

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA
SANTIAGO - CHILE



“EXPANSIÓN DE CAPAS HACIA AFUERA EN UNA
MALLA 3D SIN CONSIDERACIÓN DE COLISIONES
NI COLAPSOS”

JUAN IGNACIO ESCALONA PADILLA

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN INFORMÁTICA

Profesor Guía: Claudio Lobos Yáñez
Profesor Correferente: Nancy Hitschfeld Kahler

Julio - 2022

DEDICATORIA

Dedicada a cada uno de mis seres queridos.

AGRADECIMIENTOS

Principalmente, a nivel transversal agradezco profundamente el apoyo de mi profesor guía Claudio Lobos, ya que sin su consejo y conocimiento nada de esto hubiese sido posible. Agradezco también la constante compañía y apoyo de Carolina, mi pareja, quien supo como contenerme y apoyarme incondicionalmente, entendiendo los traspasos y dándome ánimos después de cada fracaso. Finalmente y no menos importante, agradezco a mi familia y seres queridos por la comprensión y apoyo constante.

RESUMEN

Resumen— Una malla geométrica consiste en un conjunto de elementos finitos, ya sean caras o poliedros que se relacionan entre sí generando una estructura de mayor envergadura. La generación de mallas consiste en expandir ya sea hacia el interior o el exterior una malla previamente proporcionada para lo cual existen diversos métodos que se encargan de generar dicha expansión. Sin embargo, no existen métodos claramente definidos en la literatura actual que permitan realizar expansiones hacia el exterior por lo que en este trabajo se presenta una primera aproximación a la expansión hacia afuera de mallas tridimensionales sin considerar colisiones ni colapsos enfocándose principalmente en el cálculo de las normales por nodo para identificar los vectores de expansión. Los resultados obtenidos son altamente relevantes ya que abren una nueva posibilidad de estudio respecto a la expansión exterior de mallas geométricas, sentando las bases para futuros desarrollos.

Palabras Clave— Malla;Nodo;Normal;Mallado;Volumen

ABSTRACT

Abstract— A geometric mesh consists of a set of finite elements, either faces or polyhedra that are related to each other generating a larger structure. Mesh generation consists of expanding either inward or outward a previously provided mesh, for which there are several methods that are responsible for generating such expansion. However, there are no clearly defined methods in the current literature that allow outward expansions, so in this work we present a first approach to the outward expansion of three-dimensional meshes without considering collisions or collapses, focusing mainly on the calculation of the normals per node to identify the expansion vectors. The results obtained are highly relevant since they open a new possibility of study regarding the outward expansion of geometric meshes, laying the foundations for future developments.

Keywords— Mesh;Node;Normal;Meshing;Volume

ÍNDICE DE CONTENIDOS

RESUMEN	IV
ABSTRACT	IV
ÍNDICE DE FIGURAS	VII
ÍNDICE DE TABLAS	IX
INTRODUCCIÓN	1
CAPÍTULO 1: DEFINICIÓN DEL PROBLEMA	2
CAPÍTULO 2: OBJETIVOS	4
CAPÍTULO 3: MARCO CONCEPTUAL	5
CAPÍTULO 4: PROPUESTA DE SOLUCIÓN	11
4.1 Propuesta	11
4.1.1 Tecnología utilizada	11
4.1.2 Código base	11
4.1.3 Arquitectura de la propuesta	12
4.1.4 Valores de entrada y argumentos	12
4.1.5 Limitaciones y consideraciones de valores de entrada y argumentos	12
4.1.6 Preprocesamiento de las entradas	13
4.2 Proyección de normales	15
4.3 Creación de nuevos puntos	17
4.4 Parámetros Opcionales	19
4.4.1 -l : El número de capas a expandir	19
4.4.2 -e : La distancia entre capas a expandir	20
4.4.3 -x : La tasa de distancia entre capas	20
4.4.4 -k: Mostrar las caras o elementos no expandidos	21
4.4.5 -f: <i>Whitelist</i> para expandir caras específicas	22
4.5 Código	24
4.5.1 Lectura de inputs	24
4.5.2 Identificación del tipo de malla	25
4.5.3 Reparación de normales	25
4.5.4 Calcular normales por nodo	26
4.5.5 Asignar nuevos elementos	27
4.5.6 Construir estructuras de salida	29
4.6 Consideraciones de la solución	29
4.7 Resultados esperados	30

CAPÍTULO 5: VALIDACIÓN DE LA SOLUCIÓN	31
5.1 Resultados obtenidos	31
5.2 Mallas de prueba	31
5.3 Resultados generales	32
5.4 Resultados particulares	33
5.4.1 Cantidad de capas expandidas	33
5.4.2 Distancia entre capas expandidas	34
5.4.3 Factor de distancia de expansión	35
5.4.4 Expansión selectiva	36
5.4.5 Consideración de malla original en la salida	37
5.5 Pruebas con malla de gran magnitud	38
5.5.1 Malla de superficie	38
5.5.2 Malla de volumen	40
5.6 Análisis de resultados	43
CAPÍTULO 6: CONCLUSIONES	45
6.1 Críticas al método	45
6.2 Posibles mejoras	46
REFERENCIAS BIBLIOGRÁFICAS	49

ÍNDICE DE FIGURAS

1	Expansión de una malla correspondiente a un cerebro humano.	3
2	Diagrama de Voronoï, con sus regiones coloreadas	7
3	Algunas iteraciones del proceso de <i>advancing front</i> en 2D.	8
4	Cortes de malla de superficie de una lengua	9
5	Tres caras de un hexaedro que comparten el punto p . En dicho caso se debe calcular tres prenormales, una por cada cara involucrada.	16
6	Expansión de una cara para una malla de superficie, teniendo en consideración el cálculo previo de las normales por nodo para todos los nodos de esta	18
7	Visualización del <i>wireframe</i> correspondiente a una expansión de tres capas equidistantes una de otra, con la malla original como superficie solida a modo de referencia.	20
8	Expansión para tres capas con un factor de distancia entre capas de 1,5, se observa que cada capa de expansión está más distanciada que la anterior.	21
9	Expansión generada con el parámetro -k presente, a la izquierda la malla generada como sólido y a la derecha un corte transversal para comprobar la inclusión de la malla original	22
10	Expansión generada sin el parámetro -k, a la izquierda la malla generada como sólido y a la derecha un corte transversal para comprobar la ausencia de la malla original	22
11	Expansión de caras específicas de la malla original, permitiendo seleccionar de manera personalizada las superficies a expandir.	23
12	Muestra de mallas utilizadas. de izquierda a derecha: corte transversal de malla de volumen, superficie de malla utilizada y corte transversal de malla de superficie. . . .	32
13	Resultado general sin aplicar parámetros opcionales al momento de ejecutar el algoritmo.	33
14	Cortes transversales para una expansión de la malla de pruebas utilizando desde una hasta tres capas de izquierda a derecha, respectivamente.	34

15	Resultado obtenido al generar mallas a distinta distancia inter-capas. De izquierda a derecha: utilizando distancia 1; mezcla de ambas mallas para graficar la diferencia de distancia de expansión; utilizando una distancia 2.	35
16	Resultado al aplicar un factor de expansión 1,5, aplicado para tres capas.	36
17	Muestra de expansión selectiva, utilizando diferentes configuraciones de caras a expandir y presentando resultados. Se agrega en cada caso el <i>wireframe</i> de la malla original a modo de referencia.	37
18	Malla resultante con adición de malla original ingresada en el input, malla completa post expansión sin cortes, malla resultante sin consideración de la malla original en la salida, respectivamente.	38
19	Malla superficial de corteza cerebral a la cual se le realizará la expansión, junto con un corte transversal.	39
20	Malla de corteza cerebral con sus expansiones correspondientes.	40
21	Mallas con distinto grado de refinación. A mayor refinación, mayor cantidad de elementos en su composición. Mallas con grado de refinación 3(a), 4(b), 5(c) y 6(d). .	41
22	Malla cs6, con cortes para visualizar su interior.	42
23	Expansión de cs6 sin parámetros adicionales, con cortes para visualizar su interior. .	42
24	Expansión de cs6 con parámetro adicional para conservar la malla original en el archivo de salida, con cortes para visualizar su interior.	43

ÍNDICE DE TABLAS

1	Tabla de referencia para argumentos opcionales.	12
2	Desempeño del código desarrollado para la expansión de una malla superficial de corteza cerebral.	39
3	Características de una serie de mallas volumétricas correspondientes a una corteza cerebral utilizadas para realizar ejecuciones del código desarrollado.	41
4	Desempeño del código desarrollado para la expansión de una serie de mallas volumétricas correspondientes a una corteza cerebral sin considerar la malla original en el archivo de salida.	42
5	Desempeño del código desarrollado para la expansión de una serie de mallas volumétricas correspondientes a una corteza cerebral considerando la malla original en el archivo de salida.	43

INTRODUCCIÓN

Una rama importante dentro del área de ciencias de la computación se encuentra el procesamiento y modelado de elementos finitos, en particular asociado para este trabajo con mallas geométricas tridimensionales.

Este trabajo aborda un proceso nuevo de generación de mallas, el cual consiste en una expansión hacia el exterior dada una malla de entrada, es decir, se generan nuevos elementos de volumen sin considerar colisiones ni colapsos a partir de una malla tridimensional de entrada. Dicha expansión se realiza utilizando como foco principal las normales de los puntos asociados a las caras superficiales de la malla de entrada, junto con permitir una serie de personalizaciones respecto al archivo de salida lo que establece las bases para abrir una serie de líneas de desarrollo y optimización.

En lo que al documento mismo expone, en primer lugar se encuentra presentada la definición del problema junto con los objetivos, fijando en una primera instancia el alcance de lo desarrollado. A continuación se encuentra el marco conceptual, donde se realiza una contextualización de los conceptos utilizados a lo largo del trabajo junto con algunos de los métodos de generación de mallas actualmente presentes en la literatura. Se comienza a enlazar al final del capítulo el procedimiento desarrollado con el actual estado del arte, indicando que la solución propuesta tiene bases teóricas en la generación de mallas de tipo *advancing front*.

Continuando, se expone la solución propuesta. Dicho capítulo es el que incluye en gran medida todo el desarrollo llevado a cabo, problemas encontrados en la marcha y sus soluciones propuestas. A lo largo de éste se encuentran diversas figuras de cada subsección con el objetivo de ilustrar de forma gráfica lo mencionado en cada fragmento del documento. Se encontrará también secciones de código extraídas de la solución real con el objetivo de entregar detalles más técnicos para el lector.

Una vez presentada la propuesta de solución, continúa el capítulo que se encarga de validar dicha solución. En este capítulo es donde se encuentran gran parte de los resultados obtenidos, tanto con una malla de prueba generada artificialmente como con una malla real correspondiente a un cerebro. Finalmente se presentan métricas asociadas al desempeño del código para un caso real.

Para finalizar el documento se presenta un análisis más crítico de la solución propuesta junto con mejoras posibles y trabajos a futuro asociados a este desarrollo. A su vez, se explica brevemente cuales fueron las condicionantes que se pueden mejorar con el tiempo y el cumplimiento de objetivos.

CAPÍTULO 1

DEFINICIÓN DEL PROBLEMA

Dentro del área de estudio correspondiente a las ciencias de la computación existe una rama que se dedica a investigar y desarrollar soluciones asociadas a la generación de un conjunto de elementos finitos. Una de las labores principales dentro de este campo consiste en desarrollar diferentes métodos y mecanismos para generar los elementos finitos requeridos, dentro de los cuales se encuentran las mallas geométricas. Una malla geométrica (para efectos de este trabajo, no estructurada) es una clase de elementos finitos que está compuesta principalmente por puntos y elementos o caras. Dichas mallas se pueden clasificar a grandes rasgos en dos grandes tipos: mallas de superficie y mallas de volumen; radicando su diferencia en relación al tipo de elementos que componen su estructura.

En lo que a computación gráfica se refiere existe toda una rama de estudios que se dedica a investigar y desarrollar diversos métodos de generación de mallas. Este tipo de trabajos se vuelve relevante, entre otros motivos, para realizar simulaciones y modelados especialmente en ocasiones donde no se puede realizar el experimento o prueba deseada con modelos reales (edificios o puentes, seres vivos o secciones de ellos, o un vehículo, por ejemplo). Siguiendo este lineamiento es que se requiere, para algunos estudios, realizar una generación de mallas mediante la expansión de capas aplicado sobre mallas de superficie para por ejemplo, simular la piel de una extremidad dada su estructura ósea o muscular.

Con el objetivo de conseguir lo anterior es que nacen diversos métodos de generación de mallas, sin embargo todos los métodos documentados hasta el momento tienen un foco distinto a la expansión exterior, dado que se encargan principalmente de construir una malla de volumen sin huecos a partir de una malla cuya estructura no es completamente rellena, permitiendo así realizar algunos trabajos de simulación. Es por este motivo que el presente trabajo busca explorar la expansión de mallas geométricas hacia afuera en lugar de hacia adentro, en una primera instancia sin considerar colisiones ni colapsos, estudiar su comportamiento, resultados obtenidos y un breve análisis de efectividad en cuanto al método.

Tal y como se mencionó anteriormente, existen dos grandes clasificaciones de mallas, por lo que la solución propuesta deberá contar con la compatibilidad para ambos tipos de mallas, es decir, el código a desarrollar deberá ser capaz de recibir como entradas tanto mallas de superficie como mallas de volumen, involucrando todo el preprocesamiento correspondiente con el objetivo general de poder generar una malla de volumen correspondiente a la expansión de la malla original. Sumado a lo anterior se obtendría otro problema que se busca solucionar junto con la expansión, que consiste en la rigidez de la malla de salida en relación al poder de personalización que tiene el usuario respecto al archivo de salida. Para esto, se desarrollarán una serie de parámetros opcionales que permiten, bajo ciertas limitaciones, personalizar la malla resultante.

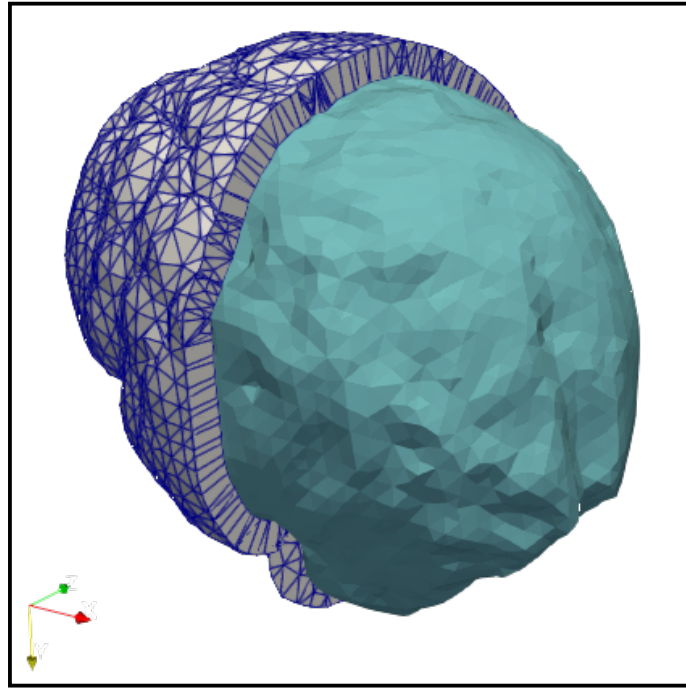


Figura 1: Expansión de una malla correspondiente a un cerebro humano.

El proceso a desarrollar plantea un desafío y la apertura a nuevas perspectivas en relación a la generación de mallas, ya que no existen trabajos previos que realicen lo que se busca obtener. La posibilidad de abrir toda una línea de investigación al respecto lo vuelve un problema a resolver de gran importancia para el campo, sentando las bases para ampliar la expansión de mallas hacia afuera de un conjunto de elementos finitos respetando su estructura.

CAPÍTULO 2

OBJETIVOS

A modo de síntesis, este trabajo tiene como objetivo principal desarrollar de manera satisfactoria un programa que genere una malla de volumen sin considerar colisiones ni colapsos a partir de un archivo de entrada, el cual puede corresponder tanto a una malla de superficie como a una malla volumétrica.

