1. Modelagem do Problema

A modelagem deve transformar a complexidade física de Ouro Preto em uma estrutura matemática tratável por algoritmos de busca.

Definição do Grafo

- **Vértices (V):** Representam as interseções e pontos de interesse (ex: Praça Tiradentes, Terminal Rodoviário, Campus UFOP).
- **Arestas (E):** Representam os segmentos de vias. Devem ser direcionadas para refletir as mãos de direção (ex: Rua São José é sentido único).

Formulação da Função de Custo (W_e)

Diferente de um GPS comum, o custo de um arco em Ouro Preto não é apenas a distância (L_e), mas uma composição de fatores de impedância:

$$W(e) = L_e \cdot (1 + \phi(S_e)) \cdot (1 + \rho(R_e)) \cdot \left[1 + \beta \left(\frac{V_e}{C_e}\right)^{\gamma}\right]$$

- Essa função faz sentido?

Onde:

- $\phi(S_e)$: Penalização pela declividade (S_e). Ladeiras acima de 15% devem ter pesos significativamente maiores, especialmente em dias de chuva. (API do google para buscar inclinação) declividade e distância
- $\rho(R_e)$: Coeficiente de rugosidade. O pavimento de pedras (pé de moleque) aumenta o tempo de viagem em até 57% comparado ao asfalto. (Teríamos um coeficiente fixo para o centro já que é o "mesmo tipo de pedra"?)
- V_e/C_e : Relação volume/capacidade, que simula o atraso gerado pelo congestionamento em tempo real. (Existem simuladores mas podemos "gerar a simulação"?)

2. Teste/Implementação

Os testes devem validar se a IA realmente "foge" dos congestionamentos e interdições.

Cenários de Simulação

1. **Cenário de Evento (Semana Santa/Evento na Praça Tiradentes/Festival):** Aplicar "peso infinito" na Rua Diogo de Vasconcelos e Praça Tiradentes. O sistema deve sugerir rotas via Rua do Pilar ou Rua Xavier da Veiga.
2. **Cenário Climático:** Aumentar os pesos das ladeiras íngremes (ex: Ladeira da Barra) em 200% para simular baixa aderência em dia de chuva.

Métricas de Avaliação

- **Throughput (Vazão):** Quantos veículos completam o trajeto por hora.
- **Estatística GEH:** Comparar o volume de tráfego simulado com as contagens reais da Ourotran para validar o modelo.
- **Latência de Re-roteamento:** Tempo que a IA leva para sugerir uma nova rota após um bloqueio (deve ser < 100ms para uso em tempo real).

Referências encontradas

1. Simulação de Tráfego em Cidades Históricas: Um estudo de caso https://www.encontrodesaberes.ufop.br/gerar_pdf.php?id=3335
2. (PDF) Implementation of D* Lite Algorithm for Dynamic Pathfinding in a Street Environment https://www.researchgate.net/publication/395552314_Implementation_of_D_Lite_Algorithm_for_Dynamic_Pathfinding_in_a_Street_Environment
3. os desafios da mobilidade urbana nas cidades históricas: elaboração de plano de mobilidade para ouro preto https://fau.ufal.br/evento/pluris2016/files/Tema%203%20-%20Mobilidade%20e%20Transportes/Paper643.pdf

Algoritmos de busca em grafos vantagens e desvantagens

1. Algoritmo de Dijkstra: A Busca Exaustiva

O algoritmo de Dijkstra é a base clássica para encontrar o caminho mínimo em grafos com pesos não negativos. Ele funciona de forma exaustiva, explorando todos os caminhos possíveis a partir da origem até encontrar o destino.

- **Funcionamento:** Ele mantém uma estimativa de distância para cada nó e, iterativamente, escolhe o nó com a menor distância confirmada para explorar seus vizinhos ("relaxamento das arestas").
- **Vantagens:** Garante o caminho mais curto absoluto e é matematicamente robusto.
- **Desvantagens:** É uma busca "cega" (não direcionada), o que gera um alto custo computacional em mapas grandes. Além disso, se o peso de uma única rua mudar (ex: um novo congestionamento na Rua São José), o Dijkstra exige

que todo o cálculo seja refeito do zero.

2. Algoritmo A* (A-Estrela): A Busca Heurística

O A* é uma evolução do Dijkstra que introduz "inteligência" à busca através de uma função heurística, tornando-a muito mais rápida.

- **Funcionamento:** Ele utiliza a função f(n) = g(n) + h(n) para decidir qual nó explorar :
    - g(n): O custo real acumulado do início até o nó atual.
    - h(n): Uma estimativa (heurística) do custo restante até o objetivo (como a distância em linha reta).
- **Vantagens:** Reduz a expansão de nós em 30% a 50% comparado ao Dijkstra, focando a busca na direção do destino. É ideal para cenários estáticos onde o mapa não muda durante o trajeto.
- **Desvantagens:** Assim como o Dijkstra, ele não lida bem com mudanças em tempo real. Se um obstáculo aparecer enquanto o veículo está em movimento, o A* precisa recalcular o caminho inteiro do zero, o que é ineficiente para sistemas de tráfego dinâmicos.

3. Algoritmo D* Lite: O Replanejamento Incremental

O D* Lite (Dynamic A*) é o estado da arte para navegação em ambientes onde as condições mudam constantemente, sendo a escolha ideal para o cenário de Ouro Preto.

- **Funcionamento:** Introduzido em 2002 por Koenig e Likhachev, ele é um algoritmo **incremental**. Em vez de recalcular tudo, ele reutiliza os dados da busca anterior e atualiza apenas os nós que foram afetados pela mudança de peso na via. Ele realiza uma busca "reversa" (do destino para a origem), o que facilita o replanejamento enquanto o agente se move.
- **Vantagens:**
    - **Alta Adaptabilidade:** Reage instantaneamente a ruas fechadas para procissões ou obras sem travar o processamento.
    - **Eficiência de Memória:** Processa apenas a parte do grafo que sofreu alteração de custo.
    - **Estabilidade:** Garante rotas ótimas mesmo com mudanças frequentes no ambiente.
- **Desvantagens:** É significativamente mais complexo de implementar do que o A*. Ele também depende de um monitoramento constante e preciso do ambiente para ser eficaz.