Map Matching: Uma análise de Dados *Streaming* de trajetórias de GPS no Transporte Público

Tiago Stapenhorst Martins[1],**Nádia P. Kozievitch**[1],**Tatiana Gadda**[1], **Marcelo Rosa**[1]
Matheus B. Gutierrez[1]

[1]Federal University of Technology of Paraná (UTFPR), Curitiba, PR, Brazil ,

tiagosmx@gmail.com,{nadiap, tatianagadda, mrosa}@utfpr.edu.br,

{matheusgutierrez}@alunos.utfpr.edu.br

Abstract. *With the cost-effectiveness and installation of IoT devices, embedded devices, sensors and GPS in public buses, a large amount of data can be generated and used as a basis for decision-making. However, if the data is affected by errors and uncertainties, such analyzes may be invalid. This work presents a prototype with a Map Matching method to detect and extract Streaming data from GPS trajectories in a transport network and its bus stops. The article uses concepts from Geographic Information Systems, Map Matching and Smart Cities. Tests using Curitiba's public transport GPS trajectory Streaming data illustrated the efficiency of the Map Matching algorithm methodology (using the GPS trajectories of vehicles and their bus stops), besides indicating factors for improvement of the data.*

Resumo. *Com o barateamento e a instalação de dispositivos IoT, embarcados, sensores e GPS em ônibus públicos, uma grande quantidade de dados pode se gerada e utilizada como base para tomadas de decisão. Entretanto se os dados forem afetados por erros e incertezas tais análises podem ser inválidas. Este trabalho apresenta um protótipo com um método de Map Matching para detectar e extrair dados de Streaming de trajetórias GPS em uma rede de transporte e seus pontos de ônibus. O artigo utiliza conceitos de Sistemas de Informações Geográficas, Map Matching e Cidades Inteligentes. Testes realizados em dados de Streaming de trajetórias de GPS de transporte público de Curitiba ilustraram a eficiência da metodologia do algoritmo de Map Matching (usando as trajetórias de GPS de veículos e seus pontos de ônibus), além de indicar fatores para melhoria dos dados.*

1. Introdução

As companhias que gerenciam o transporte público estão cada vez mais utilizando tecnologias de transmissão de dados em *streaming* para monitorar a sua frota e trazer mais dados aos usuários e gestores. Dentro dos Grandes Desafios da Computação no Brasil já foi mencionado que o descobrimento de padrões, integração de dados, estatísticas de dados abertos e conectados são desafios comuns na área[1].

Em busca de atrair novos usuários e satisfazer os atuais usuários de transporte público, melhorando a qualidade do serviço, empresas e pesquisadores em transporte

[1]http://www.sbc.org.br/documentos-da-sbc/send/141-grandes-desafios/802-grandesdesafiosdacomputaono-brasil

Figura 1. Trajeto feito pelo ônibus BA125 da linha 216 no dia 03/05/2019 segundo os dados abertos da URBS.

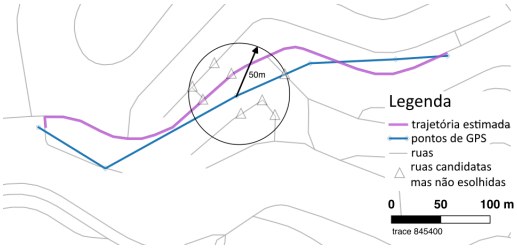

Figura 2. Algoritmo de *Map Matching* em ação sobre pontos de GPS. Modificado de [Millard-Ball et al. 2019].

se voltam aos Sistemas de Informações Geográficas (SIGs), Sistemas de Transporte Inteligentes (STIs), aliados aos Sistemas de Navegação por Satélite (GNSS, *Global Navigation Satellite System*), comumente chamados de *Global Positioning System* (GPS) [I. Meneguette et al. 2018].

Na cidade de Curitiba, todos os ônibus da frota de transporte público são operados por empresas privadas através de licitação, porém sua gestão é feita pelo município através da empresa estatal Urbanização de Curitiba (URBS), a qual equipou os ônibus, desde 2012[2], com dispositivos GPS atrelados à um computador de bordo. A partir de 2017, estes mesmos dados passaram a ser disponibilizados ao público, sendo processados, armazenados em arquivos diários e acessíveis na internet sem necessidade de qualquer tipo de cadastro ou custo[3] (como ilustra a Figura 1).