La expansión generada no considera el estudio y verificación de colisiones ni colapsos entre las diferentes caras de la expansión y se remite sólo a realizar la expansión de una malla superficial, sin embargo se plantea como objetivo secundario que el algoritmo sea aplicable a la mayor cantidad de casos de estudio a nivel de archivos de entrada, por lo que debe ser capaz de recibir tanto una malla de superficie como una malla de volumen y el resultado debe ser consistente.

Dentro de los objetivos secundarios, se tiene lo siguiente:

- Leer más de un tipo de archivo de mallas de entrada.
- El algoritmo debe aceptar tanto mallas volumétricas como superficiales y el resultado entregado debe ser consistente en ambos casos.
- Calcular las normales de los nodos considerando los casos generales, es decir, caras con la posibilidad de poseer más de una normal y caras no necesariamente co-planares.
- Asegurar consistencia entre las caras de la malla, sin importar la naturaleza de la topología de la malla.
- Incluir variadas opciones de personalización para el archivo de salida.

CAPÍTULO 3

MARCO CONCEPTUAL

Dentro de las ciencias de la computación e investigación que trabaja con elementos finitos existen una variedad de estudios y herramientas útiles que permiten desarrollar diversos trabajos e investigaciones. La rama de estudio que se aborda en esta memoria hace referencia particularmente a la generación de mallas.

La generación de mallas se puede clasificar en dos tipos dependiendo de su naturaleza de origen, siendo éstas la generación estructurada y no estructurada. Las mallas estructuradas son aquellas en las que la ubicación de cada elemento está dado por alguna herramienta matemática, función o un sistema de ecuaciones, donde no importa conocer la ubicación exacta de cada punto para conocer la topología y estructura de la malla; mientras que por otro lado se tiene la generación de mallas no estructuradas, donde se requiere la ubicación de cada punto de manera previamente establecida para poder identificar la topología y propiedades de la malla en cuestión.

Para entender cómo se construyen y generan dichas mallas se requiere realizar una serie de definiciones presentadas a continuación.

Punto/Nodo/Vértice Un punto se entiende como un conjunto de n valores reales, donde n corresponde a la dimensión donde existen. Para los fines de este trabajo donde se hará referencia todo el tiempo a estructuras tridimensionales cada punto está compuesto por tres valores reales, los cuales se encuentran situados en algún punto del espacio tridimensional.

Cara/Superficie Una cara es una superficie acotada por conjunto de p puntos. La cantidad de puntos que componen una cara determina la cantidad de aristas que posee y su orden de presentación guarda especial relevancia ya que la interpretación de todos los *software* de visualización de mallas utilizan dicho orden para orientar y construir los elementos involucrados. Si el orden no se encuentra bien establecido, al momento de realizar la visualización se observarían cuerpos inconsistentes. Por otro lado, el orden horario o antihorario al momento de definir los puntos que conforman una cara determina la orientación de su normal respectiva, lo cual guarda especial relevancia para este trabajo. Generalmente las caras a trabajar son caras compuestas por tres o cuatro puntos, es decir, observamos en su mayoría triángulos y cuadriláteros.

Normal La normal corresponde al producto vectorial entre dos puntos. Como se mencionó anteriormente, la normal al ser un vector posee dirección, sentido y magnitud; y se representa mediante un punto tridimensional. La normal de las caras superficiales de una malla y su correcta orientación es crucial para el desarrollo del trabajo que se quiere llevar a cabo, dado que se calculará la normal por nodo utilizando las normales por cara.

Elementos de volumen Los elementos de volumen corresponden a todo cuerpo volumétrico sólido compuesto por puntos. Al igual que las caras, un elemento de volumen se encuentra compuesto por puntos ordenados, sin embargo su estructura es ligeramente más compleja ya que estos elementos deben almacenar datos adicionales con el objetivo de eliminar incongruencias. Dichos datos adicionales permiten identificar de manera clara y consistente sobre qué elemento se está trabajando. En general, los elementos de volumen más comunes corresponden a **pirámides**, **tetraedros**, **prismas** y **hexaedros** los cuales poseen cuatro, cinco, seis y ocho puntos respectivamente. Para cada elemento de volumen, al igual que para la definición de caras, el orden en el que se declaren los puntos es crucial. En el caso que los elementos de volumen no se encuentren con sus puntos en un orden correcto, se obtendrán cuerpos extraños e indeseados, en lugar de los resultados esperados.

Concavidad En el contexto de generación de mallas, la concavidad tiene relación con la presencia o ausencia de secciones “hundidas” desde la vista del observador. Para este trabajo si bien se utilizan tanto mallas convexas como cóncavas, en el caso de las mallas cóncavas se producen colisiones y colapsos entre expansiones las cuales no están cubiertas dentro del alcance de la memoria. La concavidad determina características en las normales de las caras que proyectan eventuales choques entre ellas, mientras que en una malla convexa las normales están todas estructuradas de tal forma que se mitigan las colisiones dada la estrategia utilizada en el desarrollo.

Generación de elementos Dada una malla original con un conjunto de puntos y un conjunto de elementos o caras, existe la posibilidad mediante escritura del archivo de salida agregar y/o modificar los elementos a generar. La generación de elementos nuevos consiste en definir secuencias de puntos con su respectivo orden para posteriormente agregar los puntos nuevos a la lista de puntos y los elementos nuevos a la lista de elementos.

Malla Una malla (o *mesh*, en inglés) consiste en un conjunto de elementos de volumen (para las mallas de volumen) o de caras (para las mallas superficiales) relacionados entre sí con el objetivo de conformar una estructura de mayor magnitud. Ambos tipos de mallas difieren principalmente en el tipo de estructuras que utilizan para ser construidas, además de utilizarse para distintos objetivos. Mientras una malla de superficie se utiliza principalmente para modelado y visualización, una malla de volumen se utiliza para realizar simulaciones, ya que al poseer mayor cantidad de elementos en su interior permite emular situaciones reales como por ejemplo impactos entre dos objetos.

Cascarón Cascarón hace referencia a la estructura superficial de una malla. Para el caso de una malla de superficie el cascaron corresponde a la misma malla íntegra, mientras que para una malla de volumen corresponde a su capa superficial o corteza externa.

Como es de esperarse, dentro de la literatura se han desarrollado diversos métodos y algoritmos que se encargan de generar mallas tanto bidimensionales como tridimensionales, sin embargo todos se especifican en generación de mallas interiores por sobre exteriores. Algunos de los métodos de mallado presentes en la literatura son los siguientes:

Triangulación de Delaunay La triangulación de Delaunay esta basada en el diagrama de Voronoï, el cual postula la existencia de regiones dentro de una malla tal que cualquier punto de la malla en cada región se encuentre más cerca de un punto P_i que a cualquier otro punto. Luego, la unión de estos puntos P_i par a par construyen la triangulación de Delaunay, la cual tiene la característica de que es óptima en relación a la regularidad de los ángulos de cada uno de los triángulos generados con la unión de los puntos para mallas 2D, mientras que para mallas 3D requiere un procesamiento extra de los ángulos de los tetraedros formados para asegurar regularidad.

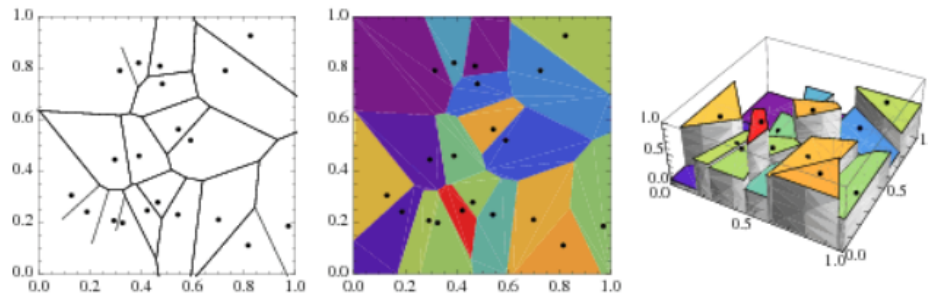


Figura 2: Diagrama de Voronoï, con sus regiones coloreadas

Fuente: <https://mathworld.wolfram.com/>.

Advancing front La generación de mallas mediante *Advancing front* recibe como input una malla de superficie con el objetivo de realizar una expansión interior utilizando las caras proporcionadas por la iteración anterior (siendo la primera iteración el input inicial) con el objetivo de construir una malla de volumen. Este método considera una regularidad al momento de expandir el frente hasta que comienza a detectar colisiones entre ellos, momento en el que comienza a adoptar elementos geométricos como prismas o pirámides para poder adaptar de mejor manera el resultado final, teniendo en consideración los casos de colisiones correspondientes.

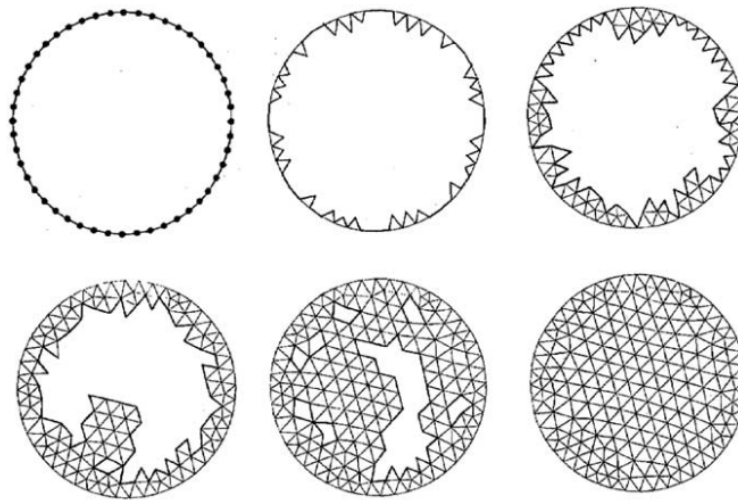


Figura 3: Algunas iteraciones del proceso de *advancing front* en 2D.
Fuente: [Davies *et al.*, 2007].

Normal offset *Normal offsetting* es una variante de *Advancing Front* con la salvedad de que asegura que en emplazamiento sera paralelo al borde original. Para aplicar este método de generación de mallas se debe tener una serie de consideraciones para sus datos de entrada, como lo son el conjunto que representa el frente a expandir, la altura del primer elemento, un factor de expansión y número máximo de capas. Tal y como se mencionó anteriormente, lo que se busca conseguir al realizar *normal offsetting* es que las capas generadas sean paralelas a las capas iniciales y para conseguir este objetivo se requiere obtener los puntos imágenes de los previamente existentes y las normales de las caras correspondientes.

Métodos basados en Octree Los métodos basados en *Octree* utilizan una estructura de datos de árbol, lo que permite resolver búsquedas en tiempos logarítmicos utilizando rangos [Jaillet y Lobos, 2021]. *Octree* es una extensión de una estructura llamada *Quadtree*, donde para un espacio en dos dimensiones se divide el espacio en cuadrantes. En el caso de *Octree*, al tratarse de una estructura para representar tridimensionalidad, el espacio se divide en octantes. Este método es sumamente útil cuando se desea discretizar un modelo permitiendo ajustar en nivel de detalle según se desee.

Dentro de la literatura existente, podemos encontrar una metodología de generación de mallas basada el *Advancing front* [Valdés, 2012], la cual recibe como input una malla de superficie y entrega como output una malla de volumen, generando todo el mallado al interior de la superficie inicial utilizando tetraedros, prismas, pirámides y hexaedros dentro de su composición como figuras elementales para el caso tridimensional. Dicho método tiene por objetivo expandir hacia el interior del objeto una serie de frentes hasta obtener una malla de volúmenes consistente con las restricciones

correspondientes. Para llevar a cabo lo anterior, dicho método utiliza las normales por cara para realizar la proyección, indicando debidamente su dirección de propagación de frentes.

Uno de los nichos de estudio habituales en la generación de mallas corresponde al área biomédica, donde existe una gran cantidad de aplicaciones utilizando información de entrada como las resonancias magnéticas. Dichas imágenes son procesadas para luego generar una malla de órganos o secciones biológicas como por ejemplo en [Lobos *et al.*, 2010] y [Samani *et al.*, 2001] y finalmente suelen ser suavizadas para simular de mejor forma el órgano en cuestión.

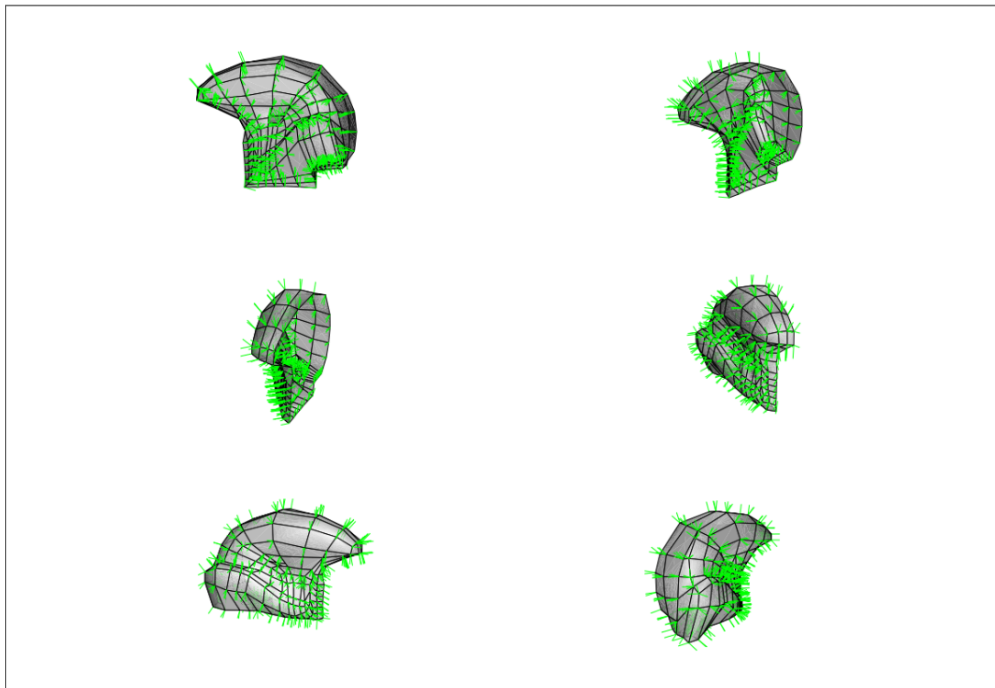


Figura 4: Cortes de malla de superficie de una lengua
Fuente: [Valdés, 2012].

La generación de una malla geométrica es una parte crucial dentro del área de estudio en cuestión y de vital importancia para el algoritmo a desarrollar dado que el método requiere una malla inicial para realizar la expansión, la cual además puede ser tanto de superficie como de volumen. Esta malla inicial es generada por alguno de los algoritmos existentes en la literatura de generación de mallas o generada a partir de datos en bruto obtenidos, por ejemplo, desde una resonancia magnética [Samani *et al.*, 2001], entre otros métodos.

Actualmente no existen en la literatura trabajos que se enfoquen en la generación de mallas hacia el exterior, ya que la totalidad de los trabajos similares se enfocan en la expansión de mallas hacia adentro con el objetivo de obtener una malla de volumen a partir de una malla cualquiera. Dichos trabajos utilizan un método llamado *Advancing front* [Valdés, 2012], el cual recibe como input una malla de superficie y entrega como output una malla de volumen, generando todo el mallado al

interior de la superficie inicial utilizando tetraedros, prismas, pirámides y hexaedros dentro de su composición como figuras elementales. Dicho método tiene por objetivo expandir hacia el interior del objeto una serie de frentes hasta obtener una malla de volúmenes consistente con las restricciones correspondientes. Para llevar a cabo lo anterior, dicho método utiliza las normales por cara para realizar la proyección, indicando debidamente su dirección de propagación de frentes con un mecanismo de control de colisiones y colapsos que permiten obtener una malla resultante óptima a nivel de elementos.

Otros trabajos asociados a la generación de mallas se encargan de refinar una malla [Bänsch, 1991] previamente proporcionada, permitiendo generar estructuras de diversa densidad de elementos generando divisiones en triángulos mas pequeños cada vez con el objetivo de obtener un modelo más apegado a la realidad.

