Simulação 3D do Comportamento Molecular dos Gases: Uma Abordagem Baseada na Teoria de Bernoulli com Inteligência Artificial.

O projeto implementa uma simulação do comportamento das partículas de gás dentro de um recipiente fechado, aplicando princípios da teoria cinética dos gases.

Prof: Felipe Mondaini

Integrantes: Adson Torino

Emanuel Cabral

Gabriel Santos

Gustavo Kanji

Nícolas Henriques

Thales Fortes

Vicente Zanatta

Introdução

Daniel Bernoulli (1700-1782) foi um físico e matemático suíço, membro de uma renomada família de cientistas. Ele é mais conhecido por suas contribuições à mecânica dos fluidos e por formular o Princípio de Bernoulli, que descreve o comportamento dos fluidos em movimento.

Além da hidrodinâmica, Bernoulli fez contribuições significativas à termodinâmica e à probabilidade. Ele foi um dos primeiros a aplicar conceitos de teoria cinética dos gases, explicando a pressão de um gás em termos das colisões moleculares. Seu trabalho estabeleceu bases para a física moderna, influenciando áreas como a aerodinâmica e a termodinâmica.

Objetivo do Projeto

O objetivo do experimento de Daniel Bernoulli relacionado à termodinâmica era explicar a pressão dos gases em termos de sua natureza molecular. Ele procurou demonstrar que a pressão dentro de um gás não era apenas uma força passiva, mas o resultado dos movimentos aleatórios e contínuos das partículas gasosas.

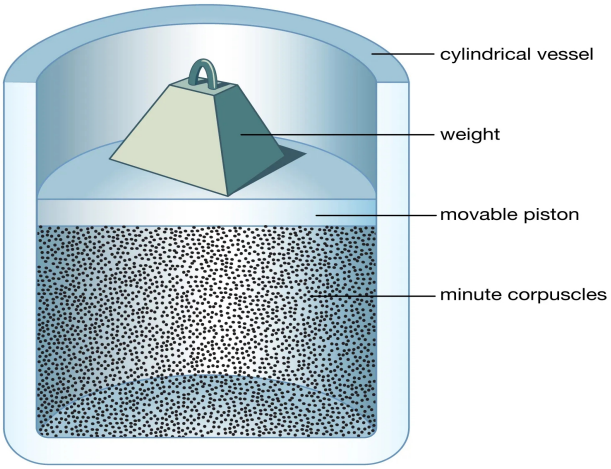
No experimento imaginado por Bernoulli, partículas minúsculas se movem em todas as direções e colidem com as paredes de um recipiente (ou pistão). Essas colisões repetidas exercem uma força sobre as paredes, que é percebida como pressão. Bernoulli mostrou que, ao remover ou reduzir o peso sobre o pistão, o gás expandiria, ilustrando a relação entre pressão, volume e temperatura, princípios fundamentais na termodinâmica. Esse experimento foi uma das primeiras tentativas de explicar a pressão e o comportamento dos gases em termos moleculares, estabelecendo a base para a teoria cinética dos gases e contribuindo para o desenvolvimento posterior da termodinâmica.

O projeto implementa uma simulação do comportamento das partículas de gás dentro de um recipiente fechado, aplicando princípios da teoria cinética dos gases. Nele, partículas minúsculas se movem de maneira aleatória, colidindo com as paredes do recipiente, fenômeno que resulta na pressão exercida pelo gás. Além disso, a simulação possibilita a visualização gráfica dos resultados em tempo real, incluindo um histograma que mostra a distribuição das partículas no cubo e outro que representa a distribuição das velocidades. A simulação também calcula o centro de massa do sistema e faz uma análise detalhada da distribuição das

partículas ao longo do eixo X, dividindo o espaço em fatias e calculando a velocidade média em cada fatia.