Este trabalho apresenta o emprego de um método de *Map Matching* (algoritmos usados para fixar dados de localização em uma rede espacial de estradas[Pereira et al. 2009]) para detectar e extrair dados de *Streaming* de trajetórias GPS em uma rede de transporte e seus pontos de ônibus, usando conceitos de SIG, *Map Matching* e Cidades Inteligentes. A abordagem possui as seguintes contribuições: (1) detecção de paradas de veículos em pontos de ônibus próximos em uma rua de mão dupla; (2) imprecisões do GPS (como um ônibus que passou no meio da quadra na Figura 1); e (3) horário exato da passagem do veículo em um ponto de ônibus (já que a tabela de previsão só inclui alguns pontos de checagem).

2. Trabalhos Relacionados

SIGs podem conter ferramentas para realizar operações com dados geográficos e uma das mais comuns é o cálculo da distância de *Haversine*, que compreende a menor distância entre dois pontos na superfície da Terra (ignorando as deformações naturais da superfície) [Panigrahi 2014]. A distância de *Haversine* considera a curvatura da terra, é relativamente leve (de uma perspectiva de custo de processamento) e precisa na maioria dos casos por sofrer erros em até um metro [Lawhead 2015].

[2]https://www.gazetadopovo.com.br/vida-e-cidadania/informacao-sobre-linhas-de-onibus-e-deficiente-2bd8wsjqxmv4rz5jipbfoehce/

[3]https://www.curitiba.pr.gov.br/dadosabertos/busca/

Ao se analisar a movimentação de veículos com dados de GNSS (GPS) dentro de um SIG alguns problemas ocorrem, e são eles [Millard-Ball et al. 2019]: (1) Falta de dados e (2) Imprecisões do dispositivo GNSS (GPS).

Como definido em [Pereira et al. 2009], *Map Matching* são algoritmos usados para fixar dados de localização em uma rede espacial de estradas, podendo ter diversas aplicações, sendo a mais usual para dispositivos de GPS. Na Figura 2 temos uma típica representação do problema: a trajetória estimada da linha de ônibus (linha roxa), a trajetória do veículo (linha azul) e a correspondência das mesmas (com uma diferença de 50 metros). Nota-se que ela ultrapassa os limites físicos das ruas (representadas pelas linhas pretas), percorrendo sobre canteiros e possivelmente construções. Os algoritmos de *Map Matching* resolvem este problema estimando qual seria a trajetória, mais realista, do veículo dentro dos limites das ruas (linha roxa).

O *Map Matching* pode ser realizado de diferentes formas por algoritmos diferentes. Alguns algoritmos funcionam em tempo real (*streaming*) enquanto outros funcionam em dados históricos (*offline*) ou em ambos. Os modelos por trás de um algoritmo de *Map Matching* podem ser diversos e difíceis de categorizar. Entretanto alguns autores criaram um modelo para facilitar este trabalho [Chao et al. 2019], sendo: Modelo de Similaridade, Modelo de Transição de Estado, Modelo Candidato Evolutivo, e Modelo de Pontuação. Além dos modelos e técnicas descritos na literatura, também encontram-se inúmeros *softwares* prontos para realização de *Map Matchings*, como Mapzen Valhalla [4], Open Source Routing Machine (OSRM) [5], GraphHopper [6], e pgMapMatch [7] [Millard-Ball et al. 2019].

No âmbito acadêmico, dentro do tópico do transporte curitibano, algumas teses, dissertações e artigos exploraram: as distribuições dos pontos de ônibus pela cidade em busca de detectar locais pouco abastecidos em termos de transporte [Da Silva et al. 2016], estimativas de origem e destino dos passageiros de ônibus [Diniz Junior 2017] com foco em unidades de saúde e educação [Souza et al. 2020], análise exploratória geral [Rodriguez Vila et al. 2016, Parcianello et al. 2018, Barczyszyn 2015] e desenvolvimento de software de caronas solidárias (*carpooling*) para a comunidade universitária [De Lara and Stapenhorst Martins 2016, Martins and Kozievitch 2015] (trabalhos anteriores dos atuais autores). É válido ressaltar que muitos desafios foram encontrados com o uso desses dados utilizados nos artigos referenciados aqui. Problemas como ambiguidades nos nomes das ruas, dados incorretos e inconsistências são apontados por diversos autores [Rodriguez Vila et al. 2016, Barczyszyn 2015, Parcianello et al. 2018].

