Capítulo 3

Elaboração

Neste capítulo é apresentada a elaboração do programa, constituído pelo desenvolvimento teórico, modelagem numérica, identificação de pacotes e algoritmos adicionais relacionados ao *software* desenvolvido.

3.1 Análise de domínio

A análise de domínio, como parte da elaboração, tem o objetivo de entender e delimitar conceitos fundamentais, sob os quais o *software* é construído [BUENO 2003].

O presente trabalho envolve quatro conceitos fundamentais:

1. Transferência de calor:

   Transferência de calor é uma das áreas clássicas de fenomenologia da física. É responsável por tratar das três formas possíveis de transferência de calor: condução, convecção e radiação. Este projeto trata especificamente da condução de calor.

   A condução só pode ocorrer em meio material (fluidos ou sólidos), e sem que haja movimento do próprio meio, característica da convecção ([NUSSENZVEIG 2014]).

2. Modelagem numérica:

   Métodos numéricos são algoritmos desenvolvidos com ajuda da matemática para resolver problemas complexos da natureza. São utilizados quando uma solução analítica é difícil de ser obtida, ou com condições de contorno complexas.

3. Programação orientada ao objeto com C++:

   O paradigma orientado ao objeto é um dos principais paradigmas da programação, utilizado especialmente na construção de grandes *softwares* devido à portabilidade, organização e delimitação de assuntos. C++ é uma das linguagens mais utilizadas atualmente, por ser mais rápida com muito suporte e por permitir a orientação ao objeto.

4. Renderização 3D:

   Renderização 3D é uma área com grande ascensão na indústria de jogos e *softwares* de engenharia profissional, torna prático que usuários consigam visualizar o objeto sob qualquer ótica. É necessário a utilização de vários conceitos da álgebra linear.

3.2 Formulação modelos teóricos

Apresenta-se a seguir os termos e unidades utilizados, a formulação teórica, condições de contorno, demonstrações e considerações sobre condutividade térmica variável.

3.2.1 Termos e Unidades

Os principais termos e suas unidades utilizadas neste projeto estão listadas abaixo:

- Dados relativos ao material:
  - c_p - calor específico a pressão constante $[J/kg \cdot K]$;
  - k - condutividade térmica $[W/m \cdot K]$;
  - ρ - massa específica $[kg/m^3]$.
- Dados relativos ao objeto
  - $\Delta x, \Delta y$ - distância entre os centros dos blocos, valor inicial: 1px=0.0026 m $[m]$;
  - Δz - distância entre perfis, valor inicial: 0.05 m $[m]$;
  - T - temperatura no nodo $[K]$;
- Variáveis usadas na simulação:
  - i - posição do nodo em relação ao eixo x;
  - k - posição do nodo em relação ao eixo y;
  - g - qual $grid$/perfil está sendo analisado;
  - t - tempo atual;
  - n - índice do passo de tempo;
  - ν - número da iteração.

3.2.2 Formulação teórica

A taxa de transferência de calor foi modelado empiricamente por Jean B. J. Fourier em 1822 ([FOURIER 1822]). Posteriormente a teoria foi aprimorada até chegar na equação geral da difusão de calor Eq. (3.1). O desenvolvimento teórico para chegar nesta equação, pode ser acompanhado detalhadamente no [Incropera 2008].

Portanto, a seguir é apresentada a equação geral da difusão de calor em meios tridimensionais em coordenadas cartesianascartesianos:

$$\frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right)+\frac{\partial}{\partial y}\left(k\frac{\partial T}{\partial y}\right)+\frac{\partial}{\partial z}\left(k\frac{\partial T}{\partial z}\right)=\rho c_p\frac{\partial T}{\partial t} \tag{3.1}$$

Para resolver a equação geral da difusão térmica, será utilizado o método implícito de diferenças finitas BTCS, com malha em formato bloco centrado.

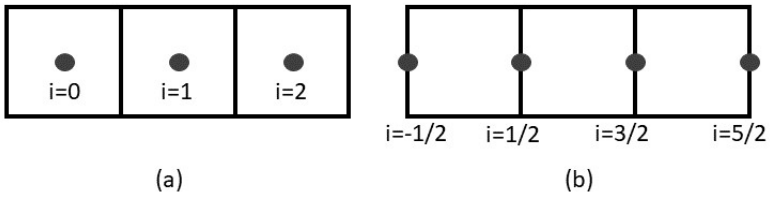


Figura 3.1: Tipos de malha, (a) bloco-centrado e (b) ponto-distribuído

Conforme a Figura 3.1, existem dois tipos principais de malha: bloco-centrado, onde os pontos analisados estão nos centros de cada bloco, e ponto-distribuído, onde os pontos analisados estão nas fronteiras de cada bloco.

Com esses conceitos em mente, a equação geral é modelada por diferenças finitas, mantendo a condutividade térmica dentro da derivada espacial. Inicialmente, será modelado somente a derivada externa:

$$\frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right)=\frac{\left(k\frac{\partial T}{\partial x}\right)_{i-\frac{1}{2},j,k}-\left(k\frac{\partial T}{\partial x}\right)_{i+\frac{1}{2},j,k}}{\Delta x} \tag{3.2}$$

Modelando as derivadas internas:

$$\frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right)=\frac{k_{i-\frac{1}{2},j,k}\left(\frac{T_{i-1,j,k}-T_{i,j,k}}{\Delta x}\right)-k_{i+\frac{1}{2},j,k}\left(\frac{T_{i,j,k}-T_{i+1,j,k}}{\Delta x}\right)}{\Delta x} \tag{3.3}$$

Com um pouco de álgebra:

$$\frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right)=\frac{k_{i-\frac{1}{2},j,k}\left(T_{i-1,j,k}-T_{i,j,k}\right)-k_{i+\frac{1}{2},j,k}\left(T_{i,j,k}-T_{i+1,j,k}\right)}{\Delta x^2} \tag{3.4}$$

Chegando na modelagem final para a derivada espacial ao longo do x:

$$\frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right)=\frac{k_{i-\frac{1}{2},j,k}T_{i-1,j,k}-\left(k_{i-\frac{1}{2},j,k}+k_{i+\frac{1}{2},j,k}\right)T_{i,j,k}+k_{i+\frac{1}{2},j,k}T_{i+1,j,k}}{\Delta x^2} \tag{3.5}$$

Como as outras dimensões são simétricas:

$$\frac{\partial}{\partial y}\left(k\frac{\partial T}{\partial y}\right) = \frac{k_{i,j-\frac{1}{2},k}T_{i,j-1,k} - \left(k_{i,j-\frac{1}{2},k} + k_{i,j+\frac{1}{2},k}\right)T_{i,j,k} + k_{i,j+\frac{1}{2},k}T_{i,j+1,k}}{\Delta y^2} \tag{3.6}$$

$$\frac{\partial}{\partial z}\left(k\frac{\partial T}{\partial z}\right) = \frac{k_{i,j,k-\frac{1}{2}}T_{i,j,k-1} - \left(k_{i,j,k-\frac{1}{2}} + k_{i,j,k+\frac{1}{2}}\right)T_{i,j,k} + k_{i,j,k+\frac{1}{2}}T_{i,j,k+1}}{\Delta z^2} \tag{3.7}$$

