Algoritmo para Detecção de Itinerários do Transporte Público Usando Dados de GPS dos Ônibus

Júlio C. N. Borges[1], **Ricardo Lüders**[1], **Thiago Silva**[1], **Anelise Munaretto**[1]

[1]Universidade Tecnológica Federal do Paraná (UTFPR)
Av. Sete de Setembro, 3165 – Rebouças – CEP 80230-901 – Curitiba – PR – Brasil

julio.2018@alunos.utfpr.edu.br, {luders,thiagoh,anelise}@utfpr.edu.br

Abstract. *Many cities have a system for tracking the movement of public transport buses. The route detection problem defined in this article consists of matching the geolocation records of the bus movement with its respective route. This article proposes an algorithm for detecting routes based on data from the GPS of buses and the location of bus stops. As this problem is affected by failures in the sending of GPS data, geolocated temporal sequences that are incompatible with the programmed itinerary need to be adjusted by the algorithm. The results show that valid routes are detected in about* 90% *of the base data, excluding invalid markings and adding times at missing bus stops through a temporal interpolation.*

Resumo. *Muitas cidades possuem um sistema de rastreamento da movimentação dos ônibus do transporte público. O problema de detecção de itinerários definido neste artigo consiste em compatibilizar os registros de geolocalização da movimentação do ônibus com seu respectivo itinerário. Este artigo propõe um algoritmo para detecção de itinerários a partir dos dados de GPS dos ônibus e da localização dos pontos das linhas de ônibus. Como este problema é afetado por falhas no envio dos dados de GPS, sequências temporais geolocalizadas incompatíveis com o itinerário programado precisam ser ajustadas pelo algoritmo. Os resultados mostram que itinerários válidos são detectados em cerca de* 90% *dos dados da base, excluindo marcações inválidas e adicionando horários em pontos de ônibus faltantes por meio de uma interpolação temporal.*

1. Introdução

O serviço de transporte público coletivo foi pouco ampliado e modernizado ao longo das últimas décadas, quando comparado ao transporte individual particular que atualmente consome a maior quantidade do espaço urbano viário disponível. Nos trabalhos [Araújo et al. 2011, Silveira and Cocco 2013, Pero and Stefanelli 2015] evidenciaram-se conflitos e contradições históricas que estão nas origens dos problemas que mais afetam os sistemas de transporte público coletivo em todo Brasil.

Visando reverter isso, atrair novos usuários e melhorar a qualidade do serviço, diversas companhias que gerenciam o transporte público coletivo e também pesquisadores dessa área [Bona et al. 2016, Curzel et al. 2019, Santin et al. 2020] estão empregando com mais frequência dados de *Global Positioning System* (GPS) de veículos, pois fornecem um meio eficiente e preciso para rastrear a posição dos veículos em um dado

horário. Desse modo, pode-se investigar o transporte público sob diversas perspectivas, examinando o comportamento dinâmico do sistema, medindo a eficiência dos serviços, identificando padrões de movimentação, capacidade de integração, horários de pico, etc.

Entretanto, trabalhar com dados reais de GPS na prática pode se tornar uma tarefa muito difícil porque os veículos falham na comunicação do GPS, gerando *gaps* que podem afetar os resultados das análises, caso não sejam tratados de maneira adequada. Outro problema é quando há necessidade de cruzar as tabelas de horários programados com a informação do GPS, pois tabelas de horários desatualizadas geram inconsistências, como os casos relatados por [Martins et al. 2022]. Desse modo, os pesquisadores são muitas vezes obrigados a procurar períodos em que haja consistência dos *logs* e das tabelas. Porém, isso restringe a pesquisa a intervalos de tempo e condições muito particulares nas quais os dados sejam íntegros.

Nesse contexto, o objetivo principal desse trabalho é desenvolver um algoritmo capaz de detectar o itinerário de um ônibus em operação empregando dados de sua geolocalização, sem a necessidade de usar informações das tabelas de horários programados de cada linha de ônibus. Além disso, o algoritmo adiciona dados faltantes devido a falhas de comunicação, interpolando valores de horários conhecidos. A base de dados utilizada contém os dados de movimentação dos ônibus do sistema de transporte de Curitiba e os resultados mostram melhora significativa em relação ao algoritmo de [Peixoto et al. 2020], o qual depende das tabelas de horários programados. Como contribuição destaca-se, além do algoritmo proposto, a análise do impacto da interpolação dos horários nos diversos tipos de linhas do sistema de transporte. Em um processo de limpeza e preparação da base de dados, este trabalho recupera um volume maior de dados brutos válidos, evitando que dados sejam descartados por não estarem associados a um itinerário válido.

O artigo é organizado da seguinte forma. A Seção 2 caracteriza o problema de detecção de itinerários, a Seção 3 descreve a base de dados utilizada, seguida de trabalhos relacionados na Seção 4. O algoritmo proposto é descrito na Seção 5, cujos resultados estão na Seção 6 e a conclusão na Seção 7.

2. O Problema de Detecção de Itinerários

Na análise da operação de uma rede de transporte público, geralmente é necessário identificar os instantes de tempo em que há passagem de ônibus nos pontos, havendo ou não parada do ônibus.

Em geral, esta identificação é feita por um algoritmo de *map matching* cruzando a informação de geolocalização do ônibus com a localização do ponto de ônibus. Um método adequado para essa tarefa foi empregado no trabalho de [Martins et al. 2022]. O método apresenta uma solução para detecção de paradas de ônibus, mesmo em vias de mão dupla, correção de imprecisões do GPS e identificação do horário exato da passagem em um ponto de ônibus.

O problema de detecção de itinerários, conforme definido neste artigo, vai além, pois consiste em compatibilizar os *logs* de GPS da movimentação de um ônibus com seu respectivo itinerário e programação horária, que é definida no planejamento da operação dos ônibus. Este problema é afetado por falhas de comunicação no envio dos dados de GPS que inviabilizam a correta detecção do itinerário do ônibus.

Uma abordagem para detecção de itinerários foi empregada no trabalho de [Peixoto et al. 2020] utilizando a tabela de horários programados para uma linha de ônibus. Neste caso, tanto o horário de saída do ponto inicial quanto o horário de chegada programado no ponto final devem ser conhecidos. Além disso, os *logs* da movimentação dos ônibus são necessários. A Figura 1 ilustra uma linha de ônibus com horários programados tanto para o percurso de ida (em laranja) quanto de volta do ônibus (em verde). O algoritmo de [Peixoto et al. 2020] identifica o itinerário quando o instante de tempo do *log* de um ônibus (10h20 ou 13h20) se encontra dentro do intervalo de tempo programado entre o horário de saída e de chegada do ônibus na linha.

