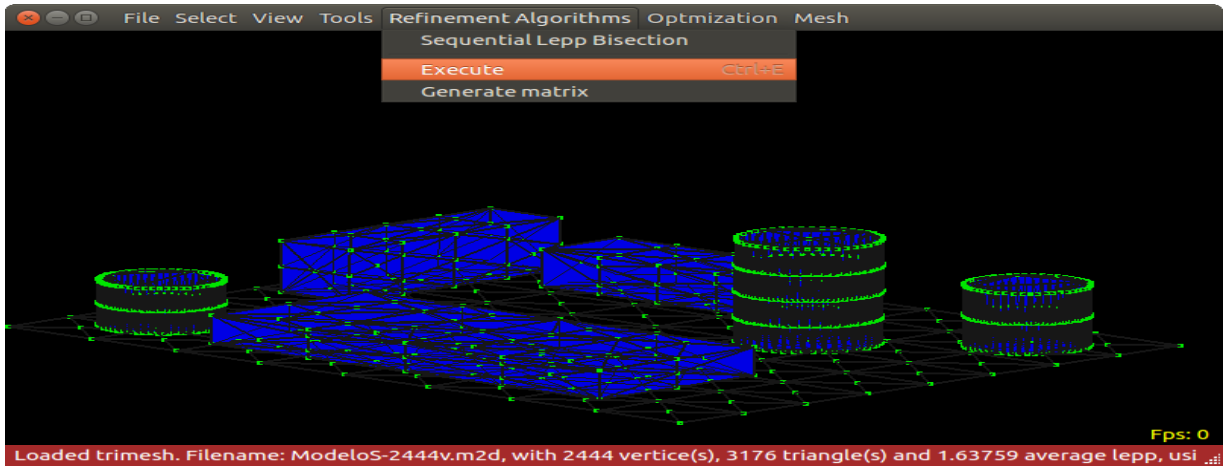
Figura 8.11: En la figura **(a)** se ingresa el valor **0,3** y se selecciona la opción **Biggest**. En la figura **(b)** se pintan todas las figuras correspondientes.

8.7. Prueba de algortimo

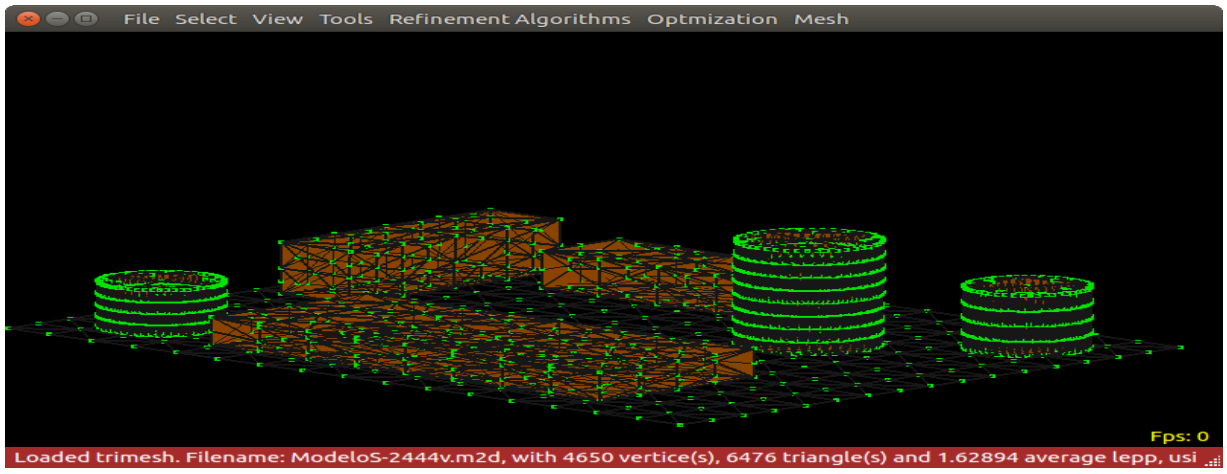
Esta sección muestra el funcionamiento del algoritmo dependiendo de los criterios utilizados en la selección.

8.7.1. LEPP Bisección

Se utiliza el algoritmo de LEPP Bisección dependiendo del tipo de selección.



(a)



(b)

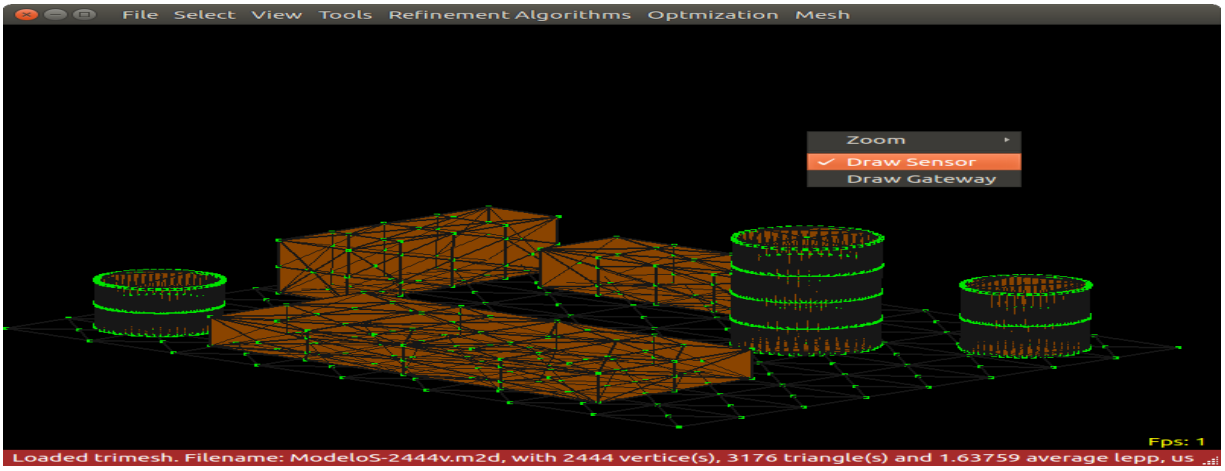
Figura 8.12: En la figura **(a)** están seleccionados todos los triángulos de acuerdo a los criterios de selección y se procede a ejecutar el algorimito. Una vez ejecutado, se aprecia en la figura **(b)** el refinamiento aplicado y la creación de nuevos triángulos.

8.8. Pruebas de interacción con el modelo

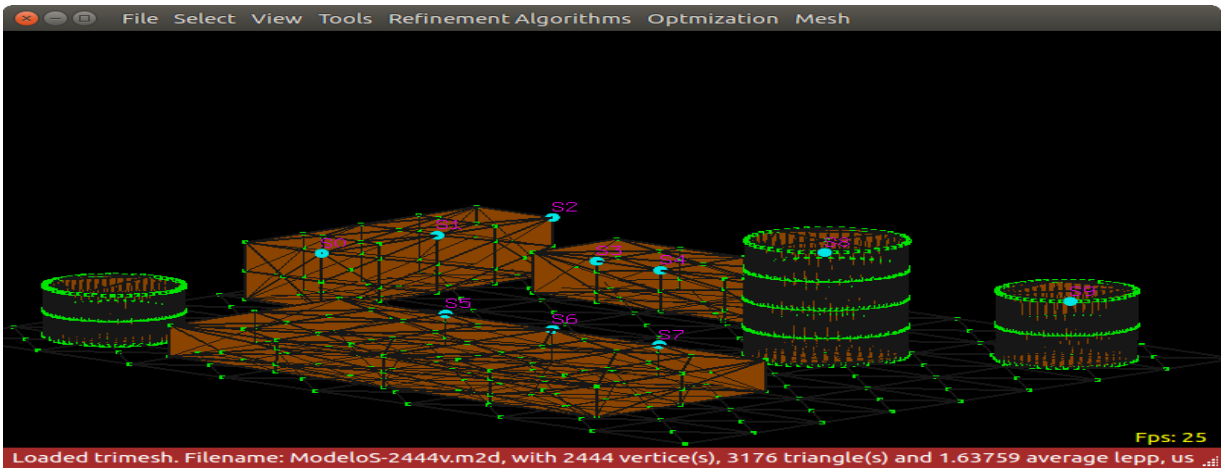
Esta sección muestra el funcionamiento de la interacción con el modelo para el despliegue de Redes de Sensores Inalámbricos a través de un menú accesible con el mouse.