A distribuição das velocidades das partículas segue a curva de Maxwell-Boltzmann, que descreve a probabilidade de uma partícula de gás possuir uma certa velocidade a uma temperatura fixa. Isso é fundamental para entender como a energia térmica de um sistema está distribuída entre suas partículas. A análise final do histograma de velocidades permite confirmar a validação da teoria de Maxwell-Boltzmann e visualizar como as partículas se distribuem em diferentes faixas de velocidade.

Teoria



Daniel Bernoulli, em 1738, foi o primeiro a entender a pressão atmosférica em termos moleculares. Ele imaginou um cilindro vertical, fechado com um pistão no topo, o pistão tendo um peso sobre ele, ambos o pistão e o peso sendo suportados pela pressão dentro do cilindro. Ele descreveu o que ocorria dentro do cilindro da seguinte forma: "imagine que a cavidade contenha partículas muito pequenas, que movimentam-se frenéticamente para lá e para cá, de modo que quando estas partículas batam no pistão elas o sustentam com repetidos impactos, formando um fluido que expande sobre si caso o peso for retirado ou diminuido ..." É triste dizer que seu relato, apesar de correto, não foi aceito de maneira geral.

A maioria dos cientistas acreditavam que as moléculas de um gás estavam em repouso, repelindo-se à distância, fixas de alguma forma por um éter. Newton mostrou que PV = constante era uma consequência dessa teoria, se a repulsão dependesse inversamente com o quadrado da distância. De fato, em 1820 um inglês, John Herapath, deduziu uma relação entre pressão e velocidade molecular dada abaixo, e tentou publicá-la pela Royal Society (a acadmeia de ciências britânica). Foi rejeitada pelo presidente, Humphry Davy, que replicou que igualando pressão e temperatura, como feito por Herapath, implicava que deveria existir um zero absoluto de temperatura, uma idéia que Davy relutava em aceitar.

Explicação do Código

Importação das bibliotecas

import numpy as np

import matplotlib.pyplot as plt

from matplotlib.animation import FuncAnimation

from mpl_toolkits.mplot3d import Axes3D

import matplotlib.cm as cm

Essas bibliotecas são fundamentais para a simulação:

- **NumPy**: Para cálculos numéricos eficientes, especialmente manipulação de arrays.
- **Matplotlib**: Para visualização de dados e animações.
- **mpl_toolkits.mplot3d**: Para gráficos 3D.
- **matplotlib.cm**: Para mapas de cores utilizados na visualização da velocidade das partículas.

Configurações iniciais

num_particulas = 1000 # Número de partículas

limites_iniciais = [-10, 10] # Limites do cubo onde as partículas se movem

velocidade_max = 1 # Velocidade máxima inicial das partículas

*volume_inicial = (limites_iniciais[1] - limites_iniciais[0]) ** 3 # Volume inicial do recipiente*

num_frames = 100 # Número de frames

num_fatias_x = 100 # Número de fatias ao longo do eixo X

As variáveis acima definem os parâmetros principais da simulação:

- **num_particulas**: Número de partículas simuladas.
- **limites_iniciais**: Define o volume cúbico dentro do qual as partículas se

movem.

- **velocidade_max**: Velocidade máxima inicial das partículas.
- **volume_inicial**: Volume do recipiente, calculado com base nos limites.
- **num_frames**: Número de frames da simulação (quantidade de passos temporais).
- **num_fatias_x**: Divisão do espaço ao longo do eixo X para análise de distribuição de partículas.

Inicialização das posições e velocidades das partículas

posicoes = np.random.uniform(limites_iniciais[0], limites_iniciais[1], (num_particulas, 3))

velocidades = np.random.uniform(-velocidade_max, velocidade_max, (num_particulas, 3))

Aqui, são geradas posições e velocidades aleatórias para as partículas:

- **posicoes**: Posições aleatórias das partículas dentro do cubo definido pelos limites.
- **velocidades**: Velocidades iniciais aleatórias dentro de um intervalo simétrico, limitado por **velocidade_max**.

Parâmetros para energia cinética, pressão e volume

tempos = []

energia_cinetica = []

pressao = []

volume_atual = volume_inicial # Volume atual começa como o volume inicial

Estas listas serão usadas para armazenar dados ao longo da simulação, como a energia cinética e a pressão.