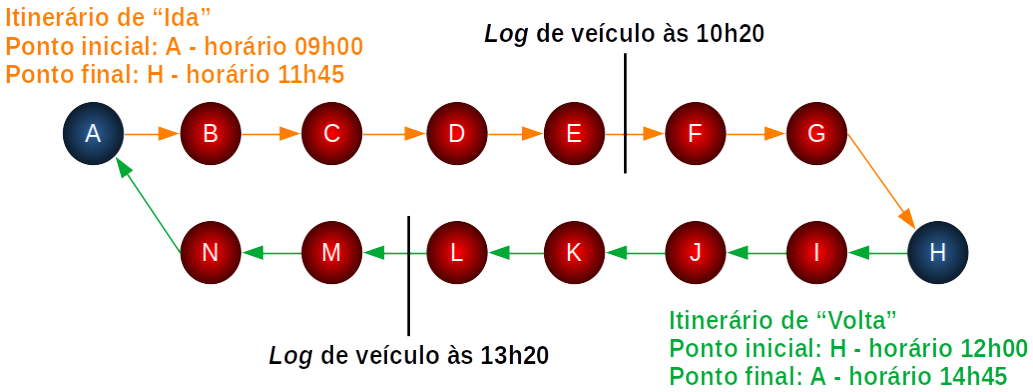

Figura 1. Programação horária de uma linha de ônibus. Tanto o horário de saída quando de chegada são conhecidos. Um *log* qualquer (10h20 ou 13h20) é associado a um itinerário quando se encontra dentro deste intervalo.

Embora este algoritmo identifique a maior parte dos itinerários, vários *logs* da monitoração dos ônibus são descartados, pois não aparecem associados a nenhum itinerário. Diferentemente do algoritmo anterior, o algoritmo proposto neste artigo não utiliza a tabela de horários programados da linha de ônibus.

3. Dados do Transporte Público

O repositório Centro de Computação Científica e Software Livre (C3SL) é reconhecidamente a principal fonte de dados para aplicações acadêmicas sobre o transporte público de Curitiba, tendo sido empregado em diversas pesquisas.

Para se tratar o problema de *map matching*, são necessários os dados geolocalizados da monitoração dos ônibus, assim como as informações estáticas da rede de transporte e da programação horária dos ônibus. O Portal de Dados Abertos da Prefeitura de Curitiba fornece um banco de dados atualizado diariamente contendo dados do transporte coletivo de Curitiba, disponibilizado via *WebService* e contendo informações como: Arquivo *General Transit Feed Specification* (GTFS), Linhas, Pontos, Itinerários, Posição dos Veículos e Tabelas de Horários. Os dados são transferidos por meio de arquivos no formato *JavaScript Object Notation* (JSON) através de uma *Application Programming Interface* (API). O dicionário de dados com a especificação das informações disponíveis na API pode ser encontrado na documentação técnica disponibilizada em [URBS 2022b].

Segundo os dados operacionais publicados em [URBS 2022a], a Rede Integrada de Transporte Coletivo de Curitiba (RIT) opera com uma frota de 1.226 veículos (desconsiderando os ônibus reserva) que atendem 250 linhas, 22 terminais e 329 estações tubo

e realizam, em média, 1.365.615 viagens por dia útil. De acordo com a documentação [URBS 2022b], esses veículos enviam sua localização periodicamente a qual é armazenada em um *log* diário que pode ser consultado via API. Como o serviço nativo não oferece requisições por data, o C3SL vinculado ao Departamento de Informática da Universidade Federal do Paraná (UFPR), criou um repositório de arquivos JSON contendo o histórico diário e completo das operações da RIT atualizado diariamente em D-1. Uma extensa análise exploratória desses dados foi desenvolvida em [Vila et al. 2016].

Para o experimento descrito neste artigo, foram empregados os seguintes arquivos de dados disponíveis no repositório C3SL[1]:

- `Linhas`: contém o código, nome, categoria de serviço, cor entre outros atributos de todas as linhas da RIT.
- `PontosLinhas`: armazena o nome, código, tipo, latitude e longitude de todos os pontos de ônibus da RIT, além de descrever a sequência correta de paradas de ônibus conforme o itinerário das linhas.
- `Veículos`: contém o histórico de coordenadas dos veículos em operação. Em média, a cada 20 segundos é amostrado a posição de um veículo com registro do dia e horário.
- `TabelaLinhas`: armazena o horário de passagem programada de ônibus nos pontos da linha, a maioria dos pontos não possui essa programação.
- `TabelaVeículos`: armazena a tabela de horários programados de passagem do ônibus em todos os trechos do seu itinerário.

4. Trabalhos Relacionados

A popularização de tecnologias de sistemas embarcados de baixíssimo custo como Arduino[2], Raspberry Pi[3] e ESP32[4] possibilitou o crescimento de pesquisas envolvendo *Internet of Things* (IoT) e transporte público coletivo. Em especial, a instalação desses equipamentos permite a criação de uma vasta gama de aplicações com potencial de promover a melhoria e eficiência do serviço. Os trabalhos de [Sridevi et al. 2017, Hakeem et al. 2022, Desai et al. 2022] exemplificam aplicações recentes em que os equipamentos embarcados coletam a trajetória de GPS dos ônibus e a centralizam em um servidor. Esses dados podem ser empregados em várias aplicações de gerenciamento do sistema de transporte. Uma revisão de literatura sobre aplicação de dados de trajetória de ônibus pode ser vista em [War et al. 2022]. Aspectos como fontes de dados e métodos de *Big Data* e IoT em transporte público coletivo são descritos em mais profundidade no *survey* de [Welch and Widita 2019]. Outras discussões sobre uso de dados de trajetórias de GPS de ônibus são apresentadas em [Singla and Bhatia 2015].