8.8.1. Despliegue de Sensores

Se selecciona la opción Draw Sensor del menú para desplegar Sensores.



(a)

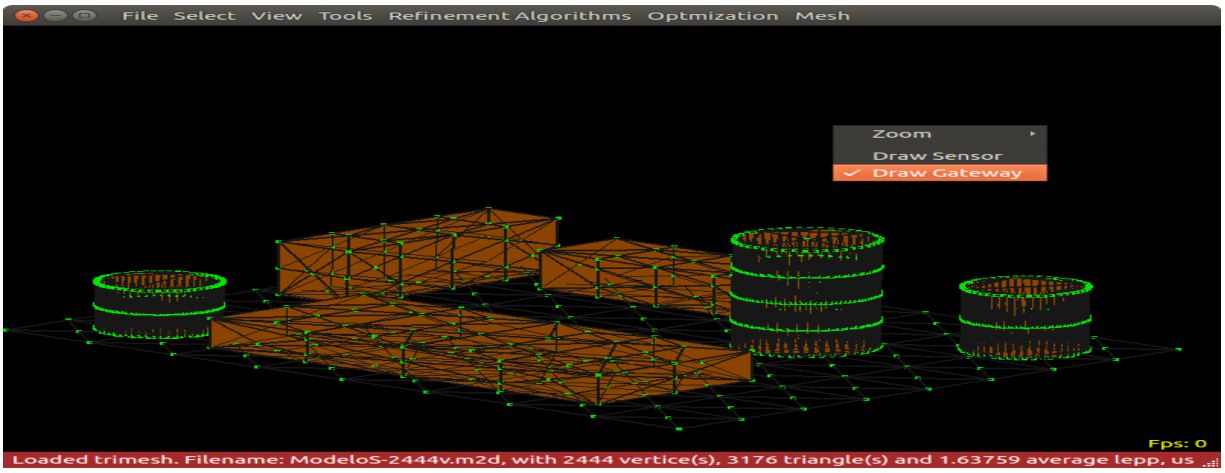


(b)

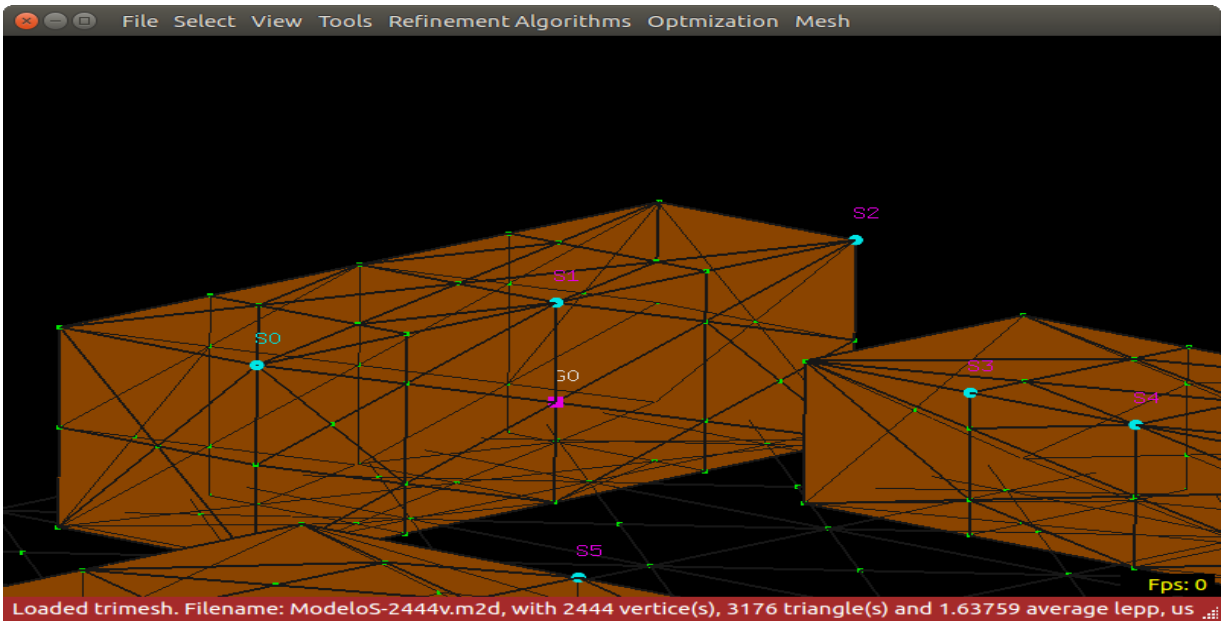
Figura 8.13: En al figura **(a)** se selecciona opción Draw Sensor. En la figura **(b)** se realiza el despliegue mediante la combinación de Shift + Click izquierdo sobre el modelo en algún vértice. Se representa por un circulo color cyan.

8.9. Despliegue de Gateways

Se selecciona la opción Draw Gateway del menú para desplegar Gateway.



(a)



(b)

Figura 8.14: En al figura **(a)** se selecciona opción Draw Gateway. En la figura **(b)** se realiza el despliegue mediante la combinación de Shift + Click izquierdo sobre el modelo en algún vértice. Se representa por un cuadrado de color lila y su identificador de letras blancas.

8.10. Despliegue de Transmisores mediante optimizador

Se selecciona la opción Execute en la sección Optimization de la barra de herramientas para realizar el despliegue. Para esto, es necesario haber realizado previamente el despliegue de Sensores y Gateways.