Cálculo da energia cinética

def calcular_energia_cinetica():

*return 0.5 * np.sum(np.linalg.norm(velocidades, axis=1) ** 2)*

*Essa função calcula a **energia cinética total** do sistema, somando as energias de todas as partículas. A fórmula usada é Ec=1/2(∑v^2), onde v é a velocidade de cada partícula.*

Cálculo da pressão

def calcular_pressao():

*return (2 / 3) * (calcular_energia_cinetica() / volume_atual)*

Aqui, a pressão é calculada usando a fórmula P = 2/3(VEc), onde Ec é a energia cinética total e V o volume do recipiente.

Cálculo do centro de massa

def calcular_centro_massa():

return np.mean(posicoes, axis=0)

Essa função calcula o **centro de massa** das partículas, encontrando a posição média de todas elas.

Função para plotar o histograma das velocidades

def plotar_histograma_velocidades():

magnitudes_velocidades = np.linalg.norm(velocidades, axis=1)

num_bins = num_fatias_x

fig, ax = plt.subplots(figsize=(6, 4))

```
        contagem, bordas, _ = ax.hist(magnitudes_velocidades, bins=num_bins, color='blue', edgecolor='black')


                norm = plt.Normalize(vmin=magnitudes_velocidades.min(), vmax=magnitudes_velocidades.max())
    cores = cm.coolwarm(norm((bordas[:-1] + bordas[1:]) / 2))


    ax.cla()
    for i in range(num_bins):
        ax.bar(bordas[i], contagem[i], width=bordas[i+1] - bordas[i], color=cores[i], edgecolor='black')


    ax.set_title('Histograma de Velocidades das Partículas')
    ax.set_xlabel('Magnitude da Velocidade')
    ax.set_ylabel('Número de Partículas')


    sm = plt.cm.ScalarMappable(cmap='coolwarm', norm=norm)
    sm.set_array([])
    fig.colorbar(sm, ax=ax, label='Magnitude da Velocidade')


    plt.show()
```

Esta função gera um **histograma das velocidades** das partículas, exibindo a distribuição das magnitudes das velocidades. Utiliza um mapa de cores para representar as diferentes faixas de velocidade.

Função para atualizar a simulação

```
def atualizar(frame):
    global posicoes, velocidades, volume_atual
```

```
    posicoes += velocidades

    colisao_menor = posicoes <= limites[0]
    colisao_maior = posicoes >= limites[1]
    velocidades[colisao_menor | colisao_maior] *= -1
    posicoes = np.clip(posicoes, limites[0], limites[1])

    magnitudes = np.linalg.norm(velocidades, axis=1)
    scatter._offsets3d = (posicoes[:, 0], posicoes[:, 1], posicoes[:, 2])
    scatter.set_array(magnitudes)

    centro_massa = calcular_centro_massa()
    centro_massa_point.set_data([centro_massa[0]], [centro_massa[1]])
    centro_massa_point.set_3d_properties([centro_massa[2]])

    if frame == num_frames - 1:
        x_fatias = np.linspace(limites_iniciais[0], limites_iniciais[1], num_fatias_x + 1)
        histograma, _ = np.histogram(posicoes[:, 0], bins=x_fatias)

        velocidade_media_fatia = []
        for i in range(num_fatias_x):
            particulas_fatia = (posicoes[:, 0] >= x_fatias[i]) & (posicoes[:, 0] < x_fatias[i
+ 1])
            if np.sum(particulas_fatia) > 0:
                velocidade_media_fatia.append(np.mean(np.linalg.norm(velocidades[parti
culas_fatia], axis=1)))
            else:
                velocidade_media_fatia.append(0)
```

velocidade_media_fatia = np.array(velocidade_media_fatia)

norm = plt.Normalize(vmin=velocidade_media_fatia.min(), vmax=velocidade_media_fatia.max())