Dentre os desafios podemos citar: (1) detecção de paradas de veículos em pontos de ônibus próximos em uma rua de mão dupla (um ponto de ônibus da linha em um sentido (“ida”) e a outra ao sentido oposto da linha (“volta”)); (2) imprecisões do GPS (como um ônibus que passou no meio da quadra na Figura 1); e (3) horário exato da passagem do veículo em um ponto de ônibus (já que a tabela de previsão só inclui alguns pontos de checagem). Este trabalho apresenta uma abordagem para resolver os três desafios através do *Map Matching*.

[4] https://github.com/valhalla/valhalla
[5] http://project-osrm.org/
[6] https://github.com/graphhopper/graphhopper
[7] https://github.com/amillb/pgMapMatch

3. Metodologia e Protótipo

Para os testes foram utilizados dados da linha 216-Cabral/Portão, com as características listadas abaixo. Detalhes adicionais dos dados podem ser encontrados em [Rodriguez Vila et al. 2016].

- "Linhas", contendo dados de código, nome, cor, categoria do serviço e se aceita somente cartão;
- "Veiculos" apresenta uma série temporal da trajetória de GPS de cada ônibus (em média a cada 20 segundos) e qual linha ele está operando;
- "PontosLinha" apresenta uma descrição dos pontos de ônibus dentro do escopo da linha de ônibus;
- "TabelaVeiculo" determina o horário teórico que cada ônibus deve passar nos pontos de ônibus principais (nem todos os pontos de ônibus tem um horário teórico);
- "TabelaLinha"descreve o horário tabelado de passagem dos ônibus por linha em seus respectivos pontos de ônibus.

As tecnologias utilizadas são: Javascript, Node-RED[8]. As bibliotecas utilizadas são: Turf.JS[9], Lodash[10], Luxon[11], Leaflet[12], lzma-native[13].

Para estimar o momento de passagem dos ônibus em seus respectivos pontos, foi preciso realizar processamentos sobre os dados já que esta informação não é explícita. Este processamento pode ser feito tanto sobre dados históricos (*offline*) como dados de *streaming* em tempo real (*online*).

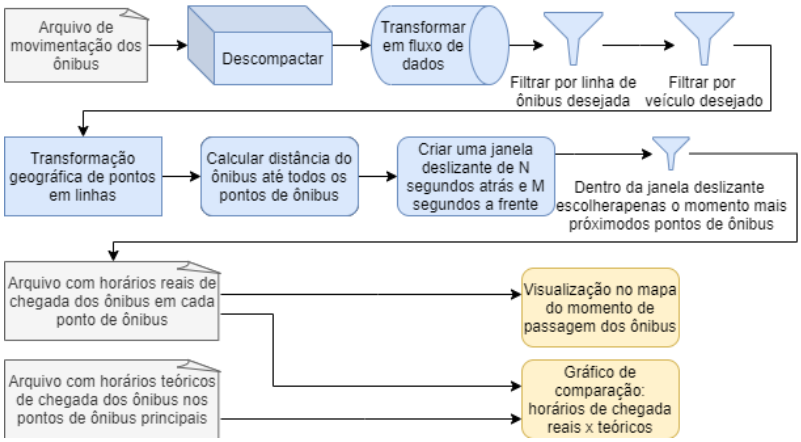

Figura 3. Fluxograma explicando a ordem do processamento de dados.

O processamento utilizado é composto por 11 etapas, descritas pela Figura 3:

1. Download do arquivo "Veiculos" do dia desejado.
2. Descompactação do arquivo "Veiculos" (formato xz).
3. Leitura do arquivo descompactado.