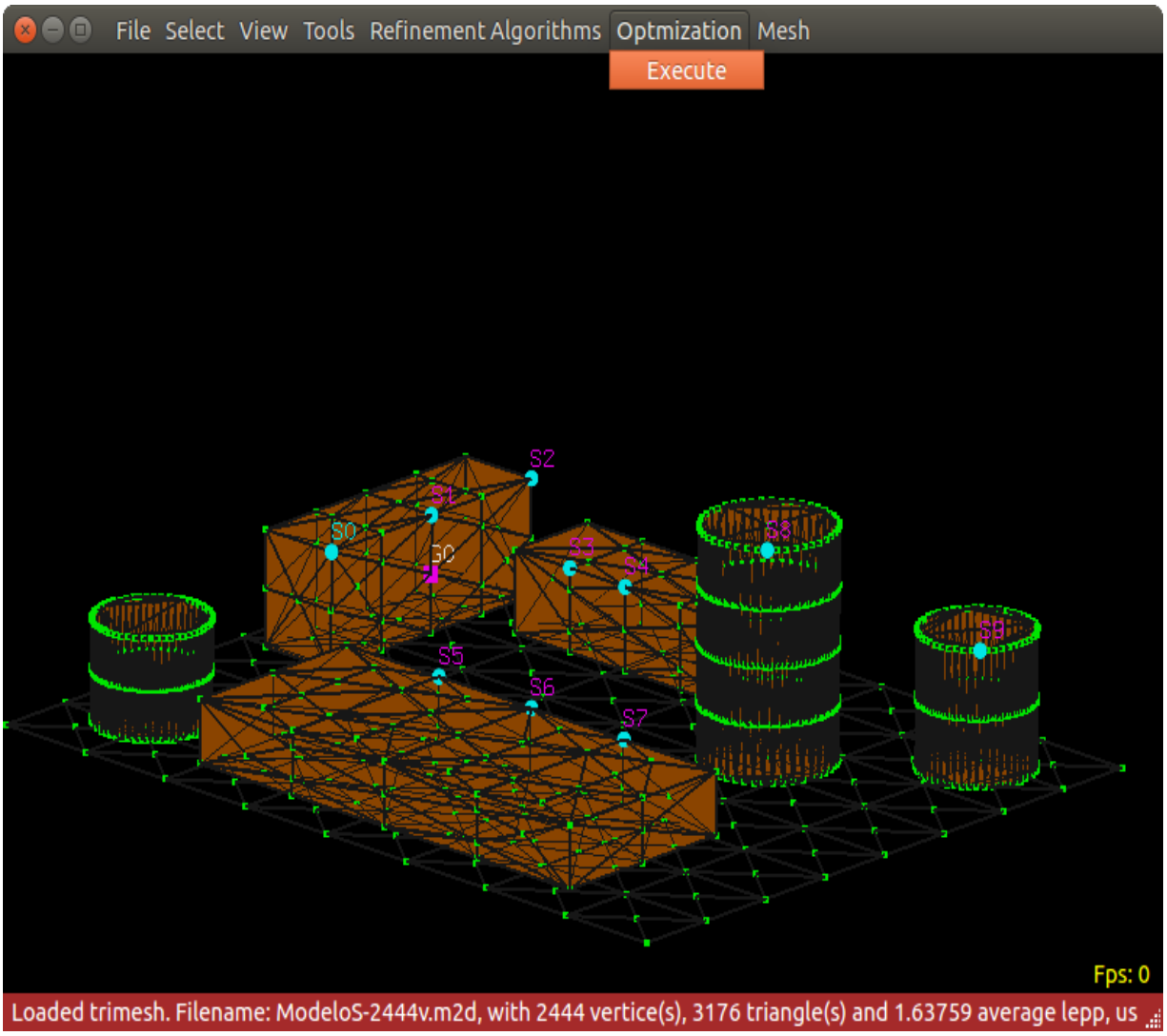


Figura 8.15: Selección optimizador

En la figura 8.15 se selecciona la opción Execute de alojada en la barra de herramientas.

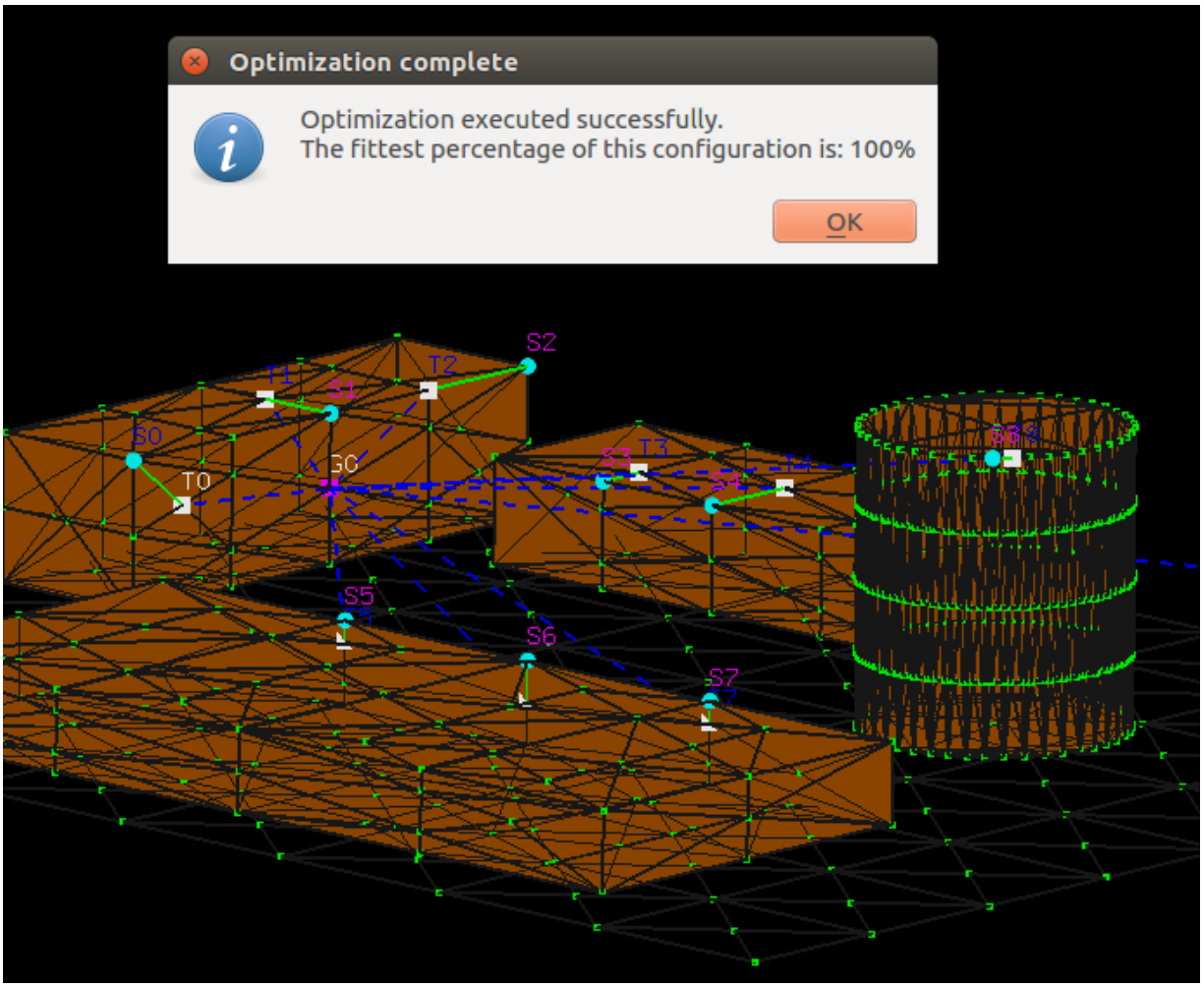


Figura 8.16: Selección optimizador

En la figura 8.16 se ha realizado el despliegue correctamente de los Transmisores, representados por un cuadrado de color blanco y su indicador de letras azules y la conexión con sus respectivos Sensores y Gateways mediante la función de optimizer. Se distingue la conexión entre Transmisor y Sensor mediante una línea verde que indica una conexión cableada y la conexión entre el Transmisor y el Gateway mediante una línea puntuada de color azul que indica conexión inalámbrica.