cores = cm.coolwarm(norm(velocidade_media_fatia))

fig_hist, ax_hist = plt.subplots(figsize=(6, 4))

for i in range(num_fatias_x):

ax_hist.bar(x_fatias[i], histograma[i], width=np.diff(x_fatias)[i], align='edge', color=cores[i], edgecolor='black')

ax_hist.set_xlabel('Posição ao longo do eixo X')

ax_hist.set_ylabel('Número de partículas')

ax_hist.set_title('Distribuição das partículas ao longo do eixo X no último frame')

fig_hist.colorbar(cm.ScalarMappable(norm=norm, cmap='coolwarm'), ax=ax_hist, label='Velocidade média')

plt.show()

plotar_histograma_velocidades()

Essa função é chamada a cada frame da animação para atualizar as posições das partículas e detectar colisões com as paredes. Também calcula o **centro de massa** e, no último frame, plota a **distribuição das partículas ao longo do eixo X** e o **histograma das velocidades**.

Configuração da animação 3D

fig = plt.figure(figsize=(6, 6))

ax0 = fig.add_subplot(111, projection='3d')

limites = limites_iniciais

ax0.set_xlim(limites)

ax0.set_ylim(limites)

ax0.set_zlim(limites)

ax0.set_box_aspect([1, 1, 1])

ax0.set_title('Colisão de Moléculas de Gás')

ax0.set_xlabel('Eixo X')

ax0.set_ylabel('Eixo Y')

ax0.set_zlabel('Eixo Z')

magnitudes_iniciais = np.linalg.norm(velocidades, axis=1)

scatter = ax0.scatter(posicoes[:, 0], posicoes[:, 1], posicoes[:, 2], c=magnitudes_iniciais, cmap='coolwarm')

fig.colorbar(scatter, ax=ax0, label='Magnitude da Velocidade')

centro_massa = calcular_centro_massa()

centro_massa_point, = ax0.plot([centro_massa[0]], [centro_massa[1]], [centro_massa[2]], 'o', color='green', markersize=10)

ani = FuncAnimation(fig, atualizar, frames=np.arange(0, num_frames), interval=100, repeat=False)

plt.show()

Aqui, a figura e o gráfico 3D são configurados, exibindo as partículas em movimento. A cor das partículas é baseada na magnitude de suas velocidades, e o centro de massa é destacado com uma cor verde. A animação é executada com a função *FuncAnimation*.

Metodologia

Ferramentas utilizadas

Para a criação dos gráficos 3D foi utilizando a biblioteca Matplotlib, que é uma biblioteca de software para criação de gráficos e visualizações de dados em geral, feita para e da linguagem de programação Python e sua extensão de matemática NumPy.

Desenvolvimento da simulação

Durante o processo de desenvolvimento da simulação foram necessárias diversas alterações no código por meio dos prompts que fornecemos à inteligência artificial utilizada (ChatGPT-4).
Prompts em inteligência artificial são instruções ou entradas fornecidas ao modelo para gerar uma resposta ou realizar uma tarefa específica. Eles atuam como guias para que a IA entenda o contexto e produza saídas relevantes. Prompts podem variar em complexidade, desde perguntas simples até descrições detalhadas de cenários ou problemas.

Os prompts utilizados inicialmente foram mais primitivos e deram bons resultados, mas nada que não poderia florescer para algo mais robusto.

Limitações da Simulação de Partículas:

- Simplificações no Modelo Físico: A simulação assume que as partículas são pontos, ou seja, elas não têm dimensão física, o que impede uma modelagem realista de colisões entre partículas. No mundo real, as partículas têm volume e massa, e interagem entre si com forças (como a de Van der Waals), algo que está ausente nesse modelo simplificado.
- Colisões com as Paredes: O modelo usa colisões perfeitamente elásticas com as paredes, o que significa que as partículas mantêm sua energia cinética após a colisão. No entanto, na realidade, colisões poderiam ser parcialmente inelásticas, resultando em perda de energia cinética devido a fatores como atrito ou dissipação de calor.
- Cálculo da Pressão: O cálculo da pressão está baseado na energia cinética média das partículas e no volume. Embora isso seja uma simplificação válida para um sistema ideal, não considera efeitos mais complexos, como flutuações locais na densidade ou interações intermoleculares, que

influenciam a pressão em um sistema real.

