UNIVERSIDAD DE LA REPÚBLICA URUGUAY

FACULTAD DE INGENIERÍA

Simulación de la transferencia de calor en entornos urbanos mediante Elementos Finitos

Informe de Proyecto de Grado presentado por

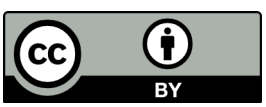
Federico Mazzoni, Rodrigo Marote

en cumplimiento parcial de los requerimientos para la graduación de la carrera de Ingeniería en Computación de Facultad de Ingeniería de la Universidad de la República

Tutor

José Pedro Aguerre

Montevideo, 22 de marzo de 2024



Agradecimientos

Agradecemos profundamente a todas las personas y entidades que contribuyeron de manera significativa a la realización de esta tesis. Sus apoyos, consejos y alientos fueron esenciales en este largo y desafiante proyecto académico.

En primer lugar, queremos expresar nuestro más sincero agradecimiento al supervisor de la tesis, José Pedro Aguerre, por su incansable orientación, apoyo y experiencia en cada paso del trabajo. Sus valiosas sugerencias, paciencia y dedicación fueron fundamentales para el éxito de este proyecto.

En segundo lugar, queremos remarcar el papel crucial de la Facultad de Ingeniería y sus distinguidos profesores. Su compromiso con la excelencia académica y su valiosa enseñanza han sido la base sobre la cual construimos esta tesis. La educación que recibimos en esta institución ha sido un pilar fundamental en nuestro desarrollo profesional.

A nuestros compañeros de carrera, les agradecemos por compartir este viaje con nosotros. Sus debates, intercambio de ideas y apoyo mutuo hicieron que esta experiencia fuera más enriquecedora y memorable. Juntos superamos desafíos y celebramos triunfos, y estamos agradecidos por cada momento compartido.

Queremos extender nuestra gratitud a Luis, Lourdes, Sebastián y Carolina, padre, madre y hermanos de Federico, así como a Julio, Fernanda, Guillermo y Juan Diego, padre, madre y hermanos de Rodrigo. Su apoyo inquebrantable y comprensión durante este tiempo fue fundamental para nuestro bienestar emocional y nos brindaron la tranquilidad necesaria para concentrarnos en nuestro proceso.

A nuestros amigos, les agradecemos por su constante aliento y por estar siempre dispuestos a escucharnos y brindarnos palabras de ánimo cuando más las necesitábamos.

Esta tesis es el resultado de un esfuerzo colectivo, y a cada uno de ustedes les debemos un profundo agradecimiento. Su apoyo constante y confianza nos impulsaron a alcanzar este logro. Esperamos que este trabajo honre sus contribuciones y que continúe siendo un testimonio de nuestra dedicación compartida a la excelencia académica y el crecimiento personal. Gracias a todos por ser parte de este importante capítulo en nuestras vidas.

Resumen

El crecimiento continuo de las ciudades ha llevado a que más de la mitad de la población mundial resida en entornos urbanos en la actualidad. En este contexto, el estudio de los intercambios térmicos en ciudades se ha convertido en un campo de investigación esencial para abordar los desafíos relacionados con la sostenibilidad y el confort en áreas urbanas.

En el contexto del estudio de la transferencia de calor en las ciudades se encuentra la tesis de posgrado publicada por José Pedro Aguerre, supervisor de esta tesis, titulada "Radiation techniques for urban thermal simulation with the Finite Element Method". Este trabajo estableció una metodología basada en el Método de Elementos Finitos y el software Cast3M para analizar los intercambios térmicos en entornos urbanos. A pesar de los avances logrados, la experiencia con el software Cast3M planteó desafíos significativos y limitaciones, lo que motiva la búsqueda de alternativas más eficientes.

El objetivo principal de esta tesis es evaluar la utilización de otros paquetes de software de elementos finitos para replicar los resultados obtenidos en la investigación mencionada. Para cumplir dicho objetivo, en primer lugar se evalúan las características de diferentes herramientas de análisis de intercambios térmicos basadas en elementos finitos, y posteriormente, se selecciona una de ellas para llevar a cabo una reconstrucción del modelo geométrico original. En segundo lugar, se ejecutan simulaciones de transferencia de calor para analizar los resultados y compararlos con los obtenidos en el estudio previo. Se estudia la eficiencia del sistema en cuanto a tiempos de ejecución y utilización de recursos computacionales, así como se evalúan los resultados desde un punto de vista analítico mediante gráficos de temperatura y termografías simuladas.

Palabras clave: Elementos finitos, Análisis térmico, Simulación computacional.

Índice general

Capítulo 1

Introducción

1.1. Contexto

El crecimiento continuo de las ciudades a nivel mundial ha provocado un cambio significativo en la manera en que las personas habitan el planeta. En 2020, las metrópolis con más de 300.000 habitantes albergaban alrededor de un tercio de la población global. Para 2035, se proyecta que las metrópolis albergarán al 39 % de la población mundial, lo que significa que casi mil millones de personas se convertirán en habitantes metropolitanos en los próximos años (United Nations, 2020). Este fenómeno ha impulsado la necesidad de comprender y abordar los desafíos que se presentan en estas áreas densamente pobladas, especialmente en lo que respecta a la sostenibilidad y el confort de sus habitantes.

Un aspecto crucial para afrontar estos retos es el estudio de los intercambios térmicos en el contexto urbano. La simulación computacional del calor, en particular, ha despertado un creciente interés en diversas disciplinas, incluyendo la física de edificios, la arquitectura y la ingeniería ambiental (Aguerre, Fernández, Besuievsky, y Beckers, 2017). Este interés ha dado lugar al desarrollo de la física urbana como una herramienta de conexión entre las disciplinas mencionadas.

Entre las técnicas que se utilizan para resolver simulaciones térmicas, destaca el método de elementos finitos (MEF), un método numérico que descompone problemas complejos en elementos discretos y resuelve las ecuaciones resultantes.

En este contexto, se sitúa la tesis de posgrado titulada *"Radiation techniques for urban thermal simulation with the Finite Element Method"* (Aguerre, 2020) realizada por el supervisor de la presente investigación. Se referirá a este trabajo como el "trabajo original" o la "tesis de posgrado". Su objetivo principal es el estudio y desarrollo de técnicas de radiación que son utilizadas en un software de elementos finitos con el propósito de simular computacionalmente la transferencia de calor en entornos urbanos.

En dicho trabajo, se estableció una metodología basada en MEF y se empleó el software Cast3M para analizar los intercambios térmicos en un entorno urbano, modelado a partir de una ubicación real. Cast3M es una herramienta de simulación que utiliza dicha metodología para realizar análisis térmicos y mecánicos en estructuras y fluidos (Di Paola, 2023). Los resultados obtenidos se contrastaron con temperaturas registradas mediante termografías urbanas. Asimismo, se aplicaron técnicas de *rendering* para generar representaciones visuales de imágenes térmicas a partir de los resultados de las simulaciones. Estas representaciones confirman la precisión de la metodología propuesta en los casos analizados. Además, se registró un desempeño computacional con un nivel de eficiencia que hace posible la realización de este tipo de estudios en computadoras personales.

1.2. Motivación y objetivos

A pesar de lograr resultados interesantes y avances significativos en el tema, la experiencia de trabajo con el software Cast3M reveló desafíos y limitaciones técnicas que motivaron la búsqueda de alternativas más eficientes.

El objetivo central de esta tesis es replicar los modelos y resultados obtenidos en la tesis de posgrado, pero utilizando un nuevo software de elementos finitos. Para esto, es necesario investigar los sistemas de software de elementos finitos disponibles, y elegir uno de ellos en base a su capacidad para cumplir los objetivos planteados, además de sortear las limitaciones técnicas y de usabilidad que presentó Cast3M.

Una vez seleccionado el nuevo software, se deberá representar el mismo caso de estudio analizado

en el trabajo original, basado en mediciones experimentales realizadas el día 23 de abril de 2017 en el callejón urbano de la calle *Rue des Tonneliers* de la ciudad de *Bayonne* en Francia.

El primer paso para representar esta realidad es construir una geometría tridimensional que represente de manera simplificada el callejón urbano. Se puede observar en la Figura 1.1 una comparación entre el distrito real y un boceto de la geometría que se desea construir.

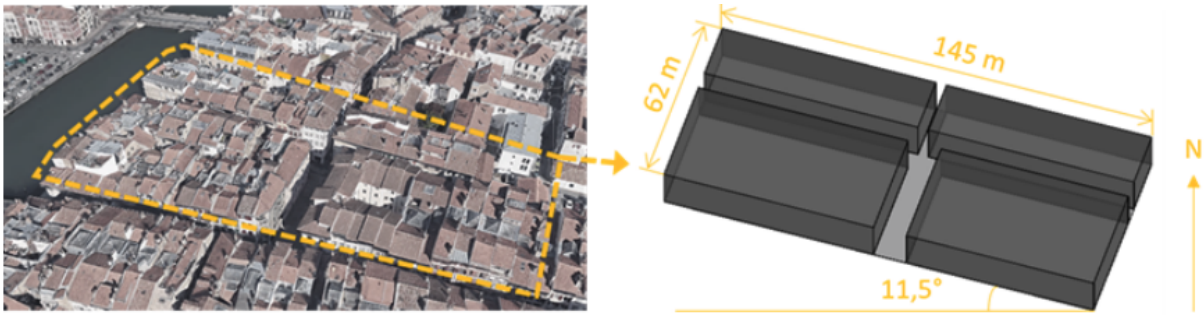


Figura 1.1: Imagen aérea del caso de estudio y su modelo geométrico (Aguerre, 2020).

El siguiente paso es la discretización de la geometría a través de una *malla de elementos*. Este concepto se detallará en secciones posteriores, junto con las características que la misma debe satisfacer. A modo de ejemplo, se puede observar la malla que se desea construir en la Figura 1.2.

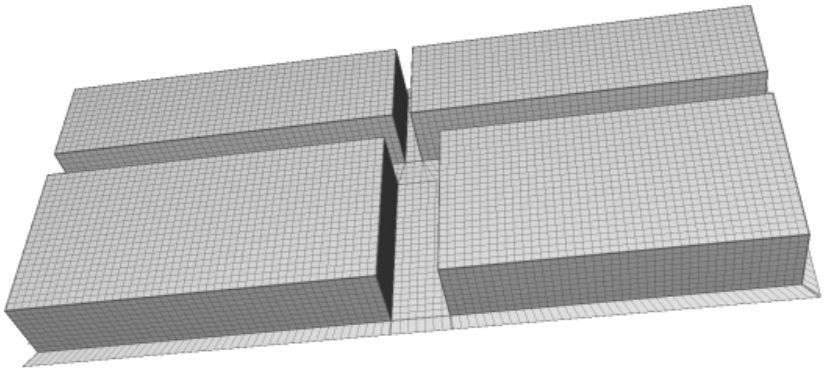


Figura 1.2: Ejemplo de malla a construir (Aguerre, 2020)

.

Posteriormente, se deben configurar los materiales y los parámetros del modelo. Dada la heterogeneidad de materiales de construcción utilizados en una ciudad real, se debe contar con la capacidad de asignar distintos materiales a distintas partes de la geometría, cada uno con sus propiedades físicas específicas. Además, se deben definir temperaturas iniciales para todos los elementos de la geometría, y valores de temperatura del cielo para cada momento del día. También se requiere configurar valores de *convección* y *radiación*. El nuevo software debe ser capaz de cargar valores de flujo radiativo obtenidos como salida del módulo de radiación implementado en la tesis original.

Con el modelo construido, el siguiente paso es ejecutar una simulación térmica. La simulación debe cubrir un lapso de tiempo de 24 horas. Los resultados obtenidos en esta simulación se deben comparar con los resultados del trabajo original, y se espera que sean similares. También resulta de interés analizar el uso de recursos del sistema por parte del software, los tiempos de ejecución de las simulaciones, y determinar el tamaño máximo de la geometría (medido en cantidad de elementos de la malla) soportado antes de que la ejecución de las simulaciones se vuelva inviable, ya sea porque el tiempo de ejecución es considerablemente extenso o porque el uso de memoria del sistema es muy elevado.

Por último, el mayor objetivo de esta tesis es proveer una evaluación final del software elegido para resolver el problema planteado, de forma de asesorar trabajos futuros en el área.

1.3. Estructura del documento

En primer lugar, se presenta un capítulo de revisión de antecedentes, en el cual se introducen conceptos esenciales para comprender el contexto de este trabajo. En este capítulo también se describe de forma detallada el trabajo que se planea replicar, se exploran diversas opciones de sistemas de software MEF potencialmente adecuadas para el desarrollo del proyecto, seleccionando una de estas opciones y justificando la elección, y por último, se analizan artículos que abordan proyectos realizados con el software *Ansys* (uno de los programas evaluados). Dichos artículos presentan trabajos que emplean la mayoría de los conceptos relevantes del tema (radiación, convección, entre otros) para el estudio de las transferencias de calor en determinados contextos.

Posteriormente, se dedica un capítulo a la implementación del caso de estudio en *Ansys*, donde se describe el desarrollo de la investigación y la construcción del modelo necesario para llevar a cabo las simulaciones. Aquí se abordan aspectos como el licenciamiento y la instalación de software, así como la construcción del modelo sobre el que se ejecutarán simulaciones de transferencia de calor.

A continuación se analizan los resultados de las simulaciones por medio de la evolución de temperatura para tres puntos en particular de la ciudad, y se contrastan con los resultados obtenidos para estos mismos puntos en la tesis original. Además, se muestran resultados de termografías simuladas y su comparación con los resultados experimentales. Por otro lado, se presentan datos estadísticos de las ejecuciones, que abarcan la utilización de recursos del sistema, la cantidad de elementos del modelo para cada simulación, el tiempo de ejecución, entre otros.

Finalmente, se destina un capítulo a las conclusiones derivadas del estudio, donde también se esbozan las posibilidades para investigaciones futuras.

Capítulo 2

Revisión de antecedentes

En esta sección, se lleva a cabo la revisión de antecedentes relacionados con el proyecto. En primer lugar, se introducen los conceptos necesarios de transferencia de calor y simulación del calor para comprender el contexto del proyecto. Luego, se describe detalladamente la tesis de posgrado de la que parte este trabajo. A continuación, se investigan las distintas opciones de software disponibles para cumplir con los objetivos, detallando el proceso de selección y justificando la elección del software Ansys como la plataforma principal para el estudio. Por último se examinan artículos que describen ejemplos de análisis de transferencia de calor con elementos finitos utilizando Ansys.

2.1. Conceptos previos

2.1.1. Transferencia de calor

El calor se define como la transferencia de energía térmica, determinada por la temperatura de los cuerpos. Dicha transferencia ocurre de tres formas principales: conducción, convección y radiación.

2.1.2. Conducción

La conducción térmica es la transferencia de calor que ocurre dentro de los cuerpos, cuando las moléculas cercanas intercambian energía sin movimiento real, o cuando los electrones libres se desplazan. Mayores diferencias de temperatura implican mayores flujos de conducción, donde el calor se conduce desde las partes calientes de un cuerpo hacia las partes frías.

Flujo de calor

El flujo de calor (*Heat flux*), también denominado flujo térmico o densidad de flujo de calor se utiliza para cuantificar la energía transferida, es una cantidad vectorial que describe el flujo de energía por unidad de área y por unidad de tiempo. Se mide en Wm^{-2}.

Ley de Fourier

El flujo de calor relacionado a la conducción térmica en una dirección se expresa por la *Ley de Fourier*:

$$q_{cond} = -k\nabla T \tag{2.1}$$

Donde q_{cond} es el vector de flujo de calor, k es la conductividad térmica del material ($W\,m^{-1}K^{-1}$), y ∇T es el gradiente de temperatura ($\frac{\partial T}{\partial x}$ en $K\,m^{-1}$).

Ecuación de calor por conducción

La ecuación de conducción de calor se deriva a partir del balance de energía en un volumen diferencial. Este análisis establece una igualdad entre la energía recibida y generada, así como entre la energía almacenada y emitida. Esta equivalencia refleja la conservación de la energía, indicando que la energía no se crea ni se destruye, sino que se transforma.

En el contexto de coordenadas cartesianas tridimensionales (x, y, z), la derivación de esta ecuación implica la aplicación de la ley de conservación de la energía en un volumen hexahédrico de control.

La ecuación de calor por conducción resultante se expresa como:

$$k\Delta^2 T + G = \rho_m c_p \frac{\partial T}{\partial t} \tag{2.2}$$

donde $\Delta^2 T$ es el Laplaciano de la temperatura y G es el término que representa el calor generado por unidad de volumen, el cual equivale a 0 en escenarios donde no hay generación interna de calor. ρ_m es la densidad de masa del material, c_p es el calor específico del material y $\frac{\partial T}{\partial t}$ es la derivada parcial de la temperatura respecto al tiempo.

Condiciones iniciales y condiciones de borde

La resolución de la ecuación diferencial de calor por conducción implica la especificación de un conjunto de condiciones iniciales y de borde. En este contexto, se pueden imponer dos tipos de condiciones de borde: aquellas que fijan la temperatura en la frontera o aquellas que establecen un flujo de calor en la misma. Por otro lado, el tiempo t se presenta como un término de primer orden en la ecuación 2.2, permitiendo que un valor de temperatura inicial sea suficiente para resolver la ecuación.

2.1.3. Convección

La convección es la transferencia de calor relacionada con el movimiento de líquidos y gases. Las moléculas que se desplazan de regiones calientes a frías llevan energía, y este fenómeno se conoce como convección natural. Su contraparte, llamada convección forzada, es causada por fuerzas externas, como el viento o las corrientes de fluidos.

Ley de enfriamiento de Newton

El flujo de calor relacionado a la convección, en su forma unidimensional, se expresa por la *Ley de enfriamiento de Newton*:

$$q_{conv} = h(T - T_a) \tag{2.3}$$

donde q_{conv} es el vector de flujo de calor, h es el coeficiente de convección ($W\,m^{-2}K^{-1}$), y T_a es la temperatura del fluido. El coeficiente de convección cuantifica la influencia de las propiedades del fluido, de la superficie y del flujo cuando se produce transferencia de calor por convección.

2.1.4. Radiación

La radiación térmica es la transferencia de calor en forma de ondas electromagnéticas a través de un medio material o el vacío. Todas las superficies con una temperatura por encima del cero absoluto emiten radiación a diferentes longitudes de onda. A nivel de la superficie de la tierra, la radiación se emite desde la superficie de un cuerpo y viaja en línea recta hasta alcanzar otra superficie, donde es absorbida, emitida, reflejada y transmitida.

Balance de radiación de la Tierra

La radiación solar es la principal fuente de energía del planeta. Es importante destacar que la longitud de onda de la radiación depende de la temperatura del cuerpo que la emite. El Sol, con una temperatura aproximada de 5700 K, emite radiación de onda corta, mientras que la Tierra, con una temperatura de aproximadamente 300 K, emite radiación de onda larga. La radiación emitida por el Sol viaja a la velocidad de la luz por el espacio hasta llegar a la Tierra. La atmósfera del planeta absorbe y refleja una fracción de la radiación que recibe, y transmite el resto hacia la superficie terrestre. La superficie eleva su temperatura debido a la radiación recibida y emite radiación de onda larga.

El balance de energía del sistema Tierra-atmósfera se puede describir con el diagrama de la Figura 2.1. Del total de energía de radiación que llega al sistema desde el Sol (100 %), el 30 % es reflejado por la atmósfera y la superficie de la Tierra, el 23 % es absorbido por la atmósfera y el 47 % es absorbido por la superficie terrestre. Dado que el sistema debe estar en equilibrio térmico, la energía absorbida (70 %) debe ser reemitida al espacio exterior. De este 70 %, el 12 % corresponde a la emisión de onda larga de la superficie terrestre que atraviesa la atmósfera, y el 58 % es la emisión de onda larga de la atmósfera. El equilibrio local entre la superficie terrestre y la atmósfera se mantiene a través de intercambios de onda larga, calor latente y calor sensible.

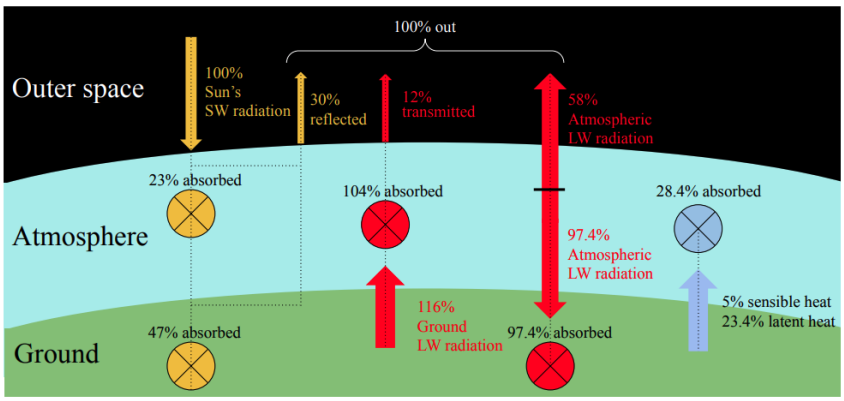


Figura 2.1: Balance de energía radiativa de la Tierra. Una fracción de la radiación solar (amarilla) atraviesa la atmósfera y llega a los objetos en la superficie del planeta. Estos objetos emiten radiación térmica (roja) hacia otras superficies y de nuevo hacia la atmósfera (Aguerre, 2020).

Los objetos en la superficie del planeta intercambian radiación entre sí. La radiación de onda corta que no es absorbida por los materiales se refleja y se dirige hacia otras superficies. La radiación de onda larga emitida es interceptada y absorbida por otros objetos. Estos intercambios ocurren en un contexto de interacciones de calor multifísicas que incluyen la conducción, la convección y otros fenómenos. Se observa un ejemplo en la Figura 2.2.

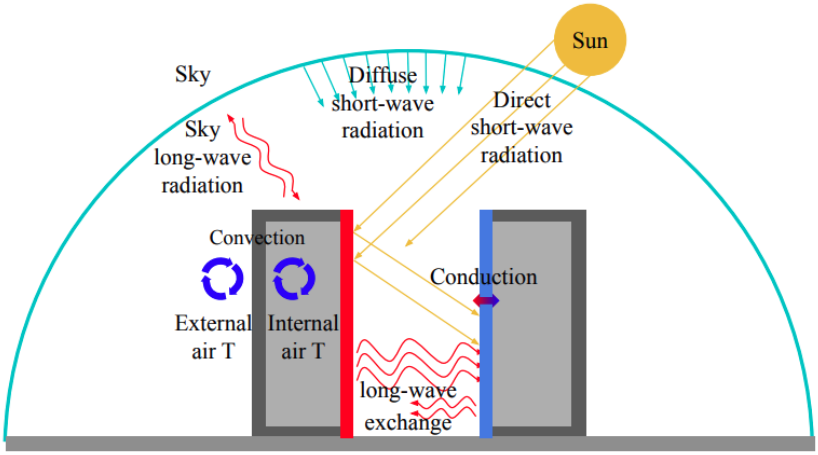


Figura 2.2: Diagrama de intercambios de calor en un estudio térmico urbano (Aguerre, 2020).

Cuerpo negro

Un cuerpo negro se define como un cuerpo ideal que absorbe toda la radiación incidente. En equilibrio termodinámico local, un cuerpo negro emite la misma cantidad de radiación que absorbe, y por lo tanto se dice que emite la máxima cantidad de radiación. En esta tesis, se asume que los materiales son cuerpos negros en la onda larga, por lo que emiten y absorven radiación pero no la reflejan.

Ley de Stefan-Boltzmann

El flujo de calor relacionado a la radiación en un cuerpo negro se expresa por la *Ley de Stefan-Boltzmann*:

$$q_{rad} = \sigma T^4 \tag{2.4}$$

donde q_{rad} es el vector de flujo de calor y σ es la constante de Stefan-Boltzmann ($5,670310^8 W\, m^2 K^4$). En los cuerpos del mundo real se emite una fracción definida por su emisividad (ϵ):

$$q_{rad} = \epsilon \sigma T^4 \tag{2.5}$$

Método de Radiosidad

El método de radiosidad (Goral, Torrance, Greenberg, y Battaile, 1984) es una técnica de iluminación global basada en MEF y diseñada para escenas compuestas únicamente por superficies difusas. El término "radiosidad"hace referencia al flujo radiativo total (emitido y reflejado) que sale de una superficie por unidad de área (W/m^2).

Se presenta en Ecuación 2.6 la "ecuación de radiosidad", donde $J(x)$ y $E(x)$ son los valores de radiosidad y emisión en el punto x, $\rho(x)$ es el coeficiente de reflectividad del material y $G(x, x')$ es un factor geométrico que relaciona la visibilidad entre dos puntos x y x'.

$$J(x) = E(x) + \rho(x) \int_{\Omega} J(x')G(x, x')dA' \tag{2.6}$$

La ecuación de radiosidad continua puede transformarse en una suma de valores al discretizar la geometría de la escena en un conjunto finito de elementos denominados parches. Esta formulación permite expresar el problema con el siguiente conjunto de ecuaciones lineales:

$$J_i = E_i + R_i \sum_{j=1}^{n} J_j F(i, j), \forall i \in \{1...n\} \tag{2.7}$$

Este conjunto de ecuaciones lineales se expresa algebraicamente en la ecuación de radiosidad discreta (Ecuación 2.8).

$$(I - RF)J = E, \tag{2.8}$$

donde:

- I es la matriz identidad.
- R es una matriz diagonal que contiene el índice de reflectividad de cada parche.
- J es el vector de radiosidad desconocido.
- E es el vector de emisión.
- F(i, j) es un número entre 0 y 1 que expresa el factor de vista entre los parches i y j. Este valor indica la fracción de la potencia luminosa que va de i a j.

La matriz de factores de vista es una matriz $n \times n$, donde n es la cantidad de parches de la escena. El cálculo de dichos factores es costoso en términos de memoria; se debe calcular la visibilidad entre cada parche, por lo que la cantidad de factores de vista a procesar es de n^2. El sistema lineal se puede resolver por medio de métodos numéricos directos que se enfocan en encontrar la inversa de la matriz de radiosidad: $M = (I - RF)^{-1}$. También es posible hallar J por medio de métodos iterativos como Jacobi o Gauss-Seidel (Aguerre, 2020).

2.1.5. Simulación del calor

La simulación del calor es una técnica para modelar y prever cómo se distribuye y transfiere el calor en un sistema. Se emplean diversos métodos para resolver la ecuación 2.2 de transferencia de calor por conducción, de los cuales destacan:

- Diferencias finitas: este método discretiza el dominio y resuelve las ecuaciones diferenciales mediante la aproximación de las derivadas a ecuaciones de diferencias.
- Volúmenes finitos: en este enfoque, el dominio se discretiza en volúmenes y se aplican las leyes de conservación en cada volumen.
- Analogías electrotérmicas: este método modela la conducción a través de un caso de estudio con una red de resistencias y condensadores. Luego, se resuelve la simulación mediante analogías entre el flujo de calor y el flujo de corriente eléctrica.
- Elementos finitos: método utilizado en la tesis original. Se describe en 2.1.7.

2.1.6. Análisis estacionarios y transitorios

Existen dos tipos de problemas estudiados por el análisis de transferencia de calor. El primero, denominado transferencia de calor en estado estacionario o *Steady State*, puede considerarse como un sistema en el cual ha transcurrido un tiempo suficiente largo para que se alcance el equilibrio de temperatura. El segundo problema, denominado transferencia de calor transitoria o *Transient*, permite la evolución de la temperatura como función del espacio y el tiempo. En este caso, las condiciones de borde pueden ser diferentes en cada paso de tiempo, resultando en un conjunto de temperaturas diferentes en cada paso.

2.1.7. Método de Elementos Finitos (MEF)

El MEF es un método numérico utilizado para resolver problemas complejos de ingeniería y ciencias aplicadas que involucran ecuaciones diferenciales parciales, incluida la discretización de la Ley de Fourier de transferencia de calor. La discretización divide el problema inicial en partes más pequeñas, conocidas como "elementos finitos". El sistema resultante de ecuaciones se resuelve mediante métodos numéricos para minimizar la función de error involucrada.