No âmbito da RIT de Curitiba, relacionam-se os trabalhos de [Bona et al. 2016, Curzel et al. 2019, Santin et al. 2020], os quais empregaram dados abertos do transporte público, como dados de trajetórias de GPS de veículos, tabelas de horários e itinerários das linhas de ônibus, na criação de modelos que permitiram expandir a compreensão das características e comportamentos da rede de transporte, a fim de promo-

[1]Repositório disponível em: `http://dadosabertos.c3sl.ufpr.br/curitibaurbs/`
[2]https://www.arduino.cc/
[3]https://www.raspberrypi.com/
[4]https://www.espressif.com/en/products/socs/esp32

ver a melhoria da eficiência do serviço. Outros trabalhos relacionados à mobilidade urbana, redes de transporte e modelos computacionais que empregam dados semelhantes à RIT são os de [Wehmuth et al. 2018, Rodrigues and Villas 2019, Maduako et al. 2019, Sadeghian et al. 2021, Li and Rong 2022]. Esses modelos fornecem um grande número de medidas de eficiência dos serviços de transporte e auxiliam na identificação de oportunidades para melhorias importantes como redução de custos, tempo de espera do usuário, economia de energia e outros recursos envolvidos na operação.

Contudo, para que os modelos funcionem de forma adequada, um processo de higienização dos dados brutos deve ser realizado para reduzir inconsistências. Por exemplo, os autores de [Martins et al. 2022] apontam vários problemas presentes nos dados reais de GPS, e por isso, desenvolveram um modelo de *map matching*. Este modelo pode servir para: i) detecção de paradas de veículos em pontos de ônibus próximos em uma rua de mão dupla, na qual um ponto serve o sentido de "ida" e o outro o sentido de "volta" da linha de ônibus; ii) imprecisões do GPS; e iii) horário exato da passagem do veículo em um ponto de ônibus. O problema da detecção das paradas de ônibus ainda foi abordado no trabalho de [Peixoto et al. 2020], onde entre outros aspectos, também foi empregado um processo de detecção do itinerário do veículo, que pode ser feito empregando dados das tabelas de horários. Outros autores também enfrentaram o desafio da detecção da compatibilidade entre a trajetória dos ônibus e seus respectivos itinerários como [Yin et al. 2014, Queiroz et al. 2019, Chawuthai et al. 2023]. Nestes trabalhos, o objetivo principal da detecção é identificar se uma trajetória de GPS de ônibus está ou não de acordo com o itinerário previsto e sinalizar uma eventual inconsistência. No trabalho de [Gallotti and Barthelemy 2015], inconsistências nos horários de parada dos veículos foram corrigidas através de um método de interpolação temporal, mas nenhuma medida de erro de interpolação foi apresentada.

O algoritmo proposto neste artigo preenche algumas dessas lacunas, pois identifica e corrige as inconsistências nas trajetórias de GPS, conforme o itinerário. Além disso, um método para medir o erro de interpolação é apresentado. Portanto, a abordagem proposta para o problema de detecção de itinerários e interpolação temporal reduz a quantidade de inconsistências dos dados brutos e fornece uma base confiável para novas aplicações.

5. Algoritmo Proposto de Detecção de Itinerários

A questão central do artigo é desenvolver um algoritmo de detecção de itinerário que seja independente da tabela de programação horária dos ônibus. Inicialmente, é necessário diferenciar os conceitos de "rede estática" e "rede dinâmica". Esses conceitos também foram empregados no trabalho de [Peixoto et al. 2020]. Uma rede estática representa a topologia da rede de transporte, ou seja, o sequenciamento dos pontos de ônibus de uma linha específica abrangendo todos os itinerários oferecidos pelo serviço, conforme considerado em [Bona et al. 2016].

Uma vez que a rede estática descreve a topologia da linha de ônibus e seus respectivos itinerários sem incluir horários, a rede dinâmica é formada à medida que um dado veículo alcança os pontos previstos no itinerário de seu serviço. O algoritmo proposto de detecção de itinerários é composto de 3 etapas:

- **etapa 1**: marcar o horário de passagem do ônibus nos pontos (algoritmo de *map matching*).

- **etapa 2**: sequenciar os pontos de acordo com esses horários (sequenciamento temporal).
- **etapa 3**: associar a sequência temporal a um itinerário conhecido, interpolando e removendo temporizações se necessário (algoritmo proposto).

Na **etapa 1**, um algoritmo de *map matching* é utilizado. Nesse trabalho, foi empregado o algoritmo de [Martins et al. 2022]. Resumidamente, para cada posição do veículo (*log*), o algoritmo calcula a distância de *Haversine* [Panigrahi 2014, Lawhead 2015] para todos pontos de ônibus da linha e atribui o *log* ao ponto mais próximo. Desse modo, obtém-se a marcação do horário de passagem do veículo nos pontos de ônibus da linha. A **etapa 2** é simplesmente a ordenação das marcações em ordem crescente de horário para obter o sequenciamento temporal dos pontos. A **etapa 3** corresponde ao algoritmo proposto, conforme descrito a seguir.

O algoritmo de detecção de itinerário tem por objetivo associar um log de eventos capturados pela movimentação de um ônibus à sequência de pontos da sua respectiva linha de ônibus cadastrada na tabela `PontosLinhas`. Assim, é possível associar o instante tempo de passagem do ônibus em todos os pontos da linha. Isso é ilustrado na Figura 2, na qual um log de eventos $log = ((l_1, t_1), (l_2, t_2), (l_3, t_3), (l_4, t_4), (l_5, t_5))$ será associado a um itinerário $iti = (p_1, p_2, p_3, p_4, p_5)$, sendo t_i o instante de passagem do ônibus na coordenada l_i e p_i um ponto do itinerário do ônibus.

Por exemplo, de acordo com a Figura 2, não há registro de passagem do ônibus no ponto p_3 e há registro de passagem do ônibus em uma posição l_4 que não corresponde a nenhum ponto cadastrado da linha.