A derivada temporal é atrasada no tempo:

$$\frac{\partial T}{\partial t} = \frac{T^n_{i,j,k} - T^{n-1}_{i,j,k}}{\Delta t} \tag{3.8}$$

Substituindo as diferenças finitas na equação geral:

$$\begin{gathered}\frac{k_{i-\frac{1}{2},j,k}T_{i-1,j,k} - \left(k_{i-\frac{1}{2},j,k} + k_{i+\frac{1}{2},j,k}\right)T_{i,j,k} + k_{i+\frac{1}{2},j,k}T_{i+1,j,k}}{\Delta x^2} + \\ \frac{k_{i,j-\frac{1}{2},k}T_{i,j-1,k} - \left(k_{i,j-\frac{1}{2},k} + k_{i,j+\frac{1}{2},k}\right)T_{i,j,k} + k_{i,j+\frac{1}{2},k}T_{i,j+1,k}}{\Delta y^2} + \\ \frac{k_{i,j,k-\frac{1}{2}}T_{i,j,k-1} - \left(k_{i,j,k-\frac{1}{2}} + k_{i,j,k+\frac{1}{2}}\right)T_{i,j,k} + k_{i,j,k+\frac{1}{2}}T_{i,j,k+1}}{\Delta z^2} = \\ \frac{T^n_{i,j,k} - T^{n-1}_{i,j,k}}{\Delta t}\end{gathered} \tag{3.9}$$

Onde a malha é homogênea na superfície, mas não entre os perfis, ou seja, $\Delta x = \Delta y \neq \Delta z$. Substituindo:

$$\begin{gathered}\frac{k_{i-\frac{1}{2},j,k}T_{i-1,j,k} - \left(k_{i-\frac{1}{2},j,k} + k_{i+\frac{1}{2},j,k}\right)T_{i,j,k} + k_{i+\frac{1}{2},j,k}T_{i+1,j,k}}{\Delta x^2} + \\ \frac{k_{i,j-\frac{1}{2},k}T_{i,j-1,k} - \left(k_{i,j-\frac{1}{2},k} + k_{i,j+\frac{1}{2},k}\right)T_{i,j,k} + k_{i,j+\frac{1}{2},k}T_{i,j+1,k}}{\Delta x^2} + \\ \frac{k_{i,j,k-\frac{1}{2}}T_{i,j,k-1} - \left(k_{i,j,k-\frac{1}{2}} + k_{i,j,k+\frac{1}{2}}\right)T_{i,j,k} + k_{i,j,k+\frac{1}{2}}T_{i,j,k+1}}{\Delta z^2} = \\ c_p\rho\frac{T^n_{i,j,k} - T^{n-1}_{i,j,k}}{\Delta t}\end{gathered} \tag{3.10}$$

Multiplicando pelo múltiplo comum:

$$\begin{gathered}\Delta z^2\left(k^{n+1}_{i-\frac{1}{2},j,k}T^{n+1}_{i-1,j,k} - \left(k^{n+1}_{i-\frac{1}{2},j,k} + k^{n+1}_{i+\frac{1}{2},j,k}\right)T^{n+1}_{i,j,k} + k^{n+1}_{i+\frac{1}{2},j,k}T^{n+1}_{i+1,j,k}\right) + \\ \Delta z^2\left(k^{n+1}_{i,j-\frac{1}{2},k}T^{n+1}_{i,j-1,k} - \left(k^{n+1}_{i,j-\frac{1}{2},k} + k^{n+1}_{i,j+\frac{1}{2},k}\right)T^{n+1}_{i,j,k} + k^{n+1}_{i,j+\frac{1}{2},k}T^{n+1}_{i,j+1,k}\right) + \\ \Delta x^2\left(k^{n+1}_{i,j,k-\frac{1}{2}}T^{n+1}_{i,j,k-1} - \left(k^{n+1}_{i,j,k-\frac{1}{2}} + k^{n+1}_{i,j,k+\frac{1}{2}}\right)T^{n+1}_{i,j,k} + k^{n+1}_{i,j,k+\frac{1}{2}}T^{n+1}_{i,j,k+1}\right) = \\ \frac{\Delta z^2\Delta x^2 c_p\rho}{\Delta t}T^n_{i,j,k} - \frac{\Delta z^2\Delta x^2 c_p\rho}{\Delta t}T^{n-1}_{i,j,k}\end{gathered} \tag{3.11}$$

Como a equação acima é complexa para ser reorganizada e resolvida por equações matriciais, será utilizado aproximações para resolver esse problema, ideia similar ao Método do Ponto Fixo (MPF). As condições de parada são: diferença entre iterações menor que

0,5ºC, máximo de iterações igual a 1.000, e mínimo de 800 iterações. Para resolver o sistema de equações, será isolado uma das temperaturas para calcular a iteração $\nu + 1$:

$$\begin{gathered} \overset{\nu+1}{T^n_{i,j}} = \\ C_1 \frac{\Delta z^2 \Delta x^2 c_p \rho}{\Delta t} T^{n-1}_{i,j} + \\ C_1 \Delta z^2 \left(\overset{\nu}{k^n_{i-\frac{1}{2},j,k}} \overset{\nu}{T^n_{i-1,j,k}} + \overset{\nu}{k^n_{i+\frac{1}{2},j,k}} \overset{\nu}{T^n_{i+1,j,k}} \right) + \\ C_1 \Delta z^2 \left(\overset{\nu}{k^n_{i,j-\frac{1}{2},k}} \overset{\nu}{T^n_{i,j-1,k}} + \overset{\nu}{k^n_{i,j+\frac{1}{2},k}} \overset{\nu}{T^n_{i,j+1,k}} \right) + \\ C_1 \Delta x^2 \left(\overset{\nu}{k^n_{i,j,k-\frac{1}{2}}} \overset{\nu}{T^n_{i,j,k-1}} + \overset{\nu}{k^n_{i,j,k+\frac{1}{2}}} \overset{\nu}{T^n_{i,j,k+1}} \right) \end{gathered} \tag{3.12}$$

Onde C_1é definido por:

$$\frac{1}{C_1} = \begin{array}{l} \frac{\Delta z^2 \Delta x^2 c_p \rho}{\Delta t} + \quad \Delta z^2 \left(\overset{\nu}{k^n_{i-\frac{1}{2},j,k}} + \overset{\nu}{k^n_{i+\frac{1}{2},j,k}} \right) + \\ \qquad\qquad \Delta z^2 \left(\overset{\nu}{k^n_{i,j-\frac{1}{2},k}} + \overset{\nu}{k^n_{i,j+\frac{1}{2},k}} \right) \quad + \Delta x^2 \left(\overset{\nu}{k^n_{i,j,k-\frac{1}{2}}} + \overset{\nu}{k^n_{i,j,k+\frac{1}{2}}} \right) \end{array} \tag{3.13}$$

Agora, é necessário definir o cálculo das condutividades térmicas nas fronteiras. Para isso, será feito um análogo com a permeabilidade de rochas em série [Rosa, Carvalho e Xavier 2006], mas utilizando as equações de calor:

$$q_x = -kA\frac{dT}{dx} = -\frac{kA}{L}\Delta T \tag{3.14}$$

Onde q_x é a vazão, k a permeabilidade, A a área, L o comprimento e ΔT a diferença de temperatura.