En el MEF, la temperatura puede variar dentro de los elementos. Esta variación se describe mediante funciones polinomiales de orden 1, 2 o superior. Las funciones que describen el comportamiento de la interpolación se conocen como funciones de forma e influyen en los resultados obtenidos mediante el MEF. Para cualquier punto del dominio, la ecuación de temperatura en función de estas funciones de forma se define como:

$$\hat{T}(p,t) = \sum_{j=1}^{N} T_j(t) N_j(p) \quad \forall p \in \Pi \tag{2.9}$$

Donde p representa cualquier punto del dominio Π, t el tiempo y $\hat{T}$ aproxima la temperatura de p. El término T_j representa la temperatura en función del tiempo, y el término N_j representa la función de forma para cada uno de los N nodos que definen el elemento al que pertenece p.

Esta aproximación se sustituye en la ecuación 2.2, se desarrolla y se aplican las condiciones iniciales y de borde vistas en la subsección 2.1.2. La obtención de la ecuación resultante se aborda detalladamente en la subsección 3.1.1 del trabajo original. Se obtiene el siguiente sistema de ecuaciones expresado en forma matricial:

$$C \left\{ \frac{\partial T}{\partial t} \right\} + K \{T\} = \{f\} \tag{2.10}$$

Donde C es la matriz de capacitancia y K es la matriz de conductividad. $\left\{\frac{\partial T}{\partial t}\right\}$ es el vector de derivadas de la temperatura con respecto al tiempo, $\{T\}$ es el vector de incógnitas, y $\{f\}$ es el vector de carga. Los valores de cada elemento son los siguientes:

$$\begin{aligned} C_{ij} &= \int_{\Pi} \rho c_c p N_j N_i \, d\Pi \\ K_{ij} &= \int_{\Pi} k \nabla N_j \nabla N_i \, d\Pi + \int_{\Gamma_q} h N_j N_i \, d\Gamma_q \\ \{f\}_i &= \int_{\Pi} N_i G \, d\Pi + \int_{\Gamma_q} N_i h T_a \, d\Gamma_q - \int_{\Gamma_q} N_i q \, d\Gamma_q \end{aligned} \tag{2.11}$$

La construcción de las matrices K y C generalmente se logra calculando matrices de elementos y luego ensamblando los resultados en las matrices globales. Las matrices de elementos se obtienen mediante la integración local sobre cada elemento. La integración global es igual a la suma de todas las integrales locales, ya que el dominio está discretizado de tal manera que los elementos no se superponen.

2.1.8. Malla de elementos

Para discretizar el dominio de un problema, el MEF se vale de una malla de elementos (también conocida como *mesh* en inglés). Una malla de elementos es una estructura compuesta por una red de puntos o nodos interconectados, formando elementos como triángulos, cuadriláteros, tetraedros o hexaedros, entre otros (ejemplo en la Figura 2.3). Los elementos finitos quedan definidos por los vértices pertenecientes a la malla. Una malla bien generada y adecuadamente refinada puede capturar detalles complejos y comportamientos locales, lo que permite obtener resultados más precisos y cercanos a la

realidad. Por otro lado, una malla mal construida o con poca resolución puede conducir a errores y resultados poco confiables.

Una característica fundamental para una malla bien construida es que sea *conforme*. Una malla conforme es aquella donde no existen intersecciones ni superposiciones, y donde ningún nodo de un elemento se encuentra en el borde o cara de otro elemento, por lo tanto, los elementos comparten bordes.

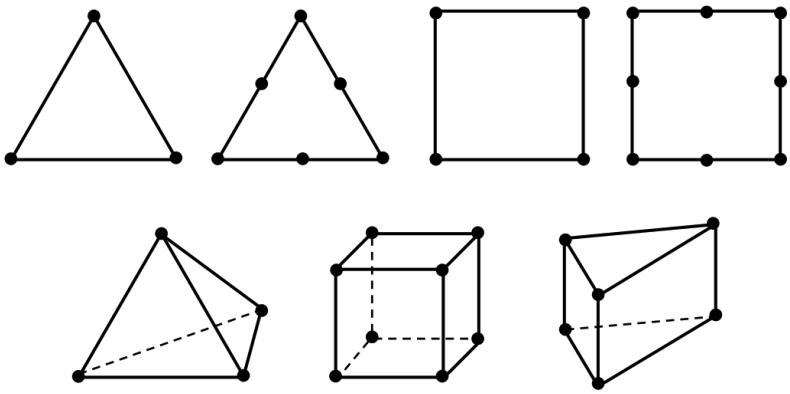


Figura 2.3: Ejemplos de elementos 2D y 3D en el MEF (Aguerre, 2020)

2.2. Resumen del capítulo 6 de la tesis original

Este trabajo fue publicado por José Pedro Aguerre en marzo de 2020 en Montevideo, Uruguay, como tesis de doctorado. En esta tesis se presenta una metodología basada en MEF para el análisis térmico de escenas urbanas, y se desarrolla un conjunto de técnicas de radiación (para radiación solar y térmica) que son integradas en el software MEF Cast3M.

Para la parte experimental, se utiliza un caso de estudio basado en una campaña de medición en la calle *Rue des Tonneliers* en la ciudad de *Bayonne*, Francia. Esta calle se encuentra en *Petit Bayonne*, uno de los distritos más densos de la ciudad de *Bayonne*. Las características de la zona hacen que el estudio de los intercambios térmicos que allí ocurren sea significativamente desafiante, principalmente debido a las múltiples interreflexiones de radiación de onda corta, pero también debido a la gran variedad de materiales y la complejidad geométrica de la escena.

2.2.1. Modelado de la geometría

El trabajo original utiliza dos modelos con distintos niveles de complejidad: uno simple y otro más detallado. Este trabajo se enfoca en el modelo simple, quedando por fuera del alcance del proyecto la replicación del modelo detallado.

El entorno de la calle Tonnelier se modela como cuatro "cajas" que representan cuatro bloques o manzanas de edificios. Con el objetivo de simplificar la geometría, los edificios se agrupan en estas cuatro cajas. El espesor de todas las paredes y techos es de 0,18m. También se unifica la altura de los edificios en 14,3m, valor que corresponde al promedio de altura de los edificios reales. Se incluye una caja adicional con un espesor de 1,0m para representar la superficie del suelo delimitada por los bloques de edificios. Se pueden observar los cuatro bloques en la Figura 1.1.

2.2.2. Propiedades de las superficies

Las superficies del entorno presentan una notable heterogeneidad de materiales que se simplifica considerando solamente cinco tipos de superficies: paredes, techos, pisos, pavimento de la calle y suelo. Las propiedades térmicas y ópticas de estas superficies se describen por medio de cuatro parámetros: coeficiente de reflexión difusa de onda corta (r), emisividad de onda larga (ϵ), conductividad térmica (k [W m^{-1} K^{-1}]) y capacidad calorífica volumétrica (c_v [J m^{-3} K^{-1}]). Sus valores se muestran en la Figura 2.4. Estos valores fueron definidos a partir de la recopilación de datos experimentales y de hipótesis basadas en observaciones *in situ*. Para simplificar los intercambios de radiación de onda larga, los elementos se consideran cuerpos negros, por lo que se fija la emisividad en 1.

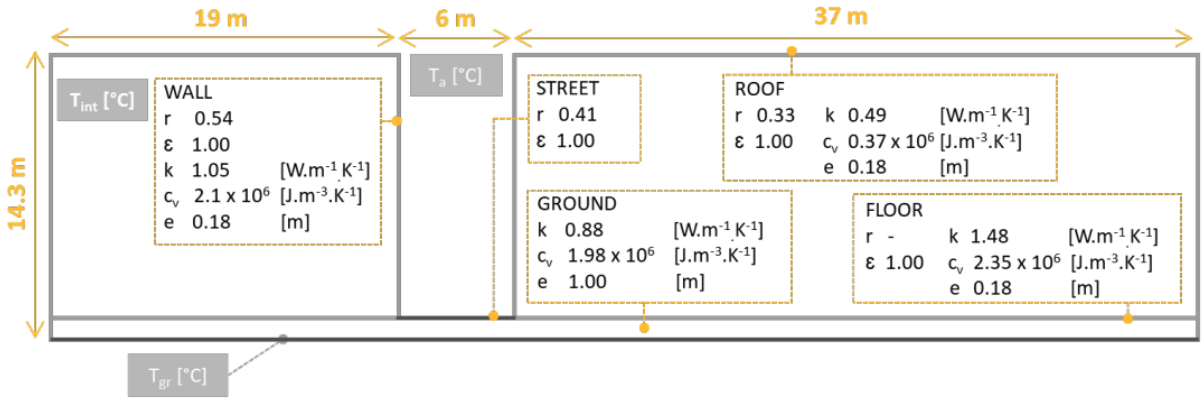


Figura 2.4: Tipos de superficies y propiedades de los materiales (Aguerre, 2020).

2.2.3. Condiciones de borde

La temperatura del aire exterior para el lapso de tiempo comprendido entre las 04:00 y las 23:00 se obtuvo de la campaña de medición, y varía entre 9,5°C y 19,8°C. La temperatura para el lapso restante comprendido entre las 23:00 y las 04:00 se obtuvo mediante interpolación lineal. También se obtuvo de la campaña de medición la humedad relativa, que varía entre 52 % y 68 %, y la velocidad del viento, que se promedió en 0,2 m/s, lo que permitió fijar el coeficiente de convección exterior en $10\,\mathrm{Wm^{-2}K^{-1}}$. Los coeficientes de convección interior a los edificios se fijaron en $0,7\,\mathrm{Wm^{-2}K^{-1}}$ para el piso, $2,5\,\mathrm{Wm^{-2}K^{-1}}$ para las paredes verticales y $5\,\mathrm{Wm^{-2}K^{-1}}$ para los techos. La temperatura interior de los edificios se fijó en 20°C. La temperatura del suelo a un metro de profundidad se fijó en 11,1°C.

La radiación de onda corta y de onda larga proveniente del cielo es anisotrópica, por lo que para calcularlas, se discretizó la bóveda celeste con una malla de 240 elementos. Su distribución se muestra en la Figura 2.5. También se modeló la distribución de la temperatura del cielo, como se muestra en la Figura 2.6.

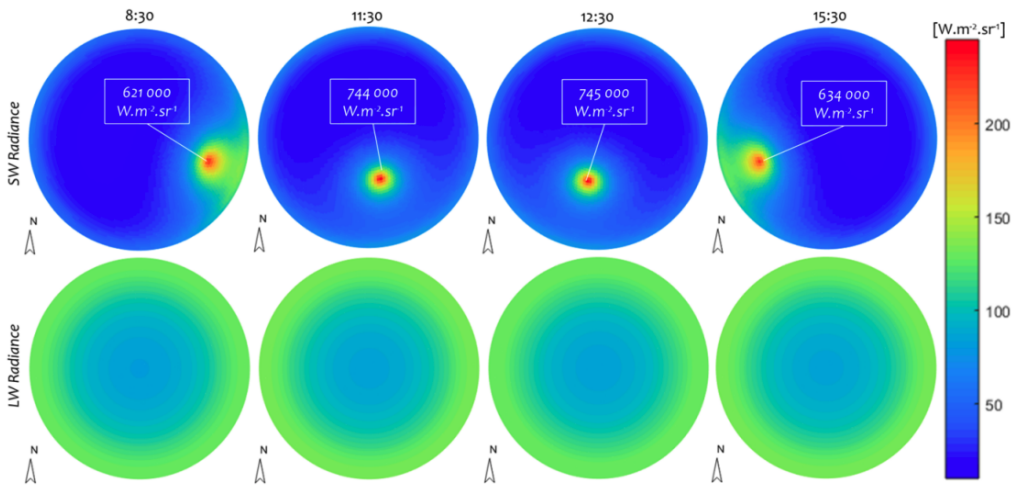


Figura 2.5: Modelado de la radiación de onda corta y onda larga para el 23 de abril de 2017 en Bayonne (Aguerre, 2020).

2.2.4. Características y construcción de la malla

La geometría se discretiza a través de una malla de elementos, la cual debe ser conforme con el objetivo de asegurar condiciones de simulación adecuadas. La malla contiene volúmenes y superficies, siendo las superficies las caras de los volúmenes. La conducción se simula sobre los volúmenes, mientras que la convección y la radiación se simulan sobre las superficies. Para los volúmenes se utilizaron hexaedros, mientras que para las superficies se utilizaron cuadriláteros.

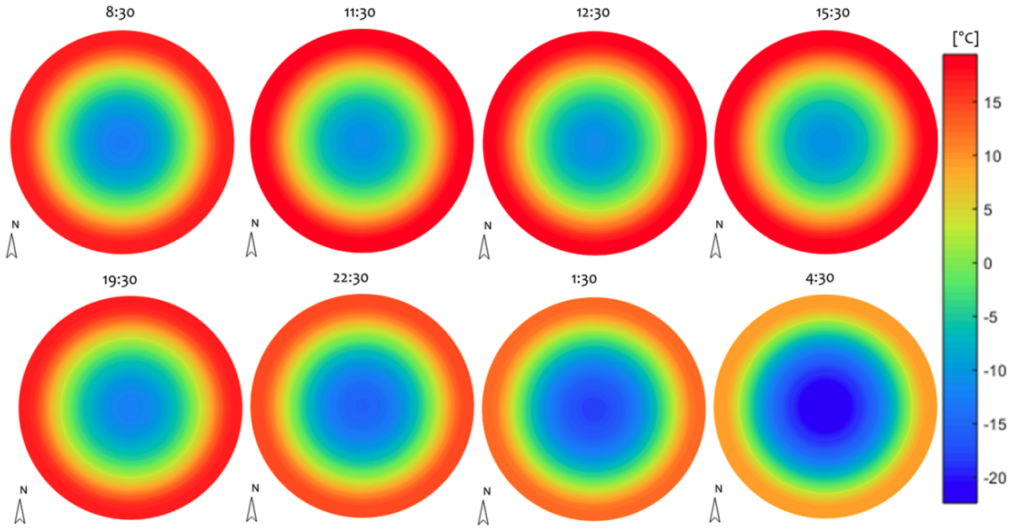
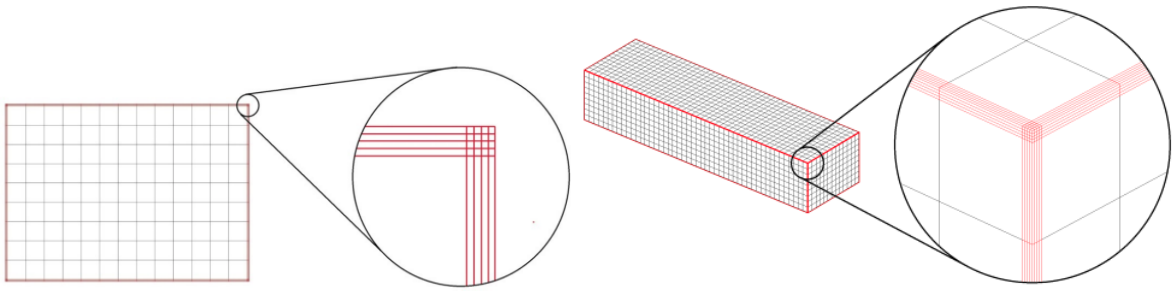


Figura 2.6: Modelado de la temperatura del cielo para el 23 de abril de 2017 en Bayonne (Aguerre, 2020).

En cuanto a las funciones de interpolación, se utilizaron elementos lineales. Sin embargo, debido que la evolución de la temperatura a través de diferentes componentes puede ser no lineal, las paredes se dividen en múltiples capas de elementos para simular correctamente las temperaturas de las superficies. Esta aproximación hace que la malla sea más compleja, pero aumenta la precisión de la simulación, razón por la cual se realizaron pruebas para determinar qué cantidad de capas utilizar para encontrar un balance entre rendimiento y precisión. Estas pruebas utilizaron de una a cuatro capas de hexaedros, obteniendo resultados convergentes a partir de las dos capas. No obstante, se procedió a utilizar cuatro para asegurar la validez de los resultados.

La construcción de la malla comienza construyendo primero las paredes de los edificios a partir de su cáscara exterior, realizando extrusiones hacia el interior, y luego uniéndolas y eliminando nodos y elementos repetidos. Se puede ver una cáscara exterior a partir de la cual se obtiene una pared en la Figura 2.7a y la unión de tres paredes en la Figura 2.7b. Luego de construida la malla de todos los edificios, se unen los bordes de las bases de las caras que están orientadas hacia el callejón urbano para construir la superficie de la calle. El volumen de la calle se construye por extrusión de su superficie. Finalmente, se añade el suelo que rodea a la ciudad, valiéndose del contorno de las bases de los bloques de edificios y de la calle. La malla del suelo se ajusta para que sea más fina cerca de los edificios y así obtener una mayor precisión en esa zona.



(a) Cáscara exterior de una pared.

(b) Volumen de un edificio.

Figura 2.7: Partes de la malla (Aguerre, 2020).

Para definir el tamaño de los elementos de la malla, se realizaron pruebas fijando distintas medidas para los cuadriláteros (lo cual aumenta o disminuye la cantidad de cuadriláteros que contiene la malla):

(a) $0,75m \times 0,75m$ - 78842 cuadriláteros.

(b) $1,5m \times 1,5m$ - 28395 cuadriláteros.

(c) $3,0m \times 3,0m$ - 12965 cuadriláteros.

(d) $5,0m \times 5,0m$ - 8017 cuadriláteros.

Las pruebas para las opciones (a) y (b) mostraron una diferencia de menos de un grado Celsius entre ellas, mientras que las opciones (c) y (d) presentaron errores mayores. Estos análisis se realizaron para tres puntos del callejón urbano: N (*north-facing point*), S (*south-facing point*) y G (*ground point*), los cuales se observan en la Figura 2.8. En base a este análisis, se optó por realizar el resto de la simulación con la opción (b), ya que los resultados son similares a la opción (a), pero la cantidad de elementos de la malla es significativamente menor. Cabe destacar que todos los cuadriláteros de la malla tienen un tamaño de $1,5m \times 1,5m$, excepto los cuadriláteros que se ubican más cercanos a los bordes, los cuales tienen un tamaño acorde al espesor de las paredes $(0,18m)$ y la cantidad de capas de cada pared (4). Las esquinas de los edificios también tienen un tamaño distinto, ya que sus cuadriláteros están determinados por el cruce entre las capas de cada pared. Estos cuadriláteros (sobre los que se hace un acercamiento en la Figura 2.7), tienen un tamaño de $0,045m \times 0,045m$.

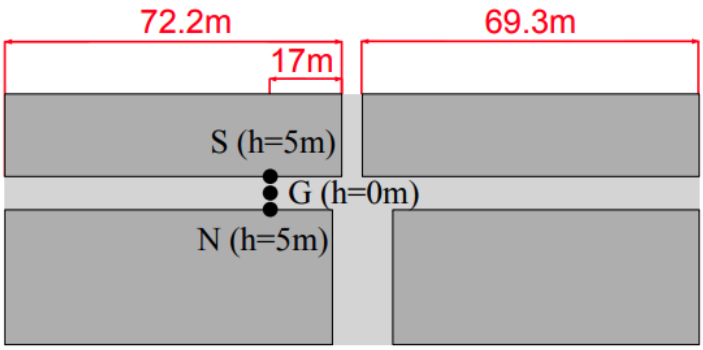


Figura 2.8: Ubicación de los tres puntos seleccionados para analizar las temperaturas para distintas medidas de la malla (Aguerre, 2020).

La malla resultante contiene 113580 volúmenes, donde su cáscara exterior está compuesta por 28395 cuadriláteros. Estas cantidades se distribuyen como sigue:

- Los cuatro edificios tienen 87904 volúmenes y 21976 cuadriláteros.
- La calle tiene 2828 volúmenes y 707 cuadriláteros.
- El suelo tiene 22848 volúmenes y 5712 cuadriláteros.

Como se menciona anteriormente, se modela un domo que representa el cielo, utilizado para simular el intercambio de radiación térmica entre la ciudad y el cielo. Este domo no tiene ningún volumen; es una cáscara compuesta por 240 cuadriláteros.

2.2.5. Pre-cómputo de radiación y condiciones iniciales

La computación de la radiación de onda corta (luz solar) es un paso fundamental en la simulación, ya que es el principal flujo externo de energía que entra en el modelo. La radiación recibida por una superficie de la ciudad se compone de la radiación emitida por el cielo y el sol, así como de la radiación reflejada por otras superficies. Ambos fenómenos son simulados a partir de un módulo implementado en la tesis original. Aprovechando que la geometría es estática, las radiaciones para cada elemento de la malla se calculan una única vez y se cargan en Cast3M como un flujo impuesto. El módulo implementado tiene en cuenta los rebotes de la radiación de onda corta, ya que los elementos que están a la sombra son considerablemente sensibles a estos rebotes. Para ello, se utiliza el método de Radiosidad, tal como se explica en (Aguerre y cols., 2017).

La computación de los intercambios de radiación de onda larga es llevada a cabo por Cast3M, que utiliza factores de forma (*form factors*) para los cálculos de radiación de onda larga sobre una misma malla. Los factores de forma se pre-calculan una única vez previo al cálculo de elementos finitos.

Las temperaturas iniciales del modelo corresponden a la solución de una simulación *Steady State*, cargando en el modelo sólamente la convección. Llevar a cabo esta simulación da como resultado las

temperaturas de todas las superficies que se utilizan como temperaturas iniciales para la simulación *Transient*.

2.2.6. Ejecución de la simulación

La solución de la simulación se obtiene para pasos de tiempo de 10 minutos. Simular 24 horas requiere de 144 pasos de tiempo.

En cuanto al rendimiento computacional de las simulaciones, se ejecutaron rápidamente, teniendo en cuenta que se utilizó una computadora personal estándar, resultando en un tiempo de ejecución total de 202,1 minutos, dentro de los que se incluyen las precomputaciones de la radiación de onda corta, precomputaciones de Cast3M, y el cálculo de la solución para cada uno de los 144 pasos de tiempo.

Sin embargo, una limitación notable fue el alto consumo de memoria RAM, que se situó en 14,2 GB. Esta limitación es importante, puesto que para geometrías con mayor nivel de detalle se tendría una mayor cantidad de elementos y por lo tanto mayor utilización de la memoria, ya que en particular la parte radiativa requiere $O(N^2)$ memoria, siendo N la cantidad de elementos. Dado que 16 GB es una capacidad estándar para computadoras personales, si aumentasen los requerimientos dejaría de ser posible la ejecución de las simulaciones en computadoras convencionales.

2.2.7. Análisis de los resultados

Los termogramas generados por la simulación se compararon con fotografías y termogramas reales obtenidos de la campaña de medición para distintos intervalos de tiempo. El modelo computacional produjo resultados convincentes, reproduciendo el mismo orden de magnitud y apariencia general de los datos experimentales. Se observaron diferencias menores, que se explican por las simplificaciones del modelo. Por ejemplo, para cierto momento del día, la temperatura simulada de la calle es mayor a la temperatura observada en los termogramas reales, lo que se debe a que los techos del modelo no tienen salientes, causando que la luz solar impacte directamente en la calle, elevando su temperatura. Otro ejemplo es la simplificación de los materiales modelados; experimentalmente, se observa que todos los elementos de madera alcanzan temperaturas más altas que en la simulación, donde este material no se modela. Por otro lado, los termogramas simulados revelan que la consolidación de edificios en bloques unitarios es una abstracción suficientemente precisa para lograr resultados comparables con los experimentos, dado que las diferencias en las temperaturas superficiales de edificios adyacentes resultaron ser poco significativas. Los termogramas reales también revelan la importancia desde el punto de vista térmico que tiene la presencia de una intersección en la calle, lo que justifica el modelado de cuatro bloques de edificios separados por dicha intersección, en lugar de simplificar el modelo utilizando solamente dos.

El análisis de los resultados se complementa con el estudio de la evolución de temperatura para tres puntos específicos de la escena, en particular, se seleccionaron dos puntos a diferentes alturas en la pared orientada al sur y uno en la pared orientada al norte. Estos puntos se observan en la Figura 2.9. Las temperaturas superficiales simuladas y observadas presentan una fuerte correlación para ambos puntos de la pared orientada al sur, con una diferencia menor a 2°C desde las 10:00 hasta las 23:00. La discrepancia es mayor en la pared orientada al norte, con una diferencia de casi 5°C durante la mayor parte del día. Esta discrepancia se explica por la suposición de que la temperatura interior de los edificios tiene un valor constante de 20°C (se presume que las temperaturas reales en el mes de abril son menores), causando que las superficies presenten temperaturas mayores debido a la conducción. También se atribuye esta discrepancia al hecho de que la radiación de onda corta sobre el punto orientado al norte es sobreestimada por causa de simplificaciones de la geometría como la mencionada ausencia de salientes en los techos o la ausencia de arcadas en los niveles cercanos a la calle.

A pesar de las discrepancias, el método de modelado presentado permite comprender los principales fenómenos de intercambio de calor en la calle al mismo tiempo que simplifica el caso de prueba y las condiciones de simulación. En resumen, permite que el caso de estudio sea instructivo y fácilmente replicable.

En el capítulo 8 de la tesis original se abordan muchas de las simplificaciones del presente modelo, implementando una geometría más detallada (Figura 2.10). Además, se elimina la suposición de que todos los elementos son cuerpos negros, y se incorporan más materiales para las superficies del modelo, entre otros cambios. No obstante, replicar este estudio térmico más preciso está fuera del alcance de este trabajo.

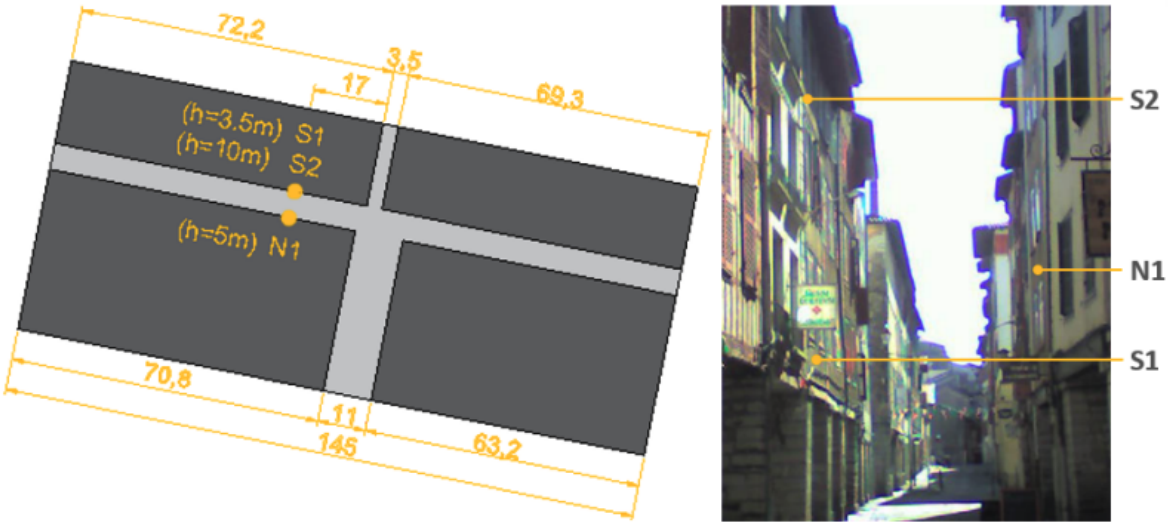
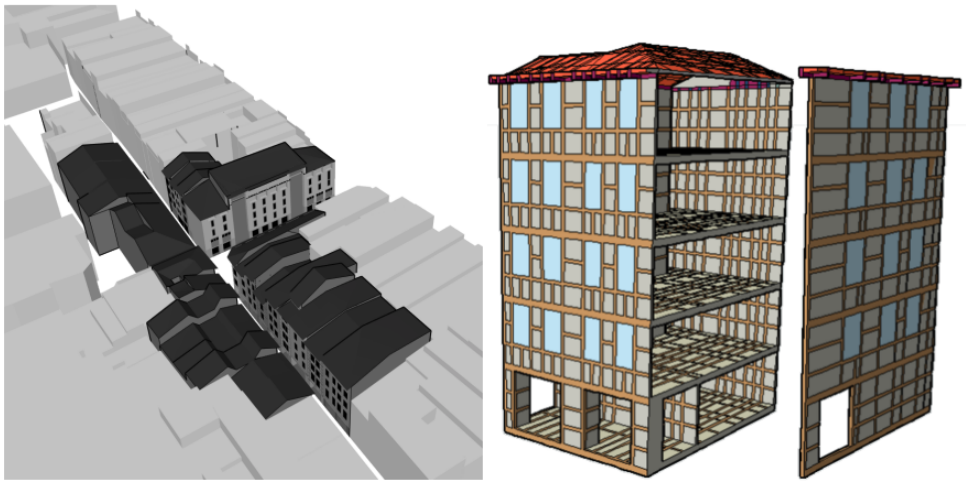


Figura 2.9: Ubicación de los tres puntos seleccionados para comparar los termogramas reales con los simulados (Aguerre, 2020).