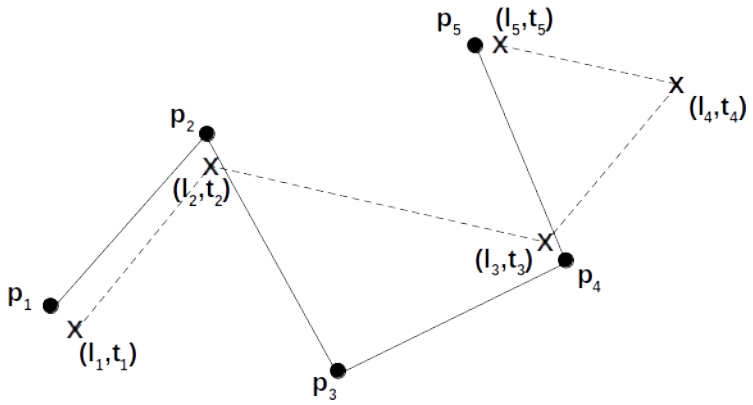

Figura 2. Exemplo de um itinerário $iti = (p_1, p_2, p_3, p_4, p_5)$ em linha cheia e um $log = ((l_1, t_1), (l_2, t_2), (l_3, t_3), (l_4, t_4), (l_5, t_5))$ em linha tracejada.

O algoritmo de *map matching* associa as localizações l_i aos respectivos pontos de ônibus pela avaliação de uma medida de proximidade espacial entre l_i e um ponto do itinerário. No caso da Figura 2, o resultado do *map matching* é o mapeamento $map = ((p_1, l_1), (p_2, l_2), (p_4, l_3), (-, l_4), (p_5, l_5))$, sendo que nenhum ponto é associado à localização l_4 e não há registro da passagem pelo ponto p_3.

Em seguida, o algoritmo proposto detecta o itinerário com informação temporal $det = ((p_1, t_1), (p_2, t_2), (p_3, \hat{t}_3), (p_4, t_3), (p_5, t_5))$, associando o instante de passagem a cada ponto da linha do ônibus. Neste caso, o instante de tempo $\hat{t}_3 = t_2 + (t_3 - t_2)/2$ é estimado pela média dos tempos t_2 e t_3 dos pontos adjacentes a p_3.

O resultado acima para detecção do itinerário $det = \{(p_i, t_j)\}$ com informação temporal pode ser generalizado em (1) para iti de dimensão n e log de dimensão m.

$$(p_i, t_i) = \begin{cases} (p_i, t_j) & : (p_i, l_j) \in map; i = 1, \ldots, n; j = 1, \ldots, m \\ (p_i, \hat{t}_i) & \text{otherwise} \end{cases} \quad (1)$$

sendo que $\hat{t}_i = \hat{t}_{i-1} + (t_{k+w} - t_k)/w$ para $k < i < (k + w)$ e $\hat{t}_k = t_k$, considerando os $w - 1$ pontos de ônibus que não foram mapeados pelo *map matching* entre os pontos (p_k, t_k) e $(p_{k+w}, t_{k+w}) \in map$. O procedimento acima é sintetizado no Algoritmo 1.

Algoritmo 1 Detecção de itinerário

Entrada: $iti = \{p_i\}, 1 \leq i \leq n; log = \{(l_j, t_j)\}, 1 \leq j \leq m$ // ordenado por pontos e tempo, respectivamente
Saída: $det = \{(p_i, t_i)\}, 1 \leq i \leq n$

$map \leftarrow \{\}$
$tempo \leftarrow -1$
for each $p_i \in iti$ **do**
  $achou \leftarrow False$
  for each $(l_j, t_j) \in log$ **do**
    if $(p_i = l_j)$ and $(t_j > tempo)$ **then**
      $map \leftarrow map \cup \{(p_i, l_j, t_j)\}$
      $achou \leftarrow True$
      $tempo \leftarrow t_j$
    end if
  end for
  if $(achou = False)$ **then**
    $map \leftarrow map \cup \{(p_i, None, None)\}$
  end if
end for
$det \leftarrow \{\}$
for each $(p_i, l_i, t_i) \in map$ **do**
  if $(l_i \neq None)$ **then**
    $det \leftarrow det \cup \{(p_i, t_i)\}$
  else
    computa w, $\Delta t = (t_{k+w} - t_k)$ e $\hat{t}_k = t_k$ // pontos faltantes entre p_k e p_{k+w} conhecidos
    $\hat{t}_i = \hat{t}_{i-1} + \Delta t/w$
    $det \leftarrow det \cup \{(p_i, \hat{t}_i)\}$
  end if
end for

O algoritmo proposto possui como requisitos a rede estática e a rede dinâmica, ou seja, é necessário fornecer como entrada tanto a estrutura (ou topologia) da rede de transporte, segundo o arquivo `PontosLinhas`, assim como os *logs* de GPS dos ônibus do arquivo `Veículos`. A vantagem em relação ao método proposto por [Peixoto et al. 2020] é não precisar da tabela de horários programados das linhas de ônibus (arquivos `TabelaLinhas` e `TabelaVeículos`).

Dado um conjunto de marcações com tempos conhecidos (p_i, t_i) e estimados $(p_i, \hat{t}_i)$, a interpolação temporal introduz um erro de estimação dado por $err_i = |t_i - \hat{t}_i|$. Neste trabalho, valores conhecidos são retirados do conjunto de dados original para obter os resultados de avaliação do erro de estimação.

6. Resultados e Discussões

Os resultados são obtidos a partir de um estudo de caso, cuja descrição é dada na Seção 6.1, a detecção do itinerário na Seção 6.2 e a avaliação dos erros de interpolação na Seção 6.3. A Seção 6.4 avalia o algoritmo, incluindo o impacto dos erros de interpolação, considerando toda a base de dados.

6.1. Configuração do Estudo de Caso da Linha 829

A linha "829 - Universidade Positivo" do tipo "Alimentador" foi selecionada para o estudo de caso. Esta linha é circular, ou seja, o ponto inicial coincide com o ponto final, e o ônibus percorre o trajeto em um único sentido. Além disso, esta linha também possui poucos pontos, o que facilita a visualização e interpretação dos dados.

A Tabela 1 fornece o itinerário programado da linha 829, contendo os nomes e a respectiva sequência de todos os pontos de ônibus da linha. O trajeto inicia em 1 no "Terminal Campo Comprido", passa por todos os pontos intermediários $[2, 10]$, retornando ao ponto inicial 1, conforme ilustra a Figura 3.

Tabela 1. Itinerário programado da linha 829 com todos os pontos de ônibus.

