

UNIVERSIDADE DE SÃO PAULO

GABRIEL FERNANDES MOTA - 11796402

LUCAS QUARESMA MEDINA LAM - 11796399

ROBERTO OLIVEIRA BOLGHERONI - 11796430

EXERCÍCIO PROGRAMA 1

Modelagem Movimento Retilíneo Uniforme

São Paulo

2021

1. INTRODUÇÃO

Esse trabalho da Disciplina MAC0209 - Modelagem e Simulação tem como objetivo estudar e modelar o Movimento Retilíneo Uniforme (MRU), a partir da análise da trajetória de um veículo em autoestrada.

Para isso, escolhemos trechos do percurso de um veículo utilizando a plataforma KartaView, sistema de coleta e compartilhamento de imagens de estradas e ruas. O sistema divide a viagem do veículo em pontos/fotos, registrando algumas informações durante suas capturas.

Para cada ponto da trajetória, coletamos três dados fundamentais: data de captura, latitude e longitude (dentro da projeção Datum WGS 84). Para calcular distâncias entre pontos a partir dos dados posicionais no globo, fizemos uso de três distintos métodos, listados a seguir:

- Fórmula de Haversine:

Dado uma esfera de raio r , a fórmula de haversine calcula a distância d entre dois pontos com latitude e longitude ϕ e λ , respectivamente, ambos em radianos.

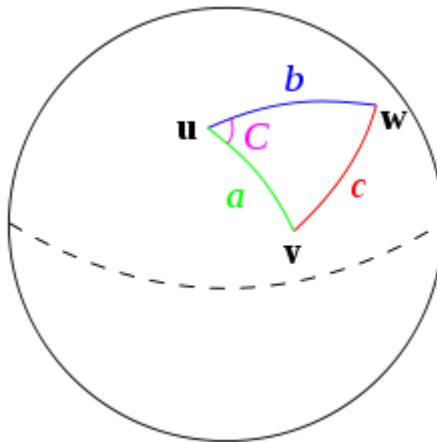
$$d = 2r \arcsin \left(\sqrt{\sin^2 \left(\frac{\phi_2 - \phi_1}{2} \right) + \cos \phi_1 \cos \phi_2 \sin^2 \left(\frac{\lambda_2 - \lambda_1}{2} \right)} \right)$$

- Uso de trigonometria esférica:

Fazendo o uso da lei esférica do cosseno, ao derivar a fórmula de Haversine, podemos usar o ângulo em grau entre os pontos para o cálculo da distância c entre dois pontos, v e w , com latitude e longitude ϕ e λ respectivamente, em uma esfera de raio r :

$$C = \sin(\phi_1) \sin(\phi_2) + \cos(\lambda_2 - \lambda_1) \cos(\phi_1) \cos(\phi_2)$$

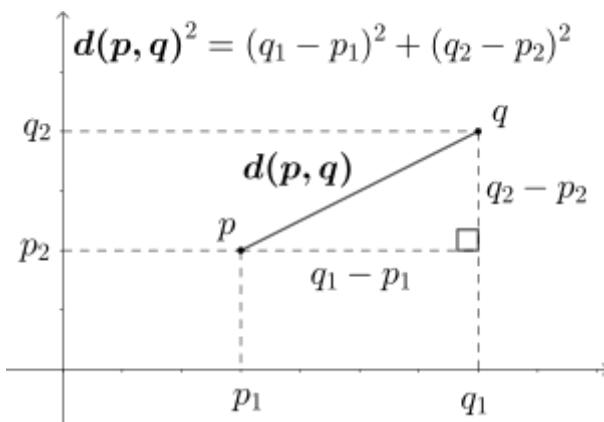
$$c = \frac{r\pi}{180} \arccos(C)$$



Fonte: https://en.wikipedia.org/wiki/Haversine_formula

- Método de Mercator:

Dado as latitudes e longitudes, em graus, transformamos para o sistema EPSG:3857, que usa a medida métrica, e então vamos usar o cálculo da distância euclidiana de dois pontos.