(a) Geometría global. (b) Modelo detallado de un edificio.

Figura 2.10: Versión detallada del modelo de la calle Tonneliers.

2.3. Selección de software

Conforme se ha expuesto en las secciones y capítulos anteriores, el objetivo central de esta tesis es el de replicar los modelos y resultados obtenidos en la tesis de posgrado, utilizando un software de elementos finitos diferente a Cast3M. Si bien existen múltiples herramientas con el potencial de cumplir este cometido, el presente estudio tiene como alcance analizar la viabilidad de los siguientes tres sistemas, sugeridos por el supervisor de este trabajo:

- Ansys Mechanical, utilizado en una gran variedad de problemas de elementos finitos (Lee, 2023).
- COMSOL Multiphysics, utilizado en problemas térmicos de elementos finitos que incluyen conducción, convección y radiación (Mayboudi, 2020).
- OpenFOAM, el cual ha sido utilizado para estudiar, entre otros tópicos, reacciones termonucleares (Guo, Li, Huang, Liu, y Wang, 2021).

A continuación, se realiza un análisis desde distintas perspectivas con el objetivo de determinar cuál se ajusta mejor a los objetivos de esta investigación.

2.3.1. Licenciamiento.

Dado que este proyecto se enmarca en el contexto de una tesis estudiantil, se prioriza la búsqueda de opciones con licenciamiento de código abierto, licenciamiento estudiantil, o licencias de pago que estén sujetas a tarifas accesibles.

- Ansys Mechanical no califica como software de código abierto, pero sí ofrece la posibilidad de un acceso limitado (el límite de nodos/elementos de la malla es de 128000) gratuito a sus funcionalidades mediante Ansys Student[1].
- COMSOL Multiphysics no califica como un software de código abierto, ni ofrece una licenciamiento estudiantil. Al 27 de setiembre de 2023, la licencia necesaria para desarrollar este trabajo tiene un costo de U$S 2590 (COMSOL, 2014). Si bien la licencia es perpetua, el alto costo resulta un factor prohibitivo.
- OpenFOAM es un software de código abierto y uso gratuito, principalmente enfocado en la resolución de problemas de dinámicas de fluidos.

2.3.2. Funcionalidades.

Dadas las motivaciones y objetivos de este trabajo, es excluyente que la herramienta seleccionada utilice metodologías basadas en el MEF y que cuente con un módulo de análisis térmico.

- Ansys Mechanical es un software basado en el MEF, y ofrece la capacidad de realizar análisis térmicos que incluyen la conducción, radiación y convección[2]. En particular dispone del módulo *Thermal Transient* para ejecutar simulaciones térmicas *Transient*.
- COMSOL Multiphysics se basa en el MEF y cuenta con la extensión *Heat Transfer Module* que posibilita el análisis de transferencias de calor mediante conducción, convección y radiación.
- OpenFOAM no está basado en el MEF, sino que está basado en el Método de Volúmenes Finitos (MVF), conforme se corroboró mediante intercambio de correos electrónicos con representantes del ESI Group, propietarios de la marca OpenFOAM.

2.3.3. Interfaz gráfica de usuario.

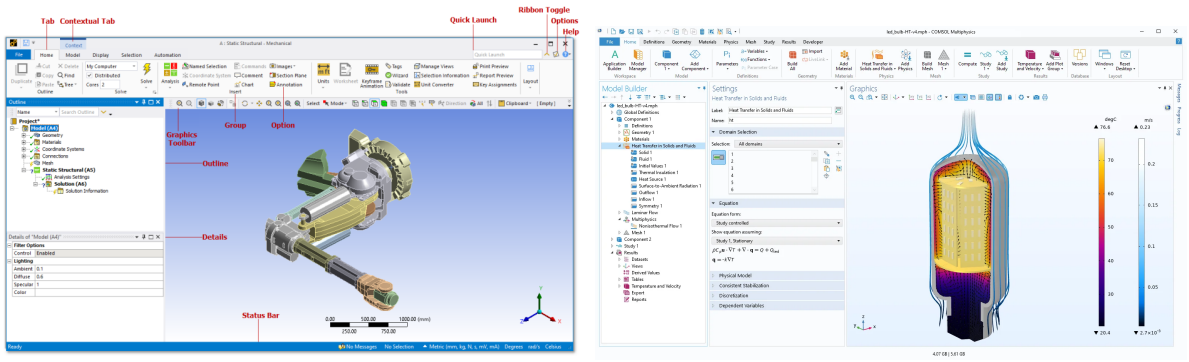
- Tanto Ansys como COMSOL proporcionan una interfaz gráfica que permite interactuar con la aplicación a través de elementos visuales. Ver la Figura 2.11.
- Para trabajar con OpenFOAM se utiliza la línea de comandos o scripts personalizados. La implementación de geometrías, mallas y la visualización de resultados se realiza a través de software de terceros, como Ennova CFD, software capaz de implementar mallas de gran tamaño (Pio y cols., s.f.), o ParaView, una herramienta de visualización y análisis de datos científicos multiplataforma y de código abierto que permite el análisis y la visualización de conjuntos de datos grandes. (Ayachit, 2019).

2.3.4. Compatibilidad con sistemas operativos.

Al evaluar la portabilidad de los sistemas de software, se prioriza la compatibilidad con Windows y MacOS debido a que son los sistemas operativos disponibles en las computadoras de los autores.

- Ansys Mechanical es compatible con Windows y sistemas basados en Linux (ANSYS, 2023). Ansys Student sólamente es compatible con Windows.
- COMSOL Multiphysics es compatible con Windows, Linux y MacOS (COMSOL, 2024).
- OpenFOAM está diseñado para utilizarse en Linux (OpenCFD, 2024), pero es posible utilizarlo en Windows a través de *Bash On Ubuntu On Windows*[3] o en MacOS a través de *Docker*, un software de código abierto para empaquetar, transportar y ejecutar cualquier aplicación como un contenedor ligero (Cuesta y González, 2016).

[1] Ansys Student https://www.ansys.com/academic/students/ansys-student
[2] Capacidades de Ansys Mechanical en: https://www.ansys.com/products/structures/ansys-mechanical#tab1-2
[3] Funcionalidad de Windows 10, que permite ejecutar una consola de Ubuntu nativamente en Windows



(a) Ansys Mechanical (ANSYS inc, 2023)

(b) COMSOL Multiphysics, módulo Heat Transfer (COMSOL, s.f.).

Figura 2.11: Interfaces de usuario de los productos de Ansys y COMSOL.

2.3.5. Documentación, tutoriales y comunidad

Es de interés verificar la disponibilidad de recursos educativos que permitan disminuir la curva de aprendizaje de cada software así como también validar la existencia de comunidades activas para cada herramienta.

- Ansys cuenta con una plataforma que ofrece cursos gratuitos[4], una comunidad activa que intercambia ideas y consejos en varios foros[5], y una plataforma con recursos de aprendizaje, que incluye casos de estudio, demostraciones y videos, entre otros[6].
- COMSOL proporciona una sección de videos con tutoriales y demostraciones[7], una plataforma de estudio con cursos gratuitos [8], foros de discusión[9] y libros relacionados con su software[10].
- OpenFOAM tiene cursos de pago[11], cuenta con una plataforma de documentación[12] que incluye información detallada sobre las funcionalidades, además de que contiene tutoriales y ejemplos. Por otro lado, debido a que es un software de código abierto, cuenta con una comunidad activa que participa en foros de discusión, como es el caso de CFD Online[13].

2.3.6. Conclusión

Luego de evaluar detenidamente las tres opciones de software consideradas se debe seleccionar una de ellas para proceder al desarrollo del proyecto.

OpenFOAM no cumple con el requisito de utilizar el MEF, por lo que se descarta esta opción. Analizando las opciones restantes, tanto COMSOL como Ansys cuentan con las funcionalidades necesarias para cumplir con los objetivos del trabajo, incluyendo metodologías basadas en MEF y análisis térmicos *Transient*. En términos de compatibilidad con sistemas operativos, ambos programas se pueden ejecutar sin problemas en Windows. En lo que respecta a la documentación, tutoriales y comunidades de usuarios, ambas opciones cumplen con los requisitos, ya que proporcionan recursos para respaldar el proceso de investigación y reducir la curva de aprendizaje de cada producto. El factor que diferencia ambas opciones es el costo. A pesar de que Ansys no es un software de código abierto, ofrece una alternativa gratis a través de su licencia para estudiantes. COMSOL, por el contrario, no dispone de una licencia estudiantil y sus costos son elevados. Por lo tanto, se tomó la decisión de utilizar Ansys para el desarrollo del proyecto.

[4] Ansys Innovation Courses https://courses.ansys.com/
[5] Foro de Ansys https://forum.ansys.com/
[6] Recursos educativos de Ansys https://www.ansys.com/academic/learning-resources
[7] Galería de COMSOL: https://www.comsol.com/videos
[8] COMSOL Learning Center: https://www.comsol.com/support/learning-center
[9] Foro de COMSOL: https://www.comsol.com/forum
[10] COMSOL-Based Books https://www.comsol.com/books
[11] Cursos de OpenFOAM https://www.openfoam.com/trainings/courses
[12] Documentación de OpenFOAM https://doc.openfoam.com/2306/
[13] Foro de OpenFOAM dentro de CFD Online https://www.cfd-online.com/Forums/openfoam/

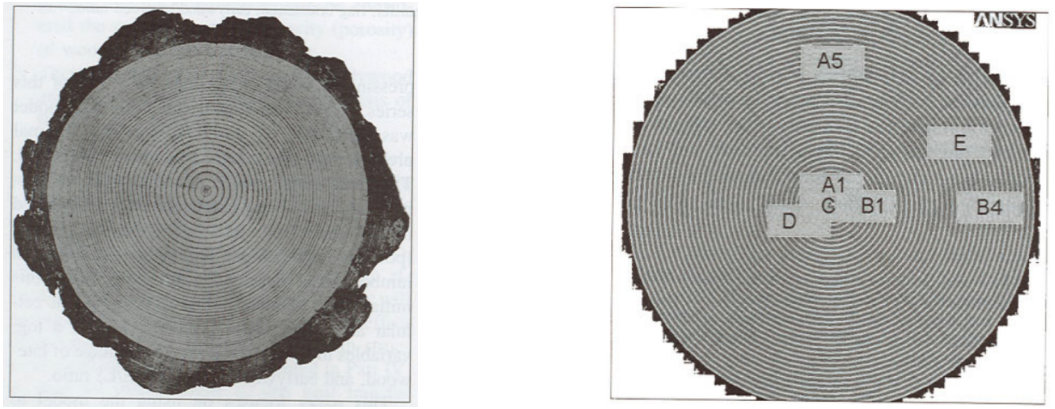
2.4. Publicaciones relacionadas

En esta sección se examina una selección de trabajos previos centrados en la aplicación del MEF para resolver problemas de transferencia de calor utilizando Ansys. Existe una larga lista de artículos que cumplen este criterio de búsqueda en la literatura, hecho que resalta la adopción de este software y respalda la decisión de emplearlo para esta investigación.

Se presentan ejemplos concretos que describen cómo Ansys ha sido utilizado en diversas aplicaciones relacionadas con la transferencia de calor. Estos ejemplos permiten ilustrar cómo Ansys permite modelar problemas del mundo real utilizando elementos finitos, y ejecutar simulaciones con resultados precisos.

2.4.1. Artículo 1 - "Two-dimensional finite element heat transfer model of softwood. Part II. Macrostructural effects" (Gu y Hunt, 2006)

Este artículo corresponde a la segunda parte de un trabajo que consiste en cuatro artículos dedicados al estudio de propiedades térmicas de la madera. El objetivo principal de esta parte es estudiar cómo determinadas propiedades de la madera afectan la transferencia de calor en dos dimensiones en dicho material. Para ello, se emplea un modelo de elementos finitos construido en la parte 1 (Hunt y Gu, 2006) utilizando Ansys, que permite generar troncos de madera de distintos tamaños, configurar propiedades como la densidad de anillos o la proporción de madera temprana/tardía y "cortar" cualquier sección de madera para analizarlas por separado (ver Figura 2.12). Dicho modelo también toma en cuenta factores como los cambios en la conductividad y la capacidad calorífica que ocurren al variar la temperatura y la convección presente en operaciones de secado de la madera.



(a) Anillos de crecimiento de un tronco de pino ponderosa. Las bandas claras son madera temprana y las bandas oscuras son madera tardía.

(b) Sección transversal de un tronco generado con Ansys. Los recuadros con letras son secciones "cortadas" para analizar transferencias de calor.

Figura 2.12: Tronco real y modelado de un tronco con Ansys.

Se llevaron a cabo una serie de simulaciones *Transient* para distintos cortes y ubicaciones en el tronco, con variaciones en las características de los anillos. Los resultados de estas simulaciones revelaron que dichas características tienen efectos significativos en las transferencias de calor en la madera.

Este artículo, si bien data del año 2006, muestra la efectividad de Ansys como herramienta para la investigación de fenómenos térmicos en materiales específicos a través del método de elementos finitos.

2.4.2. Artículo 2 - "A three-dimensional finite element analysis of temperature distribution in hot mix asphalt pothole repair"(Byzyka y cols., 2017)

En este trabajo, se lleva a cabo en Ansys un estudio de las transferencias de calor que ocurren en la reparación de baches en carreteras asfaltadas. El objetivo principal es el desarrollo de un modelo de elementos finitos que permita predecir de manera confiable la temperatura resultante de combinar material de relleno con el asfalto existente en una carretera en el contexto de una reparación en caliente, ya que dicha temperatura influye en la durabilidad de la reparación.

Se construyen geometrías y mallados con distintas medidas y cantidades de elementos para modelar porciones de carretera con zonas a reparar y la mezcla de pavimento que se utiliza en cada reparación

(Figura 2.13). Se establecen las propiedades de los materiales y se modelan condiciones de borde, dentro de las cuales se incluye la convección del aire. Las temperaturas iniciales del modelo se determinan a partir de simulaciones *Steady State*.

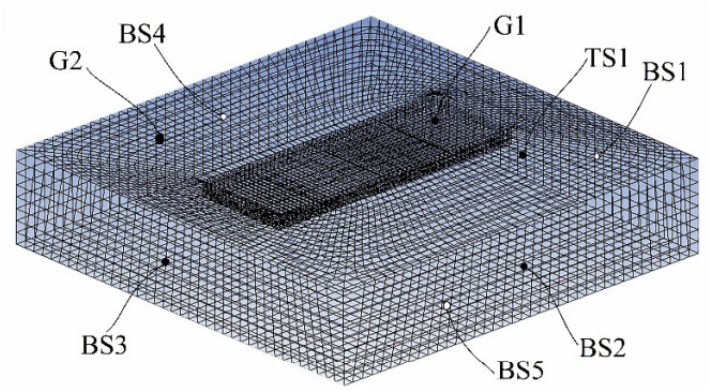


Figura 2.13: Ejemplo de mallado de pavimento asfaltado (Byzyka y cols., 2017).

Para obtener resultados de los intercambios de calor durante las reparaciones, se ejecutan múltiples simulaciones *Transient* con diversas configuraciones de profundidad del bache y temperatura del aire, de la mezcla de reparación y del asfalto preexistente.

Tras completar las simulaciones y analizar los resultados de cada configuración, el modelo de elementos finitos construido permitió extraer conclusiones sobre qué factores son los que más afectan la durabilidad de las reparaciones, cumpliendo con el objetivo inicial del proyecto.

2.4.3. Artículo 3 - "Thermal analysis on cylinder head of SI engine using FEM." (Tripathi y cols., 2014)

El estudio se enfoca en el análisis de la distribución de temperatura en la culata de un motor de combustión interna de un cilindro. Esta variable desempeña un papel crítico en la eficiencia y seguridad operativa del mismo. La realización de un análisis experimental para obtener dichos datos conlleva costos significativos, por lo que la utilización de técnicas de análisis de elementos finitos se presenta como un complemento idóneo.

El objetivo principal de la investigación consiste en medir la temperatura transitoria en diversos puntos clave del motor. Para llevar a cabo este propósito, se emplea un análisis de elementos finitos, y la herramienta seleccionada para ejecutar dicho análisis es Ansys.

Para construir el modelo, se comienza con la creación de una geometría para la culata del motor, seguido por la generación de la malla de elementos, como se muestra en la Figura 2.14.

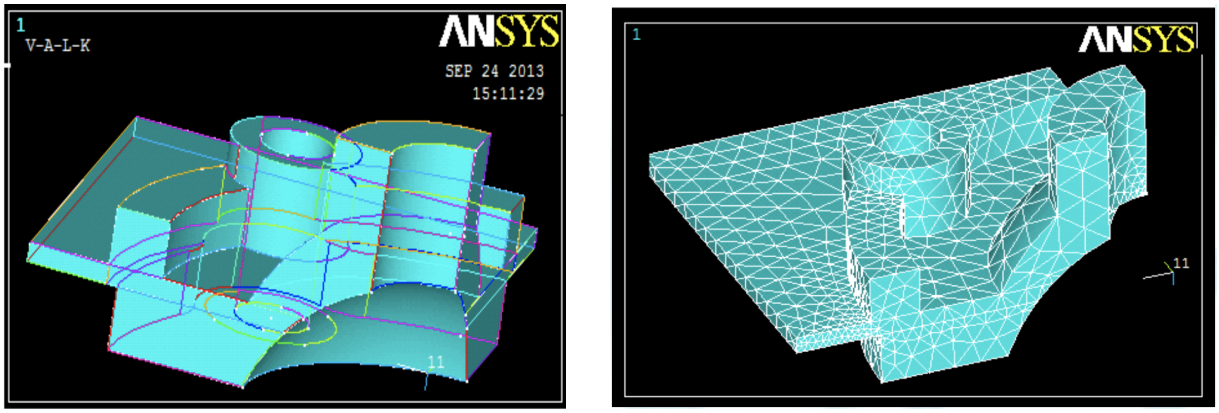


Figura 2.14: Geometría y malla de puntos.

Posteriormente, se definen las condiciones de contorno, las cuales abarcan propiedades térmicas como la conductividad térmica, el calor específico y la densidad, así como propiedades de convección, tales

como el coeficiente de convección y la temperatura ambiente para las superficies internas y externas del motor.

Una vez establecidas todas las condiciones necesarias, se ejecuta el modelo y se obtienen los resultados, los cuales se presentan en la Figura 2.15. Se valida la precisión de estos resultados, confirmando la eficacia del método de elementos finitos como una herramienta adecuada para obtener la distribución de temperatura en el motor.

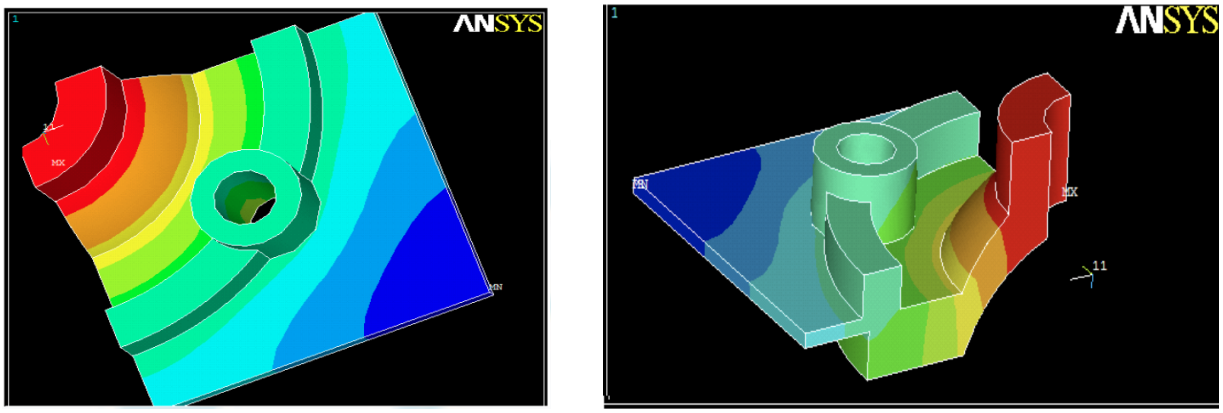


Figura 2.15: Resultados.

Capítulo 3

Implementación del caso de estudio en Ansys

El presente capítulo se dedica a la implementación del proyecto, proporcionando un enfoque detallado de todo el proceso de construcción del modelo de simulación.

El capítulo comienza con una sección que describe el proceso de instalación de Ansys, los desafíos encontrados y cómo fueron sorteados. A continuación, se presenta una sección que detalla la construcción de la geometría. Luego, se detalla la definición de los materiales utilizados en el modelo. Posteriormente, se aborda la generación de la malla de elementos y finalmente, se describe la configuración del módulo térmico.

3.1. Instalación y licenciamiento de Ansys

Las etapas iniciales del proyecto se realizaron utilizando Ansys Student, cuya instalación es sencilla siguiendo las instrucciones del instalador que se encuentra en la web de Ansys (ver Sección 2.3). Cabe destacar que este software incluye una versión limitada de Ansys Mechanical y presenta una restricción sobre la cantidad de nodos/elementos que puede contener una malla de elementos (un máximo de 128K nodos y elementos) para poder ejecutar simulaciones de cualquier tipo. Al momento de encontrarse con esta limitación, se contaba con una malla construida con elementos de 1,5m, la cual sobrepasaba este límite, por lo que para poder ejecutar simulaciones se tuvo que aumentar provisoriamente el tamaño de los elementos a 1,9m, mientras se buscaban soluciones más permanentes.

Con el propósito de superar estas limitaciones, se iniciaron gestiones para obtener una licencia de pago de Ansys que no presentase restricciones sobre la cantidad de elementos. Durante este proceso, también se exploraron las opciones disponibles dentro de la Facultad de Ingeniería (UdelaR), donde se contaba con las dos licencias siguientes: *Ansys Academic Research Mechanical* y *Ansys Academic Teaching Mechanical and CFD*. Ambas licencias cumplen con lo requerido (ANSYS inc, 2021), por lo que se intentó utilizarlas, para continuar con la implementación.

En particular, se intentó acceder a la licencia *Research Mechanical*, instalada en un servidor perteneciente a la red de la Facultad de Ingeniería, no accesible desde el exterior de la misma. El sistema de licenciamiento de Ansys opera mediante un programa denominado *Ansys License Manager*, el cual se ejecuta en una computadora cliente y se conecta a un servidor que almacena la información de las licencias. Dicho servidor debe estar ubicado en una red local o ser accesible desde internet. En una primera instancia, se intentó acceder remotamente por medio de un túnel SSH, pero este intento no tuvo éxito. Ante esta situación, se profundizó en la comprensión del funcionamiento del sistema de licenciamiento de Ansys. Como resultado de este análisis, se determinó que el proceso de licenciamiento utiliza tres puertos TCP determinados. A partir de esta información, se solicitó la apertura de estos puertos a la Unidad de Recursos Informáticos (URI) de la Facultad. Al volver a intentar el acceso mediante el túnel SSH, se logró una conexión parcial entre *Ansys License Manager* y el servidor; si bien existía una conexión, no era posible completar el proceso de licenciamiento de manera exitosa. Se realizó una investigación adicional en la que se identificó la implicación de un puerto dinámico en el proceso de licenciamiento, lo que dificultó el uso de la licencia institucional de forma remota.

Mientras se realizaban estos intentos, se continuaba el desarrollo de la tesis con la malla de 1,9m y se avanzaba en las gestiones para conseguir una licencia paga de Ansys. Como parte de estas gestiones,

se entablaron conversaciones con Engineering Simulation and Scientific Software (ESSS)[1], organización brasileña que es representante oficial de Ansys en Latinoamérica. Como resultado de estas conversaciones surge un acuerdo en el que ESSS ofrece su apoyo financiero al proyecto, permitiendo obtener una licencia de pago de Ansys por un período de siete meses de forma bonificada. Este apoyo surge del compromiso que ESSS tiene con el desarrollo del ecosistema latinoamericano y, en particular, con el avance de la investigación en Uruguay. Gracias a esta colaboración, fue posible acceder a una versión completa y sin limitaciones de Ansys Mechanical, lo que permitió seguir adelante con el proyecto.

La instalación de la licencia se realizó en una computadora Windows con las siguientes características:

- Procesador: Intel Core i7 7700k con 4 núcleos y 8 hilos.
- Memoria RAM: 16 GB DDR4.
- Tarjeta gráfica: NVIDIA GTX 1070 con 8 GB de memoria.

3.2. Ansys Workbench

Los programas incluidos en la licencia pueden ser accedidos de forma autónoma o a través de Ansys Workbench, una plataforma de integración que permite gestionar los datos de diferentes módulos o productos de Ansys e integrar múltiples análisis dentro de una sola interfaz (ANSYS inc, 2009).

El primer paso en la implementación del modelo es crear un sistema *Transient Thermal*. Un sistema está compuesto por varios elementos, entre ellos un modelo (en este caso un modelo térmico), una geometría, datos sobre los materiales utilizados por el modelo, los resultados obtenidos, entre otros. Para crear un sistema, se arrastra la opción *Thermal Transient* desde la barra de sistemas de análisis de Workbench (ubicada a la izquierda en la Figura 3.1) hacia el esquema del proyecto. El esquema final del proyecto se puede observar también en la Figura 3.1.

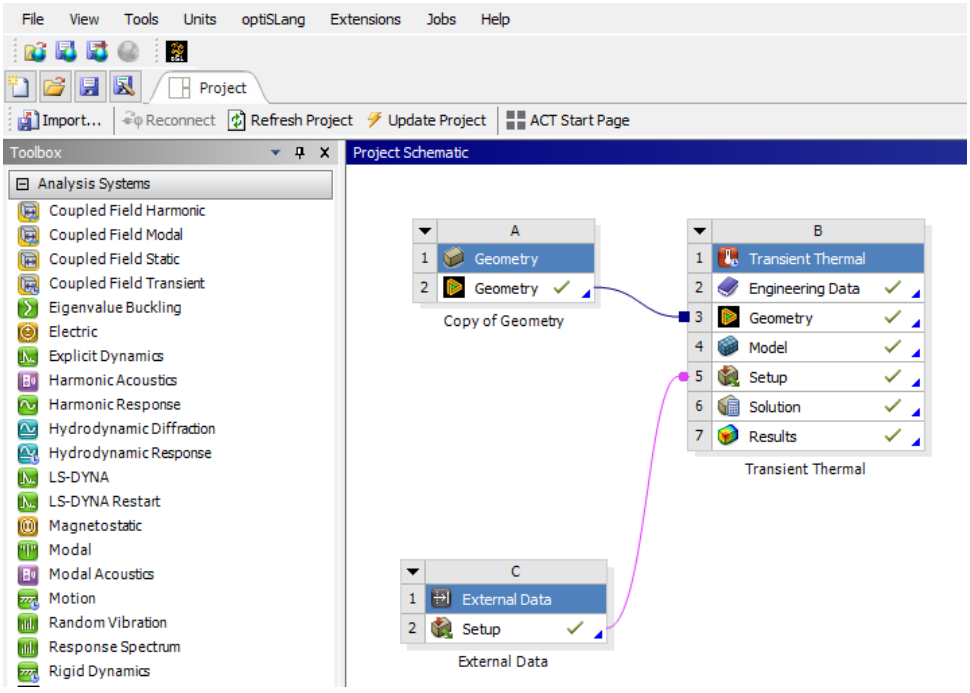


Figura 3.1: Esquema del proyecto.

3.3. Geometría

La geometría en elementos finitos es una representación tridimensional de la escena analizada en el caso de estudio. Esta representación incluye la definición de las dimensiones y forma de los objetos en cuestión y es fundamental para crear una malla de elementos.

[1] Web de ESSS: https://www.esss.co/es/

En el entorno de Ansys Mechanical, se encuentra la aplicación de modelado de geometrías *DesignModeler*, la cual provee una interfaz gráfica intuitiva y ofrece herramientas de modelado. Para acceder a esta aplicación, basta con realizar doble click en la opción *Geometry* dentro del esquema del proyecto.