Ponto de ônibus	Seq.
Terminal Campo Comprido - 829 - Universidade Positivo	1
Rua Angelo Nebosne, 75 - Cidade Industrial	2
Rua Prof. Pedro Viriato Parigot de Souza, 4716 - Cidade Industrial	3
Rua Prof. Pedro Viriato Parigot de Souza, 5136 - Cidade Industrial	4
Rua Casemiro Augusto Rodacki, 233 - Cidade Industrial	5
Rua Carlos Müller, 331 - Cidade Industrial	6
Rua Carlos Müller, 871 - Cidade Industrial	7
Rua Eduardo Sprada, 5273 - Cidade Industrial	8
Rua Dep. Heitor Alencar Furtado, 5181 - Cidade Industrial	9
Rua Dep. Heitor Alencar Furtado, 4900 - Cidade Industrial	10
Terminal Campo Comprido - 829 - Universidade Positivo	1

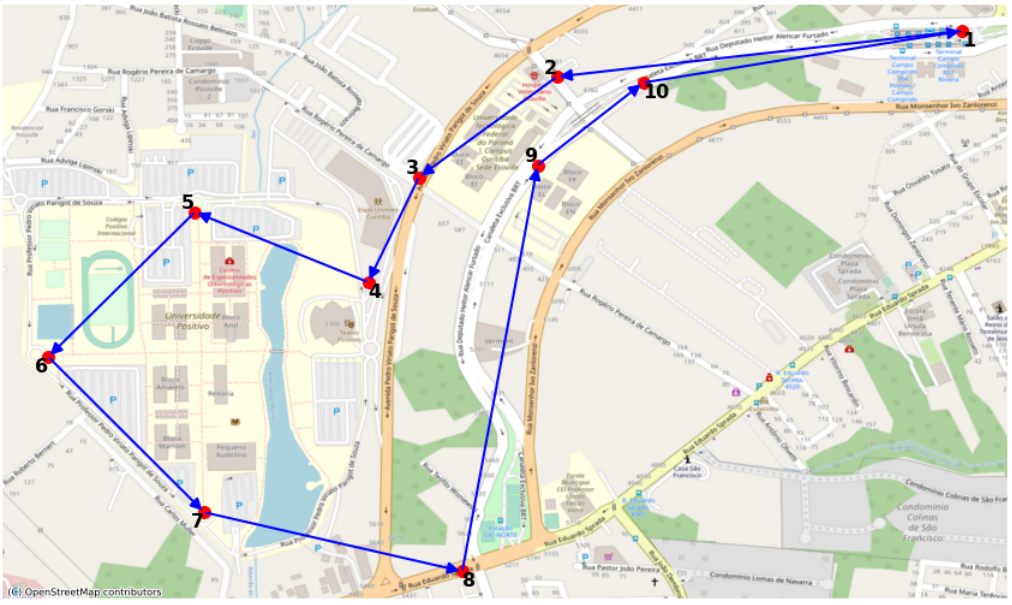

Figura 3. Trajeto programado da linha 829 que inicia no ponto 1, passa pelos pontos intermediários 2 a 10, e retorna ao ponto inicial.

O cenário foi construído empregando *logs* reais do dia 11/07/2022 e do veículo identificado pelo código BA020. Uma volta completa da linha (ida e volta) ocorreu no

intervalo das 06:04 às 06:32, durante o qual não houve perda de dados GPS. Portanto, este é um caso conveniente, não apenas para verificar a aplicação do algoritmos proposto, mas também para avaliar os erros de interpolação, simulando falhas de comunicação. Assim, alguns *logs* são excluídos dentro de intervalos de tempo específicos para que o algoritmo de *map matching* não detecte a passagem do veículo em alguns pontos e o algoritmo proposto possa recuperar esta informação com o uso da interpolação. Os pontos 3, 5 e 8 do itinerário da linha 829 foram escolhidos para serem removidos do conjunto de dados original. Os parâmetros de configuração do estudo de caso estão sintetizados na Tabela 2.

Tabela 2. Parâmetros do estudo de caso com a linha 829.

Linha	829 - Universidade Positivo
Veículo	BA020
Data	11/07/2022
Horário da volta	06:04 às 06:32
Intervalo de falhas	06:15 às 06:16 no ponto 3 06:17 às 06:19 no ponto 5 06:26 às 06:28 no ponto 8

6.2. Detecção do Itinerário da Linha 829

A Tabela 3 apresenta o resultado das **etapas 1**, **2** e parte da **etapa 3**. O procedimento adotado marcou o horário exato de passagem do ônibus (*map matching*), criou o sequenciamento temporal (**etapa 2**), localizou o itinerário e atribuiu a cada *log* um número de sequência (parte da **etapa 3**).

Tabela 3. Resultado intermediário do processamento do estudo de caso.

Ponto de ônibus	Horário	Seq.
Terminal Campo Comprido - 829 - Universidade Positivo	06:04:51	1
Rua Dep. Heitor Alencar Furtado, 4900 - Cidade Industrial	06:14:08	10
Rua Angelo Nebosne, 75 - Cidade Industrial	06:14:36	2
Rua Prof. Pedro Viriato Parigot de Souza, 5136 - Cidade Industrial	06:16:43	4
Rua Carlos Müller, 331 - Cidade Industrial	06:19:30	6
Rua Carlos Müller, 871 - Cidade Industrial	06:21:06	7
Rua Dep. Heitor Alencar Furtado, 5181 - Cidade Industrial	06:28:30	9
Rua Dep. Heitor Alencar Furtado, 4900 - Cidade Industrial	06:29:06	10
Terminal Campo Comprido - 829 - Universidade Positivo	06:31:41	1

Entretanto, o resultado da Tabela 3 contém inconsistências. Por exemplo, a marcação do ponto "Rua Dep. Heitor Alencar Furtado, 4900" às 06:14:08 está incorreta, pois este ponto está situado no final do itinerário programado. Um exame mais detalhado revelou que o trajeto do ônibus entre os pontos 1 e 2 passa muito próximo do ponto 10, conforme ilustrado na Figura 4. Neste caso, o algoritmo de *map matching* gerou um erro de marcação. Portanto, somente este algoritmo não é suficiente para tratar os *logs* de veículos de forma adequada. Ao combinar o resultado do algoritmo de *map matching* com o algoritmo proposto, este erro de marcação é identificado.

Figura 4. Região de incerteza na qual o algoritmo de *map matching* gera um erro de marcação.