Fonte: https://en.wikipedia.org/wiki/Euclidean_distance

Para os cálculos, assumimos $6,371 \times 10^6$ metros como o raio da terra. Além disso, para cálculo das fórmulas trigonométricas com a longitude e latitude de arco seno e arco cosseno, as funções escolhidas da biblioteca math retornam valores em radianos, logo devemos fazer a transformação multiplicando-os por $\pi/180$.

Durante o trabalho, será realizado o cálculo de velocidades médias de deslocamento ao longo de trajetórias definidas. Para tanto, é utilizada a relação seguinte:

$$v = \frac{\Delta s}{\Delta t}$$

Onde “v” determina a velocidade média no trajeto, “delta s” a distância percorrida e “delta t” o tempo do deslocamento. São utilizadas as unidades de medida do Sistema Internacional.

O movimento retilíneo uniforme é uma modelagem física para um movimento de aceleração nula, ou seja, cuja velocidade em cada instante permanece constante ao

longo do trajeto. Em um gráfico de posição por tempo, como o seguinte, isso está ilustrado:

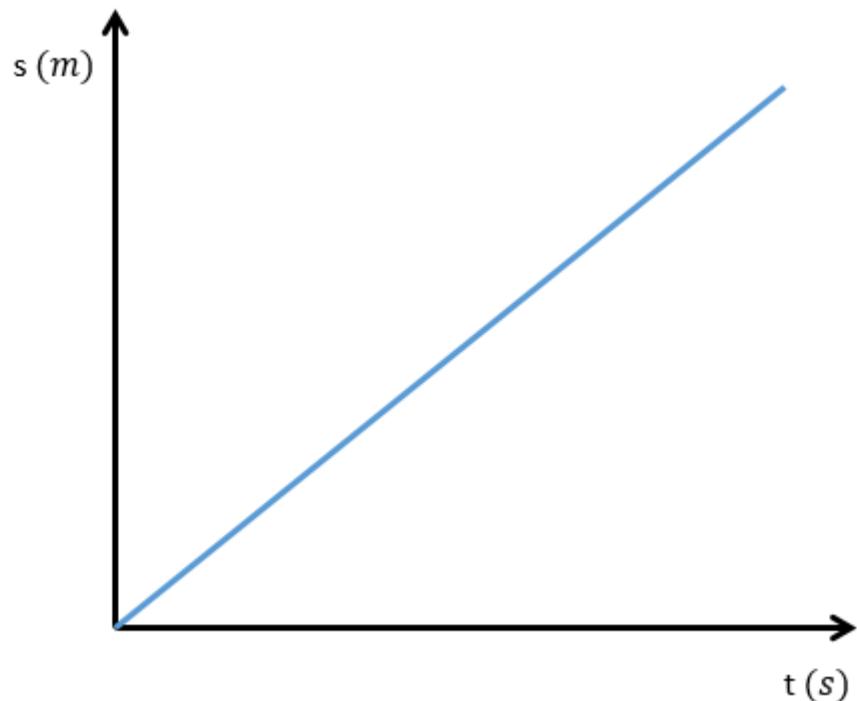


Figura 1: Gráfico de um movimento retilíneo uniforme genérico

Note que o gráfico assume forma reta, com angulação constante em relação aos eixos. Ou seja, a taxa de variação da posição em relação ao tempo (a velocidade), permanece constante.

2. MÉTODO

Cada trecho do KartaView é subdividido em pontos, os quais possuem três informações fundamentais: latitude, longitude e data de captura. Para cada trajetória, capturamos os dados de seus pontos por meio da API pública do KartaView, que, dado o ID de um trecho, retorna um arquivo JSON com uma lista de todos os seus pontos e seus dados.

Entretanto, nem todo o trecho é realizado em MRU e portanto isolamos, de cada trecho completo, um trecho de condições favoráveis para o movimento constante (trecho reto, sem curvas e desníveis). Além disso, queremos trechos delimitados por pontos fiduciais, como placas de quilometragem, de forma que seja possível comparar os resultados obtidos de velocidade e distância com as demarcações da via.

Para tanto, percorremos os trechos escolhidos e buscamos os adequados às exigências, demarcando os índices dos pontos que determinam suas extremidades iniciais e finais.

Após coletar a lista total de pontos de cada trecho pela API, realizamos um slice (retiramos uma fatia) que compreendesse exatamente esses subtrechos previamente selecionados de acordo com os índices. Além disso, removemos todas as informações provenientes da API das quais não faríamos uso nas nossas análises, efetivamente “limpando” os resultados fazendo uso de algoritmo python `photos_extract.py` fornecido durante as aulas.

Após a seleção do subtrecho e limpeza, passamos a calcular os valores de distância entre cada par de pontos sequentes entre si. Para tanto, foram utilizados três métodos de cálculo de distância a partir dos dados de posição global (latitude e longitude), os quais foram descritos na introdução deste texto. Para cada método em cada trecho, foi gerado um vetor de distâncias, totalizando 6 vetores (análogos) de distâncias entre cada par de pontos sequentes.

Note que, para os cálculos Haversine e da Tangente, é necessário um passo auxiliar de transformação dos valores de latitude e longitude de graus para radianos, exigência para utilizar as funções trigonométricas da biblioteca Mat do python.

Posteriormente, para cada trecho, foi gerado um vetor de tempos entre cada par de pontos, utilizando-se da diferença das datas de capturas entre os pontos. Munidos da distância e do tempo entre cada par de pontos, foi calculada também a velocidade média do movimento ao transitar entre eles, para cada método do cálculo de distâncias.

Para ilustrar os resultados, foram plotados dois gráficos para cada trecho: um de velocidades dos subtrechos (entre cada par de pontos) e um de distâncias dos subtrechos. Cada subtrecho está delimitado pela data de captura, então os gráficos foram de: velocidade em função do tempo; distância em função do tempo.

3. VERIFICAÇÃO DO PROGRAMA

A verificação será feita em 3 etapas, uma para cada algoritmo do cálculo da distância:

- Fórmula de Haversine:

$$d = 2r \arcsin \left(\sqrt{\sin^2 \left(\frac{\phi_2 - \phi_1}{2} \right) + \cos \phi_1 \cos \phi_2 \sin^2 \left(\frac{\lambda_2 - \lambda_1}{2} \right)} \right)$$

O código usado para modelar essa forma é o seguinte:

```
from math import asin, sin, cos, sqrt, pi

def degrees_to_rad(x):
    return x * pi / 180

def distancia_haversiana(pp1, pp2):
    raio = 6371000
    pp1 = degrees_to_rad(pp1)
    pp2 = degrees_to_rad(pp2)
    sin_lat = sin(abs(pp2[1] - pp1[1]) / 2)
    sin_lng = sin(abs(pp2[0] - pp1[0]) / 2)
    raiz = sqrt(sin_lat**2 + cos(pp1[1]) * cos(pp2[1]) * sin_lng**2)
    return (2 * raio * asin(raiz))
```

Usamos funções da biblioteca math para auxiliar os cálculos:

- $\sin(x)$: retorna o seno de x radianos;
- $\cos(x)$: retorna o cosseno de x radianos;
- $\arcsin(x)$: retorna em radianos o arco seno de x;
- \sqrt{x} : retorna a raiz quadrada de x;
- π : constante matemática PI.

Lembrando que as funções sin e cos da biblioteca math pedem ângulos em radianos, logo, fazemos a conversão de grau para radiano na função auxiliar, uma vez que os valores da latitude e longitude estão em graus.

Exemplo:

$$\begin{aligned}\varphi_1 &= 0; \lambda_1 = 0; \varphi_2 = \pi/2; \lambda_2 = \pi/2 \\ d &= 2 * r * \arcsin(\sqrt{\sin^2((\pi/2 - 0)/2) + \cos(0) * \cos(\pi/2) * \sin^2((\pi/2 - 0)/2)}) \\ d &= 2 * r * \arcsin(\sqrt{\sin^2(\pi/4) + \cos(0) * \cos(\pi/2) * \sin^2(\pi/4)}) \\ d &= 2 * r * \arcsin(\sqrt{(\sqrt{2}/2)^2 + 1 * 0 * (\sqrt{2}/2)^2}) = 2 * r * \arcsin(\sqrt{1/2}) \\ d &= 2 * r * \arcsin(\sqrt{2}/2) = 2 * r * \pi/4 = r * \pi/2 \\ \text{Sendo } r &= 6371000m, \text{ então: } d = 6371000 * \pi/2 \approx 10007543.4m\end{aligned}$$

No exemplo usamos valores em radianos para a fórmula, porém a função `distancia_haversine` recebe os pontos com latitude e longitude em graus, portanto, precisamos usar o equivalente a 0 e meio PI em graus.