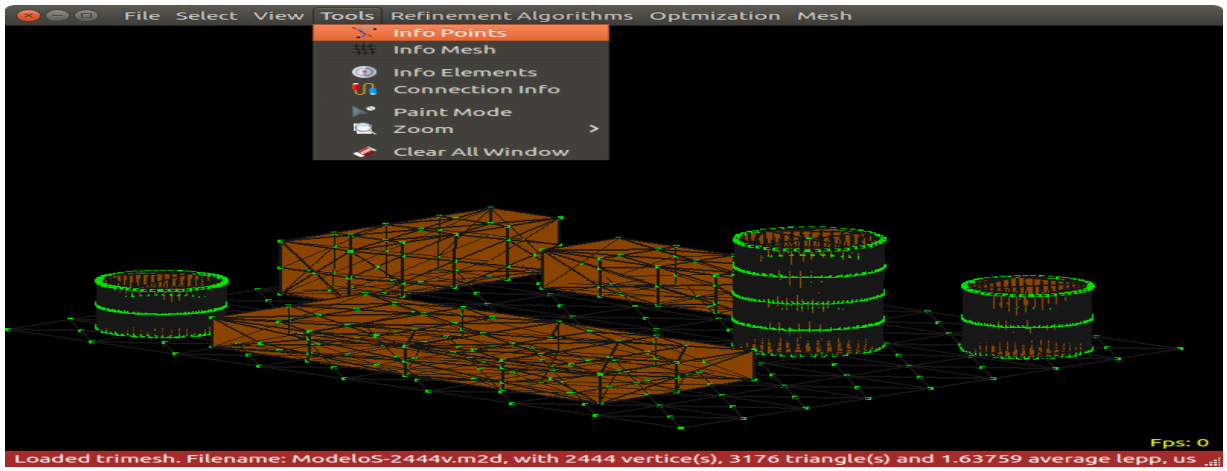
8.11. Pruebas de herramientas de interacción

Esta sección muestra el funcionamiento de las herramientas y funcionalidades para la interacción con el modelo previamente cargado. Las herramientas que se encuentran para la interacción son:

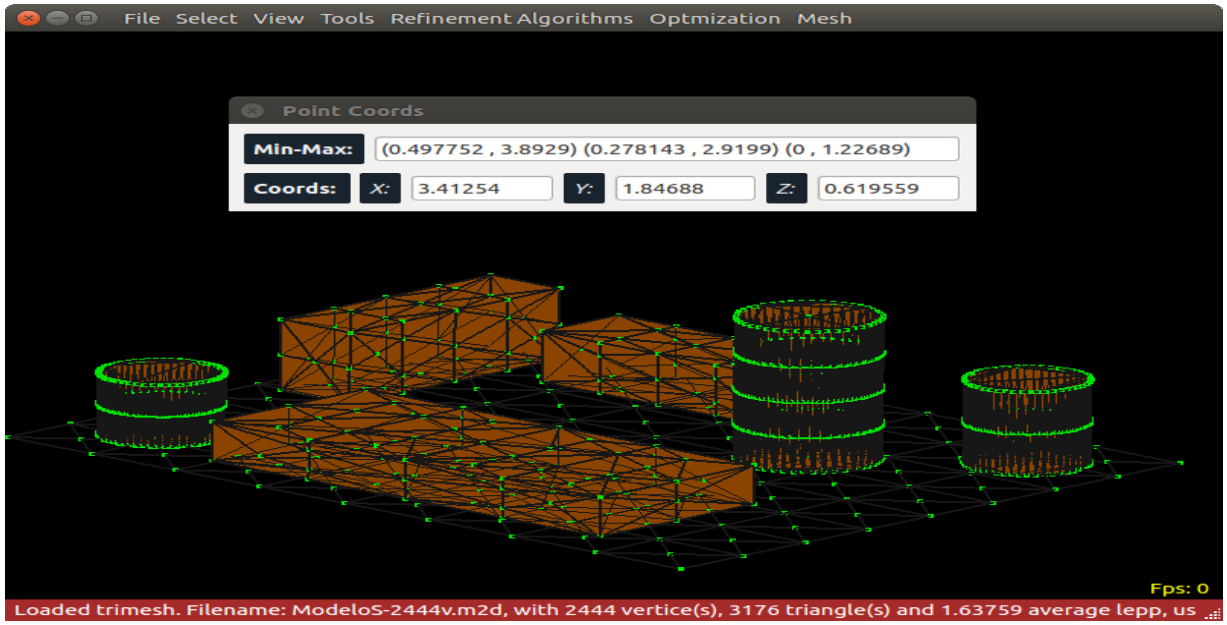
- **Info Points:** Esta herramienta muestra información referente a las medidas del modelo cargado.
- **Info Mesh:** Esta herramientas muestra información del modelo, como las medidas máximas de cada eje y la posición del puntero en el modelo transformadas a metros. También la simbología.
- **Info Elements:** Esta herramienta muestra la información de los objetos Sensor, Gateway y Transmisor, indicando su posición en coordenadas en el modelo. También permite la selección de estos y la eliminación.
- **Connection Info:** Esta herramienta muestra la información referente a la conexión existente de cada Sensor con su Transmisor y Gateway.
- **Paint Mode:** Esta herramienta ejecuta un método que desactiva la interacción con el modelo de rotación mediante el mouse.
- **Zoom:** Esta opción despliega un menú con distintas opciones para realizar Zoom (acercamiento) predeterminado.
- **Clear all window:** Esta opción limpia todos los elementos existentes sobre el modelo, limpiando las estructuras de datos correspondientes y actualizando el modelo.

8.11.1. Info Points

Esta opción se encuentra en la sección Tools de la barra de herramientas.



(a)

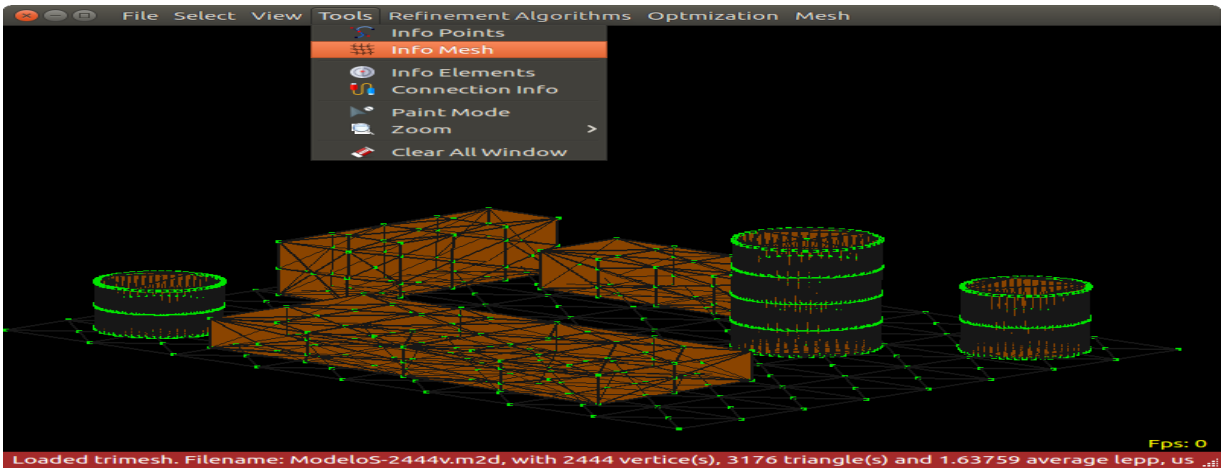


(b)

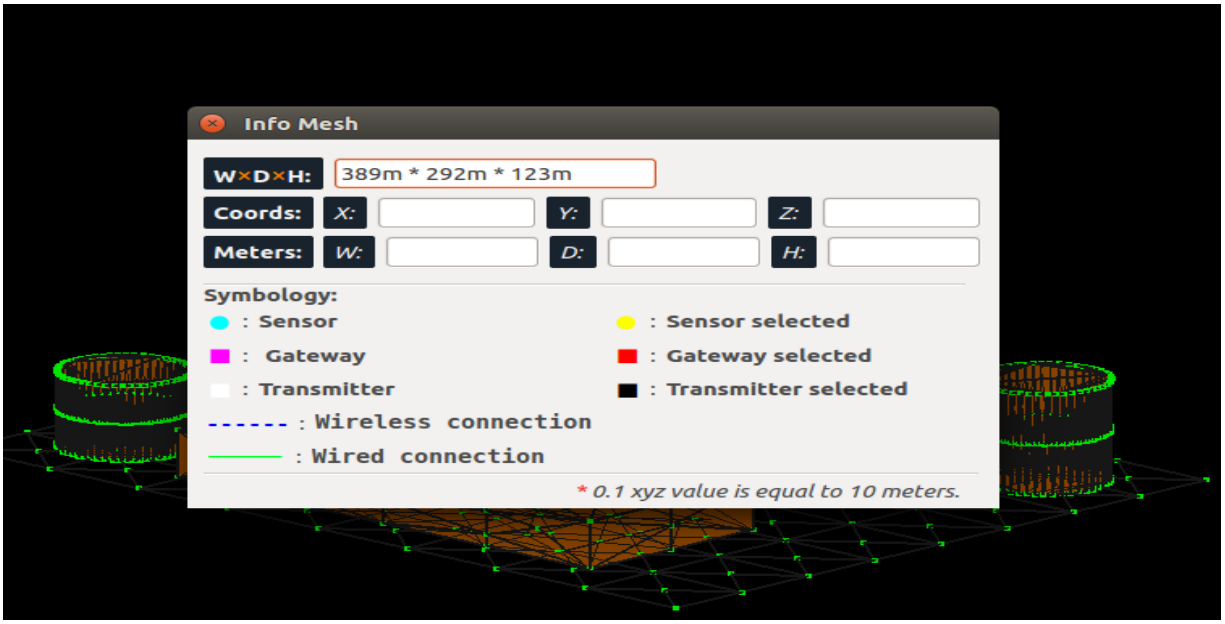
Figura 8.17: En la figura **(a)** se selecciona la opción Info Points. En la figura **(b)** se despliega la ventana con la información de las coordenadas máximas y mínimas para el eje x, y, z del modelo y otro campo llamado coords que indica la posición del punto al realizar click sobre el modelo.

8.11.2. Info Mesh

Esta opción se encuentra en la sección Tools de la barra de herramientas.



(a)

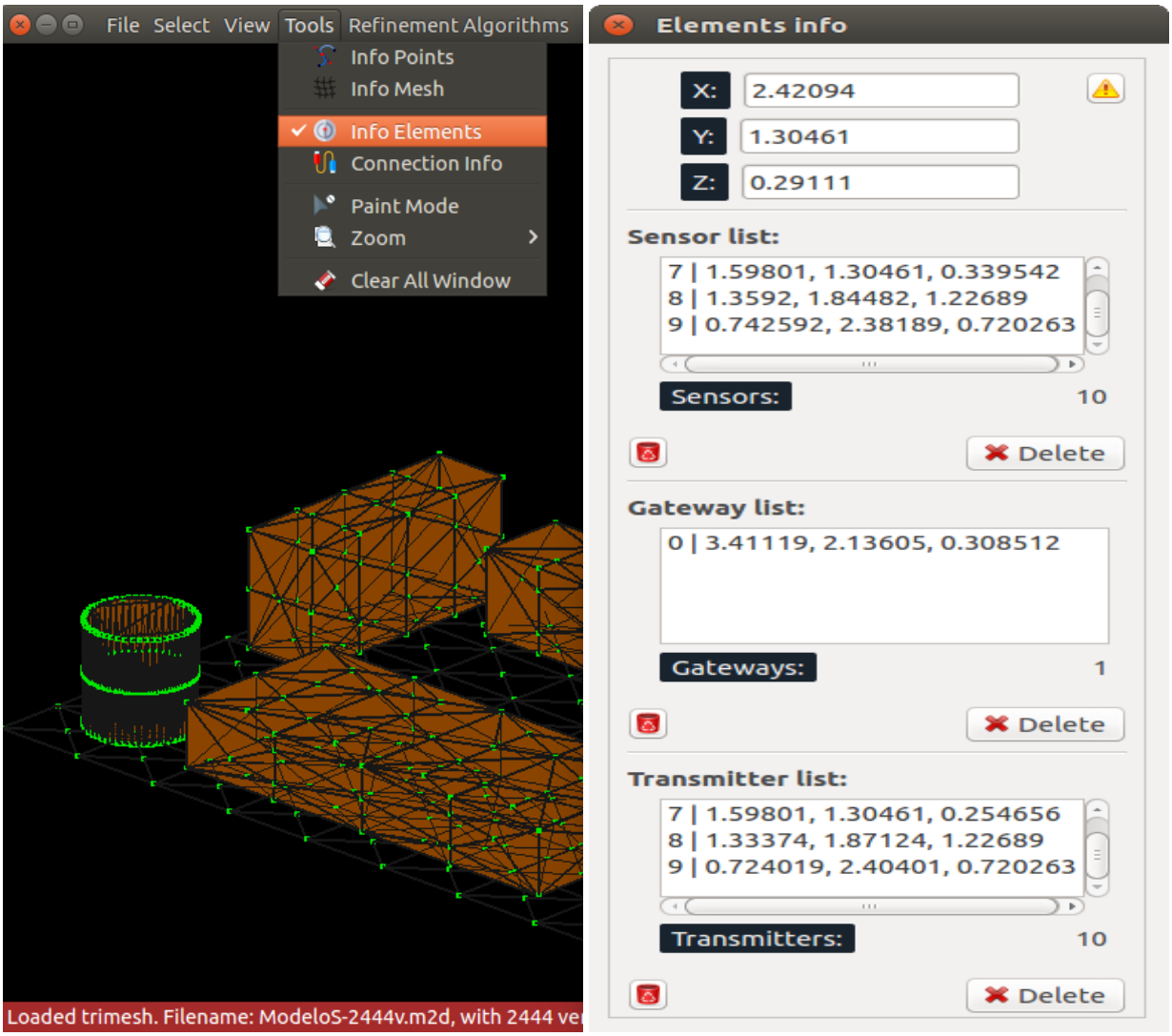


(b)

Figura 8.18: En la figura **(a)** se selecciona la opción Info Mesh. En la figura **(b)** se despliega la ventana con la información del modelo con sus medidas convertidas a metros y una sección de simbología.

8.11.3. Info Elements

Esta opción se encuentra en la sección Tools de la barra de herramientas.

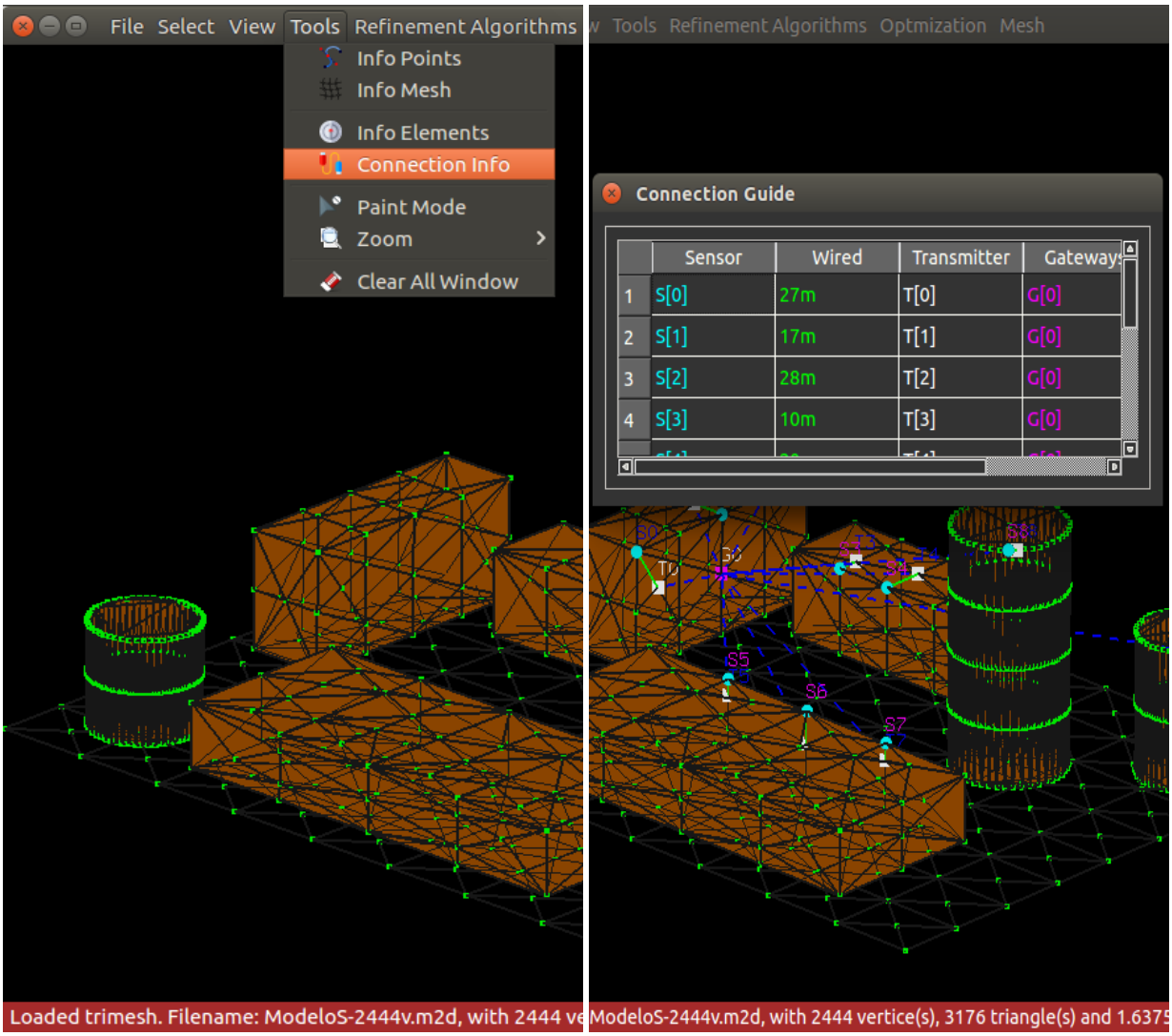


(a) (b)

Figura 8.19: En la figura **(a)** se selecciona la opción Info Elements. En la figura **(b)** se despliega la ventana con la información de los elementos desplegados en el modelo, dando la posibilidad de interactuar con ellos.

8.11.4. Connection Info

Esta opción se encuentra en la sección Tools de la barra de herramientas.

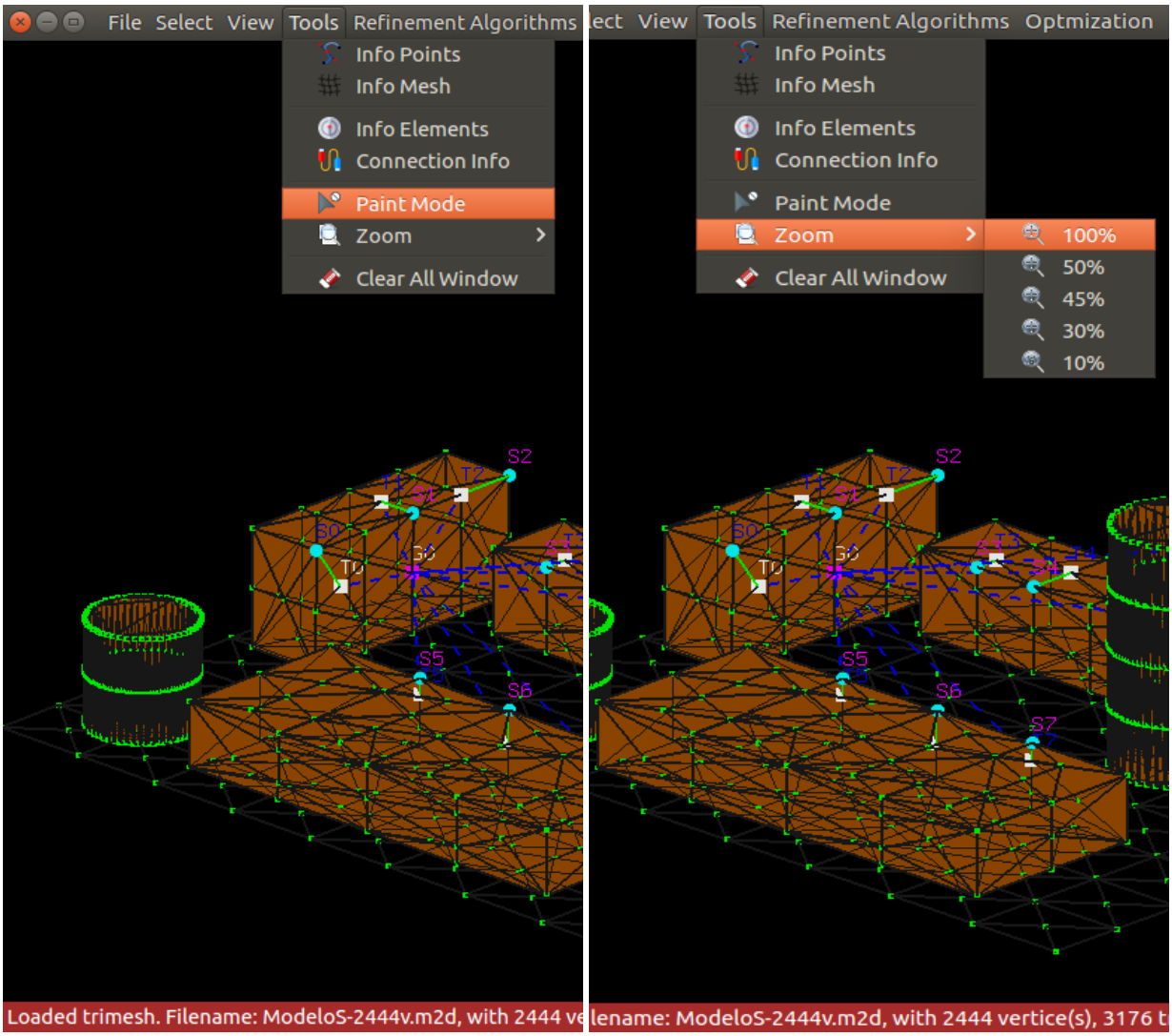


(a) (b)

Figura 8.20: En la figura **(a)** se selecciona la opción Connection Info. En la figura **(b)** se despliega la ventana con la información de los elementos desplegados en el modelo y la conexión existente entre ellos.

8.11.5. Paint Mode y Zoom

Estas opciones se encuentran en la sección Tools de la barra de herramientas.

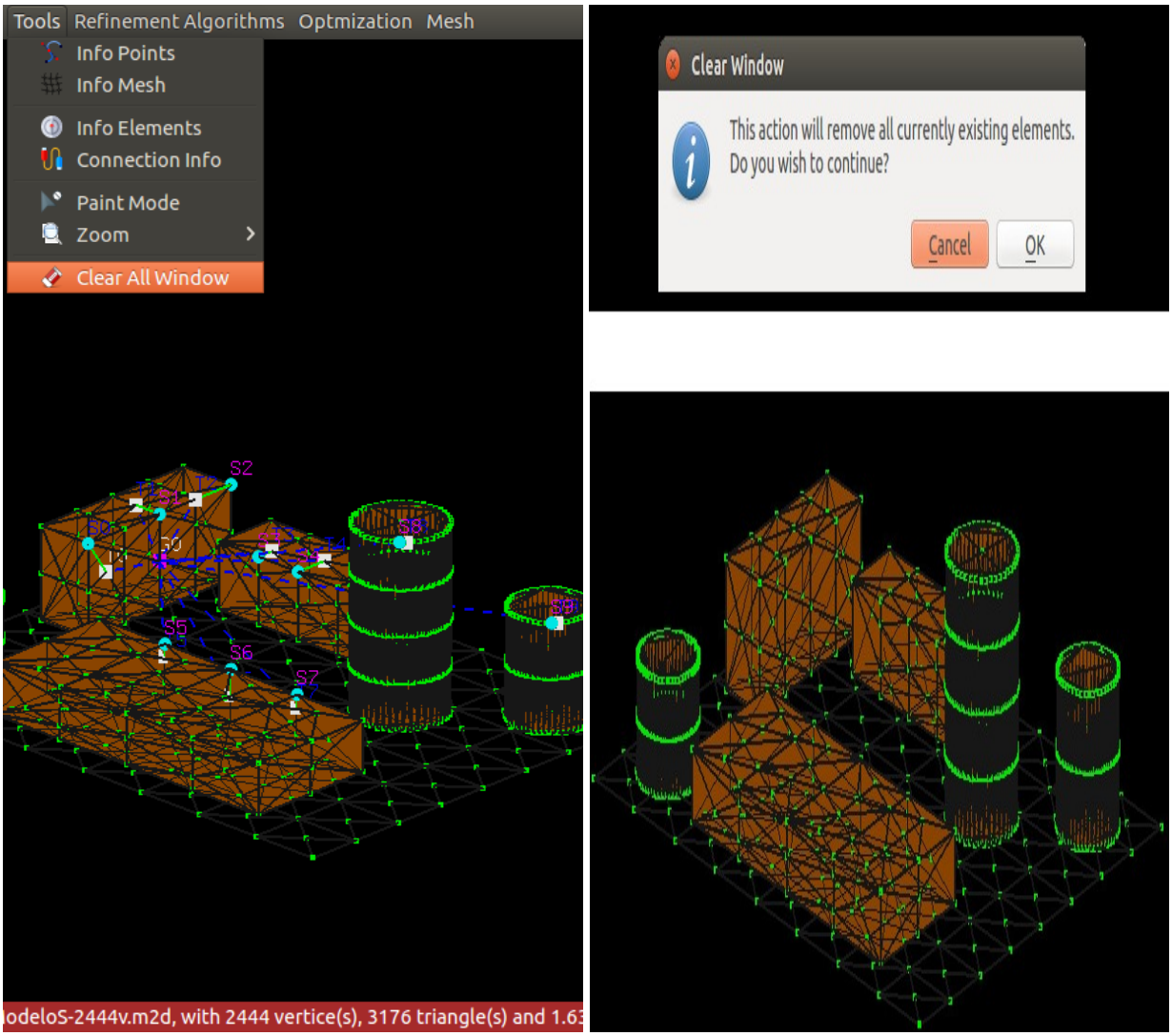


(a) (b)

Figura 8.21: En la figura **(a)** se selecciona la opción Paint Mode que desactiva la rotación del modelo mediante el mouse. En la figura **(b)** se despliega el Zoom menú para realizar un acercamiento con valores predeterminado.

8.11.6. Clear all window

Estas opciones se encuentran en la sección Tools de la barra de herramientas.



(a) (b)

Figura 8.22: En la figura **(a)** se selecciona la opción Clear all Window vacía las estructuras de datos y elimina los elementos del modelo. En la figura **(b)** se despliega un mensaje de confirmación para proceder con la eliminación.

Capítulo 9

Conclusiones

El trabajo realizado en esta memoria de título permite obtener las siguientes conclusiones finales.

Referente al desarrollo:

1. Al desarrollar software en donde se utilizan algoritmos de refinamiento, lo más adecuado para llevar a cabo estas tareas es C++, con bibliotecas de apoyo gráfico como lo es OpenGL para Qt, dado que al estar trabajando bajo este lenguaje de programación es posible obtener una mejor gestión de recursos como la memoria del sistema en donde se ejecute el software. Esto deja en evidencia la eficiencia de algunos lenguajes de programación para ciertas tareas o desarrollos en especifico, además de obtener una escalabilidad mucho mayor y el acoplamiento con bibliotecas que facilitan la tarea, haciendo posible obtener un software más completo. Estas cualidades permiten que la aplicación esté apta para futuras mejoras e integraciones con otros sistemas.

Referente al software:

1. Para el problema de visualización de ambientes industriales, modelados en tres dimensiones con la herramienta de AutoCAD(Ver sección 5.5), en forma de mallas de superficie (Ver cap. 2) es necesario cumplir algunos requisitos importantes, como la exportación en el formato indicado y la aplicación de algoritmos de conversión de

formato sobre ese modelo exportado. Una vez aplicado estos algoritmos, el modelo ya contiene la información necesaria para trabajar e interactuar con él. Este modelo se compone de vértices y triángulos los cuales son necesarios para los algoritmos de refinamiento de mallas (Ver cap. 4). Una malla de triángulos o superficie, es un conjunto de n-vértices, los cuales se unen por aristas que no se intersectan, formado una triangulación.

2. Para el refinamiento de malla, se utilizó el algoritmo de refinamiento y mejoría de malla bajo los conceptos de camino de propagación del borde más largo (Lepp, del ingles Longest edge propagation path) y arista terminal.
   - **Lepp-Bisección:** Algoritmo de refinamiento de triangulaciones por la arista más larga, donde cada triangulo t_n se refina (divide) por la arista más larga, creando dos nuevos triángulos. El proceso se repite hasta refinar el triangulo t_0 siendo este el fin del refinamiento.

3. Para solucionar el problema de despliegue de redes de sensores inalámbricos industriales (Ver cap. 2) se utilizó la biblioteca de Qt OpenGL, puesto que cuenta con los métodos necesarios para la visualización de objetos de forma gráfica. Para la creación de los objetos de tipo Sensor Transmisor y Gateway, se utilizó la estructura de datos de tipo Vector, el cual es capaz de almacenar la información asociada y única de cada objeto y a través de la biblioteca de OpenGL visualizar de forma gráfica para el usuario estos objetos mediante métodos de pintado y repintado sobre el modelo (malla) trabajado.

4. Para la funcionalidad del software se utilizó diagramas de casos de uso, donde cada uno de estos casos de uso están descritos a través de tablas donde se expone la funcionalidad de cada aspecto del software.

5. Para las pruebas de funcionalidad del software realizaron pruebas locales de funcionamiento de los algoritmos y el comportamiento que tiene de forma gráfica al estar en uso con el usuario final.

Posibles trabajos futuros:

Si bien actualmente el apoyo gráfico para la realización del despliegue de redes de sensores inalámbricos (posicionamiento de elementos sobre el modelo), se realiza en base a los vértices existentes, por lo cual para obtener más posiciones para desplegar estos elementos es necesario realizar un refinamiento completo a toda la malla (modelo) lo que causa muchas veces una sobrecarga de memoria de la maquina con la cual se está trabajando debido a la gran cantidad de figuras y vértices que se obtienen con cada refinamiento realizado.

En base a esto, se presentan unas opciones de posibles trabajos a futuros:

- Realizar un despliegue de redes de sensores inalámbricos industriales sin la necesidad de depender de los vértices del modelo para tener más opciones de posicionamiento.

- Realizar un refinamiento de malla selectivo, indicándole solo el área del modelo sobre la cual se desea realizar el refinamiento y así optimizar mucho más la aplicación en términos de tiempo de respuesta y liberar a la maquina en la que se está trabajando de una sobrecarga de memoria debido a la gran cantidad de figuras y vértices obtenidos al refinar.

Referencias

[1] Aakvaag, N., and Frey, J.-E. Redes de sensores inalámbricos. *Revista ABB 2* (2006), 39–42. 2

[2] Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., and Cayirci, E. Wireless sensor networks: a survey. *Computer networks 38*, 4 (2002), 393–422. 10

[3] Al-Yami, A., Abu-Al-Saud, W., and Shahzad, F. Simulation of industrial wireless sensor network (iwsn) protocols. In *Computer Communications Workshops (INFOCOM WKSHPS), 2016 IEEE Conference on* (2016), IEEE, pp. 527–533. 12

[4] Booch, G., Rumbaugh, J., Jacobson, I., Martínez, J. S., and Molina, J. J. G. *El lenguaje unificado de modelado*, vol. 1. Addison wesley Madrid, 1999. 44

[5] Cárcamo, H. K., Fernandez, J. M., Cartes, G. Q., and Cordero, C. V. Estrategia para la localización y posicionamiento de nodos en una iwsn. 10

[6] Chen, B., and Cheng, H. H. Interpretive opengl for computer graphics. *Computers & Graphics 29*, 3 (2005), 331–339. 77, 78

[7] Chust, A. P. Identificacion de riesgos. 34

[8] Dargie, W., and Poellabauer, C. *Fundamentals of wireless sensor networks: theory and practice*. John Wiley & Sons, 2010. 8

[9] David Rocha-Rocha, Francisco Vásquez-Salgado, C. D.-F. R. M. D. D. G. C. P. G. Genetic algorithm for the nodes deployment problem in industrial wireless sensor networks. 1–6. 13

[10] Deif, D. S., and Gadallah, Y. Classification of wireless sensor networks deployment techniques. *IEEE Communications Surveys & Tutorials 16*, 2 (2014), 834–855. 12

[11] Duran-Faundez, C., Costa, D. G., Rocha-Rocha, D., Vásquez-Salgado, F., Habib, G., and Galdames, P. On optimal deployment of industrial wireless sensor networks. *University of the Bío-Bío under Grants DIUBB 161610 2/R and GI 160210/EF, and the Lebanese University under Grant 2015/438.* (2017), 1–7. 11, 13, 14, 15, 16

[12] Gungor, V. C., and Hancke, G. P. Industrial wireless sensor networks: Challenges, design principles, and technical approaches. *IEEE Transactions on industrial electronics 56*, 10 (2009), 4258–4265. 14

[13] Güngör, V. Ç., and Hancke, G. P. *Industrial wireless sensor networks: Applications, protocols, and standards.* Crc Press, 2013. 11, 12

[14] Kanta, P., Prasad, A., and Suma, V. Area coverage redundancy and node positioning in wireless sensor networks. *International Journal of Computer Applications 111*, 5 (2015). 15

[15] Li, S., Sun, H., Nallanathan, A., Xu, L., Zhao, S., and Sun, Q. Industrial wireless sensor networks, 2014. 10

[16] Marschner, S., and Shirley, P. *Fundamentals of computer graphics.* CRC Press, 2015. 16, 17

[17] MORELLI, R. D. Aplicaciones didácticas de modelado de sólidos y vistas automáticas com autocad. In *Anales del "XVIII Simposio Nacional de Geometría Descriptiva y Diseño Técnico y VII International Conference on Graphics Engineering for Artes and Design". GRAPHICA* (2007). 17

[18] Rivara, M. C. Algorithms for refining triangular grids suitable for adaptive and multigrid techniques. *International journal for numerical methods in Engineering 20*, 4 (1984), 745–756. 26

[19] Rivara, M.-C. New longest-edge algorithms for the refinement and/or improvement of unstructured triangulations. *International journal for numerical methods in Engineering 40*, 18 (1997), 3313–3324. 26

[20] Rivara, M. C. Lepp-bisection algorithms, applications and mathematical properties. *Applied Numerical Mathematics 59*, 9 (2009), 2218–2235. 26, 29, 31

[21] Rivara, M. C., and Calderon, C. Lepp terminal centroid method for quality triangulation. *Computer-Aided Design 42* (2010), 58–66. 27

[22] Rivara, M.-C., Rodriguez, P., Montenegro, R., and Jorquera, G. Multithread parallelization of lepp-bisection algorithms. *Applied Numerical Mathematics 62*, 4 (2012), 473 – 488. Third Chilean Workshop on Numerical Analysis of Partial Differential Equations (WONAPDE 2010). 29, 31

[23] Rocha, D., and Vázquez, F. Estudio y evaluación de técnicas metaheurísticas para el despliegue de nodos en redes de sensores inalámbricos industriales. *Universidad del Bío-Bío, Concepción* (2017), 6–7. 9, 14

[24] Rosenberg, I. G., and Stenger, F. A lower bound on the angles of triangles constructed by bisecting the longest side. *Mathematics of Computation 29*, 130 (1975), 390–395. xi, 27, 28

[25] Shreiner, D., Sellers, G., Kessenich, J., and Licea-Kane, B. *OpenGL programming guide: The Official guide to learning OpenGL, version 4.3*. Addison-Wesley, 2013. 77, 78

[26] Suárez, J. P., Plaza, A., and Padrón, M. A. El problema de la propagación del refinamiento en cuatro triángulos por la arista mayor. *Revista internacional de métodos numéricos* (2006). 27

[27] Usamentiaga, R., Molleda, J., and García, D. F. Fast and robust laser stripe extraction for 3d reconstruction in industrial environments. *Machine Vision and Applications 23*, 1 (2012), 179–196. 18

[28] Zanjireh, M. M., and Larijani, H. A survey on centralised and distributed clustering routing algorithms for wsns. In *Vehicular Technology Conference (VTC Spring), 2015 IEEE 81st* (2015), IEEE, pp. 1–6. 9