[8] https://nodered.org/
[9] https://turfjs.org/
[10] https://lodash.com/
[11] https://moment.github.io/luxon/
[12] https://leafletjs.com/
[13] https://github.com/addaleax/lzma-native

4. Recorte de dados da linha e de um ônibus escolhidos.
5. Realização do *parse*, extraindo os dados de GPS contidos em formato texto para o formato apropriado que a biblioteca de processamento geográfico (SIG) consiga processar.
6. Para cada posição do ônibus no mapa, calcula-se a distância *Haversine* para todos os pontos de ônibus daquela linha e armazena-as.
7. Cálculo da janela móvel (com relação ao número de amostras) de tamanho N (pode ser escolhido arbitrariamente - recomenda-se de tamanho cinco ou mais), que se move a medida que novos dados chegam.
8. Comparação da posição i do centro da janela (terceiro, caso o tamanho da janela seja cinco) com os outros elementos. Se o elemento do centro for o ponto mais próximo de algum ponto de ônibus daquela linha e também menor que um valor arbitrário M, ele é o ponto vencedor e seu horário representa o horário de chegada daquele ônibus em um dos pontos de ônibus. Na Tabela 1, por exemplo, a leitura de id 4 apresentou maior proximidade com o ponto de ônibus 1 (PO1), logo ela será escolhida como *match*.
9. Armazenamento dos horários reais de chegada do ônibus nos pontos em arquivo (para economizar tempo em futuros processamentos).
10. Carga dos horários teóricos de chegada do ônibus nos pontos já analisados.
11. Criação e visualização do gráfico com os horários reais de chegada de ônibus juntamente com os horários teóricos, como ilustra a Figura 7. Cada linha do gráfico representa um ponto de ônibus diferente, o eixo X é o tempo, os momentos teóricos de passagem de ônibus é representado pelos círculos azuis e os momentos reais de passagem pelos círculos laranjas.
12. Criação e visualização do mapa com os momentos em que o ônibus passou pelos pontos de ônibus, ilustrado pelas Figuras 4 e 6. A interface habilita os filtros de elementos no mapa (pontos, linhas) e exibe detalhes do *match* .

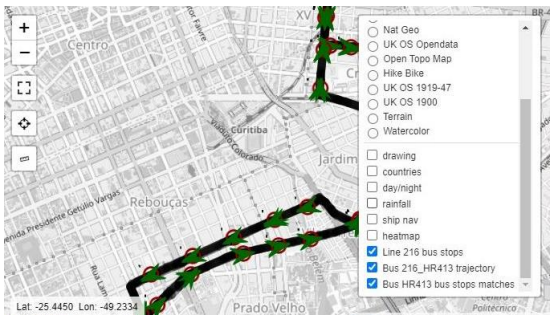

Figura 4. Resultado do Map Matching em interface da aplicação, em que o trajeto do ônibus foi coincidido com a localização real dos pontos de ônibus da linha (sobre o mapa do Open Street Map).

Um dos diferenciais da abordagem proposta é que ela realiza o *map matching* utilizando apenas dados das trajetórias GPS dos ônibus e localização dos pontos de ônibus, implicando, na prática, em uma maior economia de memória e processamento (pois não há necessidade de utilizar dados da malha viária da região analisada). Em outras palavras, é uma forma de *map matching* simplificada e adaptada para o caso específico de passagem de ônibus em seus pontos.

Id	Horário leitura	Distância Haversine até o PO 1	Distância Haversine até o PO 2	Distância Haversine até o PO 3
1	10:00:00	100m	500m	700m
2	10:00:30	60m	503m	710m
3	10:01:05	40m	510m	740m
4	10:01:35	2m	550m	770m
5	10:02:00	20m	590m	790m

Tabela 1. Exemplo da janela de tempo usada para realizar o matching de posições do ônibus com pontos de ônibus (PO).

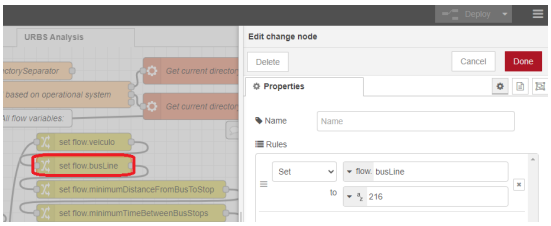

Figura 5. Os parâmetros de execução da aplicação (linha de ônibus).

Figura 6. Resultado do *Map Matching* exibidas pela interface gráfica.