- Limitação no Número de Partículas: A simulação usa um número relativamente pequeno de partículas (1000 no exemplo). No entanto, para que os comportamentos macroscópicos, como temperatura e pressão, correspondam mais aos sistemas reais, seria necessário simular milhões de partículas, o que aumentaria o custo computacional.
- Limitação de Escala Temporal: A simulação é limitada pela escala temporal dos frames. O tempo de simulação é discretizado, o que pode resultar em perda de precisão em interações rápidas ou em processos que ocorrem em escalas de tempo muito curtas.

Essas simplificações são necessárias para que o código seja computacionalmente eficiente, mas tornam a simulação uma aproximação idealizada da realidade. Modelos mais complexos, como a Dinâmica Molecular, podem fornecer uma descrição mais fiel, mas com um custo computacional muito mais elevado.

Melhorias Futuras para a Simulação

- Simulação de Interações Entre Partículas: Atualmente, a simulação ignora as interações entre partículas, o que limita o realismo em gases mais densos. Uma melhoria seria implementar um modelo de potencial de Lennard-Jones, que descreve a interação entre moléculas, considerando forças de atração e repulsão. Isso tornaria a simulação mais adequada para sistemas de gases reais e permitiria a observação de fenômenos como condensação ou formação de clusters de partículas.
- Simulação de Colisões Inelásticas: Ao incorporar colisões parcialmente inelásticas, que dissipam energia ao longo do tempo, a simulação poderia capturar a perda de energia cinética devido a efeitos como atrito ou dissipação de calor, aproximando-se mais da realidade. Isso é especialmente útil para simular sistemas com perdas energéticas, como líquidos e sólidos em transição.
- Aumento do Número de Partículas (Uso de GPUs): Uma das limitações práticas é o número reduzido de partículas. Para aumentar significativamente o número de partículas e capturar melhor as propriedades macroscópicas do sistema, seria possível utilizar unidades de processamento gráfico (GPUs) para computação paralela, permitindo simular centenas de milhares de partículas em tempo real.
- Termodinâmica Avançada (Transferência de Calor e Difusão): Adicionar

processos de transferência de calor e difusão poderia enriquecer o modelo, permitindo que diferentes regiões do recipiente tivessem temperaturas diferentes e que o calor fosse transferido entre elas. Modelos como o transporte de calor de Fourier ou o modelo de Boltzmann para transporte de partículas poderiam ser implementados.