DesignModeler utiliza los conceptos de *Plane* y *Sketch* (plano y boceto, respectivamente). Los planos son superficies bidimensionales, que se definen en función de los ejes "x", "y", y "z". Por otro lado, los bocetos son dibujos, también bidimensionales, que se crean sobre los planos. Los bocetos sirven como la base sobre la que se crean geometrías más complejas, aplicando distintas funciones como *extrude* o *revolve* para generar figuras tridimensionales.

Como se mencionó en secciones anteriores, el callejón urbano de *Rue des Tonneliers* de *Bayonne* se modela como cuatro cajas que representan bloques de edificios, un cilindro alrededor de estas cajas que representa los alrededores de la ciudad y un domo que representa el cielo. Las medidas de la geometría a construir se observan en la Figura 3.2.

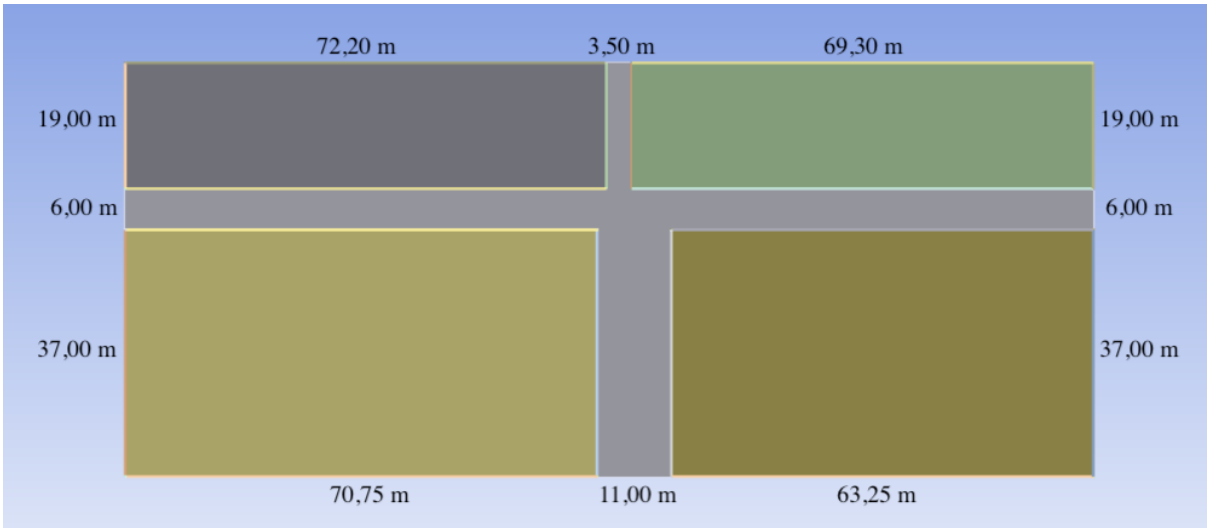
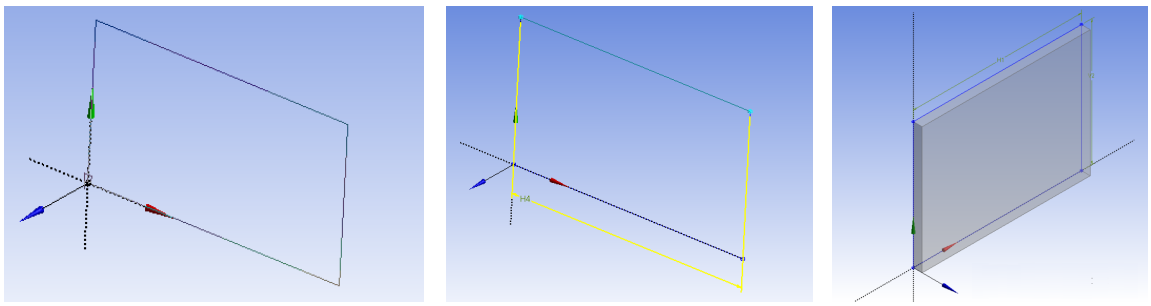


Figura 3.2: Medidas de los edificios y la calle. La altura de todos los edificios es de 14,3 metros.

3.3.1. Modelado de los edificios y la calle

El modelado comienza con una geometría vacía con un punto de origen (0, 0, 0) y tres planos XY, ZX, YZ. Para construir el primer edificio se selecciona uno de estos planos, se crea un boceto sobre su superficie y se utiliza la herramienta *Rectangle* para dibujar un rectángulo. Es posible ajustar las dimensiones del boceto antes de usar la opción *Generate*[2] para generar la superficie en la geometría. Se observa este proceso en las figuras 3.3a y 3.3b. Una vez completado el boceto, se utiliza la función *Extrude*. Esta operación transforma el boceto bidimensional en una figura tridimensional, formando así una pared (Figura 3.3c).

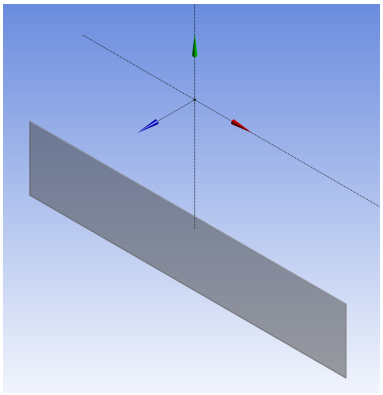


(a) Construcción de un boceto. (b) Definición de dimensiones. (c) Extrusión de un boceto.

Figura 3.3: Construcción de una pared.

[2]La geometría de DesignModeler no se crea automáticamente mientras se dibujan bocetos o se hacen extrusiones, sino que se crea o actualiza por medio de la opción *Generate*.

Este proceso se repite para cada una de las paredes del edificio, para lo cual es necesario crear nuevos planos auxiliares a partir de rotaciones y traslaciones de los planos existentes (ver Figura 3.4). De esta manera, cada plano se utiliza para crear nuevas paredes hasta completar la estructura del edificio. En la Figura 3.5 se puede ver el proceso completo. Los vértices repetidos en una misma ubicación se fusionan automáticamente.



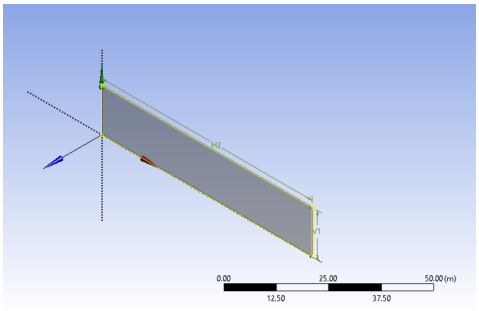
(a) Ubicación del nuevo plano.

Details View

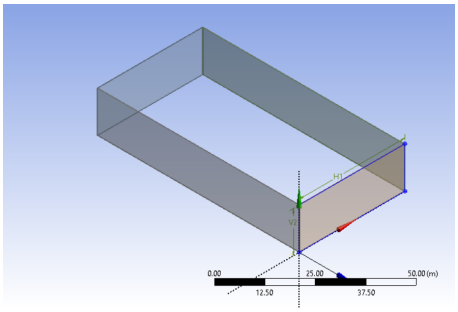
Details of Plane10	
Plane	Plane10
Sketches	1
Type	From Plane
Base Plane	XYPlane
Transform 1 (RMB)	Offset Z
FD1, Value 1	-37 m
Transform 2 (RMB)	None
Reverse Normal/Z-Axis?	No
Flip XY-Axes?	No
Export Coordinate System?	No

(b) Traslación en el eje Z del nuevo plano.

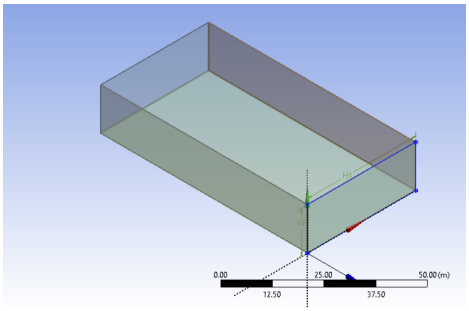
Figura 3.4: Plano creado a partir del plano XZ.



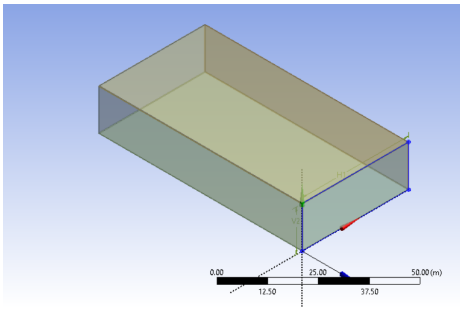
(a) Primera pared.



(b) Cuatro paredes.



(c) Cuatro paredes y piso.



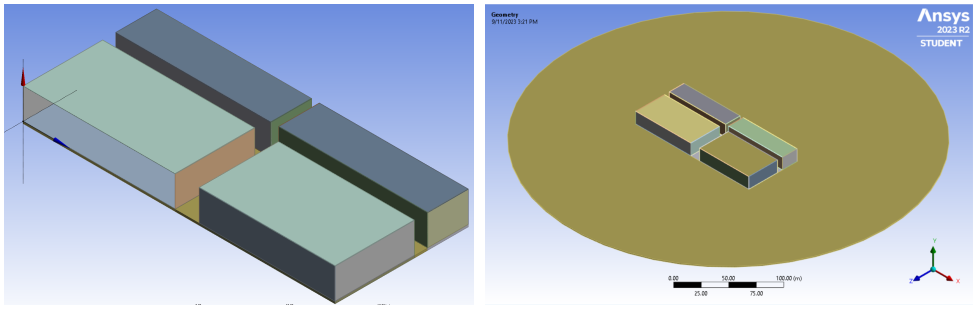
(d) Cuatro paredes, piso y techo.

Figura 3.5: Construcción de un edificio.

La calle se construye de manera similar; se dibujan varios bocetos rectangulares que luego se extruyen un metro hacia abajo de la ciudad. Se muestra la geometría con la calle construida en la Figura 3.6a.

3.3.2. Modelado del suelo y la bóveda celeste

El suelo se modela como un cilindro con un radio de 200 metros y un espesor de 1 metro. La elección de un radio de tal magnitud (significativamente más grande que la ciudad) se debe a que la superficie del suelo es tenida en cuenta para simular las reflexiones de radiación de onda corta que inciden en los edificios de la ciudad desde los alrededores de la misma. Construir el suelo sigue un proceso similar a las paredes; se crea un boceto circular de radio 200m con centro en la esquina de uno de los edificios, y luego se aplica una extrusión de un metro hacia abajo. El volumen de este cilindro se combina automáticamente



(a) Geometría de los edificios y la calle. (b) Geometría del suelo.

Figura 3.6: Construcción de los edificios, calle y suelo.

con el volumen existente del suelo; no es necesario excluir este último al momento de dibujar el boceto circular (ver Figura 3.6b).

Para modelar el cielo se construye una hemi-esfera de 424 metros de radio. Utiliza el mismo centro que el suelo y se construye a partir de un boceto de un cuarto de círculo sobre el que se aplica la función *Revolve* para formar el domo (ver Figura 3.7).

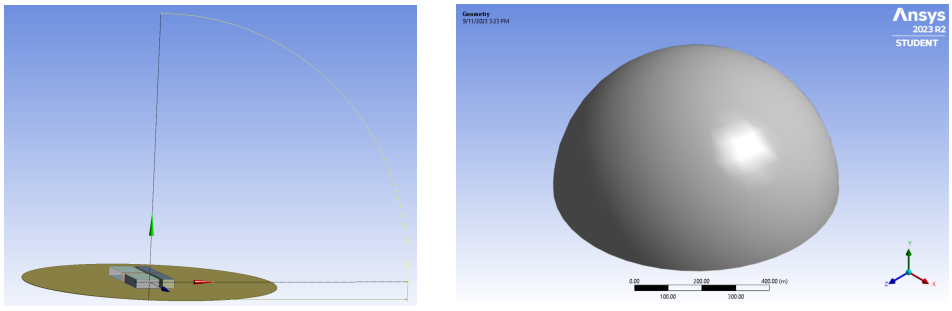
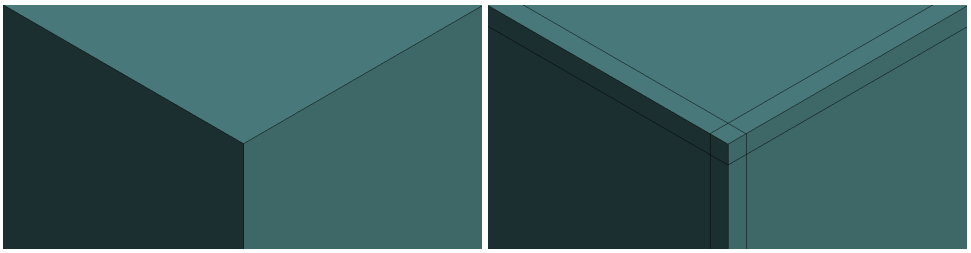


Figura 3.7: Construcción del domo.

3.4. Generación de la geometría

Cada vez que se aplica una extrusión, se crea un objeto *Extrude* en el árbol de elementos de la geometría. Dichos objetos luego se agregan en la geometría mediante el botón *Generate*. Al momento de generar, los distintos objetos extrude que coinciden en el espacio se pueden fusionar de varias maneras, dependiendo del valor elegido en la opción *Operation*. Los valores de *Operation* incluyen las opciones *Add material* (por defecto), *Add frozen*, entre otras. Utilizando *Add Material*, los volúmenes se fusionan formando uno sólo, mientras que con *Add frozen*, las intersecciones de volúmenes generan nuevos volúmenes. Utilizar la primera opción no resulta conveniente, puesto que a la hora de construir la malla de elementos es necesario diferenciar, por ejemplo, entre las paredes y su unión con los techos. La Figura 3.8 ilustra las diferencias entre estas dos opciones. Por lo tanto, todos los objetos *Extrude* se configuran con la opción *Add Frozen*.



(a) Esquina de un edificio generado con *Add material*. (b) Esquina de un edificio generado con *Add frozen*.

Figura 3.8: Comparación de geometrías generadas con *Add material* y *Add frozen*.

Luego de generar la geometría, se obtienen 31 *Bodies* o cuerpos, de los cuales 30 de ellos corresponden a sólidos y uno corresponde a la superficie del domo. Cada cuerpo constituye lo que se denomina una "parte" de la geometría. Es necesario combinar estas partes en una sola, para que la posterior construcción de la malla resulte en una malla conforme, de lo contrario, se generan mallas independientes[3]. Se observa parte del árbol de elementos de la geometría en la Figura 3.9, donde los sólidos se encuentran unidos en una sola parte.

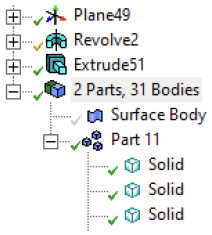


Figura 3.9: Árbol de elementos de la geometría.

3.5. Materiales

La elección y configuración de los materiales y sus propiedades es un paso fundamental para obtener resultados de simulación precisos, ya que influyen en cómo el calor se propaga en un sistema que depende del tiempo.

Mechanical provee varios conjuntos de materiales predefinidos como *Geomechanical Materials*[4], *Thermal Materials*[5] o *General Materials*[6]. Dentro de este último conjunto se encuentran materiales como el acero inoxidable, acero estructural o el concreto estructural. Se eligió el concreto estructural como el material de base para el modelo, y se modificaron algunas de sus propiedades para que coincidieran con la tesis original. En particular, se definieron cuatro tipos de concreto a partir del concreto estructural por defecto, uno por cada tipo de superficie del modelo: techo, paredes, pisos y suelo. Las propiedades modificadas se muestran en la Tabla 3.1. Los valores asignados a cada una se obtienen de diferentes formas:

- Densidad o masa específica (ρ): se utiliza el valor por defecto del concreto estructural definido por Ansys.
- Conductividad térmica isotrópica (k): se utilizan los valores definidos en la tesis original.
- Calor específico a presión constante (c_p): para obtener este valor se considera la relación entre esta propiedad y la capacidad calorífica volumétrica (c_v). La capacidad calorífica volumétrica se expresa como el producto de la capacidad calorífica específica y la densidad de un material: $c_v = c_p \cdot \rho$. Se asume que la capacidad calorífica específica es igual al calor específico a presión constante porque se está trabajando con sólidos. Entonces, $c_v = c_p \cdot \rho \Rightarrow c_p = \frac{c_v}{\rho}$ donde c_v se obtiene de la tesis original (Figura 2.4).

3.6. Generación de la malla de elementos

La construcción de la malla de elementos se lleva a cabo en el módulo *Meshing* (ANSYS inc, 2010). El acceso a este módulo está incluido dentro del modelo del sistema y es activado cada vez que se hace click en *Mesh* en el árbol de elementos del modelo. Se puede observar la interfaz de *Meshing* en la Figura 3.10.

[3] https://ansyshelp.ansys.com/account/secured?returnurl=/Views/Secured/corp/v231/en/wb_msh/msh_types_asm.html

[4] Este paquete contiene muestras de materiales de uso general para modelos geomecánicos, por ejemplo: dolomita, caliza, arenisca.

[5] Este paquete contiene muestras de materiales para modelos térmicos, por ejemplo: bronce, cobre, cobalto, oro.

[6] Paquete de materiales de uso general para análisis variados.

Material	Densidad (kgm^{-3})	Conductividad térmica ($Wm^{-1}K^{-1}$)	Calor específico a presión constante ($Jkg^{-1}K^{-1}$)
Concreto techos	394	0,49	937
Concreto paredes	1400	1,05	1500
Concreto pisos	2267	0,88	1035
Concreto suelos	1981	1,48	1000

Cuadro 3.1: Materiales y sus propiedades

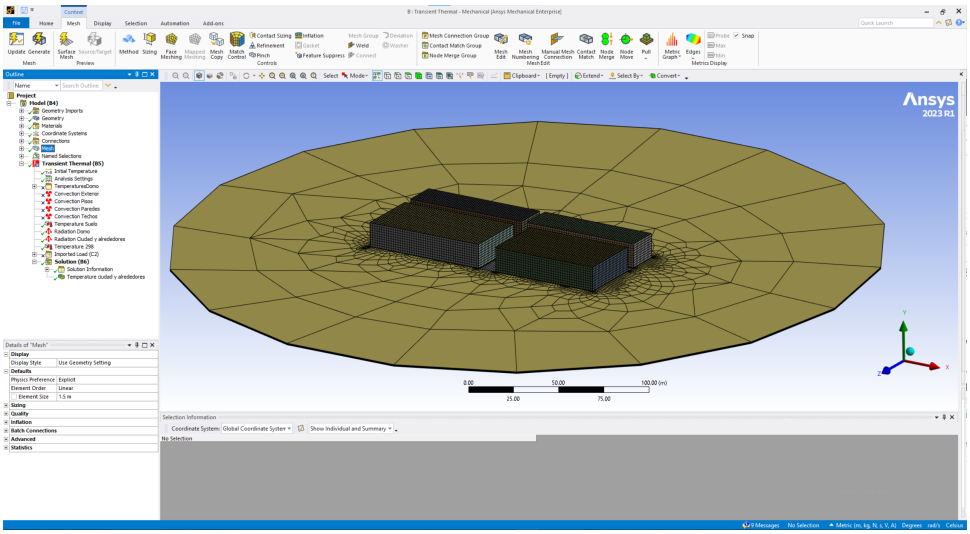


Figura 3.10: Interfaz de Ansys Meshing.

3.6.1. Requisitos

Conformalidad y estructura

La malla debe ser conforme. Cabe recordar que una malla conforme es aquella donde no existen intersecciones ni superposiciones, y donde ningún nodo de un elemento se encuentra en el borde o cara de otro elemento, por lo tanto, los elementos comparten bordes.

Una malla estructurada es una malla tal que todos sus nodos internos tienen el mismo número de elementos a su alrededor, al contrario que una malla no estructurada, la cual es irregular y no necesariamente satisface la restricción anterior (Figura 3.11). En geometrías rectangulares, la tendencia es que la malla sea estructurada, mientras que en geometrías con curvas y asimetrías, no es posible obtener soluciones de este tipo, por lo que se recurre a mallas no estructuradas. En este contexto, se utiliza mallas estructuradas para modelar los edificios, y mallas no estructuradas para la calle y el suelo.

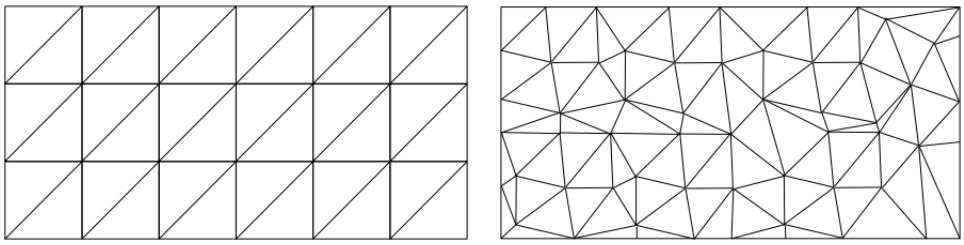


Figura 3.11: Malla 2D de triángulos estructurada y no estructurada.

Elementos

Los volúmenes deben ser en su mayoría hexaedros, y las superficies deben ser cuadriláteros. Nuevamente, esta restricción se flexibiliza al momento de generar la malla del suelo, donde es aceptable tener prismas de base triangular y triángulos.

Tamaño

Las superficies de los elementos deben ser de $1,5m \times 1,5m$, excepto para el suelo circundante, donde interesa reducir la cantidad de elementos de la malla utilizando elementos más grandes cuanto mayor sea la distancia a los edificios. En el caso del domo, éste debe tener una cantidad de cuadriláteros en el entorno de los 240 (cantidad utilizada en la tesis original), por lo que el tamaño de los mismos se deberá ajustar de manera acorde para satisfacer este requisito.

3.6.2. Configuraciones generales

Meshing ofrece una variedad de opciones de configuración globales a la malla, de las cuales interesa modificarlos siguientes valores:

- Orden de los elementos: Lineal. Seleccionar esta opción previene la generación de nodos adicionales en las aristas.
- Tamaño de los elementos: 1,5m. Este parámetro determina el tamaño de todas las aristas, superficies y volúmenes de la malla.

3.6.3. Métodos

Es posible generar una malla de forma completamente automática en la cual *Meshing* determina la estructura final de la malla. Sin embargo, como se describió anteriormente, existe una serie de requisitos que la malla debe cumplir, por lo que es deseable personalizar su generación. Con este fin, se utiliza el concepto de *Methods* (métodos). Los métodos permiten personalizar formas, tamaños y la estructura general de la malla. Se aplican sobre determinadas superficies o volúmenes, que se denominan el *Scope* o alcance del método.

Método *MultiZone*

El método *MultiZone* proporciona una descomposición automática de la geometría en regiones mapeadas (aquellas sobre las que se puede realizar un barrido) y regiones libres. Cuando se selecciona el método de malla *MultiZone*, todas las regiones se mallan con hexaedros si es posible. En los casos que no sea posible, se puede ajustar el método de manera que genere una malla barrida en las regiones estructuradas y una malla libre en las regiones no estructuradas. (Elmekawy, 2018)

Este método se utiliza principalmente para la construcción de la malla de los edificios. Para utilizarlo, se crea un objeto *Method* bajo el subárbol *Mesh* del modelo, y se configura con las siguientes opciones (ver Figura 3.12):

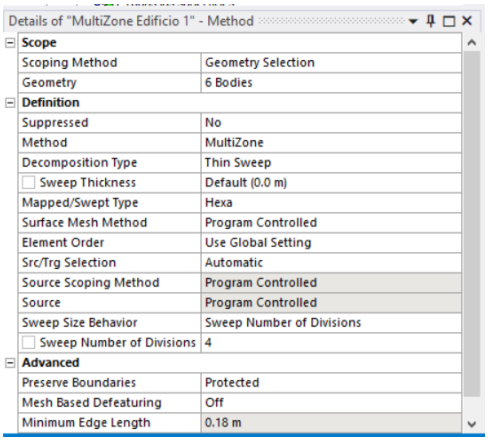


Figura 3.12: Configuración del método *MultiZone* de un edificio.

- **Scope:** Se incluyen en el alcance del método las paredes, el techo y el piso de los edificios. Se excluyen los bordes y las esquinas, como se ilustra en la Figura 3.13.
- **Decomposition Type:** El tipo de descomposición utilizado es *Thin Sweep* o barrido fino. Esta opción permite dividir la malla en capas.
- **Sweep Size Behavior:** El comportamiento utilizado es *Sweep Number of Divisions*, el cual permite que la división en capas obedezca una cantidad específica de capas, ajustando el tamaño automáticamente según sea necesario para que todas las capas tengan el mismo espesor.
- **Sweep Number of Divisions:** 4.

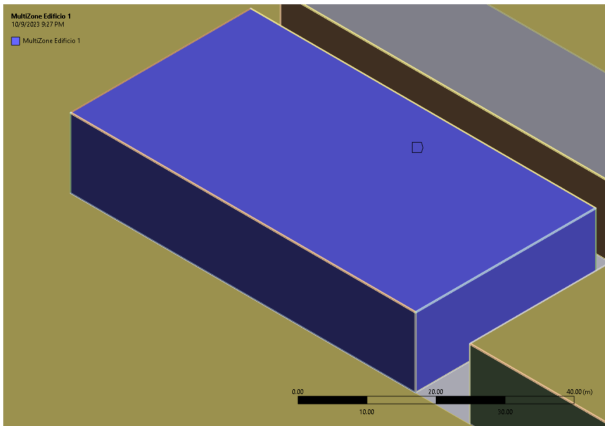


Figura 3.13: En azul, los volúmenes que forman parte del alcance del método *MultiZone* para un edificio. Notar que no se seleccionan los bordes o esquinas.

Esta configuración garantiza que cada pared, techo o piso se divida en cuatro capas utilizando hexaedros. Además, como se verá en la Sección 3.6.3, estas divisiones se propagan automáticamente hacia los bordes, ocasionando su división en capas y su intersección conforme con otros objetos. Se observa el resultado en la Figura 3.14, un acercamiento a una de las esquinas del edificio en la Figura 3.15 y un plano de corte en la Figura 3.16. Para mallar el resto de los edificios se repiten los mismos pasos.

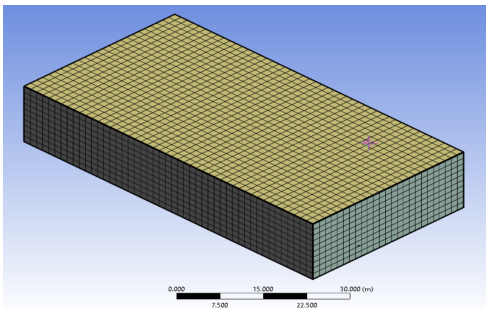


Figura 3.14: Malla de elementos de un edificio.

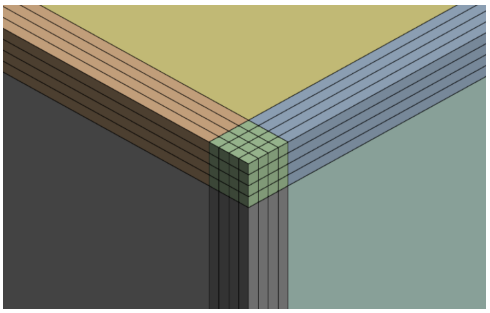
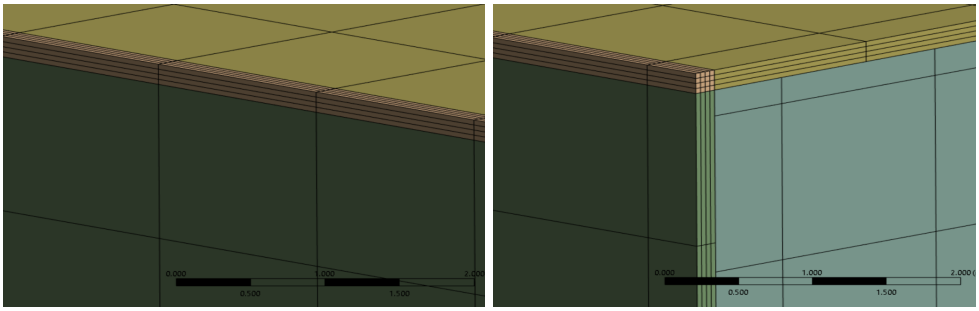


Figura 3.15: Acercamiento a la esquina de un edificio mallado.