Sendo o valor retornado pela nossa função para os parâmetros [0,0] e [90,90] é 10007543.398010284, muito próxima ao esperado.

- Fórmulas Trigonométricas:

$$C = \sin(\phi_1) \sin(\phi_2) + \cos(\lambda_2 - \lambda_1) \cos(\phi_1) \cos(\phi_2)$$

$$c = \frac{r\pi}{180} \arccos(C)$$

O código usado para a modelagem é o seguinte:

```
from math import acos, sin, cos, pi

def degrees_to_rad(x):
    return x * pi / 180

def distancia_trigonometrica(pp1, pp2):
    pp1 = degrees_to_rad(pp1)
    pp2 = degrees_to_rad(pp2)
    C = sin(pp1[1])*sin(pp2[1]) + cos(pp1[0]-pp2[0])*cos(pp1[1])*cos(pp2[1])
    raio = 6371000
    return raio * acos(C)
```

Usamos funções da biblioteca math para auxiliar os cálculos:

- $\sin(x)$: retorna o seno de x radianos;
- $\cos(x)$: retorna o cosseno de x radianos;
- $\arccos(x)$: retorna em radianos o arco cosseno de x;
- π : constante matemática Pi.

Lembrando que as funções sin e cos da biblioteca math pedem ângulos em radianos, logo, fazemos a conversão de grau para radiano na função auxiliar, uma vez que os valores da latitude e longitude estão em graus. Além disso, as fórmulas trigonométricas teóricas usam ângulos em graus, ou seja, espera que arco cosseno retorne valores em graus, portanto, como a função acos já retorna em radianos, não precisamos nos preocupar com isso, logo:

$$c = r * \arccos(C)$$

Exemplo:

$$\varphi_1 = 0; \lambda_1 = 0; \varphi_2 = \pi/2; \lambda_2 = \pi/2$$

$$C = \sin(0) * \sin(\pi/2) + \cos(\pi/2 - 0) * \cos(0) * \cos(\pi/2)$$

$$C = 1 * 0 + 0 * 1 * 0 = 0$$

$$c = r * \arccos(0) = 6371000 * \pi/2 = 10007543.4$$

O valor retornado pela nossa função com os parâmetros [0,0] e [90,90] foi 10007543.398010286, muito próximo ao valor esperado.

- Método de Mercator:

$$p_1 = (x_1, y_1); p_2 = (x_2, y_2)$$
$$d(p_1, p_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

O código usado para a modelagem é o seguinte:

```
def distancia_euclidiana(v1, v2):
    distancia = (((v2[0] - v1[0])**2) + ((v2[1] - v1[1])**2))**(1/2)
    return distancia
```

Exemplo:

$$p_1 = (0, 0); p_2 = (3, 4)$$

$$d(p_1, p_2) = \sqrt{(3 - 0)^2 + (4 - 0)^2} = \sqrt{9 + 16} = \sqrt{25} = 5$$

Com o mesmo retorno da função feita com os parâmetros [0,0] e [3,4].

4. DADOS

Seguem abaixo os resultados obtidos para os cálculos realizados programaticamente durante a pesquisa.

4.1 Tabelas

	VELOCIDADE MÉDIA NO PERCURSO (em m/s)	
MÉTODOS/PAÍSES	BRASIL	ESPAÑA
MERCATOR	22.95748783284246	28.08969036544217
HAVERSINE	21.64813771935627	21.248626030589303
TRIGONOMÉTRICA	21.648114667925835	21.248637828533116

Tabela 1: Velocidade Média no Percurso (em m/s). Relação das velocidades médias totais nos subtrechos calculados em cada método.

	DISTÂNCIA TOTAL DO PERCURSO (em m)	
MÉTODOS/PAÍSES	BRASIL	ESPAÑA
MERCATOR	3214.0482965979445	3988.736031892788
HAVERSINE	3030.739280709878	3017.3048963436813
TRIGONOMÉTRICA	3030.736053509617	3017.3065716517026

Tabela 2: Distância Total do Percurso (em m). Relação das distâncias totais dos subtrechos calculados em cada método.

4.2 Figuras

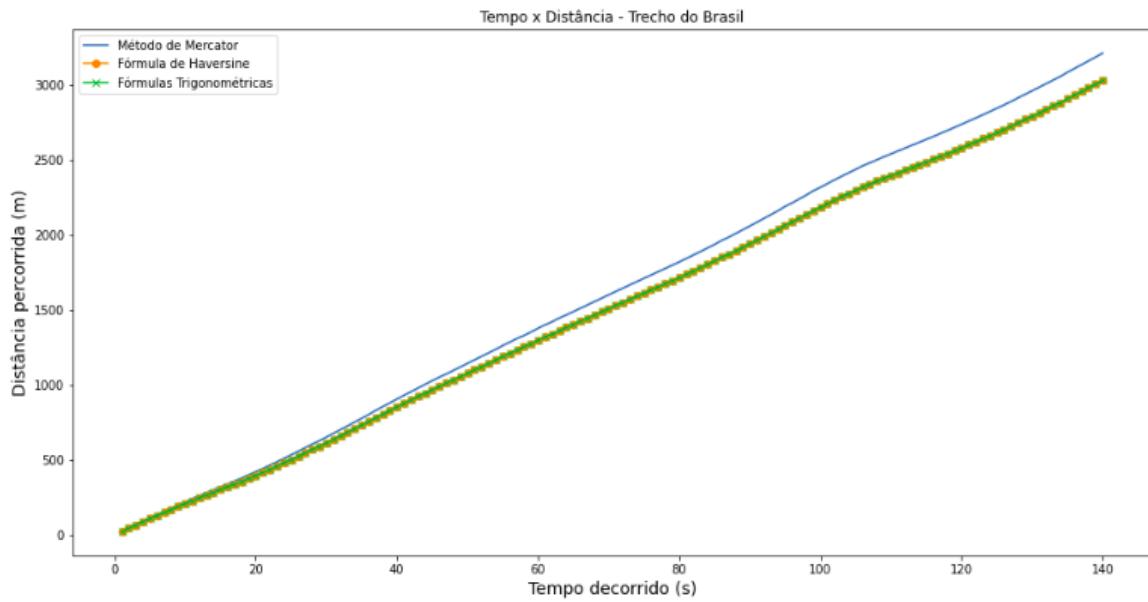


Figura 2: Gráfico da Distância total percorrida no Trecho do Brasil em função do tempo, de acordo com cada método de cálculo.

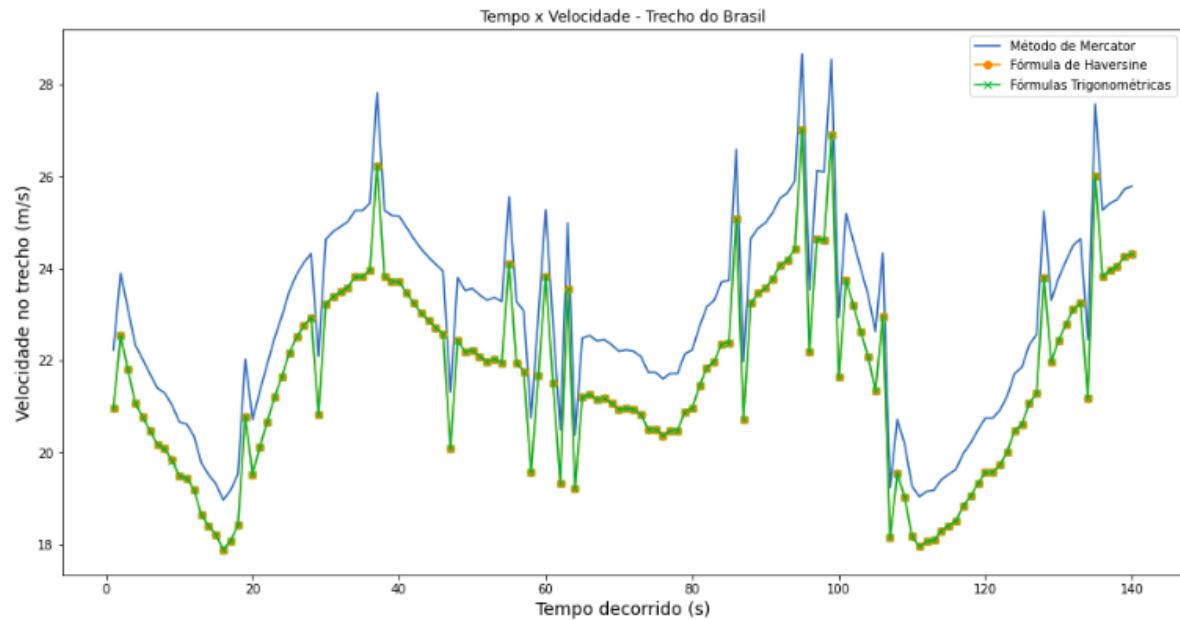


Figura 3: Gráfico da velocidade média em cada subtrecho do Trecho do Brasil em função do tempo, de acordo com cada método de cálculo.

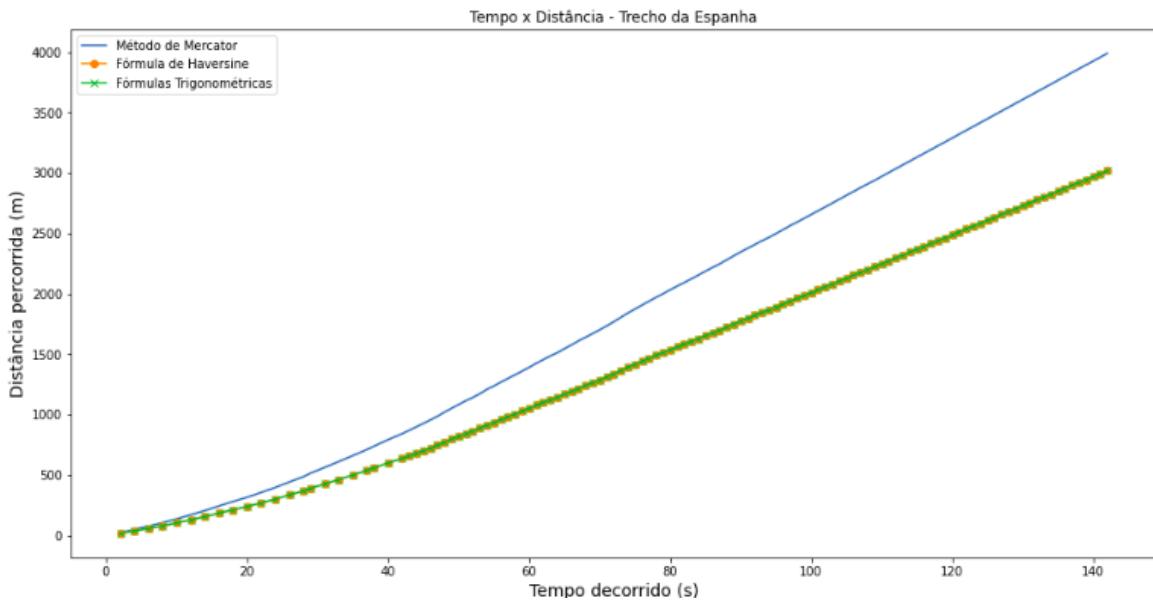


Figura 4: Gráfico da Distância total percorrida no Trecho da Espanha em função do tempo, de acordo com cada método de cálculo.

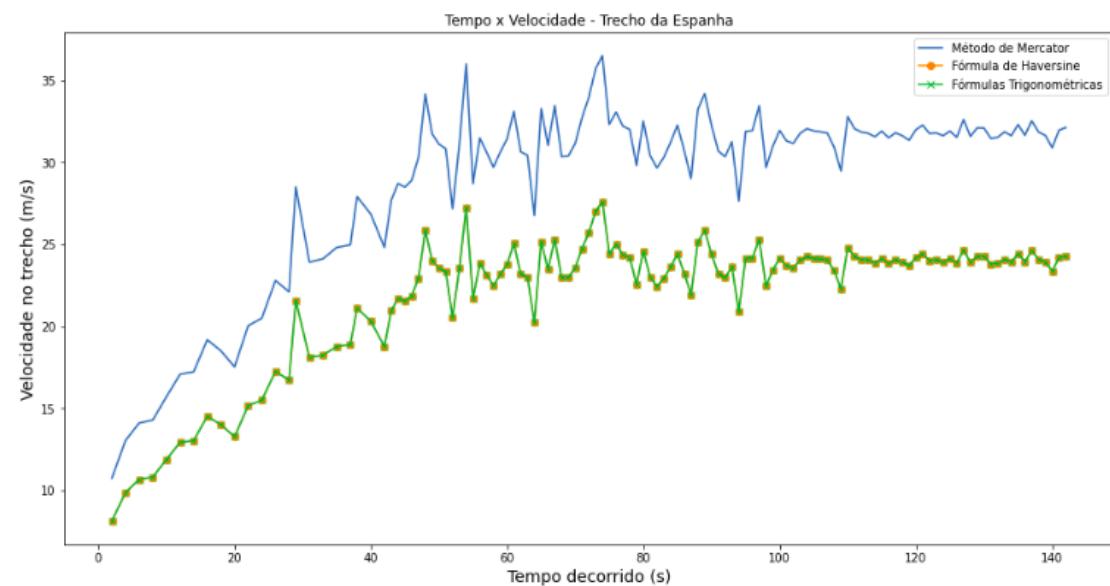


Figura 5: Gráfico da velocidade média em cada subtrecho do Trecho da Espanha em função do tempo, de acordo com cada método de cálculo.

4.3 Pontos fiduciais

4.3.1 Trecho Brasil



Figura 6. Ponto inicial do trajeto. Lê-se: “quilômetro 105”



Figura 7. Ponto final do trajeto. Lê-se: “quilômetro 102”

Distância total calculada: 3000 metros.

Velocidade média calculada: 21.42 m/s.

4.3.2 Trecho Espanha



Figura 8. Ponto inicial do trajeto. Lê-se: “quilômetro 76”



Figura 9. Ponto final do trajeto. Lê-se: “quilômetro 79”

Distância total calculada: 3000 metros.

Velocidade média calculada: 21.12 m/s.

5. ANÁLISE

De acordo com a Tabela 1, pode ser verificado que os valores de distância total, tanto para o trecho da Espanha quanto para o trecho do Brasil, são muito aproximados (diferenças menores do que 0.1%) quando calculadas nos métodos da Tangente e Haversine, enquanto o método de Mercator possui diferenças consideráveis: no trecho da Espanha, a diferença relativa de Haversine/Tangente para Mercator é de aproximadamente 30%; no trecho do Brasil, essa diferença entre os métodos é de 6%.

O mesmo princípio rege os resultados das distâncias acumuladas em cada ponto dos trechos. Tais relações de proximidade/distância dos valores estão ilustradas nas figuras 2 e 4. Veja que as linhas que correspondem aos resultados obtidos pelos cálculos de Haversine e Tangente são coincidentes, enquanto as linhas de Mercator se diferenciam das outras na maioria dos pontos.

Uma análise análoga pode ser realizada para as velocidades médias de percorrido dos trajetos. De acordo com a Tabela 2, podemos notar que as diferenças entre os resultados obtidos com o Método de Haversine e o da Tangente são insignificantes (muito menores que 0.1%), para ambos os trechos. No entanto, o método de Mercator gera diferenças com os outros valores de aproximadamente 29% no Trecho da Espanha e por volta de 5% no trecho do Brasil.