Antes do algoritmo de *matching* ser iniciado, alguns parâmetros precisam ser ajustados: 1) a distância mínima entre o veículo observado e o ponto de ônibus; 2) o tempo mínimo entre dois pontos de ônibus; 3) o tamanho da janela de amostras (um número ímpar); 4) a linha de ônibus escolhida para análise e 5) o código único do veículo. A inicialização destes parâmetros é feita alterando-se os parâmetros dos respectivos blocos de configuração na interface de configuração da aplicação, como demonstrado pela Figura 5.

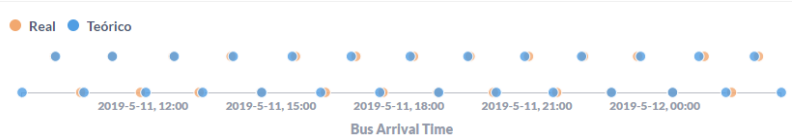

Figura 7. Horários de chegada programados/teóricos (em azul) em comparação com os reais (em laranja) para o ônibus BA600 na linha 216 no dia 11/05/2019.

A estimativa de chegada nos pontos de ônibus mostrada na Figura 4 foi realizada para o veículo BA600 da linha 216 (Cabral/Portão) no dia 11/05/2019. Os círculos vermelhos mostram os pontos de ônibus da linha, as setas verdes o *matching* da passagem de um ônibus por eles e em preto o trajeto percorrido pelo ônibus. Internamente estão armazenados dados como direção em que o ônibus se movimenta em graus (também chamado de azimute, direção ou *bearing*), o horário de passagem do ônibus naquele ponto, a distância mínima do ônibus até o ponto de ônibus referenciado, identificador do veículo e código da linha de ônibus (Figura 6).

Após o processamento do algoritmo, é possível comparar graficamente o horário teórico de chegada dos ônibus com o horário real (Figura 7). O eixo X apresenta o tempo; cada círculo colorido representa a passagem (real em laranja ou teórica em azul) do veículo em um ponto de ônibus; no eixo Y é possível distinguir qual ponto de ônibus

foi visitado. É possível notar que existe uma tendência deste ônibus passar de forma adiantada pela manhã e de atrasar de tarde e noite. A linha 216 (Cabral/Portão), apesar de ter previsão de dois pontos de checagem de horários (Terminal Portão Terminal Cabral)[14], teve, com a metodologia proposta deste trabalho, a estimativa de horário para os setenta pontos de ônibus da linha.

Dentre as inconsistências encontradas, podemos citar: (1) veículos que realizam trajetos de linhas que não estão listadas (o veículo BA125 realiza o trajeto da linha 216, mas tem programação e horários só para a linha de código 233); (2) horários cadastrados para pontos de ônibus não existentes na linha (como a linha 216 programada para o Terminal Pinheirinho); 3) trajetória dos ônibus que ultrapassam os limites físicos das ruas (atravessando terrenos, prédios, casas, etc), ou com pontos de ônibus contendo o lado incorreto das portas; e (4) Pontos de ônibus cadastrados com distâncias muito afastadas das ruas (aqui um trabalho futuro poderia incluir se o registro é devido à imprecisão do dado ou a localização do ponto em uma área). Uma forma de revisão das inconsistências é utilizar uma base de dados no formato GTFS, acessíveis através da API fechada da URBS.

De maneira resumida, através da metodologia proposta foi possível verificar: (1) detecção de paradas de veículos em pontos de ônibus próximos em uma rua de mão dupla; (2) imprecisões do GPS (como um ônibus que passou no meio da quadra na Figura 1); e (3) horário exato (com precisão de segundos) da passagem do veículo em um ponto de ônibus (já que a tabela de previsão só inclui alguns pontos de checagem). Estas informações poderiam ser usadas, por exemplo, para o serviço de manutenção dos equipamentos de GPS, a predição do horário de chegada em todos os pontos de ônibus de uma linha (com precisão de segundos), e detecção de veículos fora da rota.