Com a conclusão da **etapa 3**, o algoritmo proposto localiza as lacunas presentes na sequência. Ou seja, o algoritmo identifica a falta dos pontos 3, 5 e 8 devido a simulação da falha de comunicação. A Tabela 4 mostra o resultado final da aplicação do algoritmo proposto no estudo de caso. Nota-se que a marcação incorreta foi excluída e os pontos 3, 5 e 8 foram adicionados ao itinerário devido à interpolação temporal. Este itinerário corresponde à sequência completa de pontos cadastrada na linha 829 com a informação temporal da passagem do ônibus nos pontos da linha.

Tabela 4. Resultado final da aplicação do algoritmo proposto no estudo de caso.

Ponto de ônibus	**Horário**	**Seq.**
Terminal Campo Comprido - 829 - Universidade Positivo	06:04:51	1
Rua Angelo Nebosne, 75 - Cidade Industrial	06:14:36	2
Rua Prof. Pedro Viriato Parigot de Souza, 4716 - Cidade Industrial	06:15:39	3
Rua Prof. Pedro Viriato Parigot de Souza, 5136 - Cidade Industrial	06:16:43	4
Rua Casemiro Augusto Rodacki, 233 - Cidade Industrial	06:18:07	5
Rua Carlos Müller, 331 - Cidade Industrial	06:19:30	6
Rua Carlos Müller, 871 - Cidade Industrial	06:21:06	7
Rua Eduardo Sprada, 5273 - Cidade Industrial	06:24:48	8
Rua Dep. Heitor Alencar Furtado, 5181 - Cidade Industrial	06:28:30	9
Rua Dep. Heitor Alencar Furtado, 4900 - Cidade Industrial	06:29:06	10
Terminal Campo Comprido - 829 - Universidade Positivo	06:31:41	1

6.3. Erro de Interpolação da Linha 829

Na avaliação do erro de interpolação, foram utilizados os dados da movimentação dos ônibus da linha 829 durante um dia inteiro das 06:04 às 23:19. Pontos conhecidos foram retirados aleatoriamente do conjunto de dados original, simulando falhas de comunicação. O erro de estimação $err_i = |t_i - \hat{t}_i|$ foi calculado a partir dos valores reais conhecidos (p_i, t_i) dos pontos retirados e dos valores estimados $(p_i, \hat{t}_i)$.

As medidas de erro foram computadas em função do número de pontos de ônibus consecutivos faltantes. Nesse experimento, as medidas de erro foram calculadas em segundos para interpolações de 1 a 7 pontos consecutivos faltantes, ou $w \in \{2, 3, \ldots, 8\}$. Para cada caso, foram utilizadas 100 amostras sem reposição para gerar o resultado da Figura 5. Observa-se que o erro de interpolação aumenta com o aumento do número de pontos intermediários faltantes. Para maioria dos casos, o erro situa-se entre menos de 1 min a 2 min aproximadamente (125 segundos).

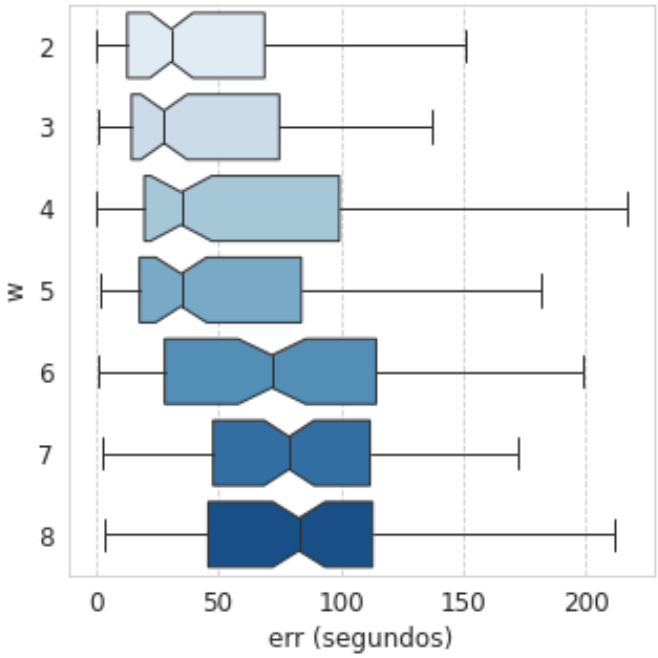

Figura 5. Erro de interpolação em segundos na linha 829 para diferentes valores de w (um ponto intermediário faltante corresponde a $w = 2$).

Esse resultado sugere que a incerteza introduzida pelo erro de interpolação é aceitável, dado que um atraso ou adiantamento de 2 min pode ser considerado tolerável em um sistema de transporte urbano de ônibus. Entretanto, um exame mais detalhado é necessário para entender melhor quais linhas são mais afetadas pelo erro de interpolação e em quais horários do dia.

6.4. Avaliação na Base de Dados Completa

Com o objetivo de avaliar a capacidade de detecção de itinerários usando toda a base de dados, o algoritmo proposto foi aplicado nos dados de movimentação dos ônibus do dia 11/07/2022. O resultado foi comparado com o algoritmo de [Peixoto et al. 2020], que usa a tabela da programação horária dos ônibus. A Tabela 5 mostra o total de marcações que cada algoritmo associou a um itinerário válido por tipo de linha de ônibus.

Observa-se que o algoritmo proposto proporciona um incremento global de $68,83\%$ para $99,33\%$ em ganho de rastreabilidade do itinerário. Ou seja, o novo algoritmo apresenta um resultado $44,31\%$ melhor que o de [Peixoto et al. 2020]. Com exceção das linhas do tipo MADRUGUEIRO, todas as linhas dos demais tipos foram favorecidas. Este resultado proporciona um aumento no volume de dados válidos da base, evitando que dados sejam descartados devido à não associação a algum itinerário. Entretanto, a diminuição observada na linha MADRUGUEIRO deverá ser melhor investigada.

Embora as marcações do algoritmo proposto da Tabela 5 estejam associadas a algum itinerário, pode ainda haver erros de sequência dos pontos ou pontos de ônibus faltantes no itinerário. A Tabela 6 apresenta a distribuição dos erros das marcações do algoritmo proposto da Tabela 5 entre i) fora de ordem e ii) pontos de ônibus faltantes. Este porcentual de $13,38\%$ de marcações contendo erros, antes do ajuste de sequência e pontos faltantes realizado ao final do algoritmo proposto, foi praticamente todo corrigido.

Tabela 5. Comparação entre o número de marcações que foram associadas a um itinerário válido de acordo com o algoritmo de [Peixoto et al. 2020] e o algoritmo proposto, segundo o tipo de linha de ônibus. Os valores porcentuais são relativos ao número de marcações atribuídas apenas com o algoritmo de *map matching*.

Tipo de linha	[Peixoto et al. 2020]		Proposto	
	Marcações	%	Marcações	%
ALIMENTADOR	160.750	70,62%	225.434	99,03%
CONVENCIONAL	52.338	62,02%	84.310	99,90%
EXPRESSO	21.986	62,41%	35.206	99,94%
JARDINEIRA	234	46,89%	499	100,00%
LIGEIRÃO	2.988	63,86%	4.677	99,96%
LINHA DIRETA	8.421	70,37%	11.879	99,26%
MADRUGUEIRO	5.659	98,26%	5.455	94,72%
TRONCAL	26.136	75,84%	34.447	99,96%
TOTAL	**278.512**	**68,83%**	**401.907**	**99,33%**

Tabela 6. Distribuição dos erros das marcações do algoritmo proposto da Tabela 5 entre i) fora de ordem e ii) pontos de ônibus faltantes.

Tipo de linha	Marcações com erro			
	i	ii	Total	%
ALIMENTADOR	15.764	21.176	36.940	16,39%
CONVENCIONAL	2.432	7.068	9.500	11,27%
EXPRESSO	487	2.139	2.626	7,46%
JARDINEIRA	0	12	12	2,40%
LIGEIRÃO	83	193	276	5,90%
LINHA DIRETA	126	162	288	2,42%
MADRUGUEIRO	1.470	283	1.753	32,14%
TRONCAL	478	1.896	2.374	6,89%
TOTAL	**20.840**	**32.929**	**53.769**	**13,38%**

O erro de interpolação por tipo de linha foi avaliado de forma semelhante à Seção 6.3, porém usando todas as linhas da base de dados - pegando apenas trajetos completos que tinham os valores reais. A Figura 6 mostra o erro de interpolação em segundos por tipo de linha. As linhas do tipo ALIMENTADOR, CONVENCIONAL, EXPRESSO, MADRUGUEIRO e TRONCAL apresentam um erro entre 0 a 1 min aproximadamente. Já os tipos JARDINEIRA, LIGEIRÃO e LINHA DIRETA são mais suscetíveis aos erros de interpolação. Os resultados das linhas JARDINEIRA e LINHA DIRETA pode ser devido à elevada separação entre os pontos de ônibus destas linhas e, portanto, mais sujeitos às condições de tráfego de veículos nas vias.

7. Conclusão

Este trabalho propôs um algoritmo de detecção de itinerários a partir dos dados de GPS da movimentação dos ônibus e da composição de pontos das linhas de ônibus. O algoritmo foi inicialmente avaliado em um estudo de caso da linha 829 do sistema de transporte de Curitiba, tendo sido capaz de identificar de forma confiável o itinerário, sequenciando

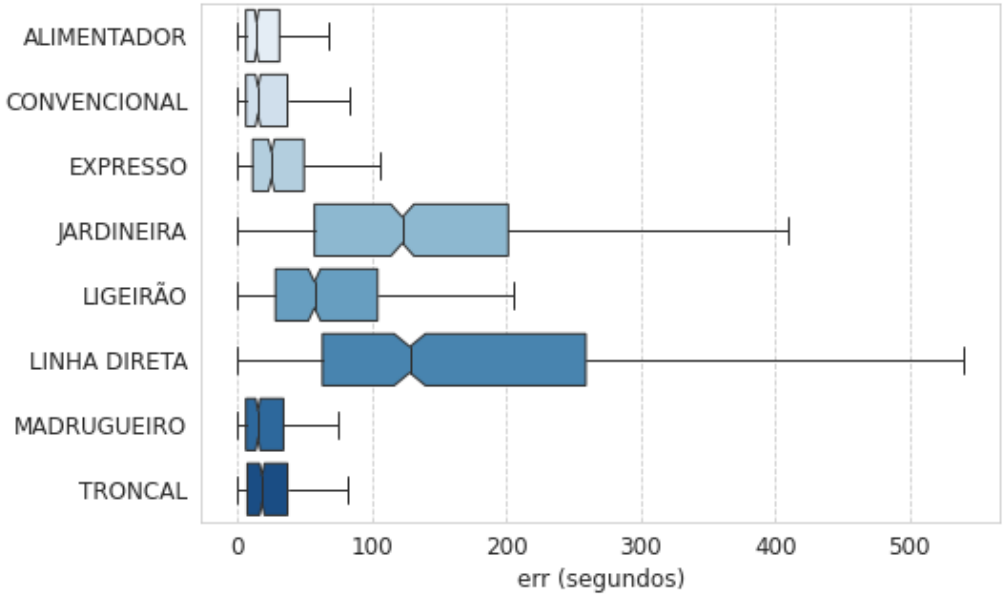

Figura 6. Erro de interpolação por tipo de linha.

corretamente os pontos, excluindo marcações inválidas e recuperando os horários de passagem dos ônibus em pontos faltantes por meio de um método de interpolação. Quando avaliado em todas as linhas de ônibus de Curitiba, os resultados do algoritmo proposto foram 44,31% melhores em comparação com outro método que emprega as tabelas de horários programados. Ao final do algoritmo, o erro de 13,38% devido à ordenação incorreta dos pontos de ônibus e horários faltantes para pontos intermediários foi praticamente todo corrigido. Cinco tipos de linhas foram identificadas como menos susceptíveis ao erros de interpolação de horários conhecidos, apresentando um erro de interpolação em torno de 1 min aproximadamente. Linhas que possuem maior separação entre os pontos de ônibus tendem a apresentar maiores erros. A abordagem apresentada contribui metodologicamente para: i) detecção de itinerário sem a necessidade de usar tabelas de horários programados; ii) correção de erros devido a marcações de pontos fora de ordem; iii) recuperação de pontos faltantes devido a falhas de comunicação do GPS. Em um processo de limpeza e preparação da base de dados, um volume maior de dados brutos válidos pode ser recuperado para uso em diversas aplicações. Como trabalhos futuros, uma melhor compreensão dos erros de interpolação nos diferentes tipos de linhas de ônibus serão considerados. Assim, as incertezas nos resultados podem ser avaliadas em diferentes aplicações. Além disso, pode-se selecionar apenas dados com erro baixo de interpolação. Outros métodos de interpolação também podem ser considerados.