El mallado del suelo debajo del piso de los edificios utiliza un *MultiZone* con las mismas características que los anteriores. En este caso, los volúmenes seleccionados como alcance son cuatro bloques de espesor un metro cuya medida coincide con el largo y ancho de los edificios.

Finalmente, el método *MultiZone* también se utiliza en la malla del domo. El domo es una cáscara ya que únicamente se utiliza para radiación y no para conducción (ver Figura 3.17). Se utiliza la siguiente configuración:

- Scope: superficie del domo.
- *Surface mesh method* (método de mallado de la superficie): uniforme. Seleccionar esta opción permite definir el tamaño de los cuadriláteros de la malla.



(a) Sin corte. (b) Con corte.

Figura 3.16: Plano de corte de un edificio.

- *Free Face Mesh Type* (tipo de malla en superficie libre): *All Quads* (todos cuadriláteros).
- *Element Size* (tamaño de los elementos): 65,0m. Este tamaño genera una malla del cielo que cuenta con 297 cuadriláteros.

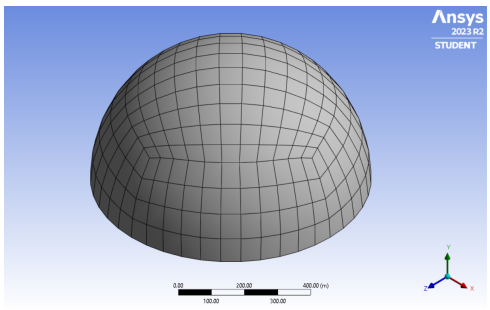


Figura 3.17: Malla de elementos del domo.

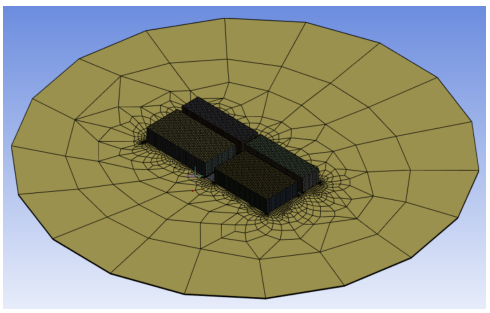


Figura 3.18: Malla de elementos del suelo.

Método *Body Sizing*

Este método permite sobreescribir el tamaño máximo de los elementos de la malla[7]. En este contexto, se aplica sobre el suelo circundante a la ciudad para que los elementos sean más grandes cuanto más se alejen de los edificios. Para lograr esto, se define un tamaño objetivo alto para los elementos del suelo. *Meshing* intenta generar elementos con estas dimensiones, pero sin perder la conformalidad con los elementos de la ciudad (que tienen tamaños reducidos de hasta 0,045m en las esquinas de los edificios), lo que ocasiona que los elementos aumenten su tamaño a medida que se alejan de los edificios, como se observa en la Figura 3.18. El método *Body Sizing* se configura como sigue:

- Alcance: Suelo circundante a la ciudad.
- Tipo: *Element Size* (tamaño de elemento).
- *Element Size*: 100m.

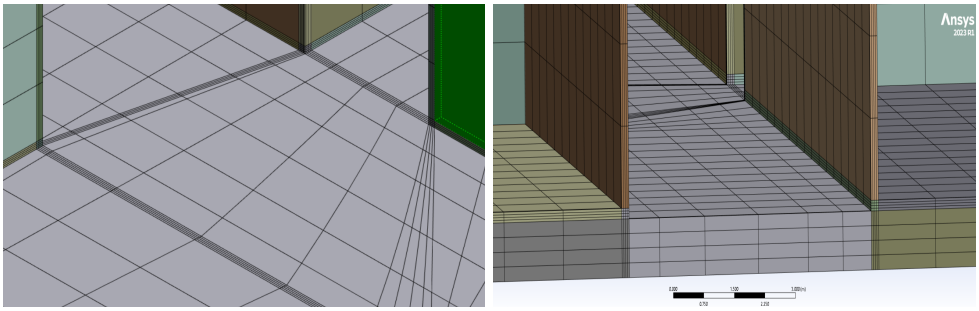
Método *Automatic*

Meshing aplica por defecto el método *Automatic* a todos los elementos que no se seleccionan en ningún método. Cuando se aplica sobre sólidos, intenta generar una malla por medio de un barrido que se origina en una superficie del sólido y lo recorre hasta llegar a otra de sus superficies[8]. Particularmente, *Meshing* aplica *Automatic* sobre los siguientes sólidos:

[7] https://ansyshelp.ansys.com/account/secured?returnurl=/Views/Secured/corp/v231/en/wb_msh/wb_msh_q1d_zhp_1w.html

[8] Descripción del método *Automatic* en: https://ansyshelp.ansys.com/account/secured?returnurl=/Views/Secured/corp/v231/en/wb_msh/msh_auto_method_option.html

- Bordes de los edificios, compuestos por las intersecciones entre paredes, techos y pisos. Como se mencionó anteriormente, la malla de estas zonas se genera automáticamente a partir de sus capas, las cuales son previamente generadas por *MultiZone*.
- Calle. Esta malla se genera a partir de barridos que inician en el suelo que está por debajo de los edificios y se dirigen hacia otro edificio, obteniendo como resultado una malla de cuatro capas conforme con los edificios, que se puede observar en la Figura 3.19.
- Suelo. Esta malla se construye con *Automatic* porque sólamente requiere ser conforme (además de contar con elementos más grandes a medida que se aleja de los edificios, lo que se resuelve con *Body Sizing*, como se describió previamente).



(a) Acercamiento sobre la intersección de la calle.

(b) Vista de la malla de la calle con un plano de corte

Figura 3.19: Conformalidad de la calle con los edificios.

3.6.4. Generación por pasos

La generación de la malla se realiza mediante la opción *Generate*, la cual utiliza todas las configuraciones y métodos activos para generar un mallado de todo el modelo. Sin embargo, para obtener los resultados deseados es necesario ordenar el proceso de generación en una secuencia de pasos en lugar de un solo paso. En Ansys, esto se denomina *Selective Meshing* o mallado selectivo.

Selecciones para *Selective Meshing*

El mallado selectivo permite generar la malla de una selección de elementos de manera "aislada", por lo tanto, el primer paso para utilizar *Selective Meshing* es crear selecciones de elementos. Las mallas de los edificios son las que deben cumplir más requisitos y restricciones, por lo que es conveniente que sean las primeras en generarse.

Para seleccionar los volúmenes de los edificios se crean *Named Selections* o selecciones nombradas, que son una selección de elementos que se guarda en el árbol del modelo para contar con un acceso rápido a determinados elementos. Se crea una selección nombrada para cada edificio (ver Figura 3.20). A continuación, se seleccionan los volúmenes de cada uno de los grupos creados (opción *Select Items On Group*) y se genera la malla mediante la opción *Generate Mesh On Selected Bodies* (que se encuentra en el menú contextual desplegado al hacer click derecho sobre los volúmenes seleccionados). Posteriormente, se utiliza *Generate* para generar la malla del resto de los elementos.

Grabación de pasos

Durante la implementación de la malla, muy a menudo es necesario regenerar la misma para evaluar diversas técnicas de construcción o comparar distintos métodos. En este sentido, resulta de utilidad registrar los pasos de la generación para que se ejecuten automáticamente, en lugar de generar manualmente la malla de cada edificio y del resto de elementos cada vez que se requiera reconstruir el mallado completo.

La grabación de los pasos se activa mediante la opción *Start Recording* (haciendo click derecho en *Mesh* en el árbol del modelo). Una vez activada, se siguen los pasos mencionados anteriormente (generar secuencialmente la malla de cada edificio y luego utilizar *Generate*), proceso que resulta en la grabación de cuatro pasos; la grabación se detiene automáticamente antes de grabar el quinto paso debido a que el

mallado del resto de los elementos es una operación por lotes (el domo conforma una parte separada del resto de la geometría y, por lo tanto, se malla por separado). Luego de finalizada la grabación, los usos subsiguientes de *Generate* generan una malla en el mismo orden que la secuencia grabada.

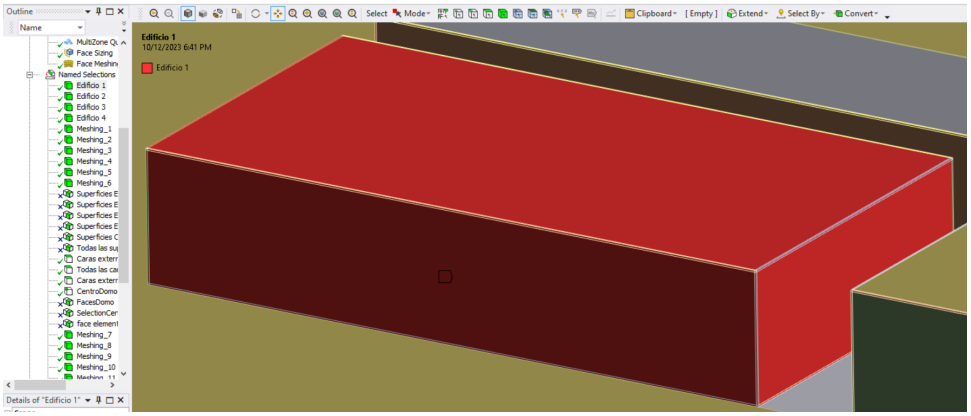
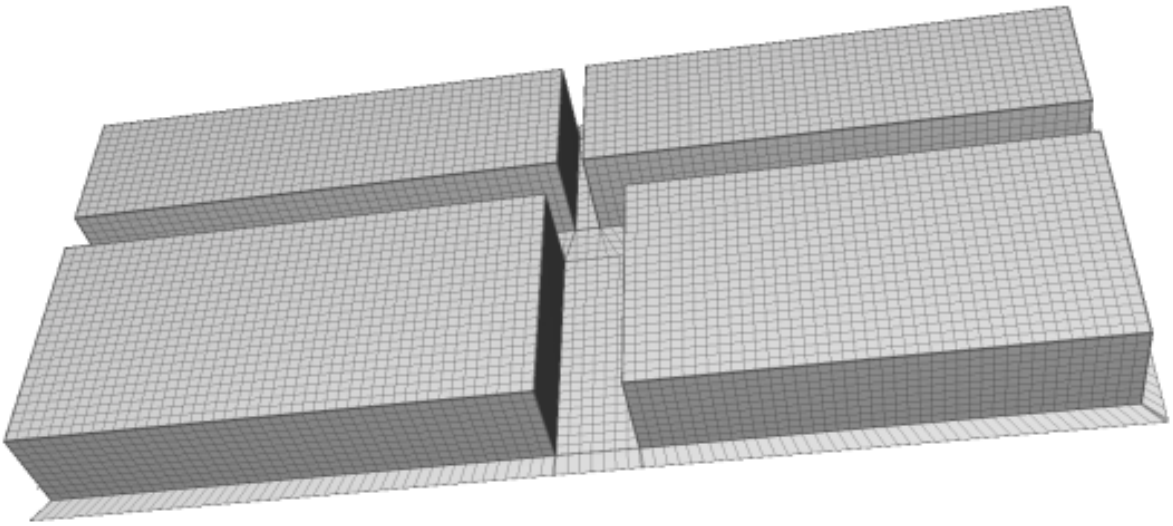


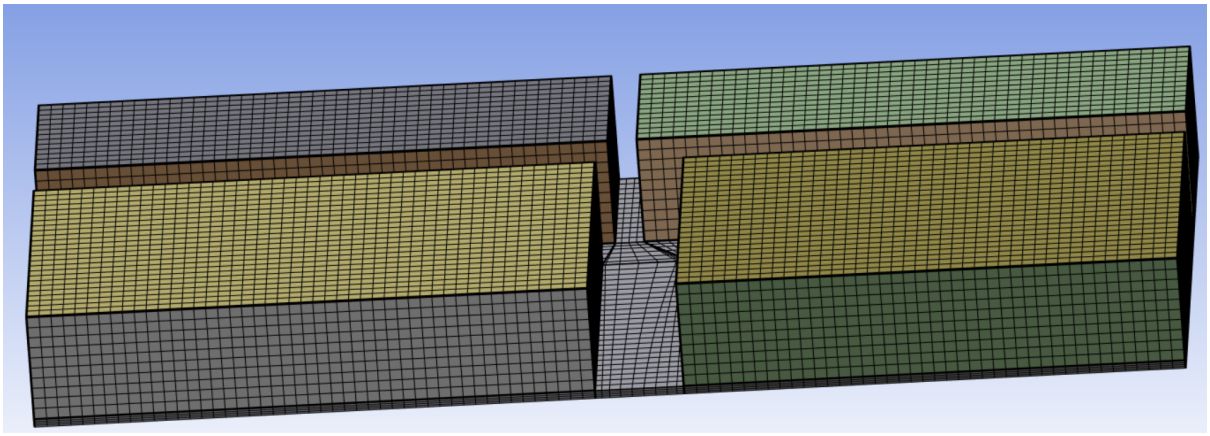
Figura 3.20: Selección nombrada del edificio 1.

3.6.5. Resultado

La malla construida es muy similar a la malla utilizada en la tesis original, como se observa en las figuras 3.21a y 3.21b), donde se comparan los mallados de la ciudad. En lo que respecta al resto de la malla, aunque no se dispone de imágenes para llevar a cabo una comparación visual directa de los resultados, la malla generada satisface todos los requisitos preestablecidos.



(a) Malla original (Aguerre, 2020).



(b) Malla construida de la ciudad (sin suelo circundante).

3.7. Configuración de las condiciones de borde

Esta sección se enfoca en la carga de las condiciones de borde del sistema, que se cargan en el módulo térmico al hacer click en *Transient Thermal* en el árbol de elementos del modelo. Este módulo contiene inicialmente dos elementos configurables: *Initial Temperature* y *Analysis Settings*, pudiéndose insertar luego otros elementos como *Temperature* o *Convection* (Figura 3.22), entre otros, para configurar temperaturas e intercambios por convección, respectivamente.

3.7.1. Condiciones iniciales

Las condiciones iniciales determinan la distribución de temperaturas bajo las cuales se encuentra el sistema en el tiempo 0 (ANSYS inc, 2019). Pueden definirse a partir de una simulación *Steady State* o se puede establecer una temperatura inicial uniforme para todos los nodos de la malla[9]. En este caso, se opta por la segunda opción. Para establecer la temperatura inicial, se configura el elemento *Initial Temperature* con las siguientes propiedades:

- *Initial Temperature*: *Uniform Temperature*.
- *Initial Temperature Value*: 14,6 ºC. Este valor corresponde a la temperatura inicial del aire en el exterior de los edificios.

[9] https://ansyshelp.ansys.com/account/secured?returnurl=/Views/Secured/corp/v232/en/wb_sim/ds_transient_thermal_analysis_type.html

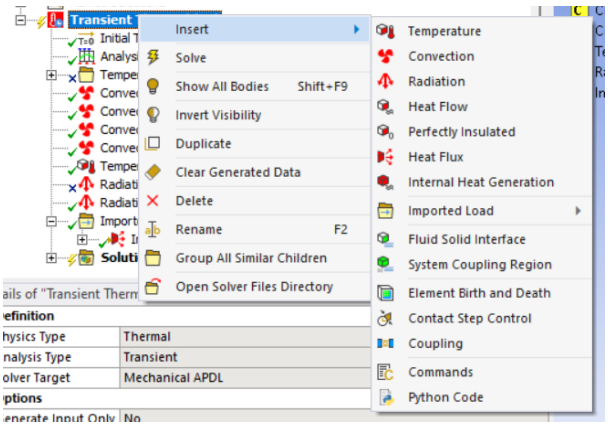


Figura 3.22: Insertar elementos en Transient Thermal.

3.7.2. Configuración del análisis

La configuración del análisis permite adaptar las opciones de solución según el tipo análisis específico. Entre estas configuraciones destacan los *Step Controls* o controles de los pasos. Un paso se define como un conjunto de cargas para las cuales se busca obtener una solución[10]. En este caso, se utiliza un sólo paso para toda la simulación. Los pasos se pueden dividir en subpasos llamados *Substeps*. La subdivisión de los pasos permite automatizar la creación de intervalos de tiempo, lo que resulta de utilidad para dividir el único paso en 144 intervalos de tiempo. El elemento *Analysis Settings* en su sección *Step Controls* se configura de la siguiente manera:

- *Number of Steps*: 1.
- *Step End Time*: 86400 s. Esta cantidad es equivalente a un período de 24 horas desde el inicio del análisis.
- *Auto Time Stepping*: *Off*. Esta función ajusta automáticamente los intervalos de tiempo para optimizar la ejecución de la simulación. Se desactiva esta opción debido a que se desean mantener intervalos de tiempo constantes.
- *Define By*: *Time*. Esta opción determina el criterio que se utiliza para definir los intervalos de tiempo, que puede ser por tiempo en segundos o por cantidad. Se selecciona la definición basada en el tiempo.
- *Time Step*: Se define un intervalo de 600 segundos, equivalente a 10 minutos.

3.7.3. Elementos *Temperature*

El elemento *Temperature* se utiliza para imponer condiciones de borde con temperaturas uniformes sobre la geometría que pueden variar en el tiempo o mantenerse constantes[11]. En este trabajo, estos elementos establecen la temperatura del suelo y de los elementos que componen el domo.

Temperatura del suelo

Para simular la temperatura del suelo a un metro de profundidad, se utiliza un *Temperature* configurado con los siguientes parámetros:

- *Scoping Method*: *Geometry Selection.*
- *Geometry*: se seleccionan las seis caras inferiores del suelo: una por cada edificio, una para la calle y una para el suelo circundante (ver Figura 3.23).
- *Magnitude*: Se establece una temperatura constante de 11,1 ºC.

[10] https://ansyshelp.ansys.com/account/secured?returnurl=/Views/Secured/corp/v232/en/wb_sim/ds_Steps_Substeps_Eq.html?q=steps

[11] https://ansyshelp.ansys.com/account/secured?returnurl=/Views/Secured/corp/v232/en/wb_sim/ds_Given_Temperatures.html

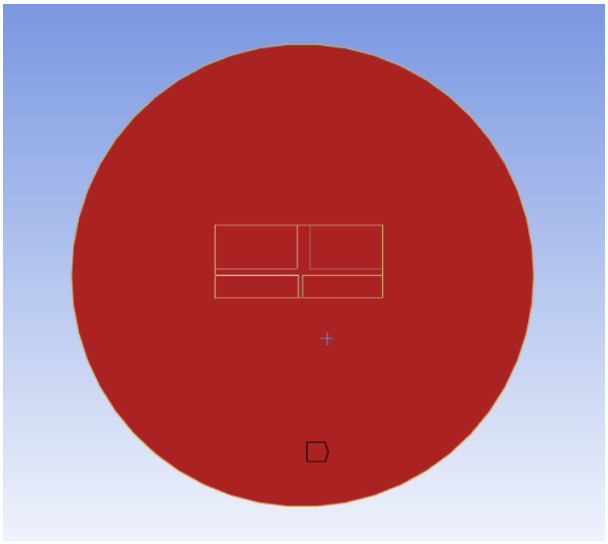


Figura 3.23: Alcance del elemento *Temperature* del suelo, observando la ciudad desde abajo.

Temperatura del domo

La temperatura del cielo varía en tiempo y espacio dada la distribución utilizada en la tesis original (Figura 2.6), lo que se representa mediante la inserción de un elemento *Temperature* de magnitud variable por cada cuadrilátero del domo. Por lo tanto, se utilizan 297 *Temperature*, cada uno configurado con los siguientes parámetros:

- *Scoping Method*: *Geometry Selection*.
- *Geometry*: un *Face Element*[12] determinado del domo.
- *Magnitude*: *Tabular Data*. Esta opción permite definir datos tabulares en función del tiempo. Se observa un ejemplo en la Figura 3.24. La obtención de estos datos tabulares se describe a continuación.

Tabular Data

	Steps	Time [s]	Temperature [°C]
1	1	0.	-6.01
2	1	1800.	-6.15
3	1	3600.	-6.31
4	1	5400.	-6.41
5	1	7200.	-6.51
6	1	9000.	-7.11

Figura 3.24: Ejemplo de datos tabulares para la temperatura de un elemento del domo.

Los datos de temperatura del cielo provienen de la campaña de medición estudiada por el trabajo original y se presentan en un archivo de formato CSV. Este archivo se organiza en once columnas, de las cuales una corresponde al registro del tiempo y las diez restantes corresponden al registro de las temperaturas en función de la inclinación respecto al zenit, con la primera columna representando la temperatura en 0 grados de inclinación (en el zenit) y la última columna representando la temperatura en 90 grados de inclinación (en el horizonte). Todas las temperaturas están expresadas en grados Kelvin.

[12] Un *Face Element* es el elemento bidimensional más pequeño de la malla. En este trabajo un *Face Element* puede ser un triángulo o un cuadrilátero.

Cada fila corresponde a intervalos de treinta minutos, empezando a las 00:00 y finalizando a las 23:30, con un total de 48 filas.

Al momento de integrar los datos de este archivo con los elementos *Temperature* de Ansys, se presentan algunos problemas a resolver. En primer lugar, se deben convertir las temperaturas a grados Celsius, y el tiempo a segundos. En segundo lugar, como sólamente se cuenta con registros de temperatura para diez valores de inclinación específicos, se requiere interpolar linealmente para obtener temperaturas en otras ubicaciones con diferente inclinación. En tercer lugar, es necesario encontrar los grados de inclinación respecto al zenit de cada cuadrilátero de la malla del domo para asignar las temperaturas a la ubicación correcta. Luego de realizar estas tareas, se deben cargar 48 filas de datos de temperaturas para cada uno de los 297 elementos del domo.

Carga automatizada de temperaturas del domo

Para simplificar este proceso, se utiliza la funcionalidad *Mechanical Scripting*, herramienta que permite la ejecución de código en lenguaje Python en Ansys con el objetivo de automatizar operaciones comunes como la creación de elementos en el árbol del modelo, la modificación de configuraciones del análisis, entre otras.

Se desarrolla un script python que realiza tres tareas principales:

- Lectura del archivo CSV: se guardan los datos del archivo CSV en memoria y se convierten las temperaturas a grados Celsius.
- Cálculo de la temperatura de cada cuadrilátero del domo: se itera sobre los *Face Elements* del domo y se utiliza trigonometría para calcular el ángulo cenital del centroide del cuadrilátero, dato utilizado para obtener la temperatura del cuadrilátero mediante interpolación lineal.
- Creación de elementos *Temperature*: se crea un elemento *Temperature* por cada cuadrilátero, y se realizan las conversiones de formato necesarias para cargar los datos tabulares de temperatura en función del tiempo. En particular, los objetos *Temperature* funcionan con *Inputs* y *Outputs*. En este caso, la entrada es el tiempo y la salida es la temperatura. Tanto el tiempo como la temperatura se declaran utilizando la función *Quantity* (Crowley, 2023).

Se presenta el código en Apéndice A. Cabe notar que previo a la ejecución del script es necesaria la creación de dos *Named Selections*: "FacesDomo" (que contiene los 297 *Face Elements*) y "CentroDomo" que contiene el vértice correspondiente al centro del domo.

3.7.4. Convección

La convección se simula mediante el elemento *Convection*. Se añaden cuatro de estos elementos en total: tres de ellos para configurar la convección en las caras internas de los edificios, y uno para configurar las caras exteriores.

Convección en caras interiores

Para simular los intercambios de calor por convección en los pisos, paredes y techos se utilizan tres elementos *Convection* configurados con los siguientes parámetros:

- *Scoping Method*: *Geometry Selection.*
- *Geometry*: se seleccionan las caras internas los pisos, paredes y techos de cada edificio según corresponda.
- *Film Coefficient*: se establece un coeficiente de convección de $0,7\,\mathrm{Wm^{-2}K^{-1}}$ para los pisos, $2,5\,\mathrm{Wm^{-2}K^{-1}}$ para las paredes y $5\,\mathrm{Wm^{-2}K^{-1}}$ para los techos.
- *Ambient Temperature*: se establece la temperatura ambiente en 17 $^{\mathrm{o}}$C para los tres tipos de superficies.

Convección en caras exteriores

En el caso de las superficies exteriores, se utiliza un elemento *Convection* configurado con los siguientes parámetros:

- *Scope*: *Named Selection.*

- *Named Selection*: se utiliza una selección que contiene todas las superficies exteriores de cara al cielo (ver Figura 3.25).
- *Film Coefficient*: $10\,\mathrm{Wm^{-2}K^{-1}}$.
- *Ambient Temperature*: *Tabular Data*. Se cargan manualmente los datos de temperatura ambiente utilizados en la tesis original.

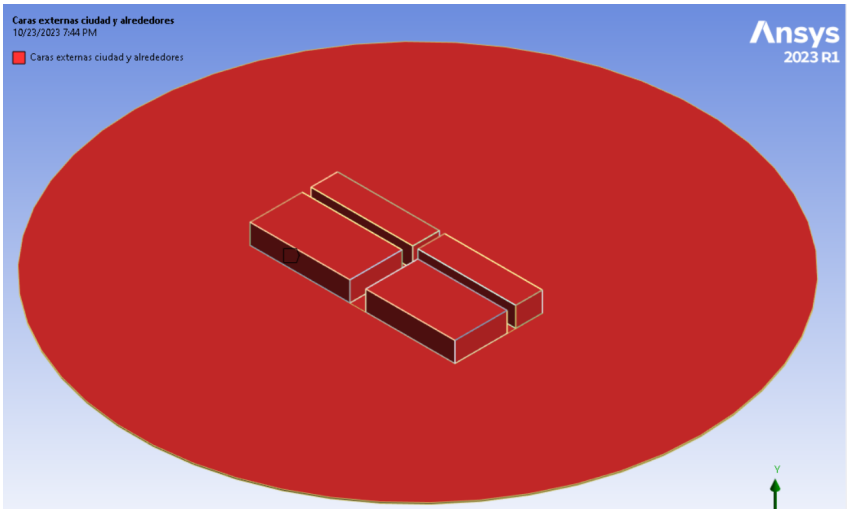


Figura 3.25: *Named Selection* utilizada para la convección exterior.

3.7.5. Radiación de onda larga

La aplicación de condiciones de borde que simulan la transferencia de calor por radiación de onda larga está dada por el elemento *Radiation*[13]. Se configuran dos elementos de este tipo: uno para la cara inferior del domo y otro para las superficies de los edificios y sus alrededores.

El elemento de *Radiation* del domo se configura de la siguiente manera:

- *Scoping Method*: *Geometry Selection*.
- *Geometry*: se selecciona el domo (Figura 3.7).
- *Shell Face*: se selecciona *Bottom* para utilizar la cara que apunta hacia la ciudad.
- *Correlation*: *Surface to Surface*. Esta propiedad asegura que los intercambios de calor por radiación se produzcan entre superficies y no al ambiente.
- *Emissivity*: 1. Los elementos se modelan como cuerpos negros.
- *Enclosure*: 1. Esta propiedad determina el cerramiento en el que ocurren los intercambios de calor por radiación. Tanto el domo como la ciudad deben estar en el mismo cerramiento.
- *Enclosure Type*: *Perfect*. Esta opción genera un cerramiento perfecto en el que el intercambio de calor por radiación sólamente actúa entre las superficies, sin afectar el ambiente.

Para el caso del elemento *Radiation* correspondiente a la radiación en las superficies de la ciudad, se asignan los siguientes parámetros:

- *Scoping Method*: *Named Selection*.
- *Named Selection*: se reutiliza la selección utilizada en la convección exterior, correspondiente a la ciudad y los alrededores (Figura 3.25).
- *Correlation*: *Surface to Surface*.
- *Emissivity*: 1.

[13] https://ansyshelp.ansys.com/account/secured?returnurl=/Views/Secured/corp/v231/en/wb_sim/ds_Radiation.html

- *Enclosure*: 1.
- *Enclosure Type*: *Perfect*.

Una vez definidos estos dos elementos de radiación, se obtiene un comportamiento en el que se simula el intercambio radiativo entre el cielo (que tiene una temperatura predefinida) y los edificios. De esta forma, se modelan comportamientos como el enfriamiento nocturno de la superficie urbana por estar expuesta al frío del cielo despejado. A su vez, también se simula el intercambio radiativo de onda larga entre los edificios, el cual tiene un efecto importante cuando existen superficies con grandes diferencias de temperatura. Por ejemplo, una pared que recibe luz solar directa eleva su temperatura rápidamente, y luego emite radiación que lentamente calentará las paredes de enfrente, que están en sombra.

3.7.6. Radiación de onda corta

La radiación de onda corta se modela como un flujo impuesto para cada elemento de la superficie exterior de la ciudad. Este flujo se pre-computa utilizando un código externo (el mismo desarrollado en la tesis de doctorado original), pero ejecutado sobre la geometría generada en ANSYS. Dicho módulo recibe como entrada un archivo que describe una malla de elementos en formato ASCII y produce como salida 144 archivos CSV compuestos por aproximadamente 20000 filas y una columna cada uno. Cada archivo CSV describe los valores de radiación incidentes a las superficies del modelo para un paso de tiempo determinado de la simulación. Las filas corresponden a los elementos de la malla y la única columna corresponde al valor de radiación de onda corta que incide sobre ellos en un momento determinado del día.

El proceso para generar e importar los datos de radiación al modelo de *Mechanical* consta de cinco pasos:

1. Exportación de la malla de elementos del modelo.
2. Depuración de la malla y clasificación de sus superficies.
3. Exportación de la malla depurada.
4. Computación de la radiación.
5. Carga de la radiación en el modelo.

Exportación de la malla de elementos

Mechanical permite exportar la malla de elementos en los siguientes formatos: STL (extensión .stl), ANSYS msh (de extensión .msh) o CGNS (de extensión .cgns) (ANSYS inc, 2010). Esta funcionalidad es accesible dentro del modelo del sistema seleccionando el menú *File* y las opciones *Export* y *Mesh*.

El módulo de radiación toma como entrada archivos de texto en formato .ac [14], por lo que se requiere una conversión de formatos. No es de utilidad exportar la malla en formato .stl debido a que este formato usa triángulos. Por otro lado, se presentaron dificultades a la hora de intentar exportar la malla completa de la ciudad a formato .cgns. Además, no se encontraron bibliotecas que permitiesen parsear directamente .msh a .ac. Luego de explorar varios formatos (.dae, .obj, .ply, x3d, .off), se acordó la utilización del formato .obj como formato intermedio entre .msh y .ac. Se utilizó la librería *meshio* [15], la cual permite convertir archivos .msh a .obj por medio del comando: *meshio convert mesh.msh output.obj*.

Depuración de la malla de elementos y clasificación

La depuración de la malla es el proceso de eliminación de elementos de la misma con el objetivo de conservar únicamente las superficies relevantes para la computación de la radiación de onda corta, la cual actúa sobre la cáscara exterior de los edificios de la ciudad y sus alrededores. Además, es esencial clasificar las superficies para poder identificarlas y asignar el coeficiente de reflexión adecuado a cada una.

Para llevar a cabo este procesamiento, se utiliza *Blender*, una suite de creación 3D de código abierto utilizada por artistas y diseñadores de todo el mundo (Hosen, Ahmmed, y Dekkati, 2019). Las razones

[14] https://www.inivis.com/ac3d/man/ac3dfileformat.html
[15] https://pypi.org/project/meshio/

principales para utilizar este software son su capacidad de trabajar con archivos .obj como el obtenido en el paso anterior, y su interfaz de usuario intuitiva, desde la cual resulta sencillo depurar la malla. Previamente, se intentó utilizar *Gmsh*, generador de mallas de elementos finitos (Geuzaine y Remacle, 2009), donde se logró depurar la malla de los edificios, pero resultó imposible importar la malla completa de la ciudad con el suelo circundante debido a que la misma se compone de una parte estructurada y una parte no estructurada. Importar una malla de tales características no está soportado por Gmsh.

Las superficies se clasifican mediante la asignación de materiales. Se deben crear cuatro materiales, uno para cada superficie que interesa catalogar: *Roof*, *Wall*, *Street* y *Ground*. Estos materiales se crean utilizando el menú *Material Properties* ubicado en la barra lateral derecha (ver Figura 3.26). Para asignar materiales a las superficies se deben seleccionar cuadriláteros primero. Tras la exploración de múltiples métodos de selección de elementos, el proceso que se describe a continuación resultó ser el más simple y el menos susceptible a errores no intencionados (por ejemplo, clicks erróneos que anulen toda la selección):

1. En la ventana principal, hacer click en *Modeling*, de esta manera se muestra la malla de la geometría.
2. Fijar el modo de selección *Face Select* en la parte superior izquierda de la ventana principal.
3. Selecciona un cuadrilátero cualquiera de la superficie que interesa clasificar.
4. Desplegar el menú *Select* y elegir las opciones *Select Linked* y luego *Linked Flat Faces*. Esto provee una selección de todas las caras conectadas entre sí que se encuentran en el mismo plano que la selección inicial (ver ejemplo en la Figura 3.27).
5. Para asignar un material: seleccionar uno de ellos y hacer click en *Assign*.

Repitiendo los pasos 3, 4 y 5 sucesivamente se pueden seleccionar todas las superficies de interés, asignando los materiales correspondientes con cada nueva selección.

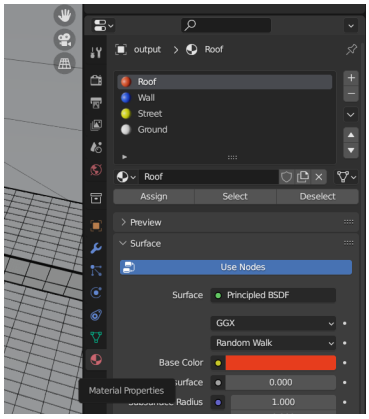


Figura 3.26: Sección de materiales de *Blender*.

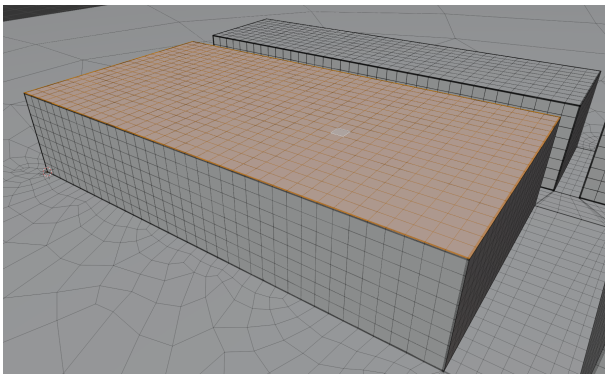


Figura 3.27: Selección de los elementos del techo de un edificio.

Es necesario eliminar todos los elementos que no forman parte de las selecciones anteriores. El proceso de depuración resulta sencillo si se utilizan las asignaciones de materiales creadas en el paso anterior de la siguiente manera:

1. Seleccionar todos los elementos de la malla presionando la tecla “A”.

2. Deseleccionar todos los elementos asignados a los materiales existentes mediante la opcion *Deselect* en la sección de materiales.

3. Borrar todos los elementos haciendo click derecho y seleccionando la opción *Delete Faces*.

Finalmente, la nueva malla se exporta desde el menú *File* > *Export* > *Wavefront (.obj)* y habilitando la opción *Export* en la sección *Materials*. Se obtiene como resultado un archivo de 40002 líneas, las cuales están agrupadas por material.

Cómputo de la radiación

Para computar la radiación de onda corta se trabajó en conjunto con el supervisor del proyecto, quien se encargó del parseo del archivo .obj obtenido en la parte anterior y la posterior ejecución del módulo de radiación. El módulo produce 144 archivos CSV, donde cada uno consta de alrededor de 20000 filas y una sola columna. Es importante destacar que el orden de las filas en estos archivos no coincide completamente con el orden de los elementos en el archivo .obj: en los archivos CSV se listan primero los triángulos y luego los cuadriláteros. No obstante, el orden relativo entre triángulos y entre cuadriláteros se mantiene.

Carga de la radiación en el modelo

El volumen de los datos de radiación es significativamente mayor que los datos de temperatura del cielo, por lo que nuevamente es necesaria la utilización de un método de carga mediante scripting.

La carga de los datos precalculados de radiación de onda corta en ANSYS se implementó de dos maneras a lo largo del proyecto. En primer lugar, se intentó utilizar *Mechanical Scripting*, aunque se encontraron limitantes de eficiencia computacional que obligaron a buscar otra alternativa. En esta sección se describe lo logrado, de forma que quede un registro técnico de los problemas encontrados y las soluciones propuestas.

Se requiere implementar un script que sea capaz de:

- Asociar cuadriláteros en Ansys con filas de los archivos CSV.
- Cargar los valores de radiación para cada elemento y cada paso de tiempo.

La principal dificultad para mapear elementos de Ansys con las filas del archivo .obj es que el orden de los elementos en este último es completamente distinto al orden de los elementos que devuelve la API de Ansys en *Mechanical Scripting*.

El primer paso para resolver este problema es la generación de un archivo CSV auxiliar que, teniendo en cuenta la estructura de un archivo .obj, reúna las coordenadas de los vértices de las caras en una sola fila para cada cuadrilátero. Con este fin, se implementa el código presentado en Apéndice B, cuya ejecución genera un archivo CSV compuesto por 20001 filas correspondientes a las caras exteriores de la malla y 13 columnas que describen el índice de la cara y las coordenadas de cada vértice (cuatro vértices de tres coordenadas cada uno).

La solución para asociar los cuadriláteros de Ansys consiste en iterar sobre ellos, buscando una coincidencia de coordenadas con las filas del CSV auxiliar. Se presenta en Apéndice C una versión simplificada del código de carga de radiaciones que solamente tiene en cuenta los techos de la ciudad. Previo a la ejecución de este script simplificado se debe crear en Ansys una selección de *Face Elements* llamada *RoofsFaces* que contenga a los cuadriláteros del techo.

El script se encarga en primer lugar de iterar sobre los elementos del techo, y a partir de la información geométrica provista por la API de Ansys, encontrar la fila del CSV que corresponde a cada elemento. Esta búsqueda requiere tener presente que: Si bien la selección se crea a partir de las caras de los elementos que componen una superficie, el objeto iterable que provee Ansys para iterar sobre estas selecciones es un volumen tridimensional, no una cara bidimensional. Esto dificulta la búsqueda de las filas pertinentes en el CSV; no es posible comparar directamente los nodos de las filas con los nodos del objeto iterable porque se trata de elementos con diferente dimensión. Se resolvió considerar que una fila es la correcta en caso de que todos sus nodos estén contenidos en los del iterable. Debido a diferencias de redondeo entre Ansys y el archivo .obj generado por Blender, se debe definir una tolerancia al evaluar la igualdad de coordenadas. Se definió una tolerancia de 0,001.

En segundo lugar, para cada fila encontrada en la operación anterior, se obtienen los valores de radiación para cada paso de la simulación a partir de los CSVs de radiación, y se genera en memoria una estructura de datos de diccionario para guardar las asociaciones de elementos con sus datos tabulares de radiación. Esta operación aprovecha que los elementos de estos CSVs tienen el mismo ordenamiento que el CSV auxiliar. Los CSVs se cargan en memoria para evitar lecturas a disco.

En tercer lugar, se crean objetos *Heat Flux*[16] a partir de los datos guardados en la operación anterior. La creación de *Heat Fluxes* debe referenciar tanto un elemento como un índice de cara dentro de ese elemento. Esta asociación elemento-índice de cara es accesible desde el código, explorando los atributos de las selecciones. Asimismo, se utiliza esta asociación como identificador al momento de cargar en memoria los datos de radiación en el paso anterior.

Nótese que la búsqueda de elementos en el CSV a partir de la inclusión de los nodos de la fila en los elementos tridimensionales de Ansys puede causar la asociación de más de una fila en el caso de elementos esquineros. Por esta razón se limita la búsqueda a las filas correspondientes al techo. Para extender el método al resto de la malla, se deben identificar los índices de las filas que corresponden a cada superficie dentro del CSV, y limitar la búsqueda de elementos dentro de dicho rango.

La carga de radiaciones se realizó por tandas, empezando por el techo. Si bien los valores se cargaron correctamente para cada *Face Element*, la carga fue excesivamente lenta, demorando en el entorno de las dos horas. Además, la creación de cada *Heat Flux* se ralentiza incrementalmente con cada elemento creado, por lo que no resulta viable insertar 20001 elementos. En total, se lograron insertar aproximadamente 4700 elementos, correspondientes al techo y algunas paredes verticales, en un período de tiempo cercano a las diez horas, antes de que Ansys finalizara su ejecución debido a una insuficiencia de recursos ocasionada por la gran cantidad de elementos en el árbol del modelo. Esta limitante motivó la búsqueda de otras alternativas para importar la radiación de onda corta.

Búsqueda de alternativas

Para intentar solucionar el problema del largo tiempo de cómputo al cargar radiaciones, se contactó al área de soporte de ESSS, además de crear un post en el foro oficial de Ansys[17]. Las respuestas por ambas vías confirmaron que el árbol del modelo no está diseñado para manejar correctamente la cantidad de elementos que se intentaron cargar, y propusieron dos opciones:

1. La utilización de comandos en Ansys Parametric Design Language (APDL). Esta opción se basa en la creación de un elemento *Commands* donde se escriben comandos en APDL para la creación de elementos *Heat Flux*. Se probó esta opción con casos sencillos pero se descartó que debido a la complejidad del lenguaje y la dificultad para programar iteraciones.

2. La utilización de un sistema *External Data* para leer archivos externos y mapearlos a condiciones de borde dentro del modelo. Esta opción resultó sencilla de implementar, fácil de escalar y adaptable a distintos archivos externos con cantidades variables de datos de radiación, por lo que se siguió este camino, que se describe a continuación.

Carga de radiación de onda corta con *External Data*

Los sistemas *External Data*[18] permiten importar archivos externos a Ansys y mapear sus datos a condiciones de borde dentro del modelo. Es posible configurar el formato esperado de estos archivos y la manera en que se interpreta su contenido. Estos sistemas se pueden crear mediante click derecho en el *Project Schematic* y seleccionando dentro del menú contextual *New Component Systems* la opción *External Data*.

Para seguir la sugerencia en el foro de ansys, se requiere generar un nuevo archivo CSV que unifique los datos de radiación con los datos geométricos de la malla en un sólo archivo. Adicionalmente, se requiere contar con las coordenadas del centroide de cada cuadrilátero, en lugar de las coordenadas de los nodos esquineros. Para obtener este CSV se ejecutan varios scripts Python que realizan las siguientes tareas:

[16] https://ansyshelp.ansys.com/account/secured?returnurl=/Views/Secured/corp/v231/en/wb_sim/ds_Heat_Flux.html

[17] https://forum.ansys.com/forums/topic/loading-heatflux-temperature-too-slow/

[18] https://ansyshelp.ansys.com/account/secured?returnurl=/Views/Secured/corp/v232/en/wb2_help/wb2h_MechLoadapp.html

- Reordenado de *faces_vertices.csv* para listar primero los triángulos y a continuación los cuadriláteros. De esta manera, se obtiene un archivo *reordered_faces_vertices.csv* cuyas filas se ordenan de la misma manera que los CSVs de radiación. Ver código en Apéndice D.
- Lectura de los CSVs de radiación, lectura de *reordered_faces_vertices.csv*, cálculo de centroides a partir de los vértices de cada cara y posterior generación de un nuevo CSV *all_fluxes.csv* con la siguiente estructura: una fila por cuadrilátero, y columnas (x, y, z, hflux1, hflux2, ..., hflux144) donde x, y, z son las coordenadas del centroide de cada cuadrilátero y los valores "hflux" son los valores de flujo radiativo para cada paso de tiempo de la simulación. Ver código en Apéndice E.

Para utilizar los datos de *all_fluxes.csv*, se debe configurar el módulo *External Data*. Para ello, se abre la pantalla de configuración haciendo doble click en *Setup* en el esquema del proyecto. En la columna *Location* se encuentra una función de navegación para buscar el archivo deseado. Una vez seleccionado el archivo, se despliega el panel *Properties of File*, donde se configuran propiedades del archivo como el tipo de formato, (en este caso, delimitado) el caracter de delimitación (coma) y en qué fila inicia la importación. Se observan las propiedades configuradas en la Figura 3.28a.

Luego, en el panel *Table of File*, se indica para cada columna del archivo su tipo de dato, unidad y un identificador. En este caso, las primeras tres columnas representan coordenadas en x, y, z donde la unidad es el metro, mientras que las restantes 144 columnas son valores de flujo de calor y la unidad es Wm^{-2}. La información de cada columna se debe ingresar manualmente. Se observa la configuración de las primeras columnas en la Figura 3.28b. *External Data* también cuenta con una vista previa de la configuración del archivo a importar (Figura 3.28c).

Una vez configurado el componente *External Data*, se debe asociarlo al modelo del análisis térmico. Esta asociación se hace en el esquema del proyecto arrastrando el componente *Setup* del sistema *External Data* hasta el elemento *Setup* del sistema *Transient Thermal*. Luego, se debe actualizar el sistema térmico, lo que ocasiona la creación automática de un elemento *Imported Load*[19] en el árbol del modelo. Dentro de dicho elemento se debe insertar un objeto *Heat Flux*, lo que resulta en la creación de un elemento *Imported Heat Flux*, en el cual se configuran las siguientes propiedades:

- *Scoping Method*: *Named Selection*.
- *Named Selection*: se reutiliza la selección utilizada en la convección exterior (Figura 3.25).
- *Tabular Loading*: *Stepped*. Esta opción permite la carga de datos tabulares por paso de tiempo.

Por otro lado, en el panel *Data View* se deben asociar los pasos de tiempo con los identificadores de los archivos CSV creados anteriormente, es decir, se deben ingresar manualmente 144 pasos de tiempo (en segundos) y para cada uno elegir el identificador del archivo CSV que contiene los datos de radiación para dicho paso de tiempo (ver Figura 3.28d).

Una vez configuradas las asociaciones, se procede a cargar los datos haciendo click derecho en *Imported Heat Flux* y seleccionando la opción *Import Load*. Para confirmar que los datos fueron cargados correctamente desde un punto de vista geométrico, es posible habilitar la opción *Display Source Points* del elemento *Imported Heat Flux*, que permite visualizar en la malla la ubicación de los puntos que se importaron desde el archivo CSV. Todos los puntos importados se ubican en los centroides de las caras. Se observa un ejemplo en la Figura 3.29. Además, la carga de datos genera un resumen con datos estadísticos de la carga (*Imported Load Transfer Summary*) donde se puede corroborar que la totalidad de los 20001 puntos fueron cargados correctamente. No obstante, la importación de datos con *External Data* no permite en este caso la visualización de los datos tabulares de radiación de manera individual para cada cuadrilátero (sí es posible esta visualización con volúmenes menores de datos), como sí fue posible en la carga de temperaturas del domo por medio de scripting.

[19] https://ansyshelp.ansys.com/account/secured?returnurl=/Views/Secured/corp/v232/en/wb_sim/ds_imported_loads.html

Properties of File - C:\Users\Fede\Desktop\TESIS\all_fluxes.csv

	A	B	C
1	Property	Value	Unit
2	Definition		
3	Dimension	3D	
4	Start Import At Line	2	
5	Format Type	Delimited	
6	Delimiter Type	Comma	
7	Delimiter Character	Comma	
8	Length Unit	m	
9	Coordinate System Type	Cartesian	
10	Material Field Data		
11	Analytical Transformation		
12	X Coordinate	x	
13	Y Coordinate	y	
14	Z Coordinate	z	

(a) Propiedades de la importación del archivo *all_fluxes.csv*.

Table of File - C:\Users\Fede\Desktop\TESIS\all_fluxes.csv : Delimiter - ','

	A	B	C	D	E
1	Column	Data Type	Data Unit	Data Identifier	Combined Identifier
2	A	X Coordinate	m		File1
3	B	Y Coordinate	m		File1
4	C	Z Coordinate	m		File1
5	D	Heat Flux	W m^-2	HeatFlux1	File1:HeatFlux1
6	E	Heat Flux	W m^-2	HeatFlux2	File1:HeatFlux2
7	F	Heat Flux	W m^-2	HeatFlux3	File1:HeatFlux3
8	G	Heat Flux	W m^-2	HeatFlux4	File1:HeatFlux4
9	H	Heat Flux	W m^-2	HeatFlux5	File1:HeatFlux5
10	I	Heat Flux	W m^-2	HeatFlux6	File1:HeatFlux6
11	J	Heat Flux	W m^-2	HeatFlux7	File1:HeatFlux7
12	K	Heat Flux	W m^-2	HeatFlux8	File1:HeatFlux8
13	L	Heat Flux	W m^-2	HeatFlux9	File1:HeatFlux9
14	M	Heat Flux	W m^-2	HeatFlux10	File1:HeatFlux10

(b) Configuración de las primeras columnas de *all_fluxes.csv*.

Preview of File - C:\Users\Fede\Desktop\TESIS\all_fluxes.csv

	A	B	C	D
1	X Coordinate	Y Coordinate	Z Coordinate	Heat Flux
2	52.398	-0.0	15.259	0.0
3	44.004	0.0	-76.74	0.0
4	121.066	0.0	1.738	0.0
5	33.989	0.0	-65.69	0.0
6	-6.683	0.0	-5.536	0.0
7	-11.432	0.0	-43.71	0.0
8	2.724	0.0	-68.831	0.0
9	-38.99	0.0	-38.99	0.0
10	150.212	0.0	-6.031	0.0
11	-3.464	0.0	-6.299	0.0

(c) Vista previa de la configuración del archivo a importar con *External Data*.

Data View

Imported Heat Flux

	Magnitude (W/m²)	Analysis Time (s)	Scale	Offset (W/m²)
1	File1:HeatFlux1	600	1	0
2	File1:HeatFlux2	1200	1	0
3	File1:HeatFlux4	1800	1	0
4	File1:HeatFlux5	2400	1	0
5	File1:HeatFlux6	3000	1	0
6	File1:HeatFlux7	3600	1	0
7	File1:HeatFlux7	4200	1	0

(d) Configuración de las primeras filas del panel *Data View*.

Figura 3.28: Vistas de *External Data*.

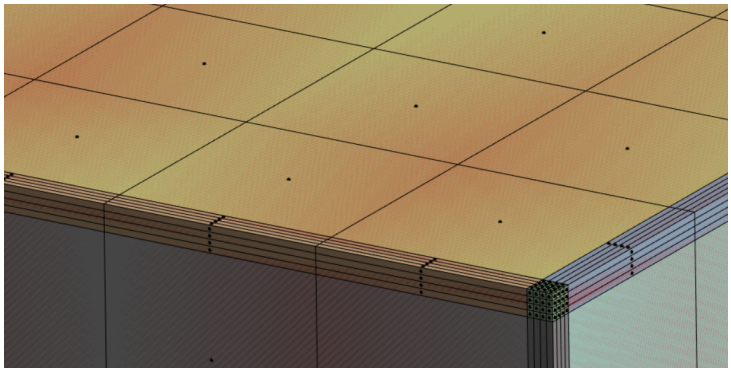


Figura 3.29: *Source Points* en los centroides de las caras exteriores de la malla.

Capítulo 4

Evaluación de los resultados

Este capítulo se dedica a la ejecución de la simulación térmica, el análisis de los resultados obtenidos y la comparación de éstos con los resultados de la tesis original.

4.1. Elementos *Temperature* y *Temperature Probe*

Para monitorear los cambios de temperatura en los elementos del modelo, se utiliza un objeto *Temperature*, el cual se crea haciendo click derecho en *Solution* en el árbol del modelo y seleccionando las opciones *Insert*, *Thermal* y *Temperature*. Es posible definir un *Scope* para limitar la visualización de los resultados de temperatura a determinadas entidades geométricas. Se utiliza la opción de alcance por defecto *All bodies* para visualizar la temperatura de toda la geometría.

Por otro lado, se crean tres objetos *Temperature Probe* para visualizar la evolución de la temperatura en tres puntos determinados N, S y G. Estos tres puntos corresponden a los puntos con el mismo nombre que se analizaron en la tesis original (ver Figura 2.8) y se utilizan para comparar directamente las evoluciones de temperatura obtenidas en cada trabajo. Los objetos *Temperature Probe* se crean haciendo click derecho en *Solution* en el árbol del modelo y seleccionando las opciones *Insert*, *Probes* y *Temperature*. Para asignar como alcance los puntos mencionados, se requiere crear un sistema de coordenadas para cada punto, donde las coordenadas de los respectivos orígenes coinciden con las coordenadas de los puntos N, S y G. Se observan los sistemas creados en la Figura 4.1.

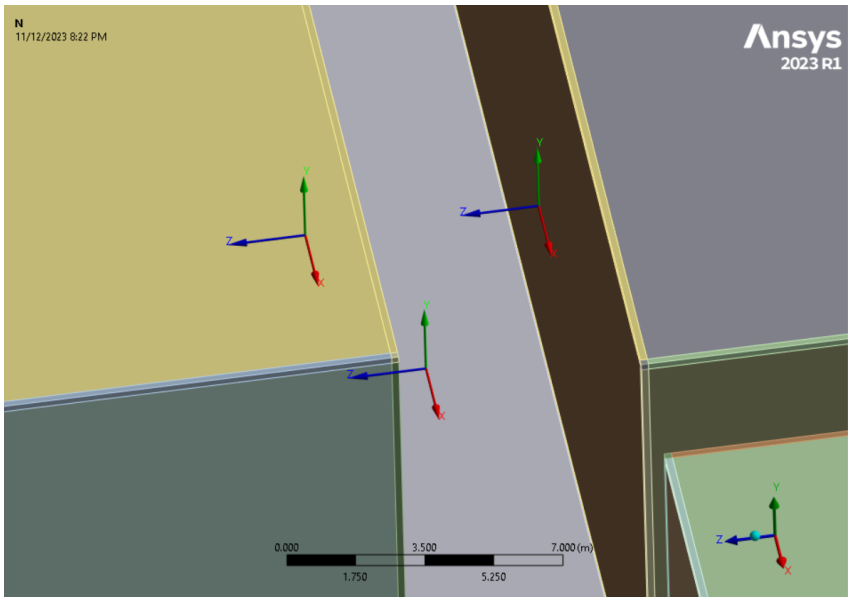


Figura 4.1: Sistemas de coordenadas donde se ubican los puntos N, G y S. Notar que el punto N (izquierda) se ubica en la pared enfrente al punto S (derecha), no en el techo.

4.2. Ejecución

Para obtener resultados es necesario resolver el problema *Transient*. Esto se lleva a cabo haciendo click en la opción *Solve* que se ubica en la parte superior de la pantalla bajo la pestaña *Home*. De esta manera, se da comienzo a la resolución del problema y el cálculo de los resultados. Durante la ejecución, se visualiza una barra que muestra el progreso de cada paso de la misma. Los pasos son:

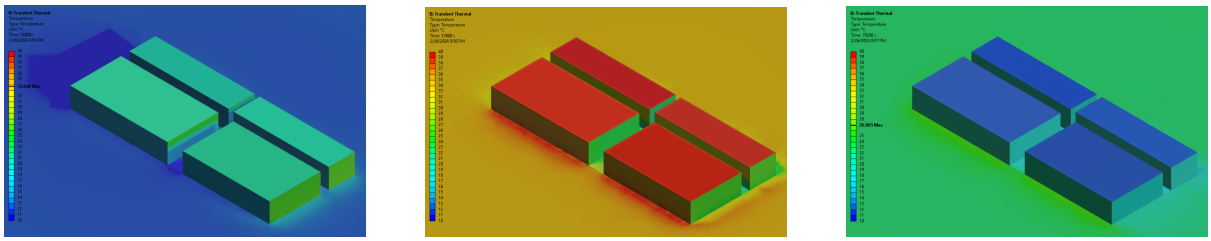
1. *Creating solver input file.*
2. *Building mathematical model.*
3. *Solving the mathematical model.*
4. *Writing results files.*

Simultáneamente, se escribe información más detallada de la ejecución en el objeto *Solution Information* que se encuentra en el árbol del modelo.
El tiempo total de ejecución es de 34 minutos y 58 segundos.