Tal y como se mencionó anteriormente, los trabajos actualmente desarrollados asociados a generación de mallas mediante la expansión se encargan de expandir frentes hacia adentro de la malla de entrada, teniendo un foco principal en el análisis de las caras para llevar a cabo el procedimiento. Por otro lado este trabajo busca un nuevo enfoque principalmente teniendo en cuenta los dos factores principales al momento de generar una malla: el sentido y los elementos de expansión que referencian la malla resultante. El sentido en el que se realizará la expansión en este trabajo es hacia afuera de la malla, por lo que en este aspecto se generará una malla diametralmente opuesta a la generación tradicional mientras que los elementos base para realizar la expansión son los nodos, mientras que en los métodos estudiados actualmente se utiliza como foco las caras al momento de realizar la expansión.

Si bien las metodologías utilizadas en esta memoria han sido estudiadas previamente y sus métodos numéricos son utilizados de manera transversal, el foco que se les da es distinto y en ello radica parte de la innovación, novedad y necesidad del estudio, dado que se busca generar mallas hacia el exterior, además de considerar el análisis y cálculo de normales por puntos en lugar de como se realiza tradicionalmente, es decir, considerando las normales por cara.

En resumen, a lo largo de los años se han utilizado diversos métodos y algoritmos para la generación de mallas, sin embargo gran parte de estos métodos corresponden a aquellos que generan el interior de una malla de superficie, es decir, realizan expansión interior de dicha malla para posteriormente entregar una malla de volumen con la cual se puedan realizar trabajos de simulación. Por otro lado, lo que se busca abordar en esta memoria es darle otro enfoque a la expansión interior planteando la posibilidad de expandir hacia el exterior de una malla de superficie, en principio pensando en el nicho de las imágenes biomédicas con el objetivo de simular tejidos externos.

CAPÍTULO 4

PROPUESTA DE SOLUCIÓN

4.1. Propuesta

La solución propuesta consiste en, utilizando una malla ya sea de volumen o superficial, lograr expandir su superficie hacia el exterior utilizando como foco principal los nodos de las caras correspondientes, donde para cada nodo superficial se realice el cálculo de la normal asociada a éste teniendo una serie de consideraciones y restricciones. Dicha expansión posee una serie de parámetros que permiten personalizar y manipular la forma en la que se expanden las caras, obteniendo así una malla de volumen.

4.1.1. Tecnología utilizada

Para el desarrollo y ejecución de la propuesta se utilizó hardware con las siguientes especificaciones relevantes:

- CPU: Intel Core i5 10400
- GPU: Geforce RTX 3070
- RAM: 16 Gb @ 2400 Mhz

Sumado al hardware mencionado, el código fue compilado utilizando `gcc version 9.4.0` bajo `Ubuntu LTS 20.04`. Todas las visualizaciones fueron obtenidas utilizando el software `Paraview` en su versión `5.9.0`.

4.1.2. Código base

Con el objetivo de mantener un orden establecido previamente y poder reutilizar estructuras ya desarrolladas en relación al tratamiento de puntos, elementos y mallas en un estado preliminar a la ejecución de los módulos nuevos se utilizó como base de trabajo un código desarrollado por el profesor Claudio Lobos, del cual se extraerían las estructuras base que componen una malla sobre lo cual se desarrolló la solución. Desde dicho código se reutilizaron las clases correspondientes a puntos y mallas junto con sus operatorias fundamentales, es decir, productos vectoriales, aritmética básica entre puntos y contención de puntos en caras, entre otras.

4.1.3. Arquitectura de la propuesta

A nivel de arquitectura, la propuesta de solución implementa 4 clases nuevas, las cuales utilizan características de las clases anteriormente mencionadas. La clase principal y que encapsula en gran medida la solución entregada en esta memoria lleva por nombre `AdvancingPoint` y se encarga de unificar todo el proceso en relación al filtro y preprocesamiento de mallas, generación de puntos y elementos nuevos. Dentro de `AdvancingPoint`, se utilizan las clases `NormalRepair`, `NodeProjection` y `NewPointRef`, obteniendo finalmente el código completo del algoritmo. La clase `NormalRepair` se encarga de realizar el preprocesamiento de la malla entregada como argumento de entrada con el objetivo de conseguir que cada cara de la malla de superficie tuviese su normal apuntando hacia afuera del cascarón. `NodeProjection` se encarga de calcular las normales correspondientes a cada uno de los nodos. Finalmente la clase `NewPointRef` almacena los datos correspondientes al cálculo de las proyecciones almacenando el sentido de relación entre el nodo original y la normal asociada al cálculo de los nuevos nodos que generarán nuevos elementos de volumen.

4.1.4. Valores de entrada y argumentos

La solución propuesta permite ajustar una serie de parámetros con el objetivo de obtener un resultado personalizado según los requerimientos de cada caso. Dentro de dichos argumentos de carácter mandatorios se encuentra: la extensión de los archivos de entrada y de salida, las rutas hacia dichos archivos (existente en el caso del archivo de entrada y archivo por crear para la salida). Por temas de alcance en esta memoria solo se abordaron los formatos de entrada `.vtk` para mallas de volumen y `.off` para mallas de superficie. Por otro lado, los argumentos opcionales y que aportan personalización al archivo de salida son los siguientes:

Argumento	Descripción	Valores admitidos
-l <val>	Cantidad de capas de expansión	Entero
-k	Opción para mostrar los elementos no expandidos	-
-e <val>	Cistancia entre capas de expansión	Real
-x <val>	Multiplicador de distancia entre capas de expansión	Real
-f <path>	Ruta hacia el archivo que posee la <i>whitelist</i> de caras	String con ruta hacia el archivo

Tabla 1: Tabla de referencia para argumentos opcionales.

4.1.5. Limitaciones y consideraciones de valores de entrada y argumentos

Dentro de las limitaciones y consideraciones asociadas a los valores de entrada y argumentos opcionales, se desarrolló todo para que pudiesen utilizarse cualquier cantidad de ellos de forma

independiente, sin embargo hay casos particulares en los que algunos argumentos carecerían de sentido y no aportarían datos extras de acuerdo a lo esperado. Es importante mencionar que si bien el programa ejecuta la expansión para mallas de volumen y superficie sin importar su concavidad, sólo para mallas convexas se asegura consistencia en el resultado obtenido, ya que existen algoritmos de reparación y depurado involucrados que no abordan casos de expansiones que colisionan o colapsan entre sí.

4.1.6. Preprocesamiento de las entradas

Dado que el algoritmo posee compatibilidad tanto para mallas de volumen como superficie, se requiere realizar una serie de procesos de sanitización y reordenamiento de datos con el objetivo de agilizar todo el proceso llevado a cabo posteriormente.

En primer lugar, es necesario algún indicador para saber que tipo de malla se esta recibiendo como input. Si la malla de entrada es de tipo superficial, el filtro de elementos internos no se debe realizar, ya que se trata desde un comienzo de un cascarón hueco el cual contiene solo los elementos precisos para realizar la expansión. Sin embargo si la malla de entrada es de tipo volumétrica se requiere realizar una limpieza de caras, ya que para fines prácticos es de utilidad solo conservar las caras de la malla que se encuentren en la superficie exterior del elemento mixto de volumen, es decir, dada una malla sólida y rellena, busquemos en primera instancia identificar y almacenar solo los índices de las caras que pertenecen solo a un elemento de volumen (ya sea un prisma, hexaedro, tetraedro o pirámide). Teniendo en consideración lo anterior es que se consigue realizar el primer filtro, obteniendo así al final de este proceso una malla de superficie que corresponde al cascarón exterior de la malla input, sin importar su naturaleza superficial o volumétrica en una primera instancia. Sin embargo, luego pueden surgir otros inconvenientes que se deben solucionar para proseguir con el correcto cálculo de las normales por nodos.

Posteriormente a la filtración de caras superficiales es importante recordar que una cara, en la práctica, es un arreglo de índices, donde cada uno de esos índices hace referencia a un punto previamente definido en un arreglo de origen, los que se obtienen desde los archivos de entrada. Teniendo esto en cuenta y considerando una cara como un arreglo ordenado de puntos, es crucial revisar el orden en el que se encuentran los puntos de las caras resultantes del proceso de filtración, ya que para las definiciones de elementos de volumen, el orden en el que están designados los puntos aseguran la correcta interpretación del elemento sin importar el orden de los puntos de cada cara. Es por lo anterior que se debe realizar una revisión a la orientación de las normales de las caras superficiales del elemento a procesar, para lo cual se desarrolló el algoritmo de reparación de normales, utilizando algunas herramientas pertenecientes al código base. Una de las consideraciones importantes al hacer uso de las propiedades de la clase `TriMesh` del código base radica en que para poder utilizarla de manera correcta se requiere un preprocesamiento en cuanto a los arreglos de entrada para construir un nuevo objeto `TriMesh`, ya que como input esta clase se diseñó para trabajar con mallas conformadas en su totalidad solamente por triángulos. Es por lo anterior que se ha realizado un particionado de las caras previo a inicializar el objeto `TriMesh`.

utilizando un algoritmo sencillo de seccionamiento, en el que fijando un punto inicial arbitrario de una cara, se comienzan a recorrer los puntos haciendo tríos hasta que se hayan recorridos todos los vértices de la cara considerando el arreglo como un arreglo circular, conformando una serie de triángulos almacenados en un arreglo conveniente para posteriormente hacer entrega de dicho arreglo al objeto `TriMesh`. Formalizando, sea L la cara poligonal de tamaño n correspondiente a una cara de la malla superficial M , el conjunto de secciones triangulares de L , L' estará definido como:

$$L' : \{[L_0, L_1, L_2], \dots, [L_i, L_{i+1}, L_{i+2}], \dots, [L_n, L_0, L_1]\} \quad \forall i \in (0, n), i \in \mathbb{N} \cup \{0\}$$

Dicho algoritmo de seccionamiento posee una limitante principal, que hace referencia a los planos donde se encuentren los puntos de una cara, es decir, el algoritmo de seccionamiento se encuentra desarrollado para dividir una cara poligonal en caras triangulares siempre y cuando los puntos que conforman el polígono original sean coplanares. En caso de que los puntos no sean coplanares el algoritmo generará caras nuevas que no pertenecían a la cara original, por lo que sus resultados no serán consistentes.

Algorithm 1: Algoritmo de reparación de normales

Result: Malla superficial con normales corregidas

```

initialization;
caraCercana ← null;
boundingBox ← (max(X), max(Y), max(Z)) + offsetPoint;
seleccionar un punto P arbitrario fuera de la bounding box;
while queden caras sin revisar do
    if  $\text{dist}(\text{CaraRevisando}) < \text{dist}(\text{caraCercana})$  then
        | caraCercana ← CaraRevisando;
    end
end
Normal ← calcularNormal(CaraCercana);
if  $\text{PointIsInMesh}(\text{Normal})$  then
    | repararNormal(caraCercana);
end
while queden caras sin revisar do
    | repararNormal(caraRevision);
end

```

El procedimiento para reparar las normales consiste en identificar en primer lugar la bounding box de la malla, es decir, encontrar los puntos $(x_{\min}, y_{\min}, z_{\min})$ y $(x_{\max}, y_{\max}, z_{\max})$ que cumplen con lo siguiente: sea M el conjunto con todos los puntos de la malla de superficie tal que P corresponde a los puntos $(x, y, z) \in M$ tal que

$$\begin{aligned} \forall P \in M : (x_{max}, y_{max}, z_{max}) - (x, y, z) &\geq 0 \\ \wedge \\ \forall P \in M : (x_{min}, y_{min}, z_{min}) - (x, y, z) &\leq 0 \end{aligned}$$

Siendo así, los puntos $(x_{min}, y_{min}, z_{min})$ y $(x_{max}, y_{max}, z_{max})$ las esquinas diametralmente opuestas de una caja que contiene completamente la malla en cuestión. Posteriormente a tener identificada la *bounding box* se debe buscar entre las caras aquella que se encuentra a una menor distancia euclidiana respecto a un punto P' que corresponde a un punto al exterior de la *bounding box*.

Tal y como se puede observar y teniendo en contexto que todo se aplica a nivel de `TriMesh` encontramos la función `PointIsInMesh`, la cual es crucial para identificar si la normal de una cara se encuentra apuntando en el sentido correcto. Esto es, si según la función `PointIsInMesh` el punto fuera de la *bounding box* se encuentra dentro de la malla quiere decir que la normal de la cara más cercana a dicho punto está invertida, lo que quiere decir que hay que reparar orden de nodos para corregir su normal. Luego, para las demás caras de la malla, se van revisando por cercanía una a una hasta que no haya ninguna cara sin revisar. Para cada cara revisada, lo que se tiene en consideración es el orden de los puntos. Por ejemplo, si la cara A (para fines del ejemplo, con su normal bien orientada) posee los puntos de índices [1,2,3] y la cara B posee [2,3,4], la cara B se debe corregir ya que para cada sucesión de puntos solo debe haber una cara que los contenga para cada orden establecido. Para el ejemplo anterior, ambas caras poseen los puntos 2 y 3 correlativos en el mismo orden, por ende esto determina que sus normales se encuentran en sentidos contrarios y dado que la normal de A se encuentra por definición bien orientada, la normal de la cara B es la que se encuentra invertida y requiere reparación. Eventualmente si la cara B tuviese bien orientada su normal, no se realizaría reparación, y solo se marcaría la cara como visitada. Al visitar todas las caras se da por finalizada la reparación de normales para la malla de superficies, dando paso así al cálculo de las proyecciones por nodo.

4.2. Proyección de normales

Una vez que se tiene la malla de superficie con las normales por cara reparadas, se puede proceder con el cálculo de las normales por nodo. Para realizar el cálculo de las normales por nodo, en primer lugar es conveniente generar una estructura que nos permita realizar una asignación por referencia entre cara-punto, ya que hasta esta instancia sólo sabemos encontrar si un punto de índice i pertenece o no a una cara C , mientras que la relación inversa se vuelve más compleja. Es por esto que se desarrolló una estructura que permite referenciar en el sentido contrario, esto es punto-cara. Es decir, dado un punto p poder saber en que caras se encuentra dicho punto, esto con el objetivo de agilizar el acceso al momento de buscar las caras para realizar el cálculo posterior.

Para realizar el cálculo de las normales por nodo, en primera instancia se debe calcular de manera independiente las prenormales, es decir, las normales asociadas a cada nodo en función de cada

cara. Una vez teniendo calculadas todas las prenormales para cada uno de los nodos, se procede a calcular su promedio simple entre ellas, componente a componente para posteriormente realizar la normalización del promedio el cual finalmente se asignará como la normal del nodo correspondiente.

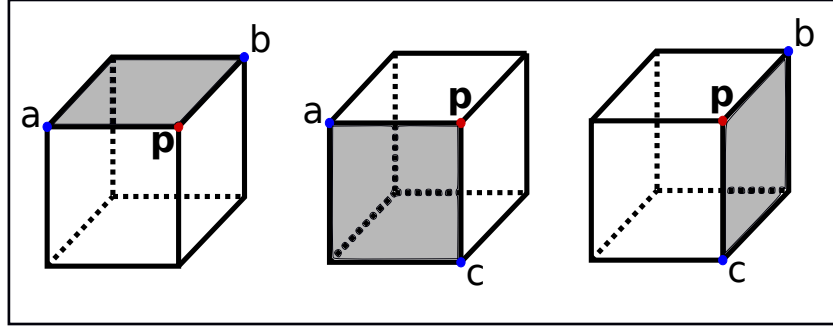


Figura 5: Tres caras de un hexaedro que comparten el punto p . En dicho caso se debe calcular tres prenormales, una por cada cara involucrada.

Fuente: Elaboración propia.

Tal y como se puede observar en la figura 5, para un hexaedro cualquiera, y teniendo el punto p como referencia, existen 3 caras del hexaedro que comparten dicho punto, lo cual indica que el punto p posee 3 puntos vecinos, siendo estos a , b y c . Como p forma parte de 3 caras superficiales, se debe calcular un total de 3 prenormales, donde cada una de estas involucra una cara correspondiente, sin embargo a nivel de cálculo lo que interesa son sus puntos vecinos de cada cara lo cual utilizando los puntos p , a , b y c de la figura nos quedarían como sigue:

$$PreNorm_1 = \frac{(\vec{b} - \vec{p}) \times (\vec{p} - \vec{a})}{|(\vec{b} - \vec{p}) \times (\vec{p} - \vec{a})|}$$

$$PreNorm_2 = \frac{(\vec{a} - \vec{p}) \times (\vec{p} - \vec{c})}{|(\vec{a} - \vec{p}) \times (\vec{p} - \vec{c})|}$$

$$PreNorm_3 = \frac{(\vec{c} - \vec{p}) \times (\vec{p} - \vec{b})}{|(\vec{c} - \vec{p}) \times (\vec{p} - \vec{b})|}$$

Una vez calculadas las prenormales, se procede a sacar el promedio lineal entre ellas, componente a componente para obtener finalmente la normal asociada a un punto. El proceso se repite para cada punto perteneciente a la malla quedando las normales asociadas a su punto p específico mediante la clase `NewPointRef`, con el objetivo de agilizar el proceso de búsqueda de la normal de cada nodo. La importancia en el cálculo de las prenormales radica en que no necesariamente todos los puntos relacionados son coplanares, por ende precalcular las normales por plano implica abordar sin problemas posteriores los casos donde las caras que compartan un punto p no sean coplanares.

y a su vez, permite entregar una normal resultante de tal forma que se asegure consistencia en el escalado y manejo de la solución final.

4.3. Creación de nuevos puntos

Una vez realizado el cálculo de todas las normales, se procede a calcular e indexar dentro de las estructuras originales los nuevos puntos y elementos. En este punto es donde comienzan a cobrar importancia los argumentos opcionales que personalizan el output del código, ya que a excepción de la *whitelist* correspondiente al argumento $-k$ todos los demás opcionales repercuten en este segmento de la solución.

El primer caso y más general indica que se expandirá sólo una capa, todos los nodos correspondientes a la malla superficial en cuestión a distancia 1 desde su ubicación original. Esto quiere decir que para cada nodo $\vec{v}$ se agregará un nodo nuevo a una distancia $\vec{v}_n$ donde dicho valor corresponde a la normal calculada y almacenada previamente a la lista de puntos general de la malla. Cabe mencionar que sobre ese arreglo de puntos no se realiza ninguna modificación a excepción de agregar información al final del mismo, ya que el orden es crucial para mantener consistencia dentro de la malla final a entregar.

Dado que el resultado esperado de salida del código desarrollado es una malla de volumen, y habiendo agregado los puntos nuevos a la lista de puntos original que será parte de la creación del output, se requiere generar de manera correcta los nuevos elementos de volumen. Para asegurar la correcta creación de los elementos de volumen se utilizan funcionalidades de la clase `NewPointRef`, la cual tiene enlace directo entre los puntos nuevos y sus puntos padres. Dicha clase es de suma utilidad dado que permite construir de manera más ordenada los nuevos elementos considerando las caras superficiales. Teniendo en consideración la lista de referencia se pueden construir de manera directa arreglos cuyos puntos correspondan a una cara de la superficie original agregando sus referencias de puntos nuevos al final en un orden correlativo respectivamente, así para una cara superficial de tres puntos cuya composición consta de los tres puntos originales, se generará un elemento de volumen prisma; mientras que para una cara de cuatro puntos, se generará un hexaedro cuyo total de puntos es ocho, correspondientes a sus cuatro puntos originales y las proyecciones de cada uno de ellos.

Dada la naturaleza del algoritmo de expansión, no existe la posibilidad de generar pirámides, ya que dicho caso sólo sería posible si las normales de cada uno de los tres hipotéticos puntos convergieran en un único punto, caso en el que implicaría que nos encontramos ante una malla de superficie cóncava, las cuales no se encuentran cubiertas en esta memoria y podrían generar resultados no esperados.

Como se mencionó anteriormente el archivo de salida corresponde a una malla de volumen, lo cual involucra un tratamiento adicional al momento de retornar el output final. Si bien en el caso ideal se expanden todas las caras superficiales en una capa a una unidad de distancia, existe la posibilidad

de que se expandan solo ciertas caras de la malla de entrada, en dicho caso para la malla resultante sólo se mostrarán los nuevos elementos creados formando una malla de volumen, sin embargo, si se habilita la opción -k como argumento opcional, se considerará tanto las caras expandidas como las no expandidas.

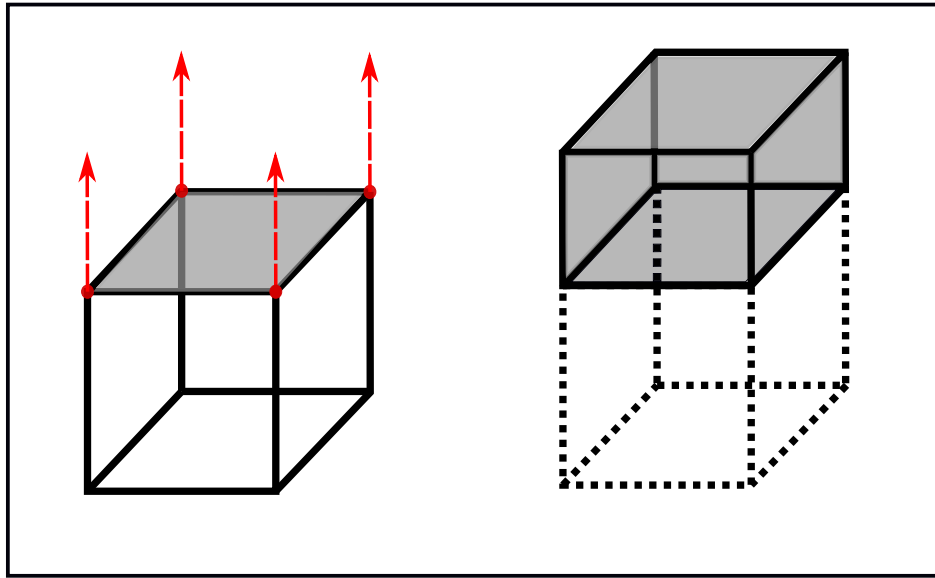


Figura 6: Expansión de una cara para una malla de superficie, teniendo en consideración el cálculo previo de las normales por nodo para todos los nodos de esta

Fuente: Elaboración propia.

En el caso que el argumento -k no se encuentre habilitado, la malla generada será hueca, formando así una especie de cascara con espesor, mientras que con el argumento -k la malla generada será sólida (en el caso que la malla original haya sido de volumen) y considerará tanto la malla original como la malla correspondiente a los nuevos puntos generados. Un caso especial de retorno, es el caso donde se reciba el opcional -k cuando el input inicial haya sido una malla de superficie. En dicho caso se consideraran tanto los elementos de volumen generados en el proceso del algoritmo como las caras planas no expandidas al momento de retornar el objeto resultante. Si bien el caso anteriormente presentado entrega una malla consistente estructuralmente hablando, esta estará compuesta tanto de elementos de volumen como elementos de superficie, lo cual podría no ser el resultado esperado, a pesar de ser un resultado consistente. En dicho caso el resultado será híbrido, considerando entre una malla exclusivamente superficial y una malla exclusivamente volumétrica, lo que se traduce en un espacio vacío dentro de las caras que encierran a la malla superficial cuyas expansiones no se llevaron a cabo.

4.4. Parámetros Opcionales

Con el objetivo de obtener un generador de mallas con un grado de personalización mayor, se consideraron una serie de parámetros opcionales ingresados por línea de comandos al momento de ejecutar el código. Dichos parámetros opcionales permiten una mayor personalización de los resultados obtenidos.

Dentro de la personalización que ofrece el generador de mallas, se encuentran las siguientes opciones adicionales:

4.4.1. -l : El número de capas a expandir

Este argumento por defecto posee el valor 1, y admite cualquier número entero mayor a cero. Al utilizar esta opción se le da al usuario la libertad de seleccionar la cantidad de capas de expansión que desea generar en la salida. Dichas capas de expansión se basan en las normales por nodo calculadas previamente y utilizan por defecto un multiplicador de valor 1 (modificable según argumento -e, cuya descripción se encuentra más adelante en esta misma sección). Para valores mayores, la distancia de los nuevos puntos se encuentra determinada como:

$$\vec{d} = (p_x, p_y, p_z) + l_i \cdot (n_x, n_y, n_z) ; l_i \in (1, k), l_i \in \mathbb{N}$$

donde k corresponde al valor entregado a continuación del argumento -l.

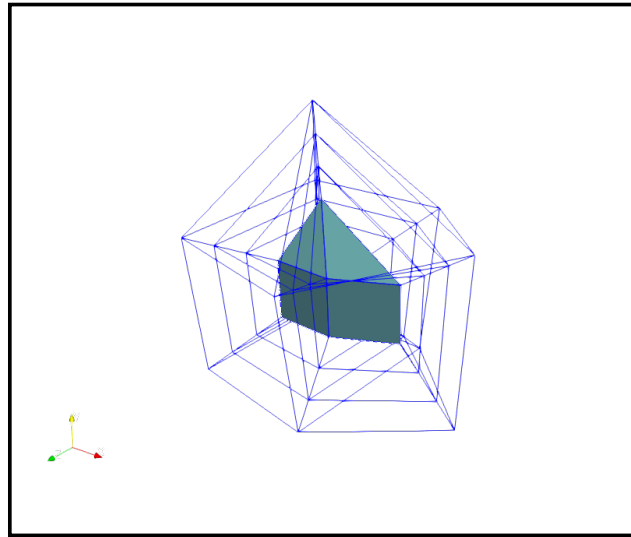


Figura 7: Visualización del *wireframe* correspondiente a una expansión de tres capas equidistantes una de otra, con la malla original como superficie solida a modo de referencia.

Fuente: Elaboración propia.

4.4.2. -e : La distancia entre capas a expandir

El argumento opcional $-e$ se recibe acompañado de un valor real, y su función dentro de la malla de salida es determinar la distancia entre capas a expandir. Por omisión la distancia entre capas es 1, sin embargo mediante este parámetro el valor puede ser fijado a conveniencia. La única consideración al momento de usar este comando es que la distancia entregada será la misma entre cada una de las capas de expansión generadas. En caso de buscar distintas distancias entre capas, se debe utilizar el argumento opcional $-x$ (no excluyente).

4.4.3. -x : La tasa de distancia entre capas

La tasa de distancia entre capas, denotada por $-x$ acompañada de un número real determina un factor de crecimiento en distancia entre capas, permitiendo que las capas sucesivas se encuentren a una distancia mayor o menor cada vez.

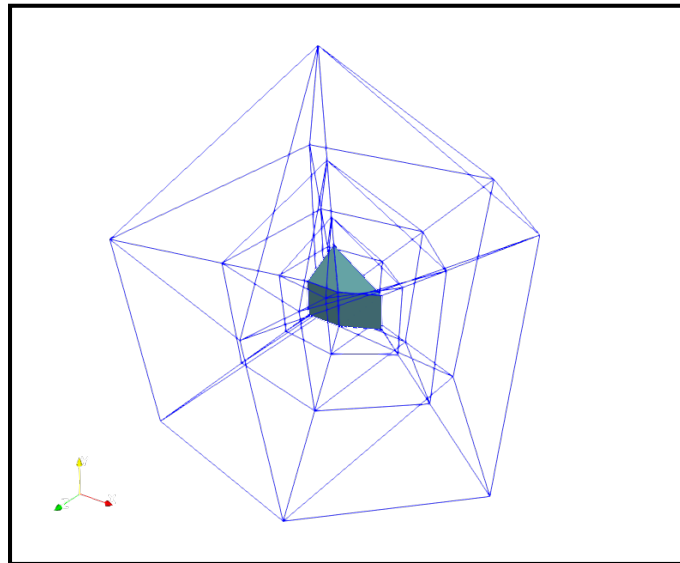


Figura 8: Expansión para tres capas con un factor de distancia entre capas de 1,5, se observa que cada capa de expansión está más distanciada que la anterior.

Fuente: Elaboración propia.

4.4.4. -k: Mostrar las caras o elementos no expandidos

Siguiendo con las opciones de personalización del output, existe la posibilidad de esconder o mostrar los elementos o caras según corresponda dependiendo del tipo de input (ya sea malla de volumen o superficie). Por defecto la malla original no se encuentra incluida dentro de la malla de salida, siendo esta última sólo el resultado de las proyecciones. Sin embargo, si se agrega el argumento `-k` a la línea de comandos, todas las caras y elementos no expandidos, al igual que los elementos originales serán incluidos en la malla resultante. En otras palabras, el parámetro `-k` permite obtener una malla de volumen completamente sólida mientras que la ausencia del parámetro `-k` indica que se generará una malla de volumen con un espacio interior no relleno.

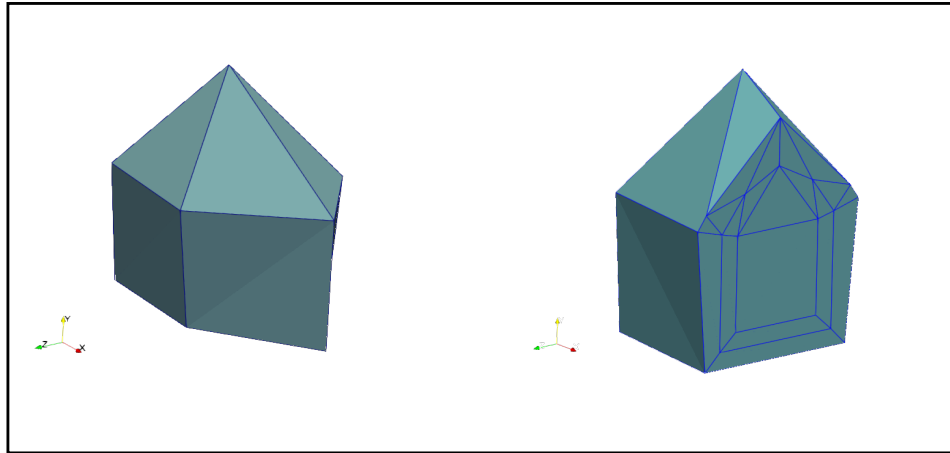


Figura 9: Expansión generada con el parámetro -k presente, a la izquierda la malla generada como sólido y a la derecha un corte transversal para comprobar la inclusión de la malla original
Fuente: Elaboración propia.

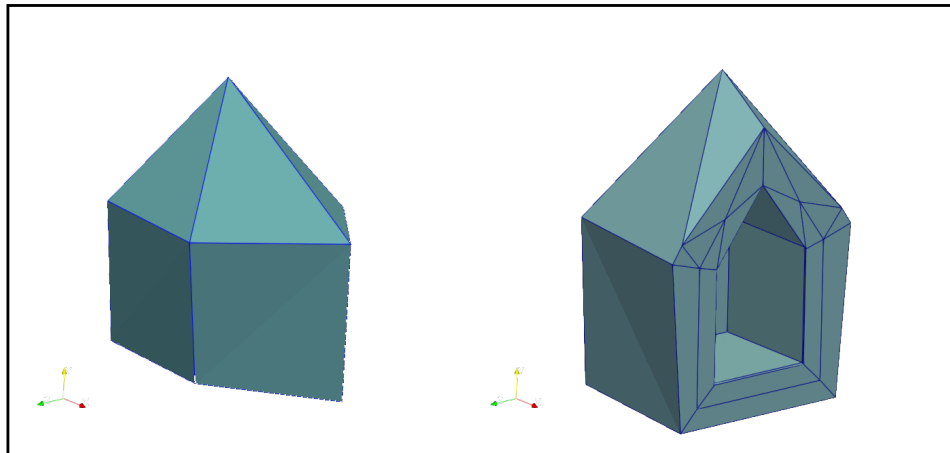


Figura 10: Expansión generada sin el parámetro -k, a la izquierda la malla generada como sólido y a la derecha un corte transversal para comprobar la ausencia de la malla original
Fuente: Elaboración propia.

4.4.5. -f: *Whitelist* para expandir caras específicas

La última personalización desarrollada para el output del generador de mallas consta de la posibilidad de entregarle un archivo con los índices de las caras a expandir. Dicho archivo debe estar en formato `.txt` y debe solo incluir los índices correspondientes a las caras que se desean expandir.

EXPANSIÓN DE CAPAS HACIA AFUERA EN UNA MALLA 3D SIN CONSIDERACIÓN DE COLISIONES NI COLAPSOS

Si bien el filtro es consistente tanto para mallas de volumen como mallas de superficie, en el caso de las mallas de volumen los resultados están limitados al conocimiento previo del usuario sobre la malla original, quedando como trabajo propuesto la evaluación del desarrollo de una implementación vía interfaz para seleccionar las caras a expandir de una malla de volumen. Por otro lado, si el input es una malla de superficie, el usuario tiene por defecto el conocimiento completo de las caras pertenecientes a la malla, permitiendo así seleccionar de manera clara y consistente las caras que se desean expandir sin ningún tipo de confusión. En caso de que las caras seleccionadas para expansión no estén conectadas por alguna de sus aristas o puntos se generará como consecuencia una malla de volumen no conexa, esto es, una malla de volumen en la cual no se puede acceder o visitar todos los demás elementos de la malla al recorrerlos de vecino en vecino.

1 1
2 3
3 4
4

Archivo 1: Ejemplo de input para whitelist.

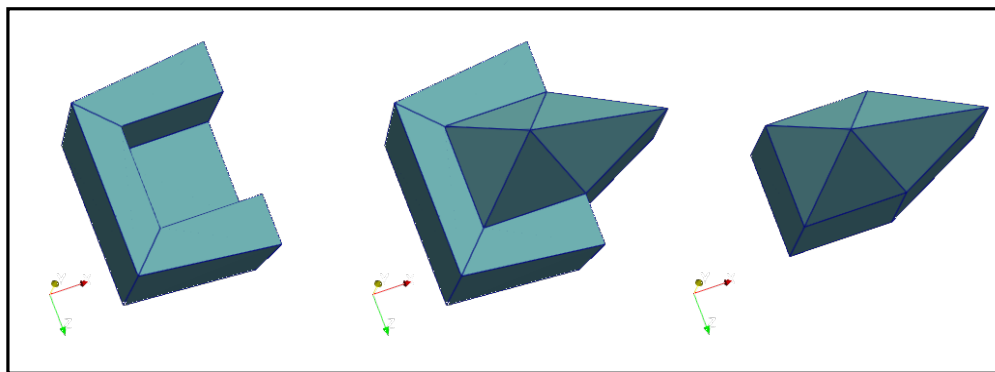
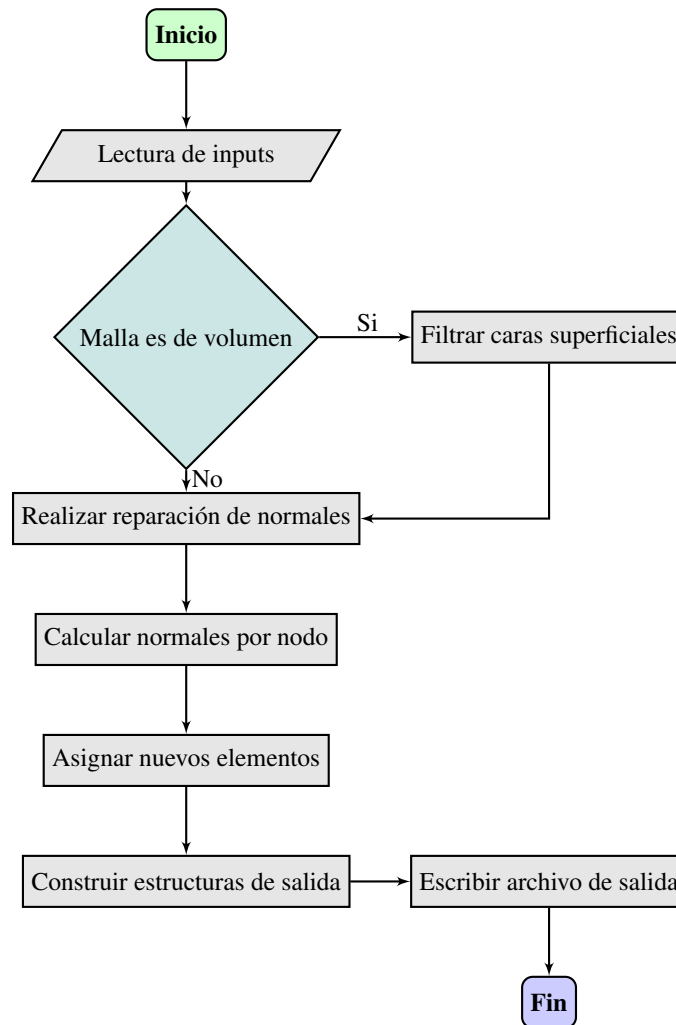


Figura 11: Expansión de caras específicas de la malla original, permitiendo seleccionar de manera personalizada las superficies a expandir.

Fuente: Elaboración propia.

4.5. Código

El código fue desarrollado en el lenguaje c++, utilizando como base de referencias un código proporcionado por el profesor Claudio Lobos Yáñez. En aspectos generales el trabajo se puede entender mediante el siguiente flujo:



4.5.1. Lectura de inputs

A nivel de mallas de entrada, el código acepta tanto superficiales como volumétricas en formatos `vtk` y `off`. Dicho input es procesado reutilizando la clase `Services` del código base con modificaciones para permitir el retorno de una mayor cantidad de valores desde el archivo, esto es, la capacidad de almacenar los `cellTypes` de los elementos además de las caras y puntos

de las estructuras completas. Dicha información sobre los `cellTypes` se utilizó posteriormente para reconocer, al momento de la escritura del archivo final, el tipo de malla ingresada junto con la topología de sus elementos originales. Adicional a lo anterior, es en este punto donde se configuran los parámetros opcionales que serán utilizados posteriormente a lo largo del desarrollo de la solución.

4.5.2. Identificación del tipo de malla

El identificador de tipo de mallas es una de las secciones cruciales dentro del preprocesamiento, ya que da paso a identificar las caras que corresponden a ser candidatas de expansión (sujetas a pertenecer a la lista de caras a expandir según el parámetro opción `-f`). En el caso de que la malla de entrada sea de tipo volumétrica, se ejecutará el siguiente algoritmo de selección que obtiene las caras que sólo pertenecen a un elemento, permitiendo así filtrar las caras relevantes para este trabajo.

```

1 for (long unsigned int fidx=0; fidx < FC.getFacesVec().size(); fidx++){
2     if(FC.getFace(fidx).numberOfElements()==1){
3
4         cantidad_superficiales++;
5         faces.push_back(FC.getFace(fidx).getPoints());
6     }
7     cantidad_caras++;
8 }

```

Archivo 2: Segmento de código que filtra las caras superficiales.

4.5.3. Reparación de normales

Como se mencionó previamente, una vez que se tiene el arreglo que contiene sólo a las caras superficiales de la malla original, se deben revisar las normales de cada una de éstas para asegurar que se encuentren apuntando hacia afuera de la malla convexa resultante del filtro anterior. Para realizar la reparación de normales se utilizó para el preprocesamiento la clase `TriMesh` del código base, sin embargo se requirió para poder crear un nuevo objeto de dicha clase que la malla entregada estuviese conformada solo por triángulos. Teniendo esto en mente, se diseñó un algoritmo de seccionamiento triangular para las caras de la malla superficial. Dado que se abordó con conceptos básicos de cardinalidad al momento de seccionar, el código funciona para cualquier polígono de n -lados siempre y cuando todos sus vértices sean coplanares.

```

1 vector <vector <unsigned int>> caras_seccionadas;
2 vector <unsigned int> caras_ref;
3 unsigned int ancla = 0;
4 for(unsigned int cara_index=0; cara_index < fv.size(); cara_index++){
5     for (unsigned int tridx = 0; tridx < fv[cara_index].getPoints().size()-2;
6         tridx++){

```

EXPANSIÓN DE CAPAS HACIA AFUERA EN UNA MALLA 3D SIN CONSIDERACIÓN DE COLISIONES NI COLAPSOS

```
6     vector<unsigned int> tmp;
7     tmp.push_back(fv[cara_index].getPoints()[ancla]);
8     tmp.push_back(fv[cara_index].getPoints()[ancla+1+tridx]);
9     tmp.push_back(fv[cara_index].getPoints()[ancla+2+tridx]);
10    caras_seccionadas.push_back(tmp);
11    caras_ref.push_back(cara_index);
12    tmp.clear();
13 }
14 }
```

Archivo 3: Seccionamiento triangular de polígonos.

Una vez seccionadas las caras y creado el objeto Trimesh, se utiliza la función `PointIsInMesh` para revisar si un punto arbitrario, que por definición está fuera de la *bounding box* respecto a la cara más cercana a éste indica que está dentro o fuera de la malla. En caso que el punto se determine adentro de la malla, entonces se requiere reparar la normal, proceso que consiste en alterar el orden de los elementos dentro del arreglo de puntos para cada cara.

```
1 void NormalRepair::repair_face(vector<unsigned int> &F) {
2     unsigned int temp;
3     if (F.size() == 3) {
4         temp = F[2];
5         F.at(2) = F[1];
6         F.at(1) = temp;
7     }
8     else if (F.size() == 4) {
9         temp = F[3];
10        F.at(3) = F[1];
11        F.at(1) = temp;
12    }
13 }
```

Archivo 4: Reparación de normal.

4.5.4. Calcular normales por nodo

Para la sección de cálculo de normales por nodo, una vez que se tienen las caras debidamente orientadas y reparadas, basta con realizar las operaciones vectoriales entre los puntos en cuestión para cada uno de los vértices, es decir, por cada vértice se aplican tres métodos que realizan cálculos vectoriales, normalizaciones y promedios simples entre las prenormales.

```
1 for (unsigned int i=0; i<Puntos.size(); i++){
2     ...
3     if (nodesInSurface[i] == 1 && pointsInWhitelist[i] == 1){
4         NodeProjection NP(i, Puntos[i], NRF, Whitelist_faces);
5
6         if (NP.getFacesInvolved().size() > 0){
```

EXPANSIÓN DE CAPAS HACIA AFUERA EN UNA MALLA 3D SIN CONSIDERACIÓN DE COLISIONES NI COLAPSOS

```
7      NP.CalcPreNormal(Puntos, 0);
8
9      NP.Normalize();
10
11     NodeProjectionVector.push_back(NP);
12
13     if(isnan(NP.getNormal().X()) || isnan(NP.getNormal().Y()) || isnan
14     (NP.getNormal().Z())){
15
16         NP.CalcPreNormal(Puntos,1);
17
18     }
19
20 }
21
22 }
23 }
```

Archivo 5: Funciones del proceso de creación de normales por nodo.

4.5.5. Asignar nuevos elementos

Una vez que se tienen las normales por nodo calculadas y almacenadas, se generan los nuevos puntos teniendo en consideración la cantidad de capas a generar en la expansión, teniendo por defecto la generación para una única capa.

```
1 for (unsigned int layer=1; layer <= num_layers; layer++){
2     for (unsigned int pointIdx = 0; pointIdx < NodeProjectionVector.size();
3     pointIdx++){
4         Point3D newPoint(
5             Puntos[NodeProjectionVector[pointIdx].getNodeIndex()].X()+
6             (layer*dist*NodeProjectionVector[pointIdx].getNormal().X()
7             *pow(distance_multiplier,layer)),
8             Puntos[NodeProjectionVector[pointIdx].getNodeIndex()].Y()+
9             (layer*dist*NodeProjectionVector[pointIdx].getNormal().Y()
10            *pow(distance_multiplier,layer)),
11            Puntos[NodeProjectionVector[pointIdx].getNodeIndex()].Z()+
12            (layer*dist*NodeProjectionVector[pointIdx].getNormal().Z()
13            *pow(distance_multiplier,layer)));
14        this->new_points.push_back(newPoint);
15
16        if (layer == 1){
17            NewPointRef NewNode(Puntos.size()+offset, NodeProjectionVector
18            [pointIdx].getNodeIndex(), newPoint);
19            NPR_arr.push_back(NewNode);
20        }
21        else{
22
23        }
```

EXPANSIÓN DE CAPAS HACIA AFUERA EN UNA MALLA 3D SIN CONSIDERACIÓN DE COLISIONES NI COLAPSOS

```
17         NewPointRef NewNode (Puntos.size()+offset, Puntos.size()+
        father_offset, newPoint);
18         NPR_arr.push_back (NewNode);
19         father_offset ++;
20     }
21     offset++;
22 }
23 }
24 }
```

Archivo 6: Creación de nuevos puntos para la expansión.

Una vez teniendo los puntos generados, se requiere construir los arreglos que darán paso a los nuevos elementos correspondientes a la malla de volumen de salida. Para lograr esto, se utilizó una estructura que permite referenciar el nodo padre con su nodo hijo (correspondiente al nuevo nodo generado) y así construir de manera iterativa, respetando el orden y consistencia los nuevos elementos de volumen a retornar en el output de la ejecución.

```
1
2 for (unsigned int face_idx=initial_faces_offset*(layer_check-1);
3     face_idx < initial_faces_offset*layer_check; face_idx++){
4     newfacePerLayer = {};
5     for(unsigned int node_face_rel=0; node_face_rel < n_nodes; node_face_rel
6         ++){
7         for(unsigned int i=0; i< NPR_arr.size();i++){
8             if (NPR_arr[i].getFIndex() == arr_faces[face_idx][node_face_rel]){
9                 this->arr_faces[face_idx].push_back (NPR_arr[i].getIndex());
10                if (layer_check < num_layers){
11                    newfacePerLayer.push_back (NPR_arr[i].getIndex());
12                }
13                break;
14            }
15        }
16
17        if (newfacePerLayer.size()>0 && layer_check < num_layers){
18            this->arr_faces.push_back (newfacePerLayer);
19        }
20
21    }
22
23    ...
24
25 vector < vector<unsigned int>> T1 = AP.getFaces();
26 for (long unsigned int i=0; i<AP.getFaces().size(); i++){
27     VUI.push_back (AP.getFaces() [i]);
28 }
```

Archivo 7: Adición de elementos nuevos al arreglo de salida.

4.5.6. Construir estructuras de salida

Para finalizar el procesamiento de los archivos y preparar las estructuras para la función de escritura, se debe revisar si se ha seleccionado la opción adicional `-k`. En caso de que dicha opción se encuentre activa, se debe agregar a la lista de elementos resultantes los elementos de la malla original. Es importante notar que a nivel de elementos el orden de estos dentro del arreglo no es relevante, dado que los elementos no dependen unos de otros, sino que dependen del orden en el que se encuentren definidos los puntos y dicho arreglo de puntos sólo sufre adiciones de puntos nuevos sin eliminar ni mover de posición bajo ningún motivo los puntos de la lista original.

```
1 if (show_previous_mesh && IsVolMesh == false){
2     for (int i=Whitelist_faces.size()-1; i >= 0; i--){
3         if (Whitelist_faces[i] == 0){
4             Tl.insert(Tl.begin(), VUI[i]);
5         }
6     }
7 }
8
9 else if (show_previous_mesh && IsVolMesh == true){
10     for(unsigned int V=OriginalMesh.size(); V>0; V--){
11         Tl.insert(Tl.begin(), OriginalMesh[V-1]);
12     }
13 }
```

Archivo 8: Verificación de inclusión para malla original.

4.6. Consideraciones de la solución

Dentro de la expansión de mallas, existen casos que no se abordaron en esta memoria debido a su complejidad, dando paso a dejar muchas de las limitaciones actuales como trabajos propuestos a futuro.

En primer lugar, en relación a las mallas de entrada, ya se ha mencionado que se reciben tanto mallas de volumen como de superficie, siempre y cuando se trate de una malla completamente conexas y convexas en su totalidad, sin importar la cantidad de caras, formas y planos involucrados. Siguiendo este lineamiento, es que también es requerimiento que si la malla de entrada es volumétrica, ésta no posea huecos en su interior y se encuentre previamente “rellena”, es decir, que sea en si misma un objeto sólido sin vacíos interiores. En caso de que la malla de entrada sea de superficie, por definición ésta no debe ser rellena, ya que sólo esta compuesta por una serie de caras en lugar de elementos de volumen.

Cabe recordar que en esta memoria no se aborda el problema de colisiones entre capas de las expansiones, por lo que queda propuesto como trabajo futuro. Sin embargo y dada la naturaleza de la expansión las colisiones considerando las limitaciones y restricciones no existe posibilidad de colisiones ni colapsos al expandir hacia el exterior considerando el enfoque en los nodos.