Nos dados acima, a distância real, medida entre os pontos fiduciais, é de 3000 metros para os dois trechos. Calculando essa mesma distância utilizando a fórmula de Haversine, encontra-se diferença de até 30 metros; já utilizando trigonometria esférica vemos diferenças de cerca de 17 metros, ambos valores abaixo de 0.1% de erro relativo ao esperado.

Note que os valores de erro para a distância total calculada usando a projeção de Mercator são de: 214 metros (7%) para o trecho no Brasil; 988 metros (30%) para o trecho Espanhol.

O mesmo padrão de erros se observa nos valores calculados para velocidade média no trajeto. Podemos averiguar que, respectivamente nos trajetos do Brasil e Espanha, os erros no cálculo da velocidade média total são: por Mercator, 7% e 32%; por Haversine e Trigonometria, 1% e 0.5%;

Ainda sobre as figuras 2 e 4, é possível observar que os gráficos que descrevem as trajetórias dos movimentos assumem formas muito próximas às de retas, mantendo angulação aproximadamente constante em relação aos eixos, na maior parte do tempo.

6. INTERPRETAÇÃO

Como esperado, os métodos de Haversine e da Tangente geram resultados significativamente distantes dos valores obtidos pela projeção de Mercator, pois consideram a Terra como uma esfera de raio fixo, enquanto, neste último método, a Terra é tomada como uma elipse.

A distância dos resultados por Mercator em relação aos valores obtidos pelos pontos fiduciais pode ser compreendida ao se levar em conta que a projeção realiza efetivamente uma distorção nas dimensões da superfície do planeta. Além disso, essa distorção não ocorre de forma uniforme em todo o globo: a distorção observada na Espanha é de fato mais impactante do que a observada no Brasil, justamente porque o país europeu está mais distante da linha do Equador.

Já os resultados de Haversine e Tangente não realizam projeção planificada, então não geram tanta distorção no trecho Espanhol em comparação com o Brasileiro. Note que estes resultados também não são iguais aos valores obtidos pelos pontos fiduciais; isso decorre da necessidade de aproximação do raio terrestre a um valor fixo para a realização dos cálculos, o que acaba por gerar um erro já que a Terra não é uma esfera perfeita.

Note que os movimentos descritos em ambos os trechos podem ser classificados como MRU, a partir de certos pontos, pois os gráficos que descrevem as trajetórias de Posição x Tempo assumem formas aproximadas às de retas.

7. CRÍTICA

Com este trabalho tivemos a oportunidade de aprender mais sobre formas variadas de modelar um problema, e também nos familiarizar com a modelagem desses sistemas físicos reais, tendo de simular-los através da implementação de algoritmos.

Para calcular as distâncias no globo terrestre, aprendemos métodos como utilizar a projeção das coordenadas esféricas no plano, a fórmula de Haversine, e a trigonometria esférica.

Conseguimos observar que dentre os diferentes métodos, existem uns mais adequados que outros para a modelagem daquele sistema em específico. Em nosso caso, os métodos mais indicados para tratar os dados e calcular a distância seriam os pela fórmula de Haversine e pela Trigonometria esférica, onde esses possuem resultados bem próximos um do outro, e com uma margem de erro aceitável do resultado real, cerca de 1%. Já utilizando o método da projeção por Mercator, vemos diferenças na faixa dos 30% do resultado real, o que acabaria por impactar grotescamente na percepção e modelagem do sistema.

Computacionalmente, aprendemos como coletar dados de APIs e tratar seus respectivos arquivos ‘.json’, fazendo manipulações e limpando todos os dados recebidos. Aprendemos também a utilizar o pacote pyproj, que faz a projeção de coordenadas esféricas do WGS84 (já levando em conta o modelo usado para o planeta) para coordenadas no plano.

8. LOG

Coletar Pontos fiduciais	2h
Limpar pontos	0h45
Organizar e comentar código	1h
Cálculos de Distâncias/Velocidades	2h
Geração dos Gráficos/tabelas	3h30
Montagem do Relatório	8h30

..