4.3. Resultados

4.3.1. Visualización en Mechanical

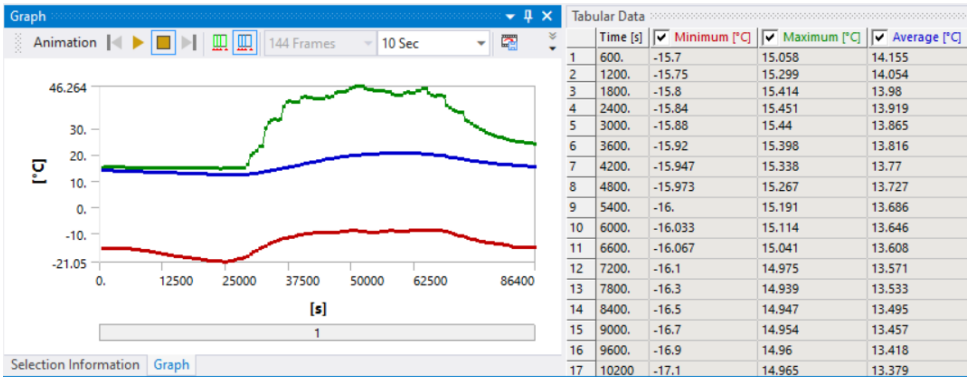
Al hacer click en el elemento *Temperature* dentro de *Solution* en el árbol del modelo, la geometría se pinta con colores en un rango entre azul y rojo. Es posible modificar el máximo, mínimo y la cantidad de valores del rango. En la Figura 4.2 se observa la ciudad representada con un rango de colores que corresponde a temperaturas que van desde los 10 ºC hasta los 40 ºC para varios momentos del día.



(a) 10:00 AM. (b) 4:00 PM. (c) 10:00 PM.

Figura 4.2: Temperatura de la ciudad, visualización en Mechanical.

Por otro lado, el elemento *Temperature* también contiene datos tabulares del mínimo, máximo y promedio de temperatura global, además de una gráfica con los mismos valores (ver Figura 4.3).



	Time [s]	Minimum [°C]	Maximum [°C]	Average [°C]
1	600.	-15.7	15.058	14.155
2	1200.	-15.75	15.299	14.054
3	1800.	-15.8	15.414	13.98
4	2400.	-15.84	15.451	13.919
5	3000.	-15.88	15.44	13.865
6	3600.	-15.92	15.398	13.816
7	4200.	-15.947	15.338	13.77
8	4800.	-15.973	15.267	13.727
9	5400.	-16.	15.191	13.686
10	6000.	-16.033	15.114	13.646
11	6600.	-16.067	15.041	13.608
12	7200.	-16.1	14.975	13.571
13	7800.	-16.3	14.939	13.533
14	8400.	-16.5	14.947	13.495
15	9000.	-16.7	14.954	13.457
16	9600.	-16.9	14.96	13.418
17	10200	-17.1	14.965	13.379

Figura 4.3: Datos tabulares y gráfica del mínimo, máximo y promedio de temperatura global.

Por último, Mechanical provee una animación de los resultados (se pueden observar los controles de la animación encima de la gráfica en la Figura 4.3). Se encuentra adjunto un video[1] de una animación

[1] https://drive.google.com/file/d/1aZwLZi4RLOj_AyvM7Glw8mTypg-EFnFe/view?usp=sharing

que consta de 144 fotogramas, donde cada uno corresponde a los resultados de la simulación en un paso de tiempo determinado.

4.3.2. Comparación en puntos N, S y G

La evolución de temperatura de los puntos N, S y G se puede observar haciendo click sobre los elementos *Temperature Probe* correspondientes. Las gráficas generadas se pueden comparar directamente con las curvas de 1,5m de las gráficas de evolución de temperatura encontradas en la tesis original para cada punto (figuras 4.4, 4.5 y 4.6).

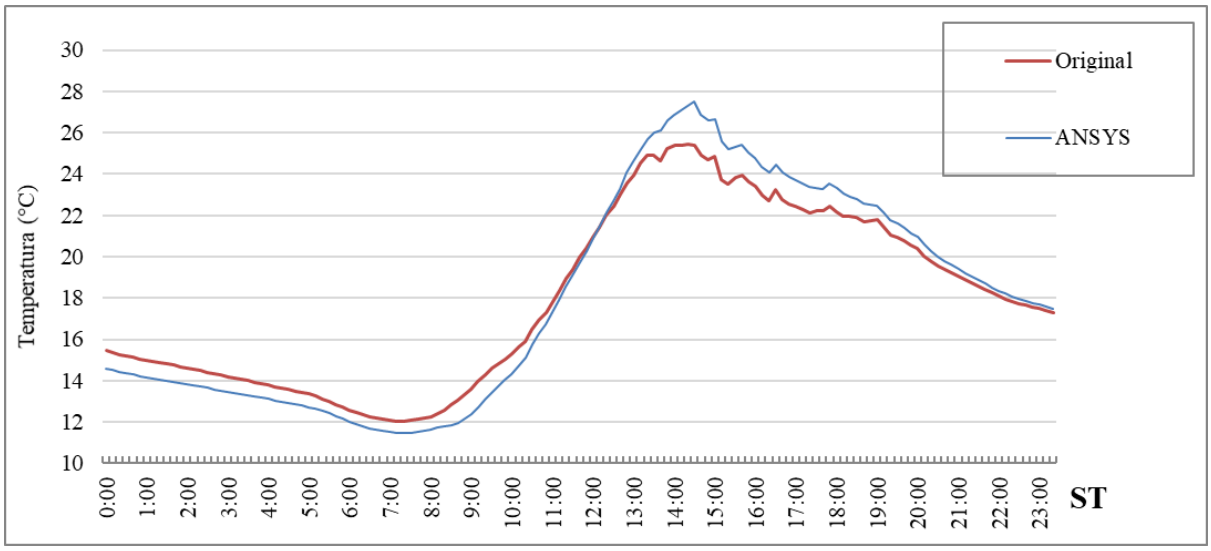


Figura 4.4: Evolución de temperatura del punto S.

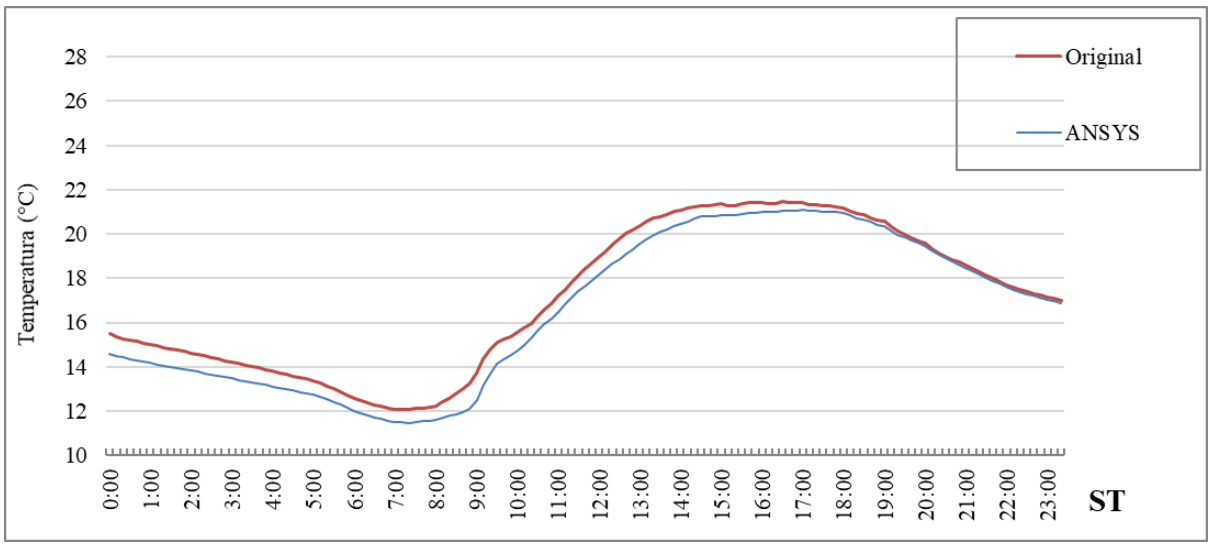


Figura 4.5: Evolución de temperatura del punto N.

Las curvas de temperatura obtenidas con Ansys denotan una gran similitud con las gráficas originales.

En el caso del punto S, la diferencia más notoria se da en el máximo de temperatura. Si bien la temperatura inicial en Ansys es menor (14,6 ºC vs. 15,489 ºC), la misma alcanza un máximo superior de aproximadamente 2 grados de diferencia con la temperatura original (27,485 ºC vs. 25.403 ºC) a las 14:30 horas.

En el punto N es donde se observan menos diferencias con la temperatura original. La temperatura inicial de Ansys es menor (14,6 ºC vs. 15,489 ºC) y se mantiene menor durante todo el día, aunque esta diferencia se hace más pequeña a lo largo del día hasta que son casi iguales sobre las 23:20 horas (16,892 ºC vs. 17,002 ºC).

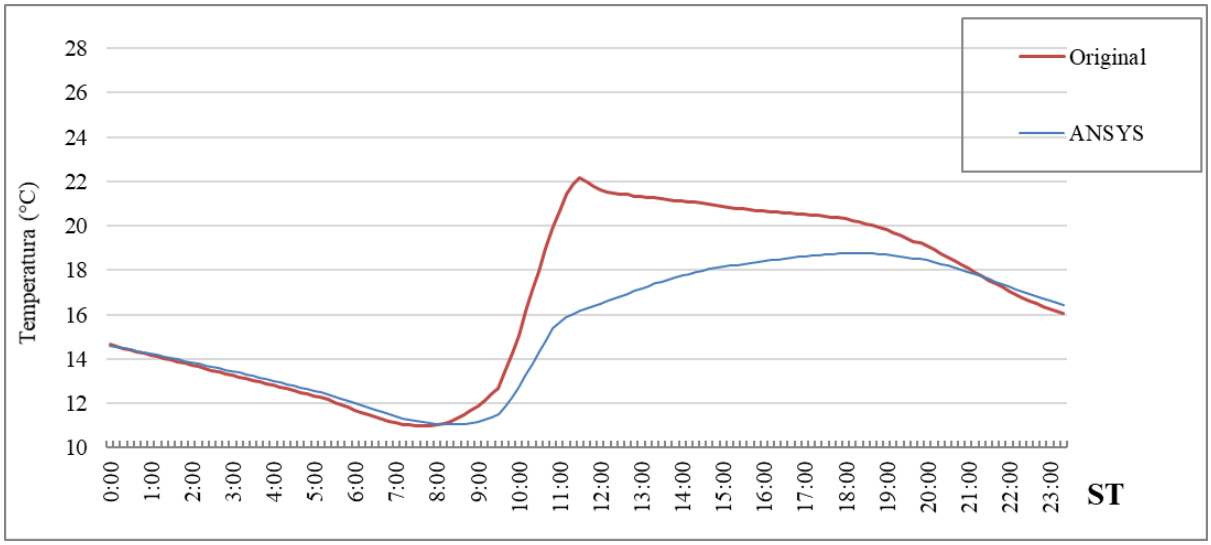


Figura 4.6: Evolución de temperatura del punto G.

En el punto G, la gráfica obtenida difiere en mayor medida de la gráfica original, en particular entre las 09:00 y las 20:00 horas. Sin embargo, como se verá en Subsección 4.3.3, la temperatura obtenida de la calle (y por lo tanto, del punto G) se acerca más a la temperatura real obtenida por la campaña de medición.

4.3.3. Termografías en Ansys EnSight

En la tesis original se realiza una comparación entre termografías reales y termografías simuladas. Estas termografías están representadas con una proyección en perspectiva, mientras que Mechanical sólamente provee imágenes con una proyección ortogonal (se observa esta proyección en la Figura 4.2). Para obtener termografías con proyección perspectiva, se utiliza Ansys EnSight[2]. Se debe ejecutar dicho programa y abrir el archivo *file.rth*, ubicado en la carpeta del proyecto creada automáticamente por Ansys, seleccionando la opción *Load all parts*.

A continuación, se seleccionan las partes cuyo nombre comience con "Part", excepto las correspondientes al domo, y se les asigna un método de coloración por temperatura haciendo click derecho en la selección y eligiendo las opciones *Color by*, *Select variable*, *Scalars*, *Temperature*. Luego, se personaliza la paleta de colores para que tenga los mismos colores que las termografías originales. Además, se ajusta el rango y la representación de los enteros de la leyenda.

Una vez configurado el entorno de EnSight, se ubica la cámara de forma manual y se toman imágenes para cada paso de la simulación. Es importante destacar que existe un desfazaje de dos horas entre la simulación y la tesis original, debido a discrepancias entre capítulos de la misma. Esto quiere decir que, por ejemplo, la termografía de las 6 PM se debe comparar con la termografía original de las 4 PM. Se puede observar la comparación de las termografías en la Figura 4.7.

A grandes rasgos, las termografías obtenidas exhiben una gran similitud con las originales, aunque requieren un análisis más detenido en algunos casos. Se evidencia nuevamente una subestimación de la temperatura de la calle en comparación con la simulación original. Sin embargo, resulta interesante notar que, al comparar con las termografías reales, se observa una mayor cercanía entre la temperatura simulada y la temperatura obtenida experimentalmente. Esto es particularmente evidente en las termografías correspondientes a las 16:00 horas.

Por otro lado, también se puede observar una sobreestimación de las temperaturas de la pared orientada al norte (derecha) en los resultados simulados originales con respecto a los resultados obtenidos con Ansys, particularmente en las termografías de las 13:00 horas. Nuevamente, los resultados de Ansys se acercan más a las termografías reales que a las termografías simuladas originales.

[2]Software de visualización de datos de simulación https://www.ansys.com/products/fluids/ansys-ensight

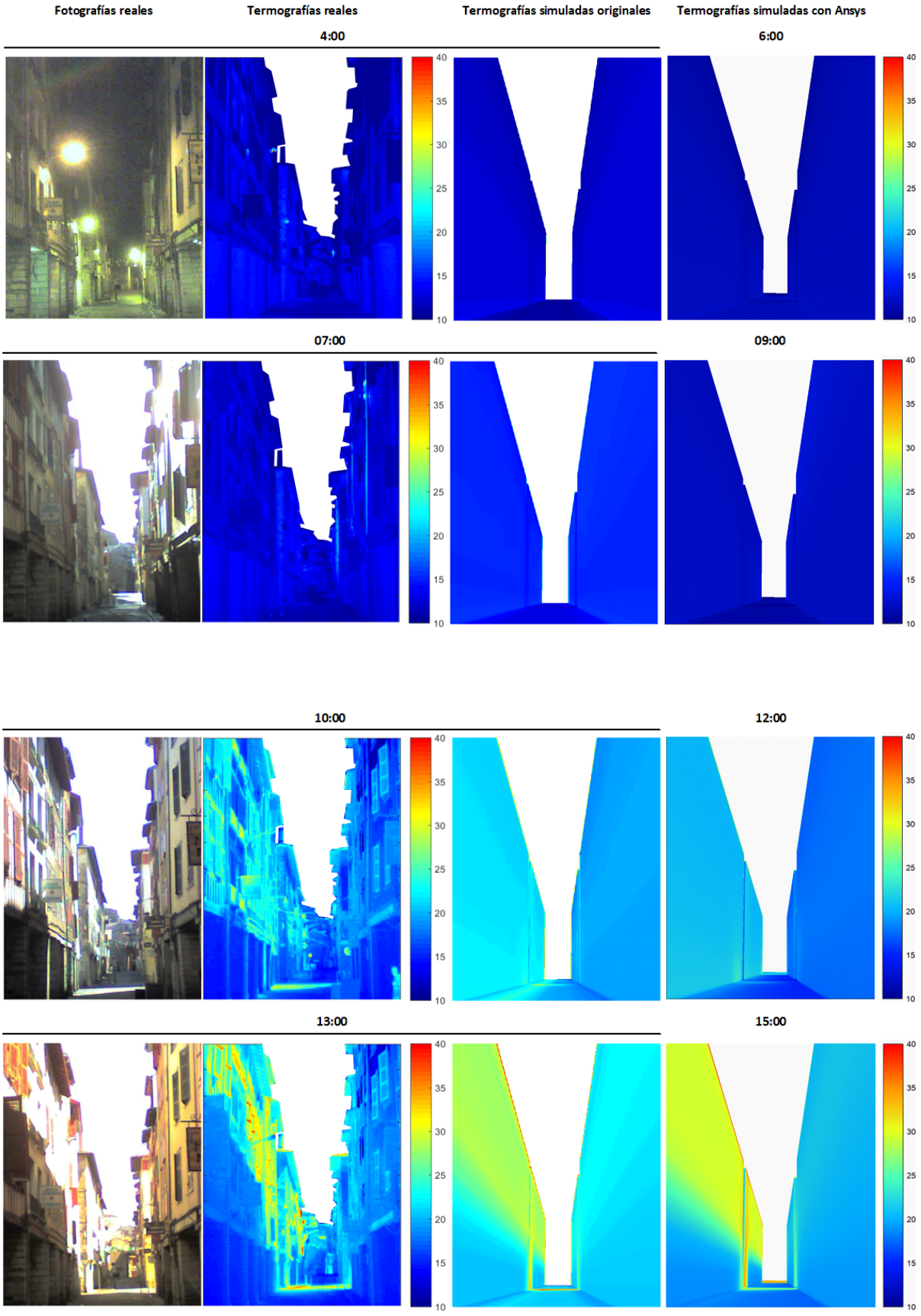


Figura 4.7: Comparación de termografías originales con termografías generadas con Ansys (Parte 2).

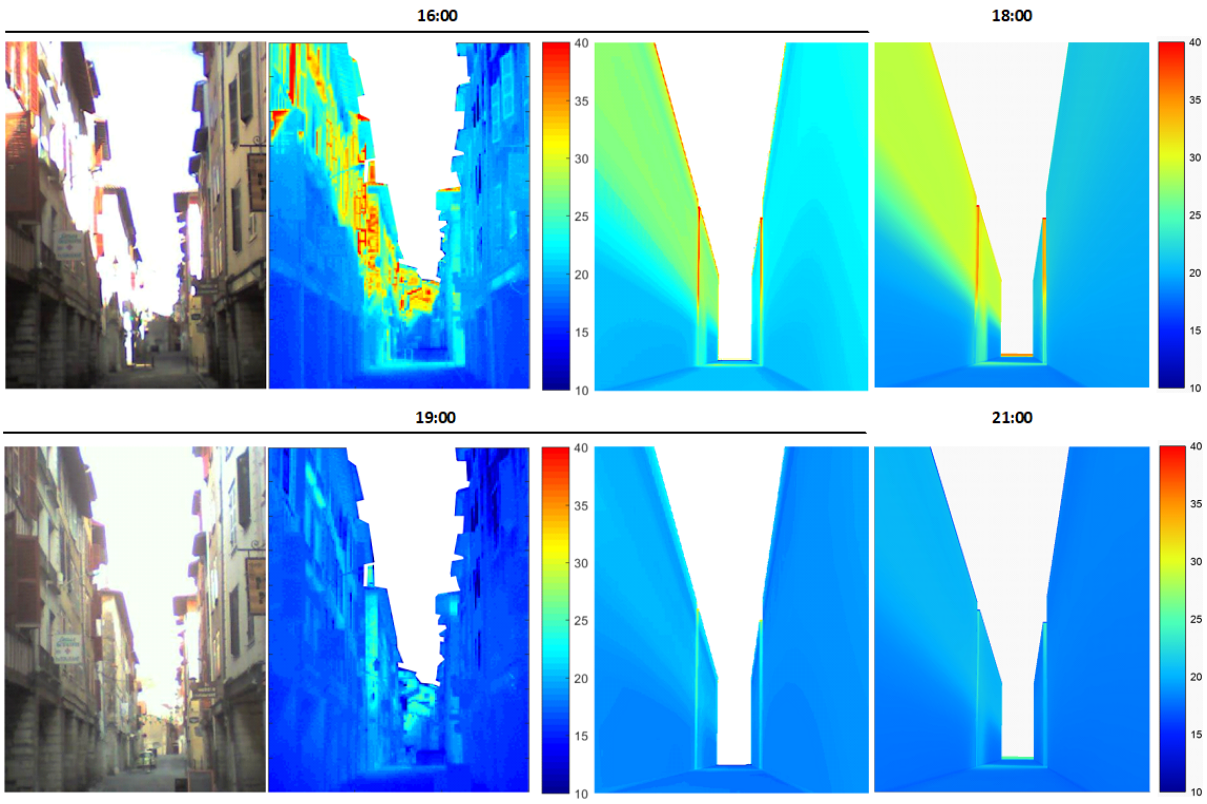


Figura 4.7: Comparación de termografías originales con termografías generadas con Ansys (Parte 3).

4.4. Datos de ejecución

En esta sección se presentan, en primer lugar, datos de ejecución de la simulación térmica del modelo construido en las secciones anteriores, tales como el uso de memoria, disco y el tiempo de ejecución.

En segundo lugar, se estudia el comportamiento de Ansys en relación a la cantidad de elementos del modelo, en particular, interesa incrementar la cantidad de elementos en los cálculos de radiación de onda larga y comprobar cuál es el límite soportado por Ansys. En estas simulaciones, sólamente se modela la radiación de onda larga.

Todas las simulaciones de esta sección se ejecutan en una computadora con las siguientes características:

- Procesador: Intel Core i7 7700k
- Memoria RAM: 16 GB
- Disco: SSD M.2 con 761 GB libres.
- Sistema operativo: Windows 10 64 bits.

Todos los datos de ejecución se obtienen del elemento *Solution Information* ubicado en el árbol del modelo. Este elemento, que se genera luego de resolver el sistema, provee varias formas de presentar la información de la solución (visibles en la opción *Solution Output* de su apartado de configuración), de las cuales se utilizaron *Solver Output* y *Solution Statistics* para obtener la información presentada en las siguientes subsecciones.

4.4.1. Ejecución del modelo completo

Se presentan los datos de ejecución del modelo completo en la Tabla 4.1. Comparando con los datos de ejecución de la simulación original en Cast3M (Figura 4.8), se observan mejoras significativas tanto en tiempo de ejecución como en consumo de memoria. La ejecución en Ansys duró aproximadamente seis veces menos que en Cast3M, con un consumo de memoria ocho veces menor para una cantidad similar de elementos (20298 contra 28635 en Cast3M). El sistema se resuelve por medio del solver *Direct Sparse Matrix Solver*, elegido automáticamente por Ansys en base a las características del sistema. Este solver está basado en la eliminación directa de ecuaciones de un sistema lineal, lo que requiere la factorización de una matriz inicial (dispersa) como producto de una matriz inferior y una matriz superior, que suelen ser simétricas (lo que se puede confirmar la salida del solver en *Solver Output*), por lo que solo se requiere la matriz triangular inferior. Si bien no es claro debido a la falta de documentación del módulo de radiación de ambos software, la gran diferencia en la utilización de memoria puede explicarse mediante la combinación de los siguientes puntos. Por un lado, la malla de elementos utilizada en ANSYS fue de aproximadamente 20 mil, mientras que en Cast3m fue de 28 mil. Dado que el consumo de memoria es de $O(N^2)$, esta diferencia no es menor. Por otro lado, probablemente ANSYS utilice precisión simple para almacenar la matriz de factores de vista, mientras que Cast3m utiliza precisión doble. Por último, si bien no está reportado en la tesis original, la lectura de disco es menor en Cast3m que en ANSYS, lo cual también puede implicar un mayor consumo máximo de memoria.

Cuadro 4.1: Datos de ejecución de la simulación térmica.

Tiempo	34m 58s
Memoria RAM	1.71 GB
Caras con radiación	20298
Nodos con radiación	20307
Total de elementos	178088
Total de nodos	123223
Lectura de disco	438.5 GB
Escritura en disco	115 GB

4.4.2. Análisis de tiempos de ejecución

Con el objetivo de analizar el uso de memoria de Ansys, se ejecutaron una serie de simulaciones con distintos tamaños de los elementos de la malla. El propósito de estas simulaciones es determinar la mayor

Task	Execution time (minutes)	Memory consumption (Gbytes)
Short-wave pre-computation	3.2	1.5
Cast3m pre-computation	16.9	12.8
Step	1.3	-
Total	202.1	14.2

Figura 4.8: Datos de ejecución de la simulación térmica original con Cast3M (Aguerre, 2020).

cantidad de elementos sobre los que se puede ejecutar una simulación con radiación de onda larga, antes de que la ejecución se detenga por falta de memoria. La radiación de onda larga se resuelve mediante el uso de una matriz cuadrática respecto a la cantidad de elementos, por lo que a medida que estos aumentan se requiere más espacio en memoria y/o disco.

Se simplifica el modelo, eliminando la convección y la radiación de onda corta. La temperatura inicial permanece en 14,6 ^{o}C y se establece la temperatura del domo en 30 ^{o}C.

Se presenta en la Tabla 4.2 la recopilación de datos de ejecución de las simulaciones. La tendencia general es el aumento de tiempo de ejecución, uso de memoria y uso de disco a medida que disminuye el tamaño objetivo de los elementos de la malla. Este resultado es esperado; disminuir el tamaño de los elementos resulta en una mayor cantidad de elementos y por consecuencia se requieren más cálculos para resolver el sistema.

Por otro lado, no se logró utilizar toda la memoria del sistema debido a que Ansys utiliza una cantidad significativa de espacio en disco para escribir los archivos de resultados mientras se resuelve el sistema. Para simulaciones con un tamaño de elementos igual o menor que 0,3m, las simulaciones finalizaron sin resolución ante una insuficiencia de espacio libre en disco.

Tamaño de la malla (m)	1,5	1,0	0,75	0,625	0,57
Tiempo (hh:mm:ss)	0:16:44	0:43:39	1:25:17	2:27:41	3:30:47
Memoria RAM (GB)	1,27	2,4	3,83	5,46	6,45
Caras con radiación	20298	35795	55254	75292	87147
Lectura de disco (GB)	275,1	534,9	622,8	1081,2	502,8
Escritura en disco (GB)	77,6	158.6	260,55	375,7	449,6
Modo	*In-core*	*In-core*	*In-core*	*In-core*	*In-core*

Cuadro 4.2: Datos de ejecución con distintos tamaños de malla (parte 1).

Tamaño de la malla (m)	0,53	0,5	0,4
Tiempo	5:39:06	5:10:06	9:35:53
Memoria RAM (GB)	7,26	7,97	12,07
Caras con radiación	99444	107623	158029
Lectura de disco (GB)	511,5	571,3	1113,2
Escritura en disco (GB)	534,4	601,3	1202,7
Modo	*Out-of-core*	*Out-of-core*	*Out-of-core*

Cuadro 4.2: Datos de ejecución con distintos tamaños de malla (parte 2).

Ansys provee diferentes estrategias de asignación de memoria, dentro de las cuales se encuentran *in-core* y *out-of-core*. El modo *in-core* aloja la matriz del sistema completamente en memoria, optimizando así el tiempo de ejecución. El modo *out-of-core*, por el contrario, sólamente mantiene submatrices denominadas matrices frontales en la memoria, pero comparte la matriz completa en el disco, causando una ejecución más lenta debido a una mayor cantidad de operaciones de lectura y escritura en dicho dispositivo.

El modo se asigna automáticamente en función de la memoria RAM disponible, utilizando *in-core* si existe memoria suficiente y *out-of-core* cuando no es posible tener la totalidad de la matriz de factores de forma en la memoria. Se observa que las simulaciones ejecutadas utilizan *out-of-core* para tamaños de malla iguales o menores que 0,53m.

Utilizando *out-of-core*, Ansys soporta mallas con tantos elementos como se desee, dado que exista espacio suficiente en disco.

Capítulo 5

Conclusiones

El método de elementos finitos es uno los métodos más usados en ingeniería y física para la aproximación de soluciones de ecuaciones diferenciales, como las ecuaciones de transferencia de calor. En la tesis de doctorado, que precede a este trabajo, el método de elementos finitos se utiliza para simular las transferencias de calor en un entorno urbano. Los resultados obtenidos en aquel trabajo, si bien fueron buenos, se obtuvieron mediante el uso del software Cast3M, el cual presentó limitaciones técnicas y dificultades respecto a su usabilidad.

El objetivo de esta tesis fue el de encontrar otro software de elementos finitos que permitiese superar las limitaciones de Cast3M y ejecutar una simulación térmica que produjera resultados similares. Tras realizar una investigación de varios software MEF, se concluyó que *Ansys* era la opción más adecuada.