4.7. Resultados esperados

Dentro de los resultados esperados, dada una malla de volumen o superficie que cumpla con las consideraciones y limitaciones expuestas anteriormente, se espera que el algoritmo sea capaz de generar una malla expandida teniendo en consideración todos los parámetros opcionales posibles en todas las combinaciones distintas de ellas. Además se debe mantener la consistencia en caso de realizar expansiones utilizando la *whitelist*, es decir, las normales calculadas por puntos tienen que considerar para su cálculo todas las caras expandidas al momento de generarse.

Todo lo anterior debe ser consistente con la malla original ingresada, teniendo la posibilidad de acoplar tanto la generación nueva como la malla preexistente, permitiendo así visualizar y obtener una malla de volumen solo con la sección nueva generada o con la sección nueva acoplada a la malla original, sea ésta de volumen o superficie.

CAPÍTULO 5

VALIDACIÓN DE LA SOLUCIÓN

5.1. Resultados obtenidos

Dentro de los resultados obtenidos se probaron cada una de las instancias y opciones disponibles de manera independiente y en conjunto. Con fines de abarcar la mayor cantidad de casos posibles se utilizó una malla de volumen y superficie específicas que contenían todas las características que permitían probar cada una de las funcionalidades implementadas. Dichas mallas son aparentemente iguales, sin embargo en el caso de la malla de volumen esta se encuentra conformada por elementos correspondientes a un hexaedro, un prisma, un tetraedro y una pirámide, siendo estos los elementos más utilizados en mallas de volumen a nivel general. Por otro lado, la malla de superficie esta compuesta tanto de cuadriláteros y triángulos en toda su extensión, siendo estos dos los polígonos esenciales para este tipo de estructuras.

Una de las características importantes dentro del proceso de generación de mallas radica en la escalabilidad, ya que basta con realizar pruebas para estructuras elegidas con los elementos esenciales (ya sean de volumen o superficie) para asumir un comportamiento similar para estructuras de mayor envergadura. Uno de los factores que se ve afectado en el caso de utilizar mallas de mayor magnitud es el tiempo de cómputo, que en líneas generales no es el foco principal de este trabajo, sino más bien generar un algoritmo base de expansión de mallas hacia el exterior de una malla siempre y cuando esta cumpla con las condiciones y restricciones impuestas desde un comienzo.

5.2. Mallas de prueba

Tal y como se mencionó anteriormente, las mallas de prueba tanto para el caso superficial como volumétrico están compuestas por los elementos esenciales presentes en estas estructuras, permitiendo presumir un escalado de la solución efectivo y sin inconsistencias según lo esperado.

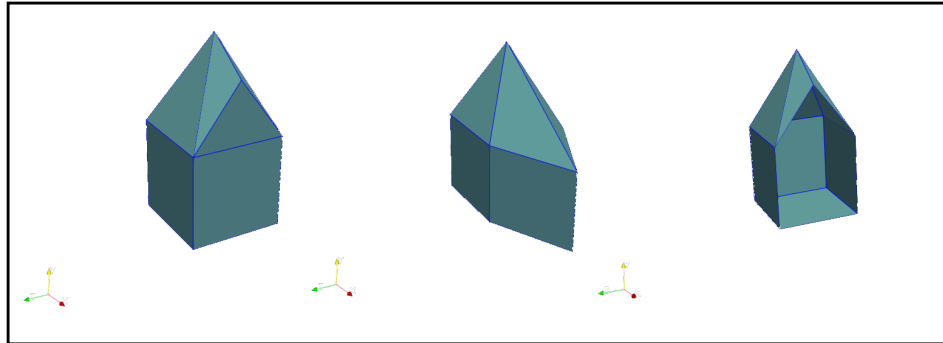


Figura 12: Muestra de mallas utilizadas. de izquierda a derecha: corte transversal de malla de volumen, superficie de malla utilizada y corte transversal de malla de superficie.

Fuente: Elaboración propia.

5.3. Resultados generales

Dentro de los resultados de manera general, es decir, sin uso de ningún parámetro opcional el resultado obtenido se comportó según lo esperado. Tanto para una malla de volumen cómo para una de superficie se obtuvo la proyección de cada normal satisfactoriamente, permitiendo expandir todas sus caras, teniendo como eje de expansión las normales de cada punto involucrado en la generación. Cabe mencionar que para el caso de la malla de volumen el resultado obtenido al no incluir ningún parámetro opcional **no** considera la inclusión de la malla con elementos originales, es decir, la malla generada será un cuerpo con volumen con un agujero en el espacio de su elemento original, lo cual se puede entender como que el resultado de expandir una malla de volumen sin parámetros adicionales resulta en un cascaron con un espesor unitario.

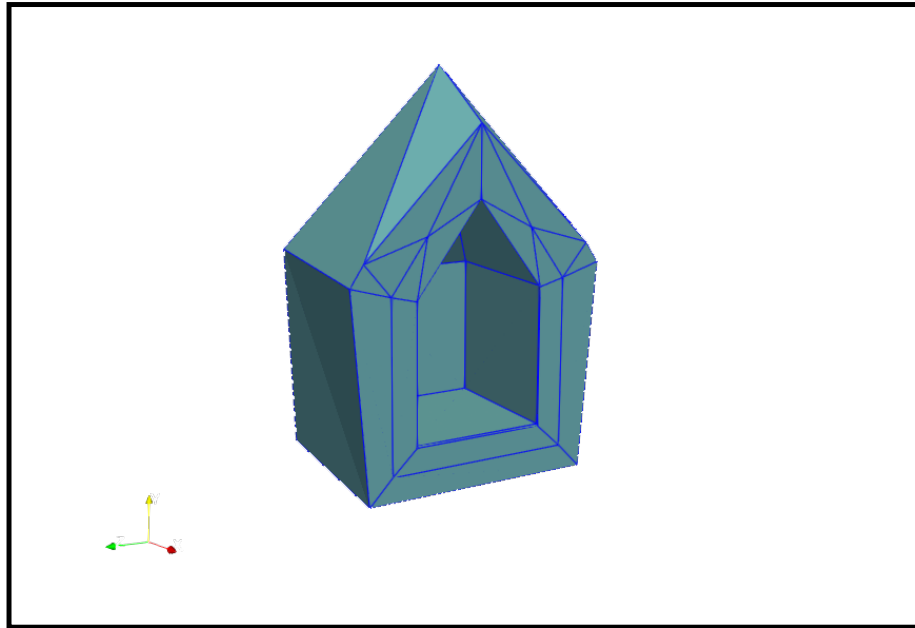


Figura 13: Resultado general sin aplicar parámetros opcionales al momento de ejecutar el algoritmo.

Fuente: Elaboración propia.

Además, para el caso de una malla de superficie el resultado obtenido será equivalente al resultado de la malla de volumen, siempre y cuando ambas tengan la misma frontera (cascarón superficial). Lo anterior se extiende incluso para casos donde la malla posea parámetros opcionales.

5.4. Resultados particulares

Para la revisión de los resultados opcionales, se requirió realizar un desarrollo exhaustivo de combinaciones de cada uno de los parámetros opcionales con los demás, entendiendo desde una primera instancia que hay parámetros opcionales que carecen de sentido por sí solos y pierden relevancia mas no consistencia.

5.4.1. Cantidad de capas expandidas

Para comenzar, se realizaron las pruebas correspondientes a la expansión de todas las caras una cantidad 1 de capas. Para el caso propuesto, los resultados esperados son la obtención de una malla con 36 elementos nuevos, todos ellos de volumen, dentro de los cuales no se encuentran

contenidos los elementos de la malla original. Lo anterior se pudo confirmar revisando la malla resultante o el archivo de salida correspondiente a la misma, teniendo en consideración que la cantidad de elementos escritos en el archivo `.vtk` de salida corresponda con la cantidad esperada. Finalmente el resultado obtenido es consistente con los resultados esperados en este primer caso de prueba particular.

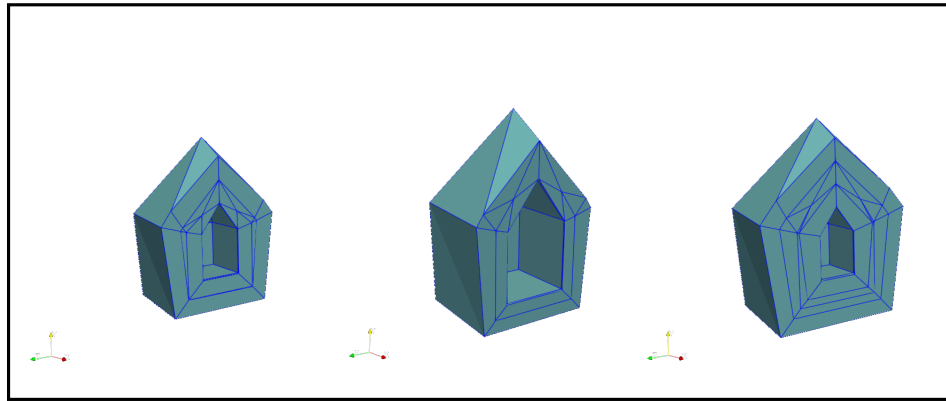


Figura 14: Cortes transversales para una expansión de la malla de pruebas utilizando desde una hasta tres capas de izquierda a derecha, respectivamente.

Fuente: Elaboración propia.

5.4.2. Distancia entre capas expandidas

Otra de las pruebas de rigor que se realizó consistió en probar el parámetro opcional que permitía modificar las distancias entre capas expandidas, todas de manera uniforme, entregando así un resultado consistente. Dicho caso de prueba al no ser de mayor complejidad (ya que sólo realiza una suma vector-escalar) no requirió por sí solo mayor análisis.

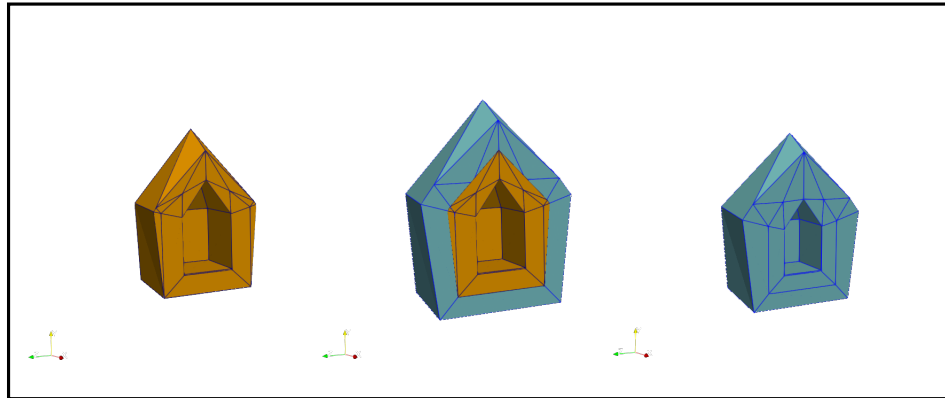


Figura 15: Resultado obtenido al generar mallas a distinta distancia inter-capas. De izquierda a derecha: utilizando distancia 1; mezcla de ambas mallas para graficar la diferencia de distancia de expansión; utilizando una distancia 2.

Fuente: Elaboración propia.

5.4.3. Factor de distancia de expansión

Continuando con los resultados obtenidos para los parámetros opcionales disponibles en el código, se cuenta con una modalidad que permite escalar el factor de expansión capa tras capa. Este parámetro opcional es redundante en el caso que se expanda solo una capa superficial, ya que si bien se expandirá según el factor entregado, existirá por aritmética un valor d que permita conseguir el mismo efecto, siendo d el parámetro opcional de distancia entre capas de expansión. El factor de distancia entre capas permitió generar de manera satisfactoria, para una cantidad de capas $l > 1$ capas sucesivas dentro de la malla tal que la distancia entre cada una de ellas fuese más (o menos) distante a la anterior según factor. En resumen, para este parámetro opcional, en el caso que se le entregue un factor $x = 1,1$ cada capa sucesiva estará un 10 % más lejana que la anterior.

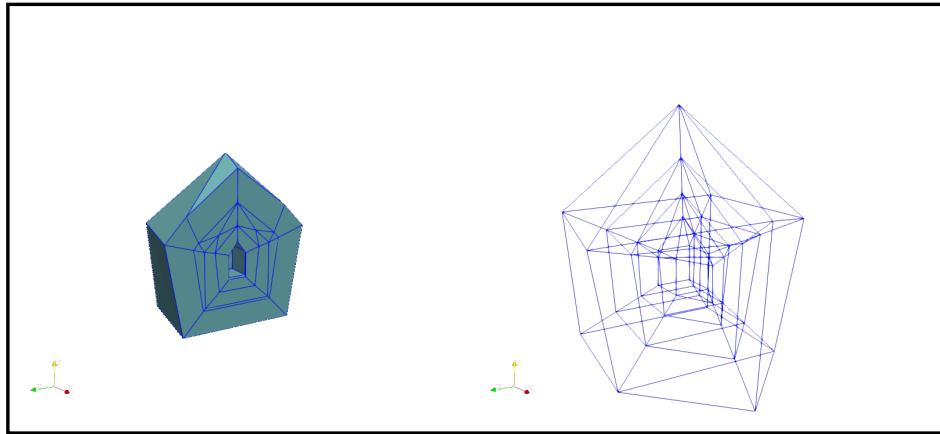


Figura 16: Resultado al aplicar un factor de expansión 1,5, aplicado para tres capas.
Fuente: Elaboración propia.

5.4.4. Expansión selectiva

Uno de los parámetros más complejos presentes en la solución ofrecida es el de la *whitelist*. El parámetro $-f$ permite seleccionar a criterio del usuario cuales son las caras que serán expandidas de la malla de entrada. Analizando los resultados obtenidos, dichas caras expandidas mantienen consistencia respecto a los cálculos realizados lo cual permite entregar como resultado consistente y bien construido según los lineamientos esperados incluso para caras que no comparten ningún elemento (en particular vértices o aristas).

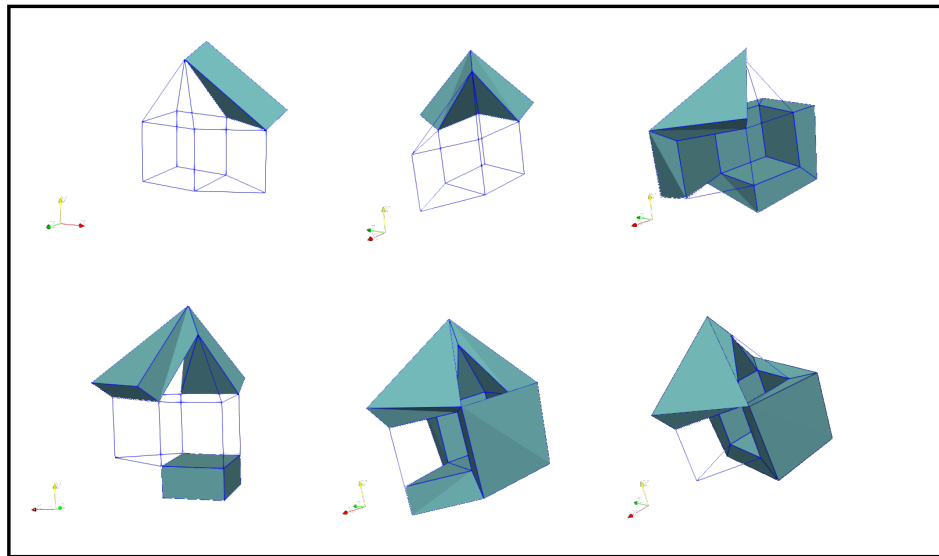


Figura 17: Muestra de expansión selectiva, utilizando diferentes configuraciones de caras a expandir y presentando resultados. Se agrega en cada caso el *wireframe* de la malla original a modo de referencia.

Fuente: Elaboración propia.

Tal y como se representa en la figura 17, existen tres grandes casos dentro de los resultados: la expansión de una cara que no comparte ningún elemento con las demás (o la simple expansión de una cara de toda la malla), la expansión de varias caras que comparten información entre sí, y finalmente la expansión mixta entre las dos versiones anteriores, es decir, dentro de una misma malla, expandir al menos dos conjuntos de caras que comparten elementos internamente pero no entre ellos.

Para el caso del parámetro $-f$ los resultados obtenidos son consistentes con los esperados, dado que se expanden solo las caras correspondientes a los índices ingresados de manera consistente y considerando las normales por nodo solo de las caras que corresponden a las ingresadas vía *whitelist*. En el caso (aunque poco relevante) que se incluyan todas las caras superficiales en la *whitelist*, el resultado obtenido será equivalente al resultado que incluye la expansión de todas las caras superficiales.