Agradecimentos

J.B. agradece o apoio do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), processo 142519/2020-0. Este trabalho é apoiado pelo Projeto *Smart City Concepts in Curitiba*, CAPES, CNPq (processo 310998/2020-4), Fapesp (processo 2023/00148-0), IPPUC e Prefeitura de Curitiba.

Referências

Araújo, M. R. M. d., Oliveira, J. M. d., Jesus, M. S. d., Sá, N. R. d., Santos, P. A. C. d., and Lima, T. C. (2011). Transporte público coletivo: discutindo acessibilidade, mobilidade e qualidade de vida. *Psicologia Sociedade*, 23(Psicol. Soc., 2011 23(3)):574–582.

Bona, A. A. D., Fonseca, K. V., Rosa, M. O., Lüders, R., and Delgado, M. R. (2016). Analysis of public bus transportation of a Brazilian city based on the theory of complex networks using the p-space. *Mathematical Problems in Engineering*, 2016.

Chawuthai, R., Sumalee, A., and Threepak, T. (2023). GPS data analytics for the assessment of public city bus transportation service quality in Bangkok. *Sustainability*, 15(7).

Curzel, J. L., Lüders, R., Fonseca, K. V., and Rosa, M. O. (2019). Temporal performance analysis of bus transportation using link streams. *Mathematical Problems in Engineering*, 2019.

Desai, S., Suthar, R., Yadav, V., Ankar, V., and Gupta, V. (2022). Smart bus fleet management system using IoT. In *2022 Fourth International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT)*, pages 01–06.

Gallotti, R. and Barthelemy, M. (2015). The multilayer temporal network of public transport in Great Britain. *Scientific Data*, 2:140056.

Hakeem, M. F. M. A., Sulaiman, N. A., Kassim, M., and Isa, N. M. (2022). IoT bus monitoring system via mobile application. In *2022 IEEE International Conference on Automatic Control and Intelligent Systems (I2CACIS)*, pages 125–130.

Lawhead, J. (2015). *Learning geospatial analysis with Python*. Packt Publishing Ltd, Birmingham, 2nd edition.

Li, T. and Rong, L. (2022). Spatiotemporally complementary effect of high-speed rail network on robustness of aviation network. *Transportation Research Part A: Policy and Practice*, 155:95–114.

Maduako, I. D., Wachowicz, M., and Hanson, T. (2019). Transit performance assessment based on graph analytics. *Transportmetrica A: Transport Science*, 15(2):1382–1401.

Martins, T., Kozievitch, N., Gadda, T., Rosa, M., and Gutierrez, M. (2022). Map matching: Uma análise de dados streaming de trajetórias de GPS no transporte público. In *Temas Emergentes: Cidades Inteligentes (XVIII SBSI)*, pages 294–301. SBC.

Panigrahi, N. (2014). *Computing in geographic information systems*. CRC Press, Boca Raton, Florida, 1st edition.

Peixoto, A., Rosa, M., Lüders, R., and Fonseca, K. (2020). Plataforma computacional para construção de um banco de dados de grafo do sistema de transporte de Curitiba. In *IV Workshop de Computação Urbana*, pages 125–137. SBC.

Pero, V. and Stefanelli, V. (2015). A questão da mobilidade urbana nas metrópoles brasileiras. *Revista de Economia Contemporânea*, 19(Rev. econ. contemp., 2015 19(3)):366–402.

Queiroz, A. R., Santos, V., Nascimento, D., and Pires, C. E. (2019). Conformity analysis of GTFS routes and bus trajectories. In *XXXIV Simpósio Brasileiro de Banco de Dados*, pages 199–204. SBC.

Rodrigues, D. and Villas, L. (2019). SMAFramework: Arcabouço para integração de dados urbanos cientes da correlação espaço-temporal. In *CTD do XXXVII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, pages 113–120. SBC.

Sadeghian, P., Håkansson, J., and Zhao, X. (2021). Review and evaluation of methods in transport mode detection based on GPS tracking data. *Journal of Traffic and Transportation Engineering (English Edition)*, 8(4):467–482.

Santin, P., Gubert, F. R., Fonseca, M., Munaretto, A., and Silva, T. H. (2020). Characterization of public transit mobility patterns of different economic classes. *Sustainability*, 12(22).

Silveira, M. R. and Cocco, R. G. (2013). Transporte público, mobilidade e planejamento urbano: contradições essenciais. *Estudos Avançados*, 27(Estud. av., 2013 27(79)):41–53.

Singla, L. and Bhatia, P. (2015). GPS based bus tracking system. In *2015 International Conference on Computer, Communication and Control (IC4)*, pages 1–6.

Sridevi, K., Jeevitha, A., Kavitha, K., Sathya, K., and Narmadha, K. (2017). Smart bus tracking and management system using IoT. *Asian Journal of Applied Science and Technology (AJAST)*, 1(2).

URBS (2022a). Características da rede integrada de transporte. Acesso 27 mar 2023, https://www.urbs.curitiba.pr.gov.br/transporte/rede-integrada-de-transporte.

URBS (2022b). Web-service: Dados públicos da rede integrada do transporte coletivo de Curitiba. https://mid.curitiba.pr.gov.br/dadosabertos/TransporteColetivo/2015-11-24_Documentação_WEB-SERVICE_–_TRANSPORTE_COLETIVO_DE_CURITIBA.pdf.

Vila, J. J. R., Kozievitch, N. P., Gadda, T. M., Fonseca, K., 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).

War, M. M., Rakhra, M., and Singh, D. (2022). Review on application based bus tracking system. In *2022 5th International Conference on Contemporary Computing and Informatics (IC3I)*, pages 876–880.

Wehmuth, K., Costa, B., Bechara, J. V., and Ziviani, A. (2018). A multilayer and time-varying structural analysis of the Brazilian air transportation network. In *Latin America Data Science Workshop*, volume 2170 of *CEUR Workshop Proceedings*, pages 57–64.

Welch, T. F. and Widita, A. (2019). Big data in public transportation: a review of sources and methods. *Transport Reviews*, 39(6):795–818.

Yin, L., Hu, J., Huang, L., Zhang, F., and Ren, P. (2014). Detecting illegal pickups of intercity buses from their GPS traces. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 2162–2167.