Utilizando Ansys, se logró reconstruir la geometría original y posteriormente generar un mallado que cumpliese con los requisitos de conformalidad, estructuralidad y divisiones en capas. La malla generada es similar a la original, excepto en el suelo circundante de la ciudad, para el cual se contó con especificaciones de su diseño, pero no con imágenes para comparar los resultados lado a lado. Desde el punto de vista de los materiales, se logró recrear la asignación de distintos materiales para cada superficie del modelo, donde cada material se configuró con las propiedades térmicas establecidas en la tesis original.

En cuanto a las condiciones de borde, se logró cargar la totalidad de las condiciones del modelo original. Más allá de la conducción, que es el fenómeno base simulado por MEF, los intercambios por convección y por radiación de onda larga resultaron sencillos de modelar, reutilizando los datos tabulares de la tesis original. Fue posible cargar una temperatura inicial uniforme en el modelo, así como una temperatura fija para debajo del suelo. Ansys provee una funcionalidad para ejecutar código Python, la cual resultó sumamente efectiva para cargar la temperatura variable por espacio y tiempo del cielo. La carga de la radiación de onda corta, si bien supuso un desafío significativo inicialmente, resultó ser relativamente sencilla de llevar a cabo con el apoyo técnico de ESSS y la comunidad de Ansys. Esto también resalta la eficacia de la comunidad vinculada a este software.

La visualización de los resultados permitió la comparación directa con los resultados originales; sin embargo, fue necesario importar los resultados en Ansys EnSight para acceder a una vista en perspectiva de la ciudad y poder obtener termografías para la comparación. Los resultados en sí son muy similares a la simulación original, con diferencias las mayores diferencias en el entorno de los 2 ^{o}C. Incluso, algunas termografías indicaron temperaturas más próximas a las observadas en las mediciones experimentales que a las simuladas en la tesis de doctorado. Esto sugiere que los resultados producidos por simulaciones térmicas en Ansys podrían ser más representativos de la realidad en comparación con las transferencias de calor simuladas por Cast3M.

La tesis original reporta el elevado consumo de memoria RAM de Cast3M, mientras que Ansys logra un consumo de memoria cerca de diez veces menor que la simulación térmica original. Además, se registraron tiempos de ejecución hasta cinco veces menores que con Cast3M. Al analizar los límites del tamaño del modelo soportado por Ansys, se encontró una alta utilización del espacio en disco para escribir los archivos de resolución de la simulación. Esta utilización de disco, y la consecuente ralentización de las simulaciones al usar un medio con menor velocidad que la memoria RAM, evitó que se lograse determinar la máxima cantidad de elementos con la que Ansys es capaz de ejecutar una simulación térmica con radiación de onda larga en términos de consumo de memoria. Aun así, se consiguió simular modelos con hasta 158029 cuadriláteros, utilizando 12 GB de RAM. Cabe recordar que Cast3M utiliza 14 GB para simular un modelo con aproximadamente 28000 superficies.

El desarrollo del proyecto se realizó durante varios meses con la versión gratuita de Ansys, lo cual

dificultó el acceso a documentación detallada y soporte técnico. Si bien es posible encontrar material audiovisual, publicaciones en foros o instructivos de acceso libre, no fue sencillo encontrar material que esclareciese dudas de implementación, recurriendo muchas veces a la resolución de problemas por medio de prueba y error.

La interacción entre programas de Ansys con otros software externos también presentó dificultades. Por ejemplo, la exportación de la malla del modelo permite seleccionar solamente dos formatos. Además, con uno de estos formatos no fue posible exportar la malla completa, por lo que, efectivamente, sólo se contó con una opción. En cuanto a la visualización de los resultados térmicos, no se encontraron opciones fuera de Ansys que permitieran obtener imágenes con proyección en perspectiva. Incluso utilizando Ansys EnSight, no fue posible abrir los archivos de resultados de la simulación. La solución a este problema fue la instalación de Ansys Student 2023 R2 para poder abrir dichos archivos y obtener termografías en perspectiva.

La curva de aprendizaje de Ansys es significativa. El tiempo necesario para obtener un dominio en el manejo de la interfaz gráfica no debe ser un factor a subestimar, y se recomienda la utilización de recursos educativos que faciliten este proceso previo a cualquier implementación.

En conclusión, Ansys es un software con todas las capacidades necesarias para replicar el trabajo original, con un manejo de recursos más eficiente, y la posibilidad de ser instalado en una red para aprovechar el uso de uno o varios computadores en pos de escalar las simulaciones, tanto en tamaño de los modelos como en velocidad de las ejecuciones. No obstante, existe una curva de aprendizaje significativa y puede presentar dificultades a la hora de integrar sus resultados con otros tipos de software. Por otro lado, los resultados obtenidos representan mejor la realidad que Cast3M.

5.0.1. Trabajo futuro

La tesis original simula un problema *Steady State* para obtener la distribución de temperatura inicial del modelo, mientras que en este trabajo, por razones de tiempo, dicha temperatura se estableció de manera uniforme en 14,6 ^{o}C. De todos modos, es posible replicar este comportamiento mediante la creación de un sistema *Steady State* y la transferencia de los resultados al sistema *Transient*[1].

Otra posibilidad es llevar a cabo una simulación extendida de tres días, manteniendo las mismas condiciones de borde para cada día. Esta estrategia permite resultados más precisos para el tercer día, dado que el paso 0 equivale a una solución convergida del día anterior.

Además, se puede realizar la simulación con una geometría más detallada, similar a la metodología empleada en el trabajo original. Esto implica la importación en Ansys de un modelo desarrollado en otro software. Esta mejora en la definición geométrica puede conducir a una representación más precisa y realista del problema estudiado.

Asimismo, se puede realizar un análisis más profundo del módulo de radiación de onda larga y explorar las diferentes variantes ofrecidas. Por ejemplo, algunos programas permiten utilizar una malla más "gruesa" en la radiación de onda larga para reducir el tiempo de cálculo. Esta técnica, conocida como "solver multi-escala", implica mapear la malla gruesa a la fina en cada paso para continuar con los cálculos posteriores.

Otra alternativa es la utilización de otro software MEF, para comparar los resultados entre Ansys, Cast3M, y un tercer software, por ejemplo, COMSOL, explorado en la Sección 2.3. COMSOL tiene la capacidad de realizar análisis térmicos *Transient*, por lo que, resulta interesante como trabajo a futuro replicar el trabajo en dicho software y comparar los resultados obtenidos.

Por último, también interesa corregir errores que se detectaron en etapas tardías del proyecto. Se identificaron pequeñas imprecisiones en las mediciones de la geometría, atribuibles a la aplicación inconsistente de redondeo en algunas de las cifras. Si bien se reconoce la importancia de la precisión en las mediciones, realizar ajustes en estas cifras requeriría una considerable cantidad de tiempo, ya que implicaría la generación de una nueva malla, su exportación, depuración, cálculo de radiación de onda corta, y la realización de todos los pasos del proceso de carga de los flujos radiativos. Dada la magnitud de estos procedimientos, se tomó la decisión de mantener estas variaciones, considerándolas dentro de los límites aceptables para los objetivos y alcances de esta investigación.

[1] https://www.youtube.com/watch?v=KhVXU1WrZro&ab_channel=Tangibleengineering

Referencias

Aguerre, J. (2020). *Radiation techniques for urban thermal simulation with the finite element method* (Tesis de doctorado, Universidad de la República (Uruguay). Facultad de Ingeniería). Descargado de https://hdl.handle.net/20.500.12008/24344

Aguerre, J., Fernández, E., Besuievsky, G., y Beckers, B. (2017). Computing urban radiation: A sparse matrix approach. *Graphical Models*, *91*, 1-11. Descargado de https://www.sciencedirect.com/science/article/pii/S1524070317300486 doi: https://doi.org/10.1016/j.gmod.2017.05.002

ANSYS, I. (2023, 7). Ansys computing platform support. Descargado de https://www.ansys.com/content/dam/it-solutions/platform-support/2023-r2/ansys-platform-support-strategy-plans-july-2023.pdf?v=123

ANSYS inc. (2009, 11). ANSYS Workbench user's guide. Descargado de https://www.researchgate.net/file.PostFileLoader.html?id=5444928ed2fd647d5f8b46b7&assetKey=AS:272120486006786@1441889987511

ANSYS inc. (2010). Ansys meshing user's guide. Descargado de https://www.researchgate.net/profile/Mohamed-Mourad-Lafifi/post/How_to_remove_warpage_angle_and_violation_of_heuristic_criterion_warning_in_LS-Dyna/attachment/5d0c0713cfe4a7968dacddf1/AS%3A771994878496770%401561069330997/download/Meshing_Tutorial_Ans.sys.pdf

ANSYS inc. (2019). Thermal analysis guide. Descargado de https://www.academia.edu/7446594/Thermal_Analysis_Guide_Copyright_and_Trademark_Information

ANSYS inc. (2021, 7). Ansys Academic product reference table. Descargado de https://www.ansys.com/content/dam/product/academic/academic-product-reference-table-2021-r1.pdf

ANSYS inc. (2023). Mechanical user's guide. Descargado de https://ansyshelp.ansys.com/Views/Secured/corp/v232/en/pdf/Ansys_Mechanical_Users_Guide.pdf

Ayachit, U. (2019, 01). The paraview guide. Descargado de https://www.mn.uio.no/astro/english/services/it/help/visualization/paraview/paraviewguide-5.6.0.pdf

Byzyka, J., Rahman, M., y Chamberlain, D. (2017). A three-dimensional finite element analysis of temperature distribution in hot mix asphalt pothole repairs.. Descargado de https://bura.brunel.ac.uk/handle/2438/18941 (Depositing User : Byzyka, J)

COMSOL. (s.f.). *Led bulb heat transfer comsol.* Descargado 2023, de https://cdn.comsol.com/product-new/comsol-multiphysics/special/full/led-bulb-heat-transfer-comsol.webp

COMSOL. (2014, 10). Comsol multiphysics® software price list. Descargado de https://techfee.fau.edu/approvedproposals/Download.cfm?sid=514&pid=439

COMSOL. (2024). *System requirements for comsol multiphysics® version 6.2.* COMSOL Website. Descargado de https://www.comsol.com/system-requirements (Accedido: 5 de marzo de 2024)

Crowley, C. (2023, 08). *A crash course in ansys mechanical scripting.* PADT Inc. Website. Descargado de https://www.padtinc.com/2023/08/08/crash-course-ansys-mechanical-scripting/ (Accedido: 7 de marzo de 2024)

Cuesta, B., y González, S. (2016, 03). Introducción a docker. Descargado de https://www.rediris.es/tecniris/archie/doc/TECNIRIS47-1b.pdf

Di Paola, F. (2023, 3). Starting with cast3m thermomechanical calculations. Descargado de https://www-cast3m.cea.fr/html/formations/Starting_with_Cast3M.pdf

Elmekawy, A. N. (2018, 11). Introduction to ansys meshing - module 01: Core skills. Descargado de https://drahmednagib.com/CAD_2018/Lecture_2_Meshing_1.pdf

Geuzaine, C., y Remacle, J.-F. (2009). Gmsh: A 3-d finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, *79*(11), 1309-1331. Descargado de https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.2579 doi: https://doi.org/10.1002/nme.2579

Goral, C. M., Torrance, K. E., Greenberg, D. P., y Battaile, B. (1984). Modeling the interaction of light between diffuse surfaces. Descargado de https://dl.acm.org/doi/pdf/10.1145/964965.808601

Gu, H., y Hunt, J. (2006, 10). Two-dimensional finite element heat transfer model of softwood. part ii. macrostructural effects. *Wood and fiber science: journal of the Society of Wood Science and Technology*, *38*.

Guo, Y., Li, Z., Huang, S., Liu, M., y Wang, K. (2021). A new neutronics-thermal-mechanics multi-physics coupling method for heat pipe cooled reactor based on rmc and openfoam. *Progress in Nuclear Energy*, *139*, 103842. Descargado de https://www.sciencedirect.com/science/article/pii/S0149197021002055 doi: https://doi.org/10.1016/j.pnucene.2021.103842

Hosen, M. S., Ahmmed, S., y Dekkati, S. (2019). Mastering 3d modeling in blender: From novice to pro. *ABC Research Alert*, *7*(3), 169–180.

Hunt, J., y Gu, H. (2006, 10). Two-dimensional finite element heat transfer model of softwood. part i. effective thermal conductivity. *Wood and Fiber Science*, *38*.

Lee, H.-H. (2023). *Finite element simulations with ansys workbench 2023: Theory, applications, case studies*. SDC publications.

Mayboudi, L. (2020). *Comsol heat transfer models*. Mercury Learning and Information. Descargado de https://books.google.com.uy/books?id=2RKvswEACAAJ

OpenCFD, L. (2024). *System requirements*. OpenFOAM Website. Descargado de https://www.openfoam.com/documentation/system-requirements (Accedido: 6 de marzo de 2024)

Pio, D. M., Rafael, C. F., Villela, P. C. S., da Silva, G. A. L., Ferro, M., y Guedes, M. J. M. (s.f.). Large eddy simulation of a lynx helicopter landing in the barroso corvette ship.

Tripathi, P. M., Prakash, S., Singh, R. K., y Dwivedi, S. K. (2014). Thermal analysis on cylinder head of si engine using fem.. Descargado de https://api.semanticscholar.org/CorpusID:124310662

United Nations. (2020). *Global state of metropolis 2020 - population data booklet* (Data Booklet ST/ESA/SER.A/417). United Nations Human Settlements Programme.

Apéndice A

Código de la carga de temperaturas del domo

```
import csv
import math
from collections import OrderedDict

csv_path = 'C:\path\ejemplo\\\BayonneTestCase.csv'
analysis = DataModel.GetObjectsByName("Transient Thermal")[0]
faces_ids = DataModel.GetObjectsByName("FacesDomo")[0].Ids
dome_center = DataModel.GetObjectsByName("CentroDomo")[0].Entities[0]
sky_temperatures = OrderedDict()
element_temperatures = {}

def add_to_sky_temperatures_dict(csv_row):
    sky_temperatures[csv_row['Official time']] = {}
    for angle in range(0, 91, 10):
        sky_temperatures[csv_row['Official time']][angle] =
            round(float(csv_row[str(angle)]) - 273.15, 2)

# se completa sky_temperatures con los datos de temperature obtenidos desde el
# archivo .csv
with open(csv_path, mode='r') as csv_file:
    csv_reader = csv.DictReader(csv_file)
    for row in csv_reader:
        add_to_sky_temperatures_dict(row)

def get_mesh_element(id):
    return ExtAPI.DataModel.MeshDataByName("Global").ElementById(id)

def calculate_zenith_angle(node):
    # el plano x z es el suelo. la componente en y de todos los vectores denota su
    # altura en la ciudad
    x0 = dome_center.X
    y0 = dome_center.Y
    z0 = dome_center.Z
    x1 = node[0]
    y1 = node[1]
    z1 = node[2]

    squared_x_diff = pow(x0 - x1, 2)
    squared_y_diff = pow(y0 - y1, 2)
    squared_y = pow (y0, 2)
```

```
41      squared_z_diff = pow(z0 - z1, 2)
42
43      # se utiliza cos(alpha) = cat_ady / h siendo alpha = 90 - angulo_cenital
44      # (x0, y0, z0) es el centro del domo. Estas coordenadas NO son iguales a
45      # (0, 0, 0)
46
47      # para calcular el cateto adyacente, se toma la diferencia de los vectores
48      # (x1, 0, z1) y (x0, y0, zo) para calcular la hipotenusa, se toma la diferencia
49      # de los vectores (x1, y1, z1) y (x0, y0, z0)
50      adjacent_leg = math.sqrt(squared_x_diff + squared_y + squared_z_diff)
51      hypotenuse = math.sqrt(squared_x_diff + squared_y_diff + squared_z_diff)
52      alpha =  math.degrees(math.acos(adjacent_leg / hypotenuse))
53      return 90 - alpha
54
55  def linear_interpolation(x1, y1, x2, y2, x):
56      return round(y1 + (((y2 - y1) / (x2 - x1)) * (x - x1)), 2)
57
58  def calculate_angle_temp(zenith_angle, time):
59      if (zenith_angle % 10 == 0):
60          return  sky_temperatures["0:00"][round(zenith_angle)]
61      else:
62          smaller_10_multiple = (zenith_angle // 10) * 10
63          larger_10_multiple = ((zenith_angle // 10) * 10) + 10
64          smaller_10_multiple_temp = sky_temperatures[time][round(smaller_10_multiple)]
65          larger_10_multiple_temp = sky_temperatures[time][round(larger_10_multiple)]
66          return linear_interpolation(
67              smaller_10_multiple,
68              smaller_10_multiple_temp,
69              larger_10_multiple,
70              larger_10_multiple_temp,
71              zenith_angle
72          )
73
74  def time_to_input(time):
75      seconds = sum(x * int(t) for x, t in zip([3600, 60], time.split(":")))
76      return Quantity('{0} [sec]'.format(seconds))
77
78  def add_model_temperatures():
79      for mesh_id in element_temperatures.keys():
80          # agregar elemento de temperatura y vincularlo con el parche actual
81          temperature_element = analysis.AddTemperature()
82          selection =
83              ExtAPI.SelectionManager.CreateSelectionInfo(SelectionTypeEnum
84              .MeshElementFaces)
85          selection.Ids = [int(mesh_id)]
86          selection.ElementFaceIndices = [0]
87          temperature_element.Location = selection
88          # cargar valores de tiempo
89          time_inputs = map(time_to_input, element_temperatures[mesh_id].keys())
90          temperature_element.Magnitude.Inputs[0].DiscreteValues = time_inputs
91          # cargar valores de temperatura
92          temp_outputs = []
93          for time in element_temperatures[mesh_id].keys():
94              temp = Quantity('{0} [C]'.format(element_temperatures[mesh_id][time]))
95              temp_outputs.append(temp)
96          temperature_element.Magnitude.Output.DiscreteValues = temp_outputs
97
```

```
98
99  for id in faces_ids:
100     face_element = get_mesh_element(id)
101     element_temperatures[str(id)] = OrderedDict()
102     for time in sky_temperatures.keys():
103         temperature_sum = 0
104         temperature_face_element =
105             calculate_angle_temp(calculate_zenith_angle(face_element.Centroid), time)
106         element_temperatures[str(id)][time] = temperature_face_element
107
108 add_model_temperatures()
```

Apéndice B

Código del mapeo de caras con vértices

```
import csv

faces = []
vertices = []

with open('mesh.obj', newline='') as csvfile1:
    reader1 = csv.reader(csvfile1, delimiter=' ', quotechar='"')
    for row1 in reader1:
        if row1[0] == 'f':
            faces.append(row1)
        if row1[0] == 'v':
            vertices.append(row1)

vertices_by_faces = []

for row in faces:
    vertex_by_face = [" ".join(row)]
    for entry in row:
        if entry != 'f':
            vertex_by_face.append(vertices[int(entry)-1][1])
            vertex_by_face.append(vertices[int(entry)-1][2])
            vertex_by_face.append(vertices[int(entry)-1][3])
    vertices_by_faces.append(vertex_by_face)

with open('faces_vertices.csv', mode='w', newline='') as file:
    writer = csv.writer(file)
    for row in vertices_by_faces:
        writer.writerow(row)
```

Apéndice C

Código de la carga de radiaciones de onda corta

```
import csv
import math
from collections import OrderedDict
from functools import partial
from itertools import islice

faces_vertices_csv_path = 'C:\path\csv\\faces_vertices.csv'
radiations_path = 'C:\path\radiaciones'

roof_elements_selection = DataModel.GetObjectsByName("RoofFaces")[0]

# los siguientes dos arreglos se relacionan por indice del arreglo, es decir que por
# ejemplo el par (roofElementsIds[0], roofElementsFaceIndices[0]) identifica una cara
roof_elements_ids = roof_elements_selection.Ids
roof_elements_face_indices = roof_elements_selection.ElementFaceIndices
roof_faces_vertices = []
roof_radiations = OrderedDict()
roof_csv_offset = 8640
street_csv_offset = 4072

all_fluxes = []
time_inputs = []

analysis = DataModel.GetObjectsByName("Transient Thermal")[0]

for i in range(1, 145):
    nombre_archivo = radiations_path + '\\flux{}.csv'.format(i)
    with open(nombre_archivo, 'r') as flux_csv:
        csv_reader = csv.reader(flux_csv)
        flux_data = [float(csv_row[0]) for csv_row in csv_reader]
        all_fluxes.append(flux_data)

def make_face_data_from_csv_row(csv_row):
    face_data = OrderedDict()
    node1 = {
        "X": float(csv_row["x1"]),
        "Y": float(csv_row["y1"]),
        "Z": float(csv_row["z1"]),
    }
    node2 = {
```

```
        "X": float(csv_row["x2"]),
        "Y": float(csv_row["y2"]),
        "Z": float(csv_row["z2"]),
    }
    node3 = {
        "X": float(csv_row["x3"]),
        "Y": float(csv_row["y3"]),
        "Z": float(csv_row["z3"]),
    }
    face_data['nodes'] = [node1, node2, node3]
    if csv_row['x4'] and csv_row['y4'] and csv_row['z4']:
        node = {
            "X": float(csv_row["x4"]),
            "Y": float(csv_row["y4"]),
            "Z": float(csv_row["z4"]),
        }
        face_data['nodes'].append(node)
    return face_data

with open(faces_vertices_csv_path, mode='r') as csv_file:
    csv_reader = csv.DictReader(csv_file)
    line_count = 0
    for csv_row in islice(csv_reader, roof_csv_offset, roof_csv_offset +
        street_csv_offset):
        face_data = make_face_data_from_csv_row(csv_row)
        roof_faces_vertices.append(face_data)

def get_mesh_element(id):
    return ExtAPI.DataModel.MeshDataByName("Global").ElementById(id)

def nearly_equal(a, b, tol=1e-3):
    return abs(a - b) <= tol

def nodes_include_face_node(nodes, face_node):
    for node in nodes:
        if nearly_equal(round(node.X, 3),round(face_node['X'], 3)) and
        nearly_equal(round(node.Y, 3), round(face_node['Y'], 3)) and
        nearly_equal(round(node.Z, 3), round(face_node['Z'], 3)):
            return True
    return False

def nodes_include_face_nodes(nodes, face):
    for face_node in face['nodes']:
        if not nodes_include_face_node(nodes, face_node):
            return False
    return True

def findIndex(flist, func):
    for i,v in enumerate(flist):
        if func(face=v):
            return i
    return -1

def find_face_csv_index(nodes, surface_type):
    if surface_type == 'roof':
        csv_index =
            findIndex(roof_faces_vertices, partial(nodes_include_face_nodes,
```

```
            nodes=nodes))
        if (csv_index) > -1:
            return csv_index + roof_csv_offset
    elif surface_type == 'wall':
        # ...
    else:
        return -1

def load_surface_radiation(surface_elements_ids, surface_elements_face_indices,
    surface_type):
    for (element_id, face_index) in zip(surface_elements_ids,
        surface_elements_face_indices):
        face_id = "{0}_{1}".format(element_id, face_index)
        element = get_mesh_element(element_id)
        nodes = element.CornerNodes
        csv_index = find_face_csv_index(nodes, surface_type)
        if (csv_index > -1):
            if (surface_type == 'roof'):
                roof_radiations[face_id] = make_radiation_steps(csv_index)
            elif surface_type == 'wall':
                # ...
        else:
            print(face_id)

def generate_time_inputs():
    for step in range(0, 144):
        time_inputs.append(Quantity('{0} [sec]'.format(step * 60 * 10)))

def create_radiations(surface_radiations):
    for face_id in surface_radiations.keys():
        heat_flux = analysis.AddHeatFlux()
        selection =
            ExtAPI.SelectionManager.CreateSelectionInfo(SelectionTypeEnum
            .MeshElementFaces)
        split = face_id.split("_")
        selection.Ids = [int(split[0])]
        selection.ElementFaceIndices = [int(split[1])]
        heat_flux.Location = selection
        heat_flux.Magnitude.Inputs[0].DiscreteValues = time_inputs
        heat_flux_outputs = []
        radiation = surface_radiations[face_id]
        for time in radiation.keys():
            flux = Quantity('{0} [W m^-1 m^-1]'.format(radiation[time]))
            heat_flux_outputs.append(flux)
        heat_flux.Magnitude.Output.DiscreteValues = heat_flux_outputs

def make_radiation_steps(csv_index):
    radiation_object = OrderedDict()
    for i in range(0, 144):
        radiation_object[i * 60 * 10] = all_fluxes[i][csv_index]
    return radiation_object

generate_time_inputs()
load_surface_radiation(roof_elements_ids, roof_elements_face_indices, 'roof')

create_radiations(roof_radiations)
```

Apéndice D

Código del reordenamiento de caras

```python
import csv

faces_vertices_csv_path = r'C:\path\a\csv\\faces_vertices.csv'
reordered_faces_vertices_csv_path = r'C:\path\a\csv\\reordered_faces_vertices.csv'

with open(faces_vertices_csv_path, 'r') as file:
    reader = csv.reader(file)
    rows = list(reader)

tris = []
quads = []
for row in rows:
    if len(row[0].split()) == 5:
        quads.append(row)
    else:
        tris.append(row)

sorted_tris = sorted(tris, key=lambda x: rows.index(x))
reordered_rows = sorted_tris + quads

with open(reordered_faces_vertices_csv_path, 'w', newline='') as file:
    writer = csv.writer(file)
    writer.writerows(reordered_rows)
```

Apéndice E

Código de unificación de archivos CSV

```
import csv

all_fluxes = []
faces_vertices = []
faces_centroids = []
from collections import OrderedDict

faces_vertices_csv_path = r'C:\path\ejemplo\\reordered_faces_vertices.csv'

def make_face_data_from_csv_row(csv_row):
    face_data = OrderedDict()
    node1 = {
        "X": float(csv_row["x1"]),
        "Y": float(csv_row["y1"]),
        "Z": float(csv_row["z1"]),
    }
    node2 = {
        "X": float(csv_row["x2"]),
        "Y": float(csv_row["y2"]),
        "Z": float(csv_row["z2"]),
    }
    node3 = {
        "X": float(csv_row["x3"]),
        "Y": float(csv_row["y3"]),
        "Z": float(csv_row["z3"]),
    }
    face_data['nodes'] = [node1, node2, node3]
    if csv_row['x4'] and csv_row['y4'] and csv_row['z4']:
        node = {
            "X": float(csv_row["x4"]),
            "Y": float(csv_row["y4"]),
            "Z": float(csv_row["z4"]),
        }
        face_data['nodes'].append(node)
    return face_data

# Lectura de CSVs de radiacion
for i in range(1, 145):
    nombre_archivo = r'C:path\ejemplo\radiacion\\flux{}.csv'.format(i)
    with open(nombre_archivo, 'r') as flux_csv:
```

```
        csv_reader = csv.reader(flux_csv)
        flux_data = [float(csv_row[0]) for csv_row in csv_reader]
        all_fluxes.append(flux_data)

# Lectura de faces_vertices para obtener informacion geometrica de los cuadrilateros
with open(faces_vertices_csv_path, mode='r') as csv_file:
    csv_reader = csv.DictReader(csv_file)
    for csv_row in csv_reader:
        faces_vertices.append(make_face_data_from_csv_row(csv_row))

# Calculo de centroides
for face in faces_vertices:
    x = 0
    y = 0
    z = 0
    for node in face['nodes']:
        x = x + node['X']
        y = y + node['Y']
        z = z + node['Z']
    x = x / len(face['nodes'])
    y = y / len(face['nodes'])
    z = z / len(face['nodes'])
    faces_centroids.append({
        "X": round(x, 3),
        "Y": round(y, 3),
        "Z": round(z, 3),
    })

all_fluxes_csv_filename = r'C:\path\ejemplo\\all_fluxes.csv'

# Trasposicion de all_fluxes para tener los elementos como filas
# y los flujos radiativos como columnas
transposed_fluxes = list(map(list, zip(*all_fluxes)))

# Agregar centroides
for i, row in enumerate(transposed_fluxes):
    faces_data = faces_centroids[i]
    row[:0] = [faces_data['X'], faces_data['Y'], faces_data['Z']]

# Agregar encabezados "x", "y", "z", "hflux1", ..., "hflux144"
headers =
    ['x', 'y', 'z'] + ['hflux' + str(i) for i in range(1,
    len(transposed_fluxes[0])+1)]
transposed_fluxes.insert(0, headers)

with open(all_fluxes_csv_filename, 'w') as csvfile:
    writer = csv.writer(csvfile, lineterminator='\n')
    writer.writerows(transposed_fluxes)
```