5.4.5. Consideración de malla original en la salida

Para finalizar, el último parámetro opcional desarrollado permite la consideración de la malla original como parte de la salida de la malla resultante, lo cual afecta en gran medida a las mallas de volumen por sobre las mallas de superficie. El parámetro k permite al usuario poder incluir de

manera satisfactoria la malla original, sin importar el origen de la misma. Sin embargo cabe tener en consideración que una malla de superficie al incluir las caras originales no expandidas generarían, aunque de manera consistente a nivel de estructura *vtk*, una especie de malla híbrida, que contiene tanto elementos de superficie como elementos de volumen, lo cual si bien es consistente, queda como trabajo propuesto la potencial utilidad de este caso particular.

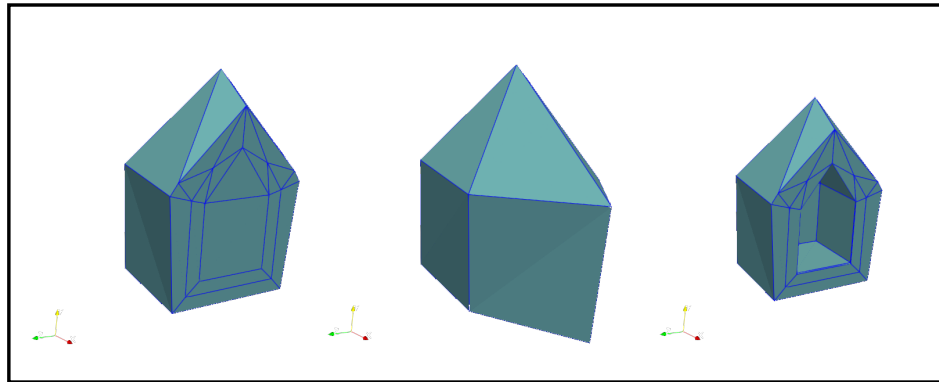


Figura 18: Malla resultante con adición de malla original ingresada en el input, malla completa post expansión sin cortes, malla resultante sin consideración de la malla original en la salida, respectivamente.

Fuente: Elaboración propia.

5.5. Pruebas con malla de gran magnitud

Luego de terminar las pruebas para una malla adaptada con el fin de revisar resultados con todos los tipos de elementos presentes, se procedió a realizar la ejecución del programa recibiendo como entrada una malla correspondiente a un cerebro tanto en su versión de malla superficial en formato *.off* mientras que por el lado de volumen se utilizó el formato *.vtk*.

5.5.1. Malla de superficie

En relación a la malla de superficie, esta se encuentra compuesta por 3152 puntos y que conforman en total 6300 elementos de superficie. El archivo de pruebas, tal y como se puede evidenciar en la figura 19 se encuentra compuesta en su totalidad por triángulos y al ser una malla de superficie todos estos elementos se encuentran visibles en su corteza. Si bien dicha malla contempla secciones cóncavas, el código puede ejecutarse sin problemas, sin embargo dada las limitaciones propuestas no se encuentra presente ningún método de reparación o comportamiento ante colapso y colisiones asociado a la expansión recurrente, es por esto que si se extrapola la cantidad de capas expandidas,

EXPANSIÓN DE CAPAS HACIA AFUERA EN UNA MALLA 3D SIN CONSIDERACIÓN DE COLISIONES NI COLAPSOS

las expansiones correspondiente a las secciones cóncavas comenzaran a colapsar generando una malla fácilmente asociada a una esfera (figura 20).

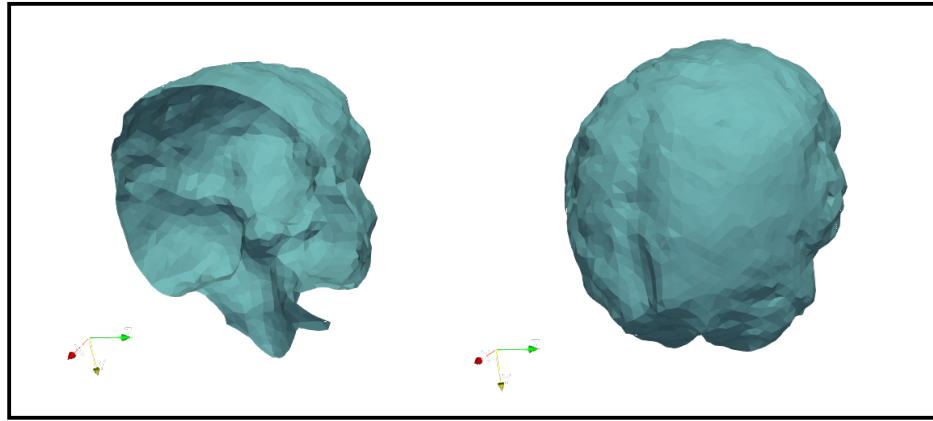


Figura 19: Malla superficial de corteza cerebral a la cual se le realizará la expansión, junto con un corte transversal.

Fuente: Elaboración propia.

Para la malla anterior, se realizaron expansiones a todas las caras del cuerpo utilizando el parámetro -1 con una, cinco, siete y diez expansiones, obteniéndose así los siguientes valores en cuanto a tiempo de cómputo y memoria utilizada.

Capas	Tiempo medido (mm:ss)	Memoria utilizada (Kb)
1	0:46.67	11340
5	6:21.40	15028
7	12:20.88	18816
10	24:18.78	24048

Tabla 2: Desempeño del código desarrollado para la expansión de una malla superficial de corteza cerebral.

Para los valores anteriores, se observó un rápido crecimiento en cuanto al tiempo de computo. Con respecto al uso de memoria, el crecimiento es de un orden lineal por lo que a priori no existen problemas de memoria para el programa. Por otro lado es importante explicar que aún con tiempos como los expuestos anteriormente el programa permite agilizar un proceso que al día de hoy solo se puede realizar por vía manual dado que no existen herramientas previas que realicen expansiones hacia afuera, por lo que a pesar de tener tiempos elevados son considerablemente inferiores a los tiempos de realizar el proceso análogamente paso a paso. Como es de esperarse los resultados para un número mayor de capas se vuelven problemas que se deben optimizar de cara al futuro,

sin embargo es importante mencionar que se debe estudiar la relevancia real de poder generar una cantidad de capas elevada en la expansión, dado que pierde sentido práctico realizar, por ejemplo, una expansión de 100 capas.

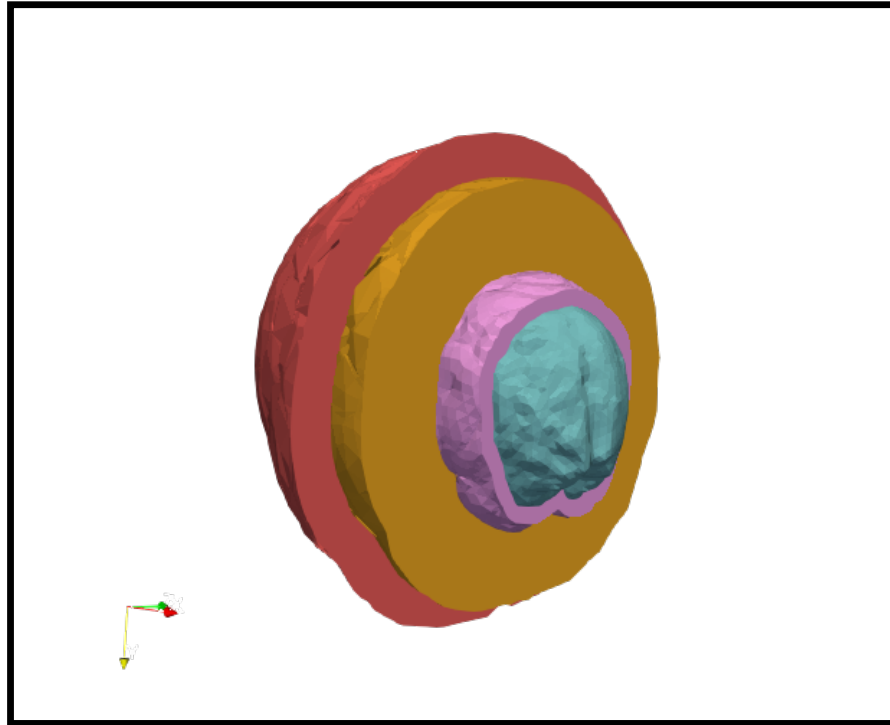


Figura 20: Malla de corteza cerebral con sus expansiones correspondientes.
Fuente: Elaboración propia.

5.5.2. Malla de volumen

Por parte de la prueba con elementos de volumen en el archivo de entrada, se utilizaron diversas mallas correspondientes igualmente a una corteza cerebral. La diferencia entre las mallas utilizadas radica en el nivel de refinamiento de cada una, es decir, qué tan denso es su nivel de elementos y por lo tanto, qué tan fiel a la realidad es el modelo en cuestión (considerando que los archivos de entrada están compuestos por elementos finitos).

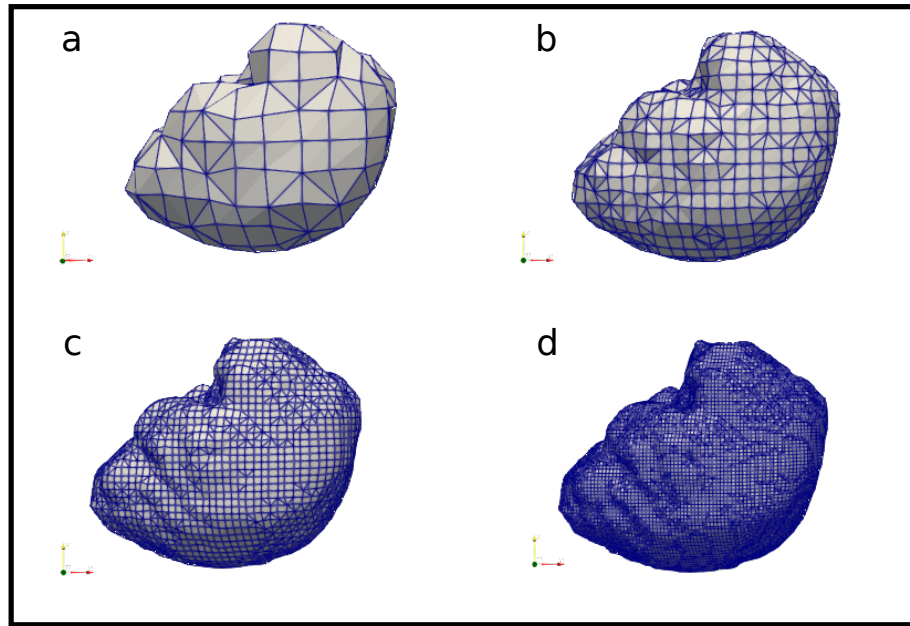


Figura 21: Mallas con distinto grado de refinación. A mayor refinación, mayor cantidad de elementos en su composición. Mallas con grado de refinación 3(a), 4(b), 5(c) y 6(d).

Fuente: Elaboración propia.

Las mallas presentadas están conformadas con la siguiente cantidad de puntos y elementos de volumen:

Malla	Puntos	Elementos
cs3	411	798
cs4	2230	5111
cs5	10941	26209
cs6	49431	119910

Tabla 3: Características de una serie de mallas volumétricas correspondientes a una corteza cerebral utilizadas para realizar ejecuciones del código desarrollado.

Tal y como se puede observar en la figura 21 dichas mallas de volumen están compuestas por elementos mixtos, esto queda claro al observar, por ejemplo la malla (a) donde se encuentran presentes tanto triángulos como cuadriláteros en su superficie. A su vez, dichas mallas al ser de volumen se encuentran completamente sólidas (a diferencia de la malla de superficie, la cual considera solo el cascarón superficial).

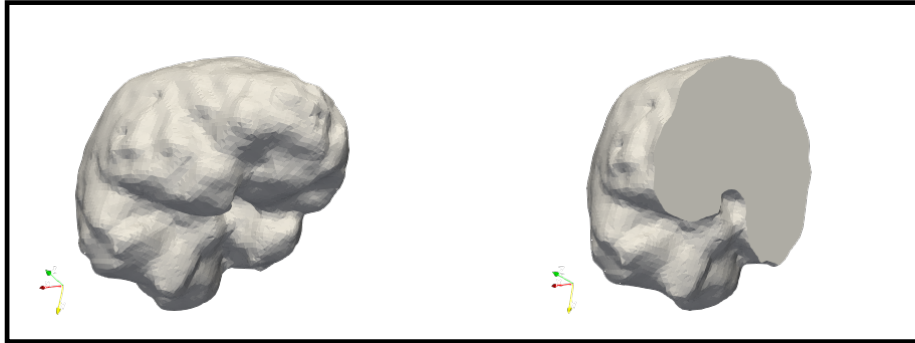


Figura 22: Malla cs6, con cortes para visualizar su interior.
Fuente: Elaboración propia.

Los resultados obtenidos para una expansión completa en cada una de las mallas de volumen proporcionadas sin parámetros adicionales fueron los siguientes:

Malla	Tiempo medido (hh:mm:ss)	Memoria utilizada (Kb)
cs3	00:00:02	6048
cs4	00:01:24	11752
cs5	00:35:03	42920
cs6	12:26:24	181220

Tabla 4: Desempeño del código desarrollado para la expansión de una serie de mallas volumétricas correspondientes a una corteza cerebral sin considerar la malla original en el archivo de salida.

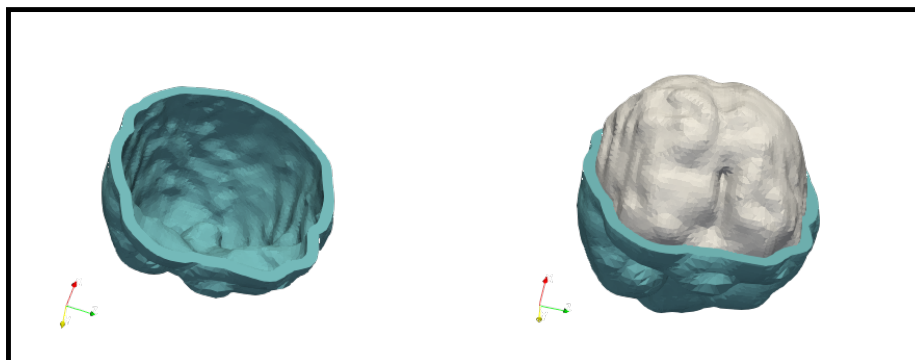


Figura 23: Expansión de cs6 sin parámetros adicionales, con cortes para visualizar su interior.
Fuente: Elaboración propia.

EXPANSIÓN DE CAPAS HACIA AFUERA EN UNA MALLA 3D SIN CONSIDERACIÓN DE COLISIONES NI COLAPSOS

Luego, para finalizar los resultados obtenidos, se repitió la ejecución con las mallas de volumen, pero agregando el parámetro $-k$, el cual permite además de generar la expansión correspondiente, agregar la malla de entrada al archivo de salida. De esta ejecución se obtuvieron los siguientes resultados:

Malla	Tiempo medido (hh:mm:ss)	Memoria utilizada (Kb)
cs3	00:00:02	5900
cs4	00:01:26	11772
cs5	00:35:49	42928
cs6	12:19:03	181344

Tabla 5: Desempeño del código desarrollado para la expansión de una serie de mallas volumétricas correspondientes a una corteza cerebral considerando la malla original en el archivo de salida.

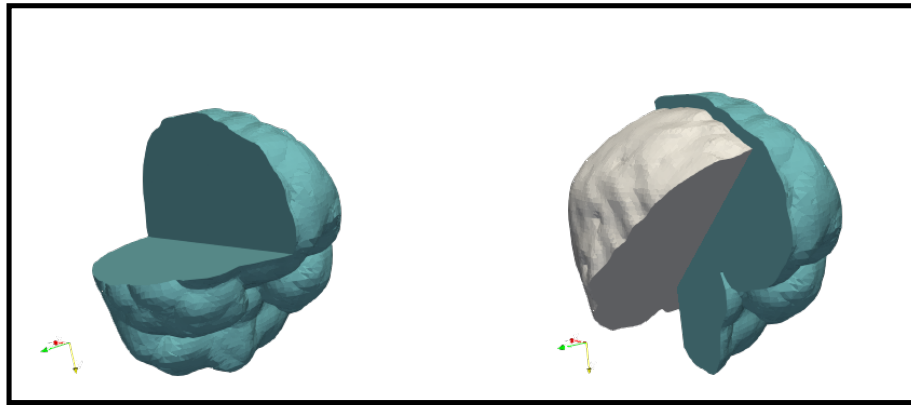


Figura 24: Expansión de cs6 con parámetro adicional para conservar la malla original en el archivo de salida, con cortes para visualizar su interior.

Fuente: Elaboración propia.

5.6. Análisis de resultados

Tal y como se ha presentado anteriormente, los resultados obtenidos para las expansiones teniendo en consideración cada uno de sus parámetros opcionales han sido satisfactorios y cumplen según lo esperado siempre y cuando se respeten las limitaciones y restricciones propuestas. Dentro del análisis final de los resultados se demuestra que para la expansión de una sección de caras conexas entre sí da como resultado una malla de volumen igualmente conexa. Continuando este lineamiento, se observó que en todos los resultados de expansiones correspondientes a mallas conexas se generan mallas igualmente conexas, es decir, no quedaban “espacios vacíos” entre elementos, asegurando así que el enfoque en nodos para las expansiones en lugar de las caras permite mantener

consistencia en las mallas resultantes adaptando de forma natural los nuevos elementos agregados sin tener que hacer un tratamiento previo para solucionar colisiones ni colapsos. Cabe mencionar que esto se cumple ya que las mallas de entrada deben ser convexas, por lo que si estas fuesen cóncavas los resultados esperados no serían los mismos y se debería tener especial cuidado con los colapsos y colisiones entre elementos generados, ya que las normales podrían estar apuntando de tal forma que si se proyectan, las capas generadas podrían colisionar y/o colapsar entre sí.

Con respecto a los resultados asociados a la expansión de una malla superficial correspondiente a una corteza cerebral se observó que los tiempos de computo aumentan en gran medida según la cantidad de capas expandidas, siendo tiempos aceptables para la generación de 10 capas de expansión o cantidades inferiores. Además, tal y como se observa en la figura 20 a medida que se expanden más capas sin tener en consideración la concavidad de la malla de entrada cuidando que ésta sea convexa, las expansiones comienzan a colapsar entre sí. Con respecto a la memoria utilizada, si bien existe un margen de optimización elevado, son cifras despreciables para el hardware actual.

Análogamente, en relación al desempeño de la versión que expande una malla de volumen asociada a la corteza cerebral, se observa que a medida que la malla generada posee una mayor cantidad de elementos, sus tiempos de computo se elevan drásticamente. Es importante notar que los tiempos y uso de memoria para las ejecuciones sin importar si la malla original se considera dentro del archivo de salida, los tiempos y memoria utilizada poseen diferencias marginales, por lo que incluir o no la malla original en el archivo de salida no significa un tiempo de ejecución adicional ni almacenamiento extra relevante.

CAPÍTULO 6

CONCLUSIONES

Como se presentó anteriormente, lo que se buscaba como objetivo principal era lograr expandir caras de una malla, sea superficial o volumétrica, lo cual se consiguió efectivamente para ambos tipos de malla. Además se generaron una serie de parámetros opcionales que permiten personalizar la malla expandida de tal forma que el archivo de salida cumpla con las características pertinentes para cada caso. Uno de los puntos importantes que se tuvieron en cuenta al momento de realizar la expansión fue la naturaleza de esta, es decir, estar enfocada en expandir teniendo en consideración la característica y estructura de los puntos por sobre las caras. Lo anterior requirió una serie de pre-procesamientos con el objetivo de entregar un resultado consistente con lo esperado, para lo cual fue necesario realizar métodos de reparación de normales e identificación de caras superficiales, sumado a la generación de puntos nuevos, y la escritura correcta del archivo de salida permitiendo conservar la malla original si así se desea. Con respecto a lo anterior, el trabajo concluye cumpliendo todos los objetivos principales de estudio, abordando los casos generales y entregando una primera aproximación para futuros trabajos utilizando como base este desarrollo.

Dentro de los objetivos secundarios se encontraban los parámetros opcionales, que permiten agregar maleabilidad a la malla de salida, permitiendo personalizar ciertos comportamientos del código. Si bien los resultados obtenidos para la malla 20 dejan entrever que queda mucho por desarrollar, se cumple con proponer un primer enfoque para abordar el problema acotado debido a una serie de restricciones que se propone a futuro ir solventando para poder entregar una solución completamente general y amplia que permita el trabajo con un mayor tipo de mallas. Lo anterior no quita el hecho de que la solución ofrecida corresponde a una vía factible y más eficiente que la realización del mismo trabajo a mano, lo que tomaría considerablemente más tiempo y estaría sujeto a errores mayores dado el factor humano al momento de realizar los cálculos.

6.1. Críticas al método

Dado que la solución propuesta es una primera aproximación que considera un enfoque en nodos por sobre caras para realizar expansiones, y sumado a que los trabajos previos tienden a realizar expansiones hacia dentro de las mallas con el objetivo de “rellenarlas” considerando las normales por cara como foco principal, queda mucho por mejorar a futuro siguiendo esta línea de investigación y desarrollo. Dentro de las críticas al método más importantes (sin embargo, necesarias para formular un alcance realista para el tiempo acotado) es la falta de implementación para permitir otras extensiones de archivos de mallas, como lo son `.mdl` y `.m3d`, por nombrar algunos. Lo cual podría haber permitido abordar una mayor cantidad de pruebas y una solución mucho más consistente de lo que actualmente es.

Otra de las críticas al método es su alta restricción respecto a la malla de entrada, dado que dicha

la malla debe cumplir con ser completamente conexa (aunque el resultado obtenido puede no serlo) dado que el algoritmo de reparación de normales realiza recorrido entre caras vecinas, la malla de entrada en el caso de ser volumétrica no debe tener espacios vacíos al interior, ya que el algoritmo de reparación de normales no está diseñado para cubrir este caso, cambiando el sentido de la normal para que apunte “hacia afuera” cuando en realidad se encuentra apuntando hacia adentro. Otra crítica, siguiendo el mismo lineamiento de las mallas restrictivas de entrada, es en relación a la naturaleza convexa obligatoria para las mallas de entrada, ya que el algoritmo de seccionamiento triangular de caras para realizar la reparación de normales no funciona como se espera cuando los puntos de una cara no son coplanares.

Finalmente, los tiempos de ejecución de carácter exponencial presentados en la sección anterior en relación a las pruebas del programa utilizando una malla de corteza cerebral como input son un punto clave a mejorar en el futuro. La creación de nuevas estructuras que permitan referenciar aún mejor los nodos proyectados hacia los nodos originales, el proceso de generación de elementos por capas y la reparación de normales pueden ser altamente optimizables en el futuro utilizando estructuras más complejas que permitan reducir los tiempos de búsqueda entre los arreglos de puntos y elementos. Lo anterior se vuelve uno de los focos principales para continuar el trabajo, sin desmerecer todo el abanico de posibilidades que abre este trabajo.

6.2. Posibles mejoras

Dentro de las posibles mejoras, y como se mencionó anteriormente, este trabajo entrega un primer enfoque para abordar el problema de expansión exterior de caras teniendo como foco principal los nodos, por lo cual la cantidad de posibles mejoras es amplia. Dentro de las mejoras principales se encuentran:

Tiempos de cómputo: Los tiempos de ejecución exponencial son aceptables para un primer enfoque de solución al problema, sin embargo se vuelve pertinente realizar optimizaciones para permitir que mallas de gran envergadura se resuelvan en tiempos aceptables. Para esto se propone llevar a cabo una revisión del código y evaluar la posibilidad de incluir nuevas estructuras complejas que permitan organizar las referencias de los objetos creados de una forma más ordenada, obteniendo así tiempos de búsqueda menores al original. Siguiendo esta misma línea, se sugiere desarrollar un mecanismo que permita trabajar con una lista acotada de puntos, permitiendo así acelerar el proceso de búsqueda, ya que el desarrollo presentado no realiza ninguna modificación o filtración a la lista de puntos, por lo que desarrollar un mecanismo de selección inteligente de puntos superficiales utilizando una estructura de datos pertinente que permita referenciar al arreglo original de puntos sin la necesidad de recorrer dicho arreglo original, buscando las referencias en esta nueva estructura. Lo anterior permitiría reducir, en teoría, los tiempos de cómputo asociados a una malla de volumen, equiparando los tiempos de cómputo entre una malla de volumen y una malla de superficie.

Ampliar horizonte de I/O: Si bien esta mejora no es más compleja que la anterior, sí requiere

cierto grado de importancia con el objetivo de otorgar una mayor relevancia al programa desarrollado. La inclusión de las extensiones faltantes entendiendo la estructura de construcción de cada una de estas, al igual que reducir la cantidad de restricciones se vuelve un próximo paso a seguir, persiguiendo el mismo objetivo de ampliar el horizonte de compatibilidad con los demás archivos y extensiones existentes en el área del modelado tridimensional. Si bien en esta memoria se aborda una de las extensiones principalmente utilizadas, se vuelve necesario trabajar en el desarrollo de las demás extensiones para poder así ampliar el uso del método desarrollado. En líneas generales, se requiere estudiar la construcción de los diferentes formatos de archivos tanto de entrada como de salida, permitiendo entregar un módulo de escritura y lectura consistente y amplio en término de soporte de extensiones, todo esto aplicado tanto a mallas superficiales como volumétricas, respetando las características de cada una de ellas.

Modelo restrictivo: Debido a que este trabajo presenta un primer enfoque para la resolución de expansiones hacia afuera teniendo el foco del cálculo en los nodos en lugar de las caras la cantidad de restricciones a tener en cuenta es considerable. La disminución de restricciones desarrollando algoritmos para cubrir los casos que no se encuentran cubiertos actualmente por la solución entregada es el próximo paso a seguir para continuar con el objetivo de dar una solución aún más sólida que pueda mantener la consistencia actual pero abordando casos que originalmente se encuentran fuera de los límites de esta memoria. Una de las restricciones a trabajar es estudiar el comportamiento de las mallas cóncavas, ya que con el mecanismo actual no cuenta con planes de corrección ante colisiones y colapsos al encontrarse dos normales apuntando de tal forma que sus planos se cortan entre sí en algún punto. Otro de los puntos de mejora posible consiste en permitir realizar expansiones en mallas de volumen que posean un vacío en su interior, es decir, mallas que tengan forma de cascaron con espesor. Dichas mallas tampoco se encuentran cubiertas en este trabajo, sin embargo se propone como trabajo a futuro. Para llevar a cabo esto, se debe realizar una reformulación de la función de filtro de capa superficial, ya que dicha capa en su algoritmo utiliza la lógica de seleccionar todas las caras que pertenezcan a un único elemento, seleccionando así tanto las caras correctas (las que realmente forman el cascaron exterior) como las caras de la zona vacía al interior de la malla. Esto afecta posteriormente al algoritmo de reparación de normales, razón por la cual se propone como trabajo a futuro. Finalmente se tiene la restricción correspondiente al algoritmo de reparación de normales, específicamente al seccionamiento triangular de las caras que posean más de tres puntos, ya que se utilizó elementos de la clase *TriMesh* la cual trabaja exclusivamente con triángulos. El problema en este caso radica en que el seccionamiento no queda correctamente definido cuando los puntos de la cara correspondiente a seccionar no son coplanares, conllevando a resultados poco consistentes con lo esperado ya que genera enlaces donde no debería haberlos, permitiendo así la creación de caras triangulares que no pertenecen en ningún caso a la malla original.

Ampliar funcionalidades manteniendo consistencia: Para continuar con el desarrollo de una solución más completa, se vuelve importante también diseñar una herramienta que permita la selección inteligente de caras para la *whitelist* ante una malla de volumen, dado que el

sistema entregado permite tener un orden claro al momento de construir una *whitelist* cuando se trata de una malla de superficie, sin embargo, al momento de realizar la operación para una malla de volumen se pueden generar resultados no esperados dado que no es posible a priori identificar con completa certeza qué cara pertenece a qué elemento, ya que la lista del archivo *whitelist* recibe los índices de las caras en lugar de los índices de los elementos, por lo que se vuelve una mejora importante a futuro el poder desarrollar una interfaz que permita seleccionar de manera consistente y sin ambigüedades las caras a expandir para una malla de volumen. Se espera que dicha herramienta incluya este trabajo dentro de sus funcionalidades con el objetivo de ofrecer una solución más completa que la actual.

Implementar paralelismo: Dado que los cálculos desarrollados a lo largo del código son en su mayoría sumas, restas y multiplicaciones repetidas sucesivamente para una serie de nodos, teniendo como resultado una tarea matemáticamente sencilla pero computacionalmente importante (sujeto a la cantidad de nodos, caras y elementos de la malla a tratar) agregar cómputo paralelo se vuelve una posible vía de desarrollo y mejora respecto a este trabajo. Mas específicamente utilizando *CUDA* se presume que se podrían reducir los tiempos de construcción de las expansiones de manera drástica para mallas de gran envergadura y cantidad de elementos. Lo anterior dado que los cálculos realizados son repetitivos, recurrentes y se aplican los mismos cálculos para cada uno de los elementos en cuestión a lo largo del código. Para llevar a cabo lo anterior, se sugiere implementar diversos *kernels* para cada uno de los procesos de cálculos matemáticos, siendo estos especialmente el cálculo de las normales por nodo y la generación de puntos nuevos. Una vez implementado lo anterior, comparar resultados con este trabajo para evaluar mejoras en el desempeño del generador de mallas.

Tal y como se mencionó anteriormente, este trabajo sienta las bases que abre una línea de desarrollo respecto a la expansión de mallas hacia el exterior teniendo como foco principal la expansión por nodos en lugar de caras, permitiendo así tener una nueva forma de abordar problemas del área, incluyendo la posibilidad de agregar a este método programación paralela y posteriormente evaluar desempeño respecto a los métodos existentes y actualmente utilizados.

Si bien este proceso se acotó para realizar expansión exterior, no se descartan líneas de investigación que le den otro enfoque, como por ejemplo, buscar una alternativa de expansión interior para generar elementos de volumen reduciendo la cantidad de colapsos y colisiones, dado que al trabajar con el foco en los puntos, la forma y naturaleza de las caras expandidas pierden importancia, permitiendo generar elementos de volumen nuevos teniendo en cuenta el origen en su espectro más amplio, esto es, analizando a nivel estructural las conexiones de cada nodo con las caras correspondientes, permitiendo trabajar con una mayor cantidad de datos y por consiguiente, información.

Finalmente, se cumplió satisfactoriamente con los resultados esperados, sentando así las bases de desarrollo para una nueva arista de investigación y desarrollo de generación de mallas con un foco distinto, aplicable a una amplia gama de campos y trabajos a futuro.

REFERENCIAS BIBLIOGRÁFICAS

- [Bänsch, 1991] Bänsch, E. (1991). Local mesh refinement in 2 and 3 dimensions. *IMPACT of Computing in Science and Engineering*, 3(3):181–191.
- [Davies *et al.*, 2007] Davies, D. R., Davies, J., Hassan, O., Morgan, K., y Nithiarasu, P. (2007). Investigations into the applicability of adaptive finite element methods to two-dimensional infinite prandtl number thermal and thermochemical convection. *Geochemistry Geophysics Geosystems*, 7207.
- [Jaillet y Lobos, 2021] Jaillet, F. y Lobos, C. (2021). Fast quadtree/octree adaptive meshing and re-meshing with linear mixed elements. *Engineering with Computers*. (in press).
- [Lobos *et al.*, 2010] Lobos, C., Payan, Y., y Hitschfeld, N. (2010). Techniques for the generation of 3d finite element meshes of human organs. En *Informatics in Oral Medicine: Advanced Techniques in Clinical and Diagnostic Technologies*, pp. 126–158. IGI Global.
- [Samani *et al.*, 2001] Samani, A., Bishop, J., Yaffe, M. J., y Plewes, D. B. (2001). Biomechanical 3-d finite element modeling of the human breast using mri data. *IEEE transactions on medical imaging*, 20(4):271–279.
- [Valdés, 2012] Valdés, M. (2012). Expansión de frente para generar una malla parcial de volumen.