Isolando a diferença de temperatura:

$$\Delta T = -\frac{Lq_x}{kA} \tag{3.15}$$

A Figura 3.2 mostra um caso de condutividades térmicas em série. Em um sistema onde não há fontes de calor, a taxa de calor (q) que entra no sistema, é igual a que sai. E a diferença de temperatura entre a esquerda (0) e a direita (2), é soma das diferenças nesse meio, ou seja:

$$T_0 - T_2 = (T_0 - T_1) + (T_1 - T_2) \tag{3.16}$$

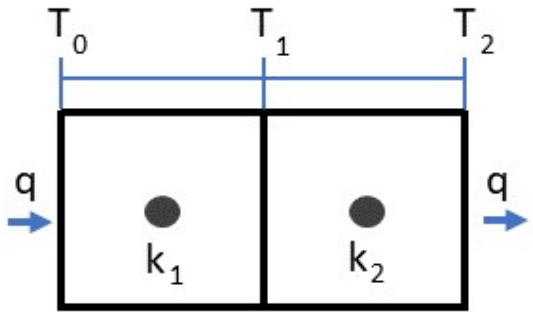


Figura 3.2: Representação de condutividade térmica em série

Logo,

$$\Delta T_t = \Delta T_1 + \Delta T_2 \tag{3.17}$$

Onde as taxas de transferência de calor são:

$$-\frac{2Lq}{k_r A} = -\frac{Lq}{k_1 A} - \frac{Lq}{k_2 A} \tag{3.18}$$

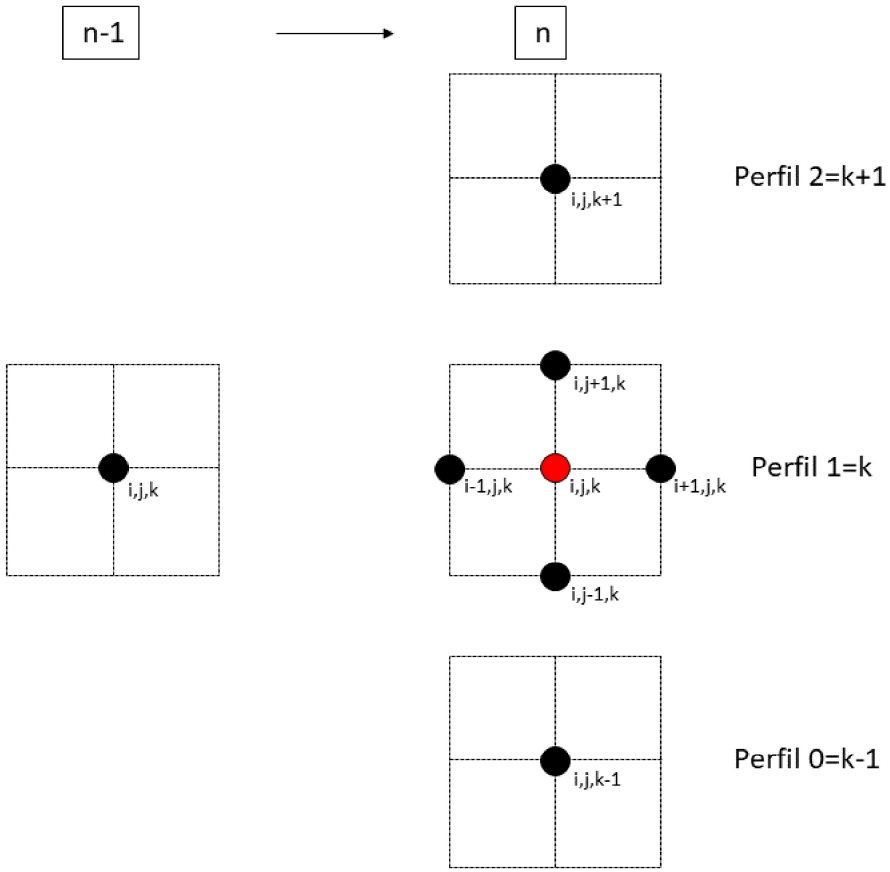
Com alguns ajustes algébricos:

$$\frac{2}{k_r} = \frac{1}{k_1} + \frac{1}{k_2} \tag{3.19}$$

Ou, simplesmente:

$$k_r = \frac{2k_1 k_2}{k_1 + k_2} \tag{3.20}$$

É importante analisar a célula computacional, ou a região que é observada quando a temperatura é calculada em um ponto específico. Para isso, é apresentada a Figura 3.3, onde a esquerda é o tempo anterior t=n-1, e o ponto calculado está no tempo presente t=n. Para calcular a temperatura no ponto vermelho, é utilizado o mesmo ponto, mas no tempo anterior, e uma célula em cada sentido.



computacionais

Figura 3.3: Malha utilizada para calcular a temperatura de um ponto, onde cada ponto é o centro dos blocos

A seguir, é resolvida a última etapa da modelagem do problema, a modelagem da condição de fronteira de Neumann.

3.2.3 Condição de fronteira

Condição de fronteira, como o próprio nome diz, é a condição onde estão os limites materiais do objeto. Nessa região, a condução térmica é diferente do interior do objeto, pois não poderá conduzir calor em todos os sentidos, mas só onde existir material adjacente.

A condição de contorno de Neumann define a taxa de troca de calor com o meio externo, no trabalho desenvolvido essa taxa será sempre nula, ou seja, a região estudada não troca calor com o meio externo. Na Figura 3.4, a fronteira está na reta vermelha e, como o método modelado utilizaria o ponto à esquerda, é necessário encontrar um substituto real para esse termo.

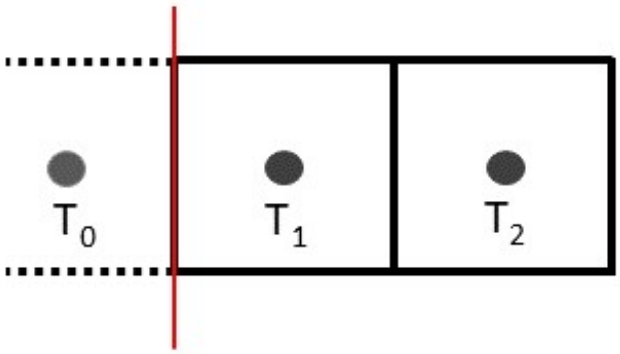


Figura 3.4: Análise da fronteira de Neumann

Por isso, é importante modelar a condição de contorno, que pode ser modelada com diferenças finitas centradas como:

$$k\frac{\partial T}{\partial x}_{i-\frac{1}{2},j,k} = \frac{T^n_{i,j,k} - T^n_{i-1,j,k}}{\Delta x} = 0 \tag{3.21}$$

A equação acima possui duas soluções:

$$\begin{cases} k_{i-\frac{1}{2},j,k} & = 0 \\ \frac{\partial T}{\partial x}_{i-\frac{1}{2},j,k} & = 0 \end{cases} \tag{3.22}$$

Resolvendo a linha de baixo:

$$\frac{\partial T}{\partial x}_{i-\frac{1}{2},j,k} = \frac{T^n_{i,j,k} - T^n_{i-1,j,k}}{\Delta x} = 0 \tag{3.23}$$

$$T^n_{i-1,j,k} = T^n_{i,j,k} \tag{3.24}$$

Todas as seis fronteiras são simétricas, então:

$$\begin{aligned} T^n_{i-1,j,k} &= T^n_{i,j,k} \\ T^n_{i+1,j,k} &= T^n_{i,j,k} \\ T^n_{i,j-1,k} &= T^n_{i,j,k} \\ T^n_{i,j+1,k} &= T^n_{i,j,k} \\ T^n_{i,j,k-1} &= T^n_{i,j,k} \\ T^n_{i,j,k+1} &= T^n_{i,j,k} \end{aligned} \tag{3.25}$$

As equações encontradas na Eq. 3.25 dizem que, se existir uma fronteira, a temperatura inexistente deve ser substituída pela temperatura do próprio ponto. De forma alternativa, como mostrado na Eq. 3.22, a condutividade térmica na fronteira deve ser zero. Quaisquer dentre as duas opções resolvem o problema da condição de contorno de Neumann.

3.2.4 Demonstrações matemáticas

Nesta parte, serão analisados dois casos para validar as modelagens. Primeiro, será utilizado um objeto formado por uma única célula isolada no espaço. Posteriormente, será analisado o caso do objeto constituído por um único material, mas bidimensional.

Começando pelo objeto de única célula, em todas as suas seis fronteiras devem ser aplicadas as condições de contorno de Neumann. Fisicamente, é esperado que o objeto, por estar isolado, não varie sua temperatura interna ao longo do tempo. Então, partindo da equação geral:

Partindo da Eq. (3.12) e, como demonstrado na Eq. 3.22, quando houver fronteira a condutividade térmica na fronteira é zero:

$$\begin{aligned} &\overset{\nu+1}{T^n_{i,j,k}} = \\ &C_1\frac{\Delta z^2\Delta x^2 c_p\rho}{\Delta t}T^{n-1}_{i,j,k} + \\ &C_1\Delta z^2\left(\cancel{k^{n\ \nu}_{i-\frac{1}{2},j,k}}^{\ 0}\overset{\nu}{T}{}^n_{i-1,j,k} + \cancel{k^{n\ \nu}_{i+\frac{1}{2},j,k}}^{\ 0}\overset{\nu}{T}{}^n_{i+1,j,k}\right) + \\ &C_1\Delta z^2\left(\cancel{k^{n\ \nu}_{i,j-\frac{1}{2},k}}^{\ 0}\overset{\nu}{T}{}^n_{i,j-1,k} + \cancel{k^{n\ \nu}_{i,j+\frac{1}{2},k}}^{\ 0}\overset{\nu}{T}{}^n_{i,j+1,k}\right) + \\ &C_1\Delta x^2\left(\cancel{k^{n\ \nu}_{i,j,k-\frac{1}{2}}}^{\ 0}\overset{\nu}{T}{}^n_{i,j,k-1} + \cancel{k^{n\ \nu}_{i,j,k+\frac{1}{2}}}^{\ 0}\overset{\nu}{T}{}^n_{i,j,k+1}\right) \end{aligned} \tag{3.26}$$

$$\begin{aligned} \frac{1}{C_1} = \ &\frac{\Delta z^2\Delta x^2 c_p\rho}{\Delta t} + \ \Delta z^2\left(\cancel{k^{n\ \nu}_{i-\frac{1}{2},j,k}}^{\ 0} + \cancel{k^{n\ \nu}_{i+\frac{1}{2},j,k}}^{\ 0}\right) + \\ &\Delta z^2\left(\cancel{k^{n\ \nu}_{i,j-\frac{1}{2},k}}^{\ 0} + \cancel{k^{n\ \nu}_{i,j+\frac{1}{2},k}}^{\ 0}\right) \quad + \Delta x^2\left(\cancel{k^{n\ \nu}_{i,j,k-\frac{1}{2}}}^{\ 0} + \cancel{k^{n\ \nu}_{i,j,k+\frac{1}{2}}}^{\ 0}\right) \end{aligned} \tag{3.27}$$

Resultando em:

$$\overset{\nu+1}{T^n_{i,j,k}} = C_1\frac{\Delta z^2\Delta x^2 c_p\rho}{\Delta t}T^{n-1}_{i,j} \tag{3.28}$$

$$\frac{1}{C_1} = \ \frac{\Delta z^2\Delta x^2 c_p\rho}{\Delta t} \tag{3.29}$$

Logo:

$$\overset{\nu+1}{T^n_{i,j,k}} = \frac{\Delta t}{\Delta z^2\Delta x^2 c_p\rho}\frac{\Delta z^2\Delta x^2 c_p\rho}{\Delta t}T^{n-1}_{i,j} \tag{3.30}$$

$$\overset{\nu+1}{T^n_{i,j,k}} = \overset{\nu}{T}{}^{n-1}_{i,j,k} \tag{3.31}$$

Mostrando que a temperatura não varia com o tempo.

Para a segunda demonstração, onde o objeto é constituído pelo mesmo material e mesma condutividade térmica, mas somente bidimensional.

O índice k referente à terceira dimensão continuará aparecendo nas equações abaixo para manter a ideia do algoritmo. Como só existe um valor para essa dimensão, pode-se considerar o valor fixo de 0.

Partindo da Eq. (3.12) e substituindo todas as condutividades térmicas nas interfaces por k, e simplificando para bidimensional:

$$\begin{gathered} T_{i,j,k}^{n} = \\ C_1 \frac{\Delta z^2 \Delta x^2 c_p \rho}{\Delta t} T_{i,j,k}^{n-1} + \\ C_1 \Delta z^2 \left(k T_{i-1,j,k}^{n} + k T_{i+1,j,k}^{n} \right) + \\ C_1 \Delta z^2 \left(k T_{i,j-1,k}^{n} + k T_{i,j+1,k}^{n} \right) \end{gathered} \tag{3.32}$$

$$\frac{1}{C_1} = \frac{\Delta z^2 \Delta x^2 c_p \rho}{\Delta t} + \Delta z^2 \left(k + k\right) + \Delta z^2 \left(k + k\right) \tag{3.33}$$

Com alguns ajustes:

$$\begin{gathered} \frac{1}{C_1} T_{i,j,k}^{n} = \\ \frac{\Delta x^2 c_p \rho}{\Delta t} T_{i,j,k}^{n-1} + \\ k \left(T_{i-1,j,k}^{n} + T_{i+1,j,k}^{n} \right) + \\ k \left(T_{i,j-1,k}^{n} + T_{i,j+1,k}^{n} \right) \end{gathered} \tag{3.34}$$

$$\frac{1}{C_1} = \frac{\Delta x^2 c_p \rho}{\Delta t} + 2k + 2k \tag{3.35}$$

Logo:

$$\begin{gathered} \left(\frac{\Delta x^2 c_p \rho}{k \Delta t} + 2 + 2 \right) T_{i,j,k}^{n} = \\ \frac{\Delta x^2 c_p \rho}{k \Delta t} T_{i,j,k}^{n-1} + \\ \left(T_{i-1,j,k}^{n} + T_{i+1,j,k}^{n} \right) + \\ \left(T_{i,j-1,k}^{n} + T_{i,j+1,k}^{n} \right) \end{gathered} \tag{3.36}$$

Chegando na equação:

$$\begin{gathered} \frac{c_p \rho}{k} \frac{T_{i,j,k}^{n} - T_{i,j,k}^{n-1}}{\Delta t} = \\ \frac{T_{i-1,j,k}^{n} - 2T_{i,j,k}^{n} + T_{i+1,j,k}^{n}}{\Delta x^2} + \\ \frac{T_{i,j-1,k}^{n} - 2T_{i,j,k}^{n} + T_{i,j+1,k}^{n}}{\Delta x^2} \end{gathered} \tag{3.37}$$

A Eq. (3.37) representa o método implícito BTCS para um sistema homogêneo bidimensional, conforme [Incropera 2008].

3.2.5 Condutividade térmica variável

A condutividade térmica, no programas desenvolvido, pode variar com o espaço, pelo objeto ser constituído por mais de um material, com condutividade térmica distinta.

Contudo, pode também variar com a temperatura e, consequentemente, com o tempo.

Serão fornecidas aos usuários, três opções para calcular essas condutividades térmicas:

- Valores constantes;
- Correlação;
- Interpolação.

Para o primeiro caso, como o nome diz, a condutividade térmica será constante ao longo de todo o tempo, variando somente com a posição.

No segundo caso, será utilizado os modelos de correlação do *handbook Thermophysical Properties* [Valencia e Quested 2008]. O modelo proposto, é calculado, em geral, como:

$$k = C_0 + C_1 T - C_2 T^2 \tag{3.38}$$

onde C_0, C_1 e C_2 são constantes da correlação, específicas para cada material.

O terceiro caso, é o cálculo pela interpolação e, como o nome diz, calcula a condutividade térmica pela interpolação linear entre valores obtidos em laboratório.

3.3 Formulação modelos computacionais

Apresenta-se a seguir considerações sobre o que é processamento paralelo, paralelismos usando múltiplas *threads* e informações sobre renderização 3D.

3.3.1 O que é processamento paralelo?

O processamento paralelo consiste em dividir uma tarefa em suas partes independentes e na execução de cada uma dessas partes em diferentes processadores ou núcleos.

A Figura 3.5 mostra um processamento serial e a Figura 3.6 sua versão paralelizada.

Note que os processadores podem estar numa mesma máquina, em um cluster ou em máquinas distribuídas.

Para poder aproveitar o poder de processamento dos novos processadores é preciso que os programas sejam desenvolvidos utilizando processamento paralelo.

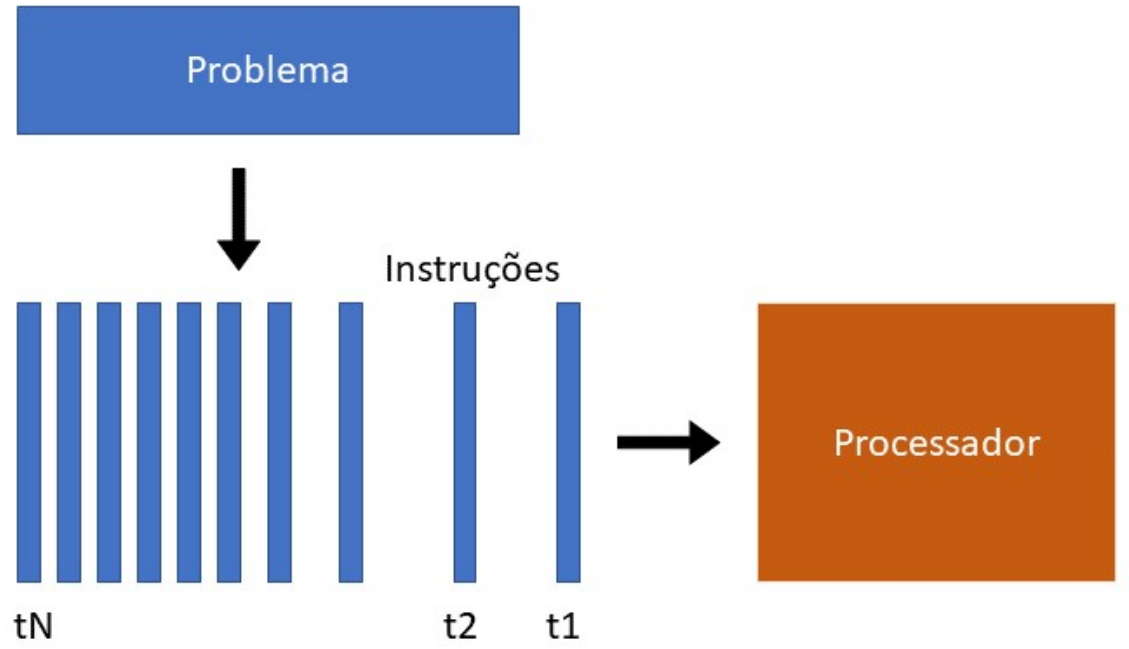


Figura 3.5: Processamento serial

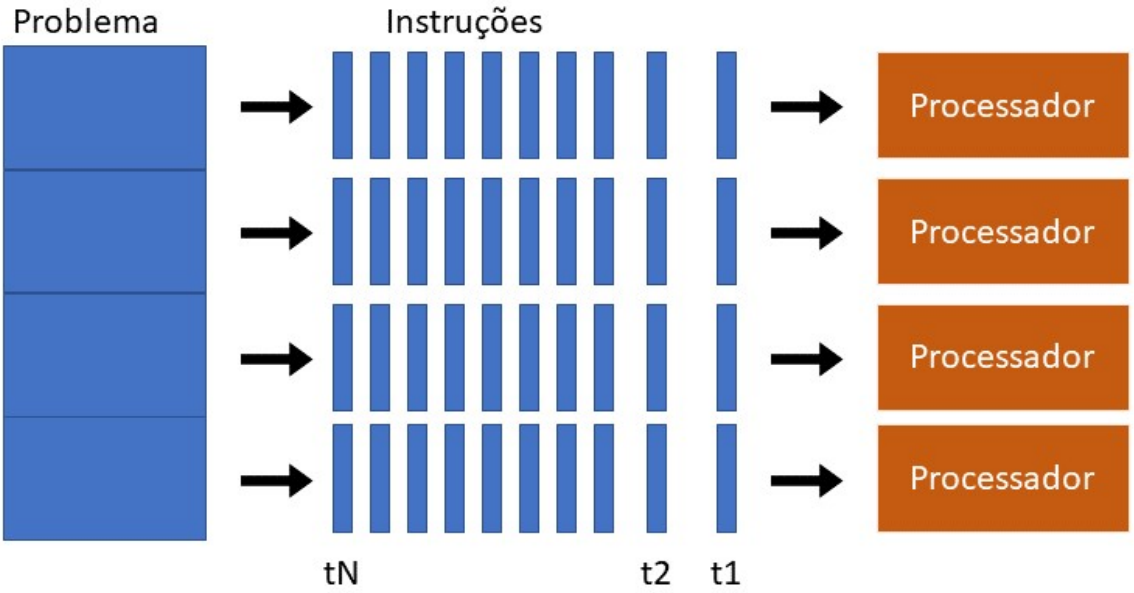


Figura 3.6: Processamento paralelo

Note que no processamento paralelo o acoplamento é mais forte, exigindo sistemas de comunicação mais velozes (como barramentos e/ou *switches* de alta velocidade). O controle dos recursos do sistema é mais refinado.

- **Vantagens:**
  - Melhor uso dos recursos disponibilizados; Maior velocidade na realização das tarefas; Maior confiabilidade.

- **Desvantagens:**
  - O controle dos recursos é de responsabilidade do programador.

3.3.2 *Processamento paralelo com múltiplas-threads - multi-thread*

Os chips dos processadores atuais são constituídos por vários processadores menores, o que permite que um mesmo processador consiga realizar tarefas (distintas ou iguais) nos processadores menores. A ideia é separar tarefas distintas, para que um processador não fique envolvido em uma única tarefa.

Uma analogia para melhorar a explicação sobre processamento paralelo é a da aplicação de um filtro sobre uma imagem bidimensional. O filtro a ser aplicado é exatamente o mesmo em toda imagem(mesmo algoritmo), sendo possível dividir a imagem em partes e entregar cada parte para um processador(*thread*) diferente executar.

Similarmente ao cenário acima, foram implementados três casos de paralelismo, por questão de didática.

1. Sem paralelismo: uma única *thread* do processador resolve todos os cálculos.
2. Paralelismo por *grid*: cada *thread* resolve uma camada do objeto. Possuí certa otimização em relação ao anterior, mas, se só existir objeto em uma camada, outras threads ficam ociosas.
3. Paralelismo total: todas as *threads* do processador resolvem os cálculos de todo o objeto 3D, intercalando a posição com base no número da *thread*.

A Figura 3.7 ilustra melhor esses três casos.

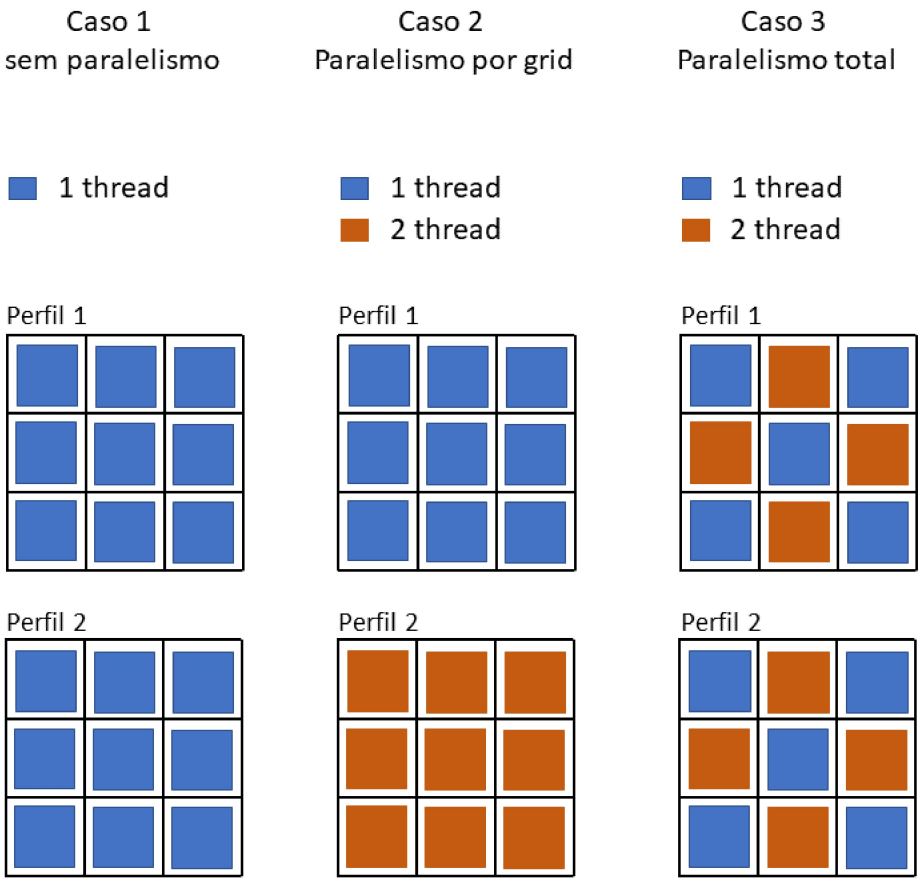


Figura 3.7: Ilustração dos três casos de paralelismo implementados para duas camadas com 9 células cada, e um processador com duas *thread*

O algoritmo utilizado para o caso 3 é:
for(int i = NUM_ THREAD; i < size; i+=MAX_ THREADS)

Esse algoritmo diz que a *thread* "i", deverá começar a resolver as equações na posição "i". Quando finalizar, deve pular para a posição "i + números de *thread*".

3.3.3 Renderização 3D

Após o usuário desenhar algum objeto no *software*, pode ser de interesse observar como seria em renderização 3D. Portanto, foram implementados algoritmos para essa renderização.

Inicialmente, é interessante observar a complexidade da renderização: um objeto 3D deve ser apresentado em uma tela 2D, com a ilusão de ótica que é um objeto com profundidade. Por exemplo, um cubo com arestas de tamanho 1 cm é mostrado nos quatro casos da figura abaixo:

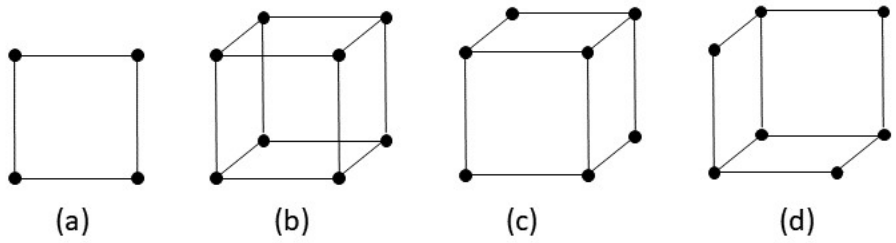


Figura 3.8: (a) Observador alinhado com uma das faces do cubo. (b) observador não está alinhado e não foram removidas arestas ocultas. O cérebro consegue interpretar que é um objeto 3D, mas fica confuso entre os casos (c) e (d)

Todos cantos do cubo da Figura 3.8 estão na mesma posição, o que mudou foi o ângulo do observador com o objeto.

Portanto, tendo definidos os pontos das arestas, seus respectivos vetores são multiplicados pela matriz de rotação [Herter e Lott] mostrada na Eq. 3.39, a qual permite rotacionar qualquer ponto a partir dos três ângulos do observador.

$$R(\alpha,\beta,\gamma) = \begin{bmatrix} cos(\gamma)cos(\beta) & cos(\gamma)sin(\beta)sin(\alpha) - sin(\gamma)cos(\alpha) & cos(\gamma)sin(\beta)sin(\alpha) + sin(\gamma)cos(\alpha) \\ sin(\gamma)cos(\beta) & sin(\gamma)sin(\beta)sin(\alpha) + cos(\gamma)cos(\alpha) & sin(\gamma)sin(\beta)cos(\alpha) - cos(\gamma)sin(\alpha) \\ -sin(\beta) & cos(\beta) * sin(\alpha) & cos(\beta) * cos(\alpha) \end{bmatrix} \tag{3.39}$$

onde α, β e γ são os Ângulos de Euler.

Ou seja, inicialmente, um cubo de aresta 3 cm, com uma distância da origem de 1 cm, pode ser mostrado na tela (monitor) com os pontos do caso (a) da Figura 3.9, onde o observador está alinhado com o objeto.

Conforme desejado, o objeto pode mudar seu ângulo com o observador, como no caso (b), onde os ângulos α e β passaram a ter o valor de 0.1 radianos. Não foi só os pontos de trás do cubo que aparecem (e mudaram seus valores), mas todos os pontos foram modificados.

Além disso, a aresta possui valor ligeiramente menor que 3, pois não é mais "de frente" que o observador está olhando, mas ligeiramente de lado. Mesmo que o objeto cubo tenha aresta de 3 centímetros.

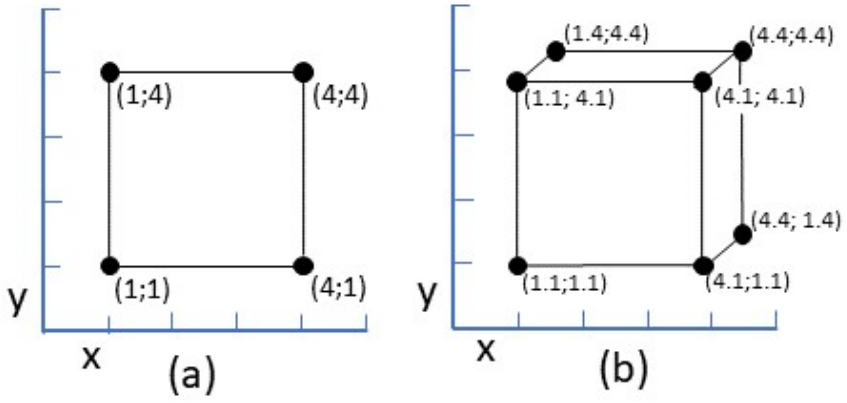


Figura 3.9: (a) o cubo está com ângulos nulos. (b) os ângulos α e β estão com valor de 0.1 radianos

Nos desenhos do simulador, cada pixel da figura, é uma célula com propriedades que serão calculadas, possuindo material, temperatura e volume. Como o usuário pode desenhar por pixel, a renderização 3D deve partir do princípio que cada pixel é um potencial objeto que deve ser renderizado.

Inicialmente, essa conclusão pode ficar vaga, pois todas as células do simulador devem ser renderizadas, mas, quando a simulação fica grande, é numerosa a quantidade de objetos renderizando ao mesmo tempo, tornando muito lenta a apresentação. Então algumas considerações são feitas no algoritmo para otimizar a renderização.

Primeiro, é desejável desenhar triângulos, e não pontos ou retas, por 2 motivos: geometria simples, possui normal e a biblioteca do Qt consegue desenhar e preencher a área com qualquer cor escolhida.

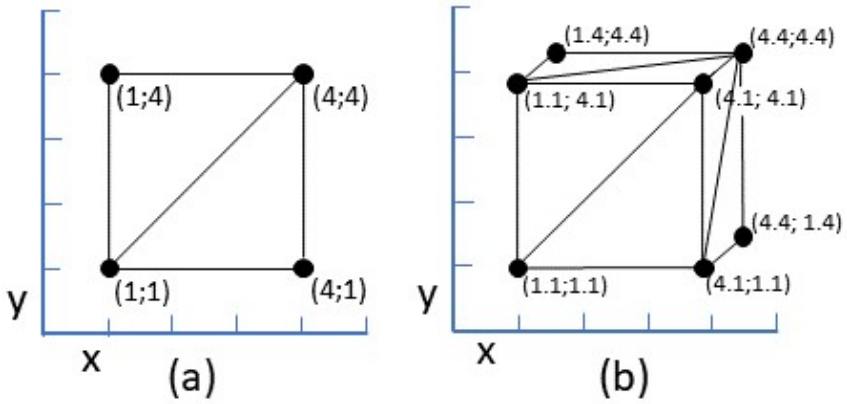


Figura 3.10: Mesmo desenho da Figura 3.9, mas agora renderizando a partir de triângulos

O segundo motivo apresentado, é o mais importante dos três. Um triângulo possui três pontos, podendo ser reduzido para dois vetores (subtraindo o ponto de origem dos

outros dois pontos) e permite-se calcular a normal dessa superfície. Com isso, são obtidos os vetores $\boldsymbol{a} = \{a_1, a_2, a_3\}$ e $\boldsymbol{b} = \{b_1, b_2, b_3\}$ permitindo a realização do produto vetorial:

$$\boldsymbol{a} \times \boldsymbol{b} = \begin{bmatrix} \boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k} \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{bmatrix} \tag{3.40}$$

Ou simplesmente:

$$\boldsymbol{a} \times \boldsymbol{b} = (a_2 b_3 - a_3 b_2)\boldsymbol{i} - (a_1 b_3 - a_3 b_1)\boldsymbol{j} + (a_1 b_2 - a_2 b_1)\boldsymbol{k} \tag{3.41}$$

Utilizando a Regra da Mão Direita[1], é possível entender a utilidade da equação 3.41: o caso (a) da figura 3.11, mostra uma normal saindo do papel, em direção ao olho do leitor, logo, é um triângulo que deve ser renderizado. O caso (b) possui uma normal no sentido contrário, e não faz sentido desenhar esse triângulo, pois está na parte de trás do objeto.
[1]

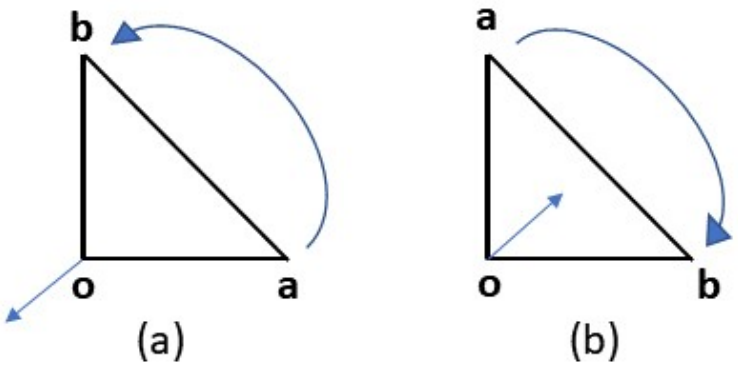


Figura 3.11: (a) mostra um caso onde a normal é na direção do leitor e (b) mostra um caso onde a normal é para dentro da folha

Essa simples operação condicional do valor positivo (apontando para o olho do observador) e negativo (apontando para dentro da tela) de **j** da normal, reduz a quantidade de objetos que devem ser renderizados, e otimiza o software em duas vezes.

Uma outra condição implementada é a de avaliar se o objeto está em contato com outro objeto. Na superfície de contato, existem 4 triângulos (2 de cada objeto), e eles não devem ser renderizados, pois estão no interior do objeto maior. Para esse cenário, pode-se pensar em blocos de montar: enquanto eles não estão juntos, é possível observar todas as superfícies do bloco mas, quando eles são encaixados, essa superfície de contato entre eles fica oculto.

[1]Para utilizar a Regra da Mão Direita, posicione o dedo polegar sobre o ponto **o**, e estique o indicador para o ponto **a**, agora, feche o indicador no sentido do ponto **b** (seta curvada mostra o sentido que a ponta do indicador deve realizar). No caso (a) da figura, o dedo polegar fica no sentido para fora do papel, e o caso (b), para dentro.

Por fim, antes de renderizar os numerosos triângulos, eles são colocadas em ordem crescente com o valor de **j** da normal. Isso serve para ser desenhado primeiro o que está atrás, e depois desenhar o que está na frente, sobrescrevendo áreas que deveriam estar ocultas, evitando a criação de figuras confusas como no caso (b) da Figura 3.8. É uma técnica lenta, mas de fácil implementação.

3.4 Identificação de pacotes

- Pacote de malhas: organiza o objeto desenhado em vetores, facilita o acesso do simulador às propriedades de cada célula.
- Pacote de simulação: nele está presente o coração do simulador: o *solver* da equação da temperatura, discretizada por métodos numéricos, e resolvida por método iterativo.
- Pacote de interpolação: utilizado para realizar interpolação com propriedades termofísicas dos materiais, é acessado pelo simulador, e retorna as propriedades do material.
- Pacote de correlação: mesma função da linha acima, mas para método de correlação.
- Pacote de interface ao usuário: utilização da biblioteca Qt, para criar interface gráfica amigável. Fornece um ambiente onde o usuário pode enviar comandos para o simulador de maneira fácil, e apresenta os resultados.
- Pacote de gráficos: utilização da biblioteca *qcustomplot*, para montar os melhores gráficos para o problema. É solicitado ao pacote de malhas os resultados da temperatura. Está presente junto com o pacote de interface.

3.5 Diagrama de pacotes

O diagrama de pacotes é apresentado na Figura 3.12.

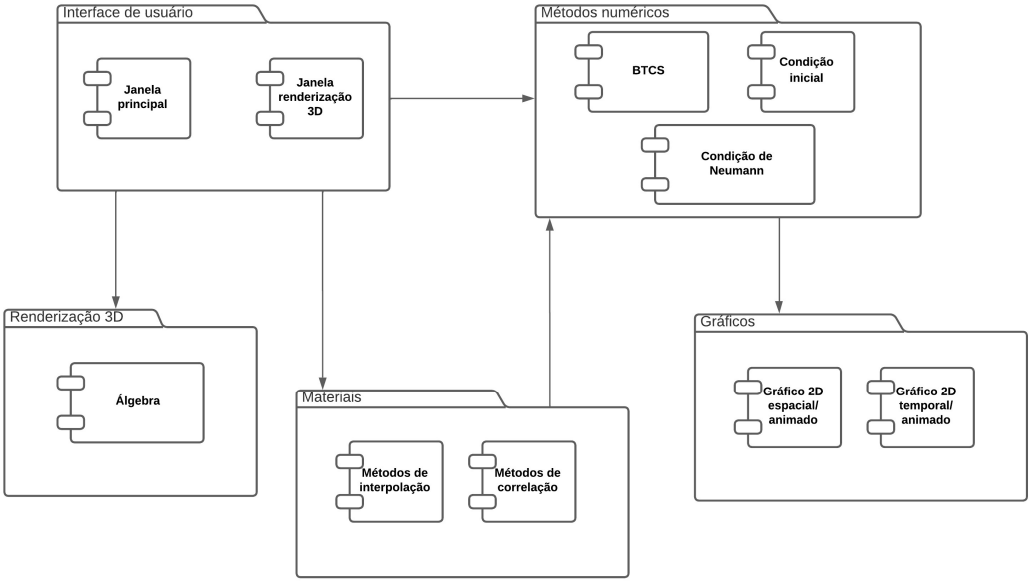


Figura 3.12: Diagrama de Pacotes