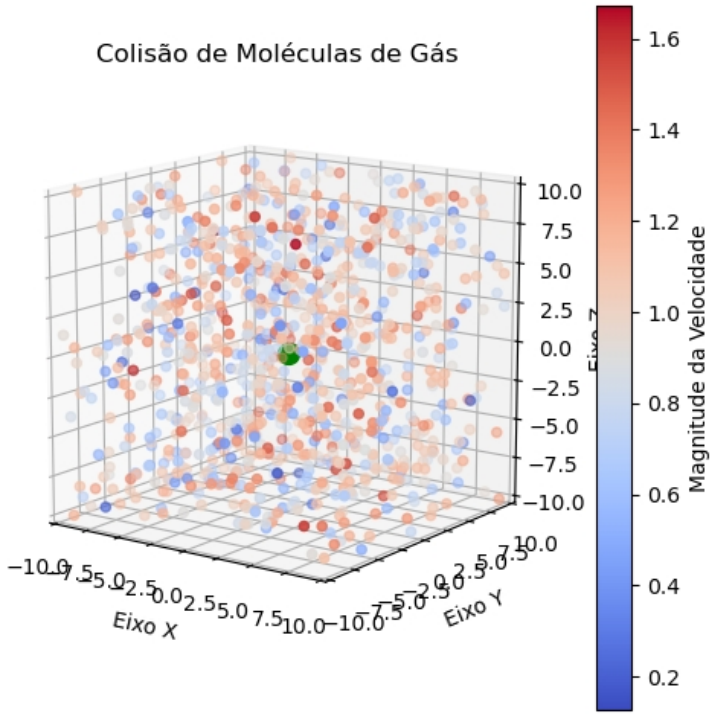
- Adição de Outras Propriedades Físicas (Viscosidade, Tensão Superficial): Para tornar a simulação mais versátil, seria interessante adicionar outras propriedades físicas, como a viscosidade do fluido e a tensão superficial, permitindo a simulação de líquidos além de gases. Isso poderia incluir a modelagem de fases líquidas ou a transição entre fases, por exemplo, de líquido para gás.
- Simulação em Alta Pressão e Altas Temperaturas: A simulação atual opera em um regime de gás ideal com baixas pressões e temperaturas moderadas. Expandir o modelo para simular gases em condições extremas (alta pressão ou temperatura) ou mesmo plasmas exigiria mudanças no tratamento de colisões e interações intermoleculares.
- Visualização Melhorada e Interatividade: Aumentar a interatividade da simulação, permitindo que o usuário manipule parâmetros em tempo real, como volume, temperatura e número de partículas, com feedback imediato sobre as mudanças no sistema. Melhorar a visualização com ferramentas como renderização em 3D com shaders gráficos pode proporcionar uma experiência mais envolvente e cientificamente precisa.

Essas melhorias, incluindo simulações multiescala e interações mais complexas entre partículas, trariam a simulação mais próxima da realidade e permitiriam a exploração de fenômenos mais variados e complexos. Implementar esses avanços exigiria mais poder computacional, mas o uso de GPUs e modelos adaptativos ajudaria a manter a eficiência do sistema.

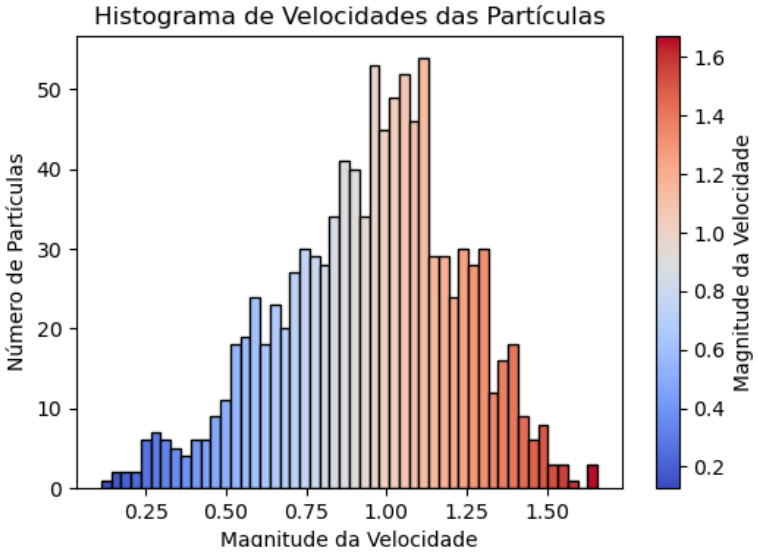
Conclusão

Este projeto implementa uma simulação eficaz do comportamento molecular dos gases, baseada nos princípios da teoria cinética, conforme descritos por Daniel Bernoulli. A simulação consegue modelar de maneira precisa o movimento aleatório das partículas de gás, suas colisões com as paredes do recipiente e as variáveis fundamentais que emergem desse sistema, como a pressão e a energia cinética. Através da análise das colisões, foi possível validar conceitos importantes da termodinâmica, ilustrando a relação entre pressão, volume e temperatura em sistemas gasosos.

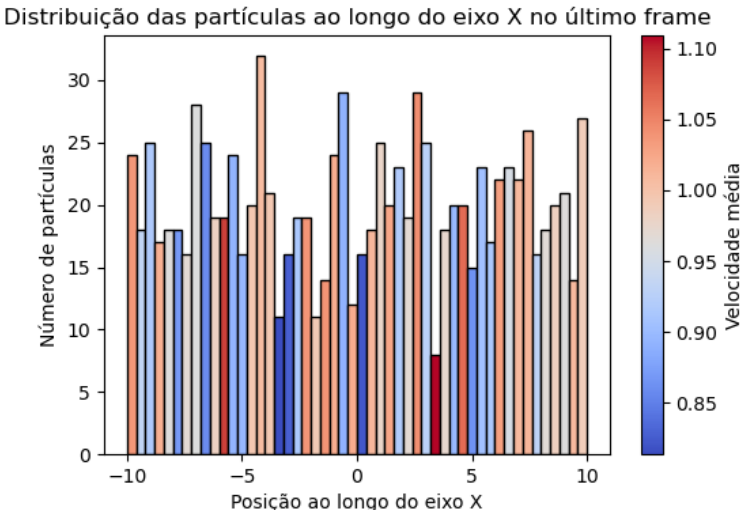
A simulação também introduziu uma abordagem visual poderosa, permitindo a visualização em tempo real da movimentação das partículas e da distribuição de suas velocidades.



O uso de histogramas para representar a distribuição de velocidades e a distribuição espacial das partículas ao longo do eixo X proporcionou uma compreensão clara dos fenômenos estatísticos que regem o comportamento dos gases, como a curva de Maxwell-Boltzmann.



Além disso, o cálculo do centro de massa e a análise da velocidade média em diferentes regiões do recipiente fornecem uma visão mais detalhada da dinâmica interna do sistema.



Além da relevância acadêmica e pedagógica, este projeto também possui grande importância prática. A simulação molecular dos gases é amplamente aplicada em áreas como a **física de fluidos** e a **engenharia**, auxiliando na modelagem de sistemas complexos, como turbinas e motores, e no estudo da dinâmica de fluidos em ambientes industriais. Em setores como o **aeroespacial**, o comportamento molecular dos gases é fundamental para projetar sistemas de propulsão eficientes e prever o comportamento dos gases em altas altitudes e em velocidades supersônicas.

Da mesma forma, a simulação molecular desempenha um papel importante nas **ciências atmosféricas**, auxiliando na previsão de padrões climáticos e na dispersão de poluentes na atmosfera. Em áreas como a **ciência dos materiais**, entender como as moléculas de gás interagem com diferentes superfícies é crucial para o desenvolvimento de novos materiais, como os usados em processos de deposição química em fase vapor (CVD).

Embora a simulação apresente algumas simplificações em relação ao mundo real, como o uso de partículas sem interação entre si e colisões perfeitamente elásticas com as paredes, ela cumpre de maneira notável o objetivo de demonstrar, com precisão, os princípios fundamentais da teoria cinética dos gases. As simplificações são justificadas pelo foco em eficiência computacional e clareza didática, sem comprometer a validade dos resultados para um sistema idealizado.

Em suma, este trabalho demonstra com sucesso os conceitos fundamentais da termodinâmica molecular, sendo uma ferramenta valiosa tanto para o ensino quanto para a compreensão dos comportamentos estatísticos dos gases. A combinação de simulação numérica, visualização em tempo real e análise estatística robusta posiciona este projeto como uma base sólida para futuras explorações no campo da física dos gases e da termodinâmica, com amplo potencial para aplicações em áreas como a engenharia, as ciências atmosféricas e o desenvolvimento de novas tecnologias.

Bibliografia

https://pt.wikipedia.org/wiki/Princ%C3%ADpio_de_Bernoulli

https://pt.wikipedia.org/wiki/Hist%C3%B3ria_da_termodin%C3%A2mica

https://www.if.ufrj.br/~bertu/fis2/teoria_cinetica/teoria_cinetica.html

https://brasilescola.uol.com.br/biografia/daniel-bernoulli.htm