4. Conclusão

Com o crescimento dos grandes centros urbanos, as organizações responsáveis por gerenciar o transporte público de ônibus precisam buscar formas de melhorar a qualidade, eficácia e eficiência do transporte público. Este trabalho apresenta o emprego de um método de *Map Matching* para detectar e extrair dados de *Streaming* de trajetórias GPS em uma rede de transporte e seus pontos de ônibus. Os testes foram realizados na linha 216-Cabral/Portão, e dentre as limitações da abordagem, podemos citar os parâmetros utilizados (como a linha de ônibus escolhida, o veículo e a janela de amostras). Em particular, a abordagem aborda os seguintes desafios: (1) detecção de paradas de veículos em pontos de ônibus próximos em uma rua de mão dupla; (2) imprecisões do GPS; e (3) horário exato (com precisão de segundos) da passagem do veículo em um ponto de ônibus. Como trabalhos futuros podemos citar a comparação com outros algoritmos, a sua otimização, a inclusão e testes com outras linhas de ônibus, a estimativa de horários teóricos mais compatíveis com os dados, a criação de uma metodologia para definição de horários teóricos para todos os pontos de ônibus da linha, entre outros.

5. Agradecimentos

Os autores gostariam de agradecer a Universidade Tecnológica Federal do Paraná (UTFPR) (DIREC01/2020), IPPUC, URBS e Prefeitura da cidade de Curitiba.

[14] `https://www.urbs.curitiba.pr.gov.br/horario-de-onibus/216`

Referências

Barczyszyn, G. L. (2015). Integração de dados geográficos para planejamento urbano da cidade de curitiba. Master's thesis, UTFPR.

Chao, P., Xu, Y., Hua, W., and Zhou, X. (2019). A Survey on Map-Matching Algorithms. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 12008 LNCS:121–133.

Da Silva, E. L. C., Rosa, M. D. O., Fonseca, K. V. O., Luders, R., and Kozievitch, N. P. (2016). Combining K-means method and complex network analysis to evaluate city mobility. In *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, volume 19, pages 1666–1671, Rio de Janeiro. IEEE.

De Lara, F. R. and Stapenhorst Martins, T. (2016). Protótipo de sistema de caronas aplicado na Universidade Tecnológica Federal do Paraná. Monografia (Bacharel em Informática), UTFPR.

Diniz Junior, P. C. (2017). Serviços Telemáticos em uma Rede de Transporte Público Baseados em Veículos conectados e Dados Abertos. Master's thesis, UTFPR, Curitiba.

I. Meneguette, R., E. De Grande, R., and A. F. Loureiro, A. (2018). *Intelligent Transport System in Smart Cities*. Urban Computing. Springer International Publishing, Cham, 1 edition.

Lawhead, J. (2015). *Learning Geospatial Analysis with Python*, volume 7. Packt Publishing, Birmingham, 2nd edition.

Martins, T. S. and Kozievitch, N. P. (2015). Os desafios de uma aplicação de Carpooling no contexto de uma comunidade universitária brasileira. *XI Escola Regional de Banco de Dados*, XI:10.

Millard-Ball, A., Hampshire, R. C., and Weinberger, R. R. (2019). Map-matching poor-quality GPS data in urban environments: the pgMapMatch package. *Transportation Planning and Technology*, 42(6):539–553.

Panigrahi, N. (2014). *Computing in Geographic Information Systems*. CRC Press, Boca Raton, Florida, 1st edition.

Parcianello, Y., Kozievitch, N. P., Fonseca, K. V. O., Rosa, M. D. O., Gadda, T. M. C., and Malucelli, F. C. (2018). Transportation: An Overview from Open Data Approach. In *2018 IEEE International Smart Cities Conference (ISC2)*, pages 1–8, Kansas City. IEEE.

Pereira, F. C., Costa, H., and Pereira, N. M. (2009). An off-line map-matching algorithm for incomplete map databases. *European Transport Research Review*, 1(3):107–124.

Rodriguez Vila, J. J., Kozievitch, N. P., Gadda, T. M. C., Fonseca, K. V. O., Rosa, M. O., Gomes-Jr, L. C., and Akbar, M. (2016). Urban Mobility Challenges - An Exploratory Analysis of Public Transportation Data in Curitiba. *Revista de Informática Aplicada*, 12(1):14.

Souza, F. X. D., Bauer, P. R., Fonseca, K., Gadda, T., and Berardi, R. (2020). Visualização de Dados de Origem-Destino - Foco em Unidades de Saúde e Educação. *Geoinfo*, 1(21):6.