International Journal of Geo-Information

MDPI

Article

Interactive Visualization for the GTFS and GTFS-RT Data of Budapest

Róbert Tóth, Márton Ispány and Marianna Zichar *

Faculty of Informatics, University of Debrecen, Kassai u 26, 4028 Debrecen, Hungary; toth.robert@inf.unideb.hu (R.T.)

* Correspondence: zichar.marianna@inf.unideb.hu

Abstract

Various platforms, such as Google Maps, provide information about the services of public transport companies worldwide. Operators publish the planned (static) timetable using the General Transit Feed Specification (GTFS) format, while the GTFS Realtime (GTFS-RT) specification provides live (dynamic) information about the services. In this paper, we present our dataset that was built by retrieving and pre-processing the data sources of the open data platform of BKK Futár, hosted by the Centre for Budapest Transport Company (BKK). The paper contains a well-detailed description of our methods for retrieving and pre-processing the data among statistical features. The dataset covers a one-year period in which the data collection mechanism used for realtime data was continuously improved from collecting only live vehicle positions to covering all the available feeds and increasing the query frequency. We merged the static data with the vehicle positions to filter them, yielding a clean set of tracked trips. As a result, more than 90% of the daily planned trips could be reconstructed from the responses. We provide an interactive web-based visualization for the analysis of the GTFS schedule's, and the GTFS-RT *Vehicle Positions* feed's, geospatial features. The dataset and also our methodology can serve as input for various research studies to investigate the common characteristics of delays and disruptions or predict real departure times based on the current vehicle positions and historical data.

Keywords: public transport; geospatial data; historical data; pre-processing; reconstruction; GTFS; GTFS-RT; visualization; Leaflet

Academic Editors: Wolfgang Kainz and Hartwig H. Hochmair

Received: 1 May 2025
Revised: 15 June 2025
Accepted: 20 June 2025
Published: 25 June 2025

Citation: Tóth, R.; Ispány, M.; Zichar, M. Interactive Visualization for the GTFS and GTFS-RT Data of Budapest. *ISPRS Int. J. Geo-Inf.* **2025**, *14*, 245. https://doi.org/10.3390/ijgi14070245

1. Introduction

We spend most of our lives in two main locations: our home and our workplace. The distance between these two can vary widely, ranging from almost zero, such as in cases where a full home office is set up or when an entrepreneur works from a space adjacent to their home, to several kilometers.

During the last few decades, several trends have emerged. Urbanization has long been used to describe the phenomenon of people leaving rural areas or smaller towns to move to larger cities. Behind people's decisions, we often find better job opportunities, improved healthcare services, and the proximity of educational and cultural institutions. However, a recent trend, known as suburban migration or suburbanization, has emerged, particularly in developed countries, where many city residents are relocating to the suburbs. They are drawn to lower property prices, cleaner air, and a calmer environment, which are found in smaller communities near larger cities [1–3].

Commuting between work and home takes time, and since we generally cannot spare this time, we seek affordable and reasonable solutions. The luckiest people can walk, bike, or scooter to their workplace. Although driving may seem convenient, it has several downsides, including fuel costs, vehicle expenses, parking challenges, and traffic jams during peak hours. The alternative offered by larger cities is the public transport system, with very different fare systems [4,5].

Smart City and Mobility

The term Smart City and related research emerged in the 1990s and have since focused on various aspects, including technological, sociocultural, political–institutional, and economic–business dimensions [6–9]. Although a precise and comprehensive definition of a Smart City is still lacking, the number of scientific publications on the topic is significant and continues to grow. Defining this term comprehensively is beyond the scope of this paper but is relevant to our discussion.

Some researchers identify two generations of the Smart City concept. Smart City 1.0 is primarily concerned with technology (including transportation) and economic factors, while Smart City 2.0 emphasizes the importance of people, governance, and policy. This second generation addresses specific issues and citizen needs, with the aim of improving well-being and public services [10]. Therefore, mobility and various services offer opportunities for intelligent and sustainable movement for residents. The IMD World Competitiveness Center has released the Smart City Index (SCI) since 2019 as a pivotal tool within the domain of smart city assessment [11]. This annual assessment includes a global assessment of 142 cities around the world, ranking them according to two key pillars. The pilar Structures has an indicator that measures satisfaction with public transportation, which highlights the importance of public transport quality in assessing smart cities.

Public transportation must appeal to users and offer a viable alternative to private mobility. Some researchers have studied the spatial distribution of public transport services in two metropolitan areas (Rome and Turin), focusing on three key dimensions: mobility, competitiveness, and accessibility. The paper's findings show that private car indexes for the key dimensions are higher than those of public transport; thus, new strategies need to consider multiple dimensions to effectively support the use of public transport [12].

2. Related Work and Our Research Objectives

Planning and maintaining a city's public transportation network is a complex task. It involves not only determining routes and stop locations but also creating a timetable that requires careful planning and effective communication. Today, many people use mobile apps or online trip planners to navigate public transportation. As a result, transport companies and transit agencies describe and share their schedules, trips, routes, and stop data using a standard format known as GTFS (General Transit Feed Specification). GTFS was created by Google and TriMet in 2005 and has since evolved based on feedback from agencies and developers. A GTFS feed consists of at least 7 and up to 13 CSV files that represent routes, stops, stop times, trip calendars, and more. These files have a `.txt` extension and are compressed into a single zip file. In contrast, GTFS-RT (General Transit Feed Specification–Real Time) provides realtime updates, including vehicle locations, arrival time predictions, and alerts about detours and cancellations. These updates are essential for immediate trip planning [13]. Since public transportation impacts many people, researchers are actively studying these file formats [14–20] and their visualization possibilities [21,22].

A fast response time is essential for effective trip planning, which relies on the structure of the database used to store and deliver GTFS feeds. As new principles in database

management systems have emerged, scholars have started to focus on these innovative systems, including NoSQL databases [23]. This trend has also appeared in GTFS research, leading to benchmarking studies to evaluate the capabilities and potential applications of these systems. In study [24], a benchmarking tool developed in Java was introduced to compare the performance of Redis and MongoDB as databases for a trip planning application that utilizes GTFS data.

New algorithms have been developed to benefit from the new data models. The method *Range Mapping Hash* is a new approach that uses Redis NoSQL to find and calculate the accessibility of a trip plan with a fixed time complexity of $O(2)$ regardless of the GTFS size [25]. Another algorithm has been designed to maintain efficiency even as the data dimensions increase, such as when considering all times of day and extensive networks [26]. Other scholars also applied the Resource Description Framework model to create a benchmark system based on Madrid's GTFS data [27].

A recent study showed that published GTFS feeds might also contain errors. The research team surveyed a significant number of US feeds using the Canonical GTFS Schedule Validator tool and found that 21% of them had at least one error. Most of these issues were related to the optional `shape_distance_traveled` field. The study was limited to identifying only specification violations and did not analyze warnings and information generated by the validator. The authors pointed out that manual investigation might uncover even more issues [28].

A tool called GTFS-Viz for pre-processing and visualizing GTFS data is introduced in [29]. The authors developed a preprocessor that generates a CSV file from the GTFS feeds only to contain the necessary information required for their visualizer. In 2020, a more sophisticated public transit operation visualizer (called PubraVis) was published that measures and dynamically visualizes public transit operation from six perspectives: mobility, speed, flow, density, headway, and analysis. This last module summarizes and shows insightful analytical information [30]. This team continued their research and released an online visualizer (G2Viz) in 2024 [31]. Although the online application has many functionalities, it does not use GTFS-RT feeds. The use of GTFS data also attracted the attention of network scientists, who explored methods for extracting abstract transportation networks from GTFS feeds. However, some manual editing is necessary because providers do not always strictly adhere to the specifications when defining parent stations [32].

Researchers in paper [33] utilized both GTFS and GTFS-RT data to examine the spatial and temporal aspects of schedule padding and delays. The methods described, which include various measures and visualizations, can assist transit planners in continuously monitoring the schedule padding situation across a transit network. GTFS Realtime data can be used to quantify transit performance by tracking the time difference between the planned schedule and the realized one. This time is significant for the passengers and the professionals who design the schedule. The development of two metrics (systematic delays and stochastic delays) was introduced in [34], whose usage was also demonstrated in a case study of the King County Metro network in Seattle, Washington. The authors collected the data into a PostGIS database and used built-in functions and QGIS for visualization. The framework allowed for segment-based analysis of transit hotspots and differentiated between the delay types, which may require different types of treatment. Another study demonstrated how traffic volume, signal location, and other spatial and temporal data can be incorporated into the pipeline to evaluate bus lane effectiveness [35]. Bus arrival and departures time updates were in focus in [36] where scholars aimed to design a processing pipeline to archive the real-time information about bus operation. The data is essential for evaluating bus operational performance.

A recent paper explores the possibility of extracting transit information from GTFS through natural language instructions [37]. The authors tested the ability of GPT-3.5 Turbo and GPT-4 to answer questions about the GTFS specification and to extract information from the GTFS feed using two benchmarks. This research work is very innovative but does not relate to our investigations.

2.1. Public Transport Services of Budapest

A city's available public transportation system is one of the most critical pillars of motorizing everyday life. Budapest is the capital of Hungary, with a population of approximately 1.75 million residents and a density of 3200 people/km^2. The population of Budapest was similar in the last decades without significant changes; however, the population of county Pest that forms the agglomeration has had a monotonous increase since 2001, according to the official reports of the Hungarian Statistical Bureau. Moreover, the number of workers hired in micro and macro-scaled businesses was 32.7% in Budapest.

The Budapesti Közlekedési Központ (in brief, BKK, in English, Budapest Transportation Center) was established in 2011 by the metropolitan municipality to manage the transportation system in the capital. This integrated network offers various modes of transport, including metro, suburban rail, tram, trolleybus, and bus services. In addition, BKK operates a public bike service called MOL Bubi, as well as taxi services.

As of the time of writing this paper, according to BKK's website, the transportation network accommodates around 1900 buses, trams, trolleybuses, metro trains, and suburban trains on a single day, collectively providing over 43,000 departures. In addition, similar to other metropolitan areas, the use of micromobility options is on the rise in Budapest. These include walking, cycling, scooters, rollerblading, e-bikes, hoverboards, and more, which complement traditional forms of transportation. Micromobility options enable access to areas of the city that are difficult to reach by car or public transport [38]. A recent study examined the satisfaction rate of citizens with public transportation in Budapest and identified cleanliness and equipment of vehicles and stops, frequency of vehicles and their correspondence to the timetable, passenger information, and barrier-free accessibility as the most important factors that determined the public's feelings about BKK's service [39].

2.2. Research Objectives

Engaging with GTFS and GTFS-RT data opens up numerous impactful research opportunities across diverse fields.

- From a database theory perspective, we have a classic relational database stored in a ZIP file, which offers an efficient storage format thanks to its encoding scheme, resulting in a significant compression ratio. The GTFS file contains trip data for the next 30 days, and, according to the specification, changes are not recommended within 7 days of the planned trip dates. This allows trip planning applications to pre-process the data before users begin to query it intensively. GTFS-RT feeds provide real-time updates on vehicle positions, arrival time predictions, and alerts for detours and cancellations. Efficient utilization and processing of this data are crucial. To create a robust application capable of delivering real-time responses to user requests, it is essential to handle these file formats together. From a scientific standpoint, building a historical database that can be analyzed in different contexts is also of great interest.
- If geoinformatics perspectives are the focus, visualization and geospatial accuracy receive greater emphasis. For instance, it is crucial to ensure that the alignment and representation of the stops and the waypoints determining the route shapes are accurate.

- As a computer science-related topic, an efficient graph search algorithm is essential for planning trips between various stations, often involving multiple services and transfers. This challenge is significant because the graphs can extend not only across a city but also across entire continents.

The main objective of our research is to fetch, pre-process, and analyze the GTFS and GTFS-RT data of public transport systems and provide a visual interface that supports understanding the data and decoding its hidden information. The research started in August 2023 by requiring our token for the BKK OpenData Portal, starting to collect the daily released timetable schedules in the GTFS format, and retrieving the live vehicle positions using BKK's custom API serving as an alternate to the *Vehicle Positions* GTFS-RT feed. Later, we started to use all three GTFS-RT feeds that are provided by the agency: the feed *Vehicle Positions* to retrieve the live vehicle positions in the standard format, the feed *Alerts* to retrieve the updates about the planned and ad-hoc service changes, and the feed *Trip Updates* to collect real-time data about canceled, updated, or added trips.

Our multidisciplinary research aims to offer several features that can be interesting to a number of professionals and scholars:

- Consistent geovisualization requires consistent data; thus after collecting data we cleaned the data in the frame of a pre-processing;
- Building a historical dataset and using actual vehicle positions, which offers identification of the common features of disruptions and changes and the prediction of live arrival times with further traffic analysis.

This paper presents our retrospective analysis of our data-collecting methods after the first one-year period of hosting our data lake service. First, we describe the used services offered by the BKK OpenData Portal, including the most important features of the available data. In parallel, we also introduce the method of data retrieval in Section 3. The second part of this paper, in the form of Section 4.1, introduces our pre-processing algorithm that was applied to the dataset with its statistical features, such as the ratio of tracked trips and classifying tracked trips based on their accuracy. We believe that our reconstructable method and supporting features in our dataset will serve as a basis for other researchers who plan to start research based on a similar methodology. Figure 1 shows that the focus is on the GTFS archives and GTFS-RT *Vehicle Positions* feed, but the also captured GTFS-RT *Trip Updates* and *Service Alerts* feeds will provide an opportunity for cross-checking and extended analysis in the near future.

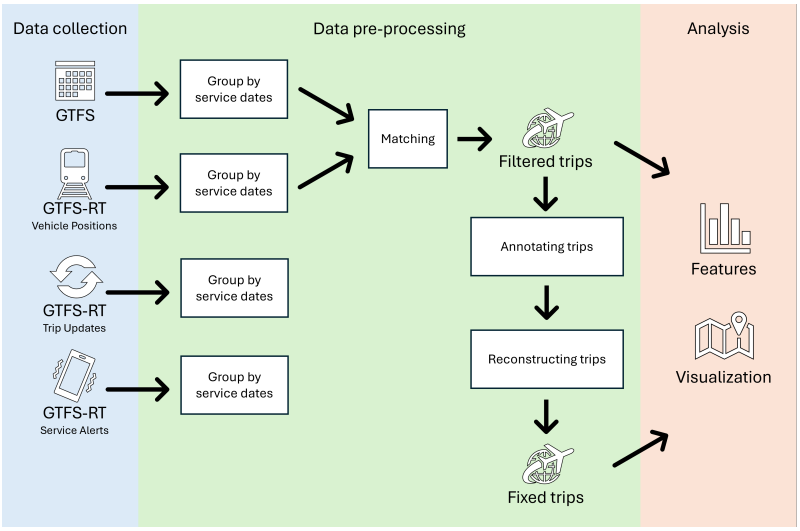

Figure 1. The proposed pre-processing stages, focusing on the planned timetables of GTFS archives and the GTFS-RT *Vehicle Positions* feed.

3. Materials and Methods

3.1. Data Sources

In this subsection, we introduce our data sources and give a brief description of their formats to understand the further sections that describe the method used for the pre-processing.

3.1.1. Static Source (GTFS)

BKK has provided its timetable schedule in the GTFS format since 2012. Currently, the daily-updated archives can be accessed via BKK's OpenData Portal (https://go.bkk.hu/api/static/v1/public-gtfs/budapest_gtfs.zip, accessed on 21 November 2024). Since the GTFS specification lets agencies customize the exact structure of their archives using optional or conditionally required fields in the files, we summarize the most important features of the seven mandatory and two optional files provided by BKK:

1. `agency.txt`—The archive introduces two agencies having identifiers `BKK` and `HEV`. Agency `BKK` operates buses, trolleybuses, and trams, the other company MÁV-HÉV has replaced BKK for operating the suburban railway lines as the subsidiary of the company Hungarian State Railways (MÁV), referred as the agency `MAV`.
2. `calendar_dates.txt`—As the GTFS specification states, the recommended way would be to specify patterns of services using entries `calendar_dates.txt` and `calendar.txt` together. However, archives use the alternate approach: each scheduled service is specified explicitly in the file `calendar_dates`. The schedule is available typically for the upcoming 1–2 month period.
3. `routes.txt`—Different routes are identified using the following numbering scheme:
   - Identifiers starting with the digits `0`, `1`, and `2` denote bus routes;
   - Identifiers starting with the digit `3` denote tram routes;
   - Identifiers starting with the digit `4` denote trolleybus routes;
   - Identifiers starting with the digit `5` denote metro routes;
   - Identifiers starting with the digit `8` denote boat services on the river Danube;
   - Identifiers starting with the digit `9` denote night bus routes;
   - Identifiers starting with the prefix `VP` (abbreviation for the Hungarian tram replacement expression) denote replacement bus services for tram routes, and identifiers starting with the prefix `MP` (abbrevation for the Hungarian metro replacement expression) denote replacement services for metro routes. Replacement services can be scheduled for various reasons, such as events and maintenance work;
   - The schedules also contain additional routes for identifying the suburban railway services, special trips such as heritage trams, etc.
4. `shapes.txt`—As one of the optional files, BKK's GTFS archives contain the segments for representing the geoinformation for all variants of routes. This allows consumers to display the routes using various visualization techniques.
5. `stop_times.txt`—This file contains one optional field: the field `shape_dist_traveled` gives the actual distance traveled along the associated shape.
6. `stops.txt`—This file contains three optional fields and one non-standard field: fields `stop_code`, `location_type`, `wheelchair_boarding`, and `location_sub_type`.
7. `trips.txt`—This file contains six optional fields: `trip_headsign`, `direction_id`, `block_id`, `wheelchair_accessible`, and `bikes_allowed`.

3.1.2. Realtime Sources (GTFS-RT)

BKK's OpenData Platform implements all three types of real-time feeds introduced by the GTFS-RT specification. In this paper, we are going to primarily use the *Vehicle Positions* feed, providing the following properties for each `VehiclePosition` message:

1. `trip`—Describes the *Trip* that the vehicle is serving. The corresponding `TripDescriptor` object contains the optional `schedule_relationship` field.
2. `position`—Describes the current position of the vehicle. The corresponding `Position` object contains the optional fields `bearing` and `speed`.
3. `current_stop_sequence`—The index of the current stop (that the vehicle is staying or is heading to).
4. `current_status`—The status of the vehicle.
5. `timestamp`—Moment at which the vehicle's position was measured.
6. `stop_id`—The ID of the vehicle's current stop.
7. `license_plate`: The license plate of the vehicle.

3.1.3. FUTÁR API

BKK also provides an easy-to-use REST API whose specification is available in the OpenAPI 3.0 format, (accessed on 21 November 2024). The various endpoints provide live information about the vehicle positions (serving as an alternative to the GTFS-RT trip updates), but trip planning and data from the public bike-sharing system Bubi are also available. Moreover, data from the GTFS files can also be queried, such as the locations of the stops.

3.2. Collecting Data

Our scheduled API requests were sent by a Python 3.12 service that was hosted on an Ubuntu virtual machine of the Faculty of Informatics, University of Debrecen. As the machine has a limited amount of disk space, we developed a workflow using GitHub's REST API version 2022-11-28 to commit the responses immediately to the dedicated repository and store data on the local disk only if the GitHub commit was unsuccessful (except the GTFS schedules because of their sizes). Using this process, we started to collect the scheduled timetables and the live vehicle positions. The vehicle positions were obtained from the FUTÁR API in format JSON. During the data collection period, we have continuously improved the frequencies and included additional endpoints. Table 1 contains a brief summary of the data collecting periods, including the available APIs, and their refresh ratios.

Table 1. Periods of data collection together with the frequencies used for the data retrieval (in s).

ID	Start Date	End Date	Vehicle Positions	Trip Updates	Service Alerts
`P1`	1 September 2023	17 September 2023	30	—	—
`P2`	18 September 2023	3 May 2024	15	—	—
`P3`	4 May 2024	20 July 2024	15	—	—
`P4`	21 July 2024	31 August 2024	5	600	600

1. Period `P1`—The initial period started on 1 September 2023. At that moment, the FUTÁR API's `vehicles-for-location` endpoint was invoked every 30 s to gather the live vehicle positions. We started to download the GTFS schedules twice a day: primarily, the state of the archive was fetched between 01:00 a.m. and 02:00 a.m.; however, a secondary fallback operation was also introduced between 01:00 p.m. and

02:00 p.m. In most of the cases, our service was able to download both versions; additionally, they were identical. In the case of differences, we kept the afternoon version of the schedule to consider the latest released content as next day's schedule.

2. Period `P2`—After a two-weeks long experience using the REST endpoint and our environment, we increased the frequency of the requests to 15 s, doubling the number of API calls within the same period starting on 18 September 2023.
3. Period `P3`—On 29 April 2023, BKK deployed a new version of its API, introducing various limits on the queries. Thus, our previous calls—which queried the full area of Budapest for the vehicle positions—could not be used anymore. Realizing this change of the API, we discontinued the use of the REST API. Starting 4 May 2023, we replaced the old method by fetching the *Vehicle Positions* GTFS-RT feed for retrieving the live vehicle positions in its PB format instead of consuming the REST API's JSON responses. This also introduced changes in the fetched data: additional properties are served using the standard format, and all the minor differences could be post-processed and unified between the multiple data sources.
4. Period `P4`—Starting 21 July 2024, we increased the retrieval frequency by replacing the previous value of 15 s to 5 s. In parallel, we established the collection of the other two GTFS-RT feeds: the frequency of 600 s was introduced for the feeds *Vehicle Positions* and *Service Alerts*.

We must note that the data fetching service was updated on the day preceding the official start date of the given period. Thus, the last day of the previous period could already collect data using the increased frequencies. This feature will affect the further figures and tables since Table 1 contains the first dates that used only the new version of the script.

3.3. Pre-Processing Data

In this subsection, we give an overview of our pre-processing stages that were performed on the collected data. All the steps were designed based on the experimental analysis of the BKK dataset, using the special characteristics of the available data.

3.3.1. Cleaning the GTFS Schedules

As we mentioned in Section 3, each `calendar_dates.txt` file contains entries for the next couple of weeks. As a consequence, a huge subset of the records is unnecessary if we want to analyze a date based on the latest available schedule from the previous day. Denote the current date as d_A, denote the previous day as d_P, and the next day as d_N. To use the latest available schedule d_A, we have to consider the schedule published on the preceding day d_P. Moreover, the metainformation stored in the route identifiers can also be used to determine what entries can be tracked using the GTFS-RT feed *Vehicle Positions* and what routes are untrackable—thus, irrelevant for further research stages:

- The bus, trolleybus, and tram lines served by the `BKK` agency serve as the basis for tracked routes; thus, we must keep and process them in the further stages.
- The suburban railway lines are operated by the `MÁV-HÉV` agency and provided as a standalone service not sharing any tracks with other routes; thus, we keep only the routes served by the `BKK` agency only.
- The tram-replacement routes having the prefix `VP` and the metro-replacement routes having the prefix `MP` can also have scheduled trips that can be tracked and analyzed identically to the regular bus, tram, and trolleybus routes.
- Since the metro lines are untrackable, we can eliminate all their trips from the schedule.
- Additional lines such as the boat service, heritage lines, etc., can be considered as noise and they must be also eliminated.

Based on the previous ideas, we clean all the GTFS archives and retrieve the schedule timetable for d_A by performing the next algorithm on the archive of d_P:

1. Remove all the records from file `calendar_dates.txt` that do not belong to d_A. Yield the S' set of filtered services.
2. Remove all the records from file `trips.txt` that are not added for any service of S' and keep only the bus, tram, trolleybus, and replacement routes. Yield the R' set of filtered routes, the T' set of filtered trips, and the SH' set of filtered shapes.
3. Remove all the records from file `routes.txt` that cannot be found in R'.
4. Remove all the stops from file `stops.txt` that are not referred to in T'. Yield the ST' set of filtered stop IDs.
5. Remove all the records of file `stops.txt` that cannot be found in ST'.
6. Remove all the records of file `shapes.txt` that are not referred to in SH'.
7. Files `pathways.txt`, `feed_info.txt`, and `agencies.txt` can be removed since they do not contain any relevant information for the further research phases.

Further sections will refer to the output as the cleaned schedule for d_A.

3.3.2. Grouping Vehicle Positions

The vehicle positions were queried with the frequencies of 30, 15, and 5 s. Performing the next stage of the pre-processing, we group the tracked vehicle positions by their trips. On the other hand, there are trips that run partially or exclusively on the next day of the service date. The precise management of these trips is crucial and should be done carefully since the operators can reuse the same trip identifiers on consecutive days. The association of these trips should be deterministic. As a solution to this problem, we introduce the following rules:

1. A trip that belongs to service date d_A with its departure time scheduled for d_A as well belongs to date d_A. In other words, the scheduled departure time for the departure stop determines what day should belong to the given trip.
2. A trip that belongs to service date d_A with its departure time scheduled for d_N belongs to d_N. In other words, a trip will be processed as part of the next service date if it departs after midnight.

In this stage, we group the trips by their (d_S, d_D) pairs, where d_S denotes the service day, while d_D denotes the first departure time of the trip. Collecting the trips in these groups technically help the multi-thread or multi-process implementation of the implementation, since there is no concurrent write access for the corresponding archives.

3.3.3. Matching Vehicle Positions and Scheduled Timetables

In this stage, we match grouped trips reconstructed from the live vehicle positions with their scheduled timetables, yielding a single archive for each date containing all the trips that must depart on the same day. Due to the nature of the dataset and the already applied stages, the non-scheduled trips can be detected without using the *Trip Updates* feed because no matching planned trip can be found for them. As a consequence, three groups of trips can be yielded:

1. The group of scheduled trips that could be tracked;
2. The group of scheduled trips that could not be tracked;
3. The group of non-scheduled trips that could be tracked.

Later—by fetching and processing the *Trip Updates* feed—the second group can be divided into two subgroups:

- The group of scheduled trips that could not be tracked and their cancellation was confirmed;
- The group of scheduled trips that could not be tracked and their cancellation was not confirmed.

Based on the GTFS-RT specification, the second group can be considered as the group of trips for which live-time tracking is unavailable.

3.3.4. Annotating and Validating Trips

During our research, we manually detected anomalies in the case of many trips. The following three cases can be considered common anomalies, resulting in noises:

- Multiple vehicles were tracked using the same trip identifier (in most of the cases, their services were redesigned at their origins before departing; in other cases, the original vehicle canceled its trip, and a replacement vehicle departed from one of the intermediate stops);
- The GPS locations could be inaccurate, resulting in "jumping" vehicles. This behavior could be detected especially at origins, before departing: in these cases, the system detected that the vehicle was already heading to any of the next stops, then was reassigned to the initial stop;
- Vehicles did not log off from their trips after reaching their terminus. This resulted in extra entries that were not active, just parking or moving around the terminuses or bus depots.

To automatically detect and later eliminate these anomalies, we define the function `ANNOTATE()`, which gets the sorted sequence of a tracked trips' vehicle positions. The function returns a label that describes the status of the trip. The function checks the labels in the following order and assigns the first matching one to the trip (can be applied on both the clean and reconstructed trips):

1. `EMPTY`—if the trip has no entries (only applicable in the case of reconstructed trips);
2. `VEHICLE`—if multiple vehicles logged in for completing the same trip;
3. `UNKNOWN`—if an entry does not have a sequence number;
4. `MIXED`—if the tracked sequence numbers do not form a monotonous increasing sequence;
5. `HEAD_2`—if the vehicle could not be tracked while heading to or staying at the origin or its subsequent (first) stop;
6. `TAIL_2`—if the vehicle could not be tracked while heading to or staying at the terminus or its preceding (last) stop;
7. `GAP_2`—if the vehicle could not be tracked while heading to or staying at two consecutive stops, but the preceding and subsequent stops were tracked;
8. `HEAD_1`—if the vehicle could not be tracked before leaving its origin;
9. `TAIL_1`—if the vehicle could not be tracked while heading to or staying at its terminus;
10. `GAP_1`—if the vehicle could not be tracked while heading to or staying at a stop, but its preceding and subsequent stops were tracked.

Based on the introduced annotations, we define two different methods for the validation of the tracked trips:

1. Strong validation—Using this technique, we assign each trip to category `INVALID` if any of the labels representing anomalies are present. The category `NON_TRACKED` contains those trips that did not appear in the *Vehicle Positions* feed, while the category `VALID` contains those trips that do not have any of the labels.
2. Weak validation—Using this technique, we assign each trip to category `INVALID` if any of the labels representing anomalies are present except the `HEAD_1`, `TAIL_1`, and `GAP_1` ones. The category `NON_TRACKED` contains those trips that did not appear in the feed

Vehicle Positions, while the category `VALID` contains those trips that do not have any of the labels. As a consequence, this validation technique accepts those trips whose stop sequence does not lack more than one consecutive stop.

3.3.5. Reconstructing Trips

To resolve the errors detected during the validation, we define the function `RECONSTRUCT()`, which gets the sorted sequence of a tracked trip's entries. The function returns the sequence of reconstructed entries by applying the following steps:

1. Traverse all the entries and yield the vehicle that appears the most frequently;
2. Remove e_i if it does not have the most frequent vehicle;
3. Remove e_i if its stop sequence is unknown;
4. Remove e_i if its stop sequence is greater than the stop sequence of e_{i-1};
5. Remove e_0 if its stop sequence is the same as the stop sequence of e_0 and both of the *stop distance* fields are 0. (The vehicle stays at its origin. In this context, origin means the first detected stop, which can differ from the real one.) Repeat this step while the condition is fulfilled.
6. Remove e_{-1} if its stop sequence is the same as the stop sequence of e_{-2} and both of the stop distance fields are 0. (The vehicle stays at its terminus. In this context, terminus means the last detected stop, which can differ from the real one.) Repeat this step while the condition is fulfilled;

In normal cases, the *stop distance* values form a monotonous decreasing sequence as the vehicle approaches its next stop. However, traffic deviations can change this feature; thus, this feature is neither validated nor reconstructed.

4. Results and Discussion

In this section, we share the dataset's most important statistical features and the impact of the reconstruction algorithm. First, we focus on the availability and the correctness of the trips; second, we introduce our interactive visualization that is designed to support the analysis of the dataset.

4.1. Statistical Features of the Dataset

4.1.1. Availability of Our Data Fetching Service

Both the API and GTFS-RT feeds of BKK are robust: the downtime of the data collection technique was our virtual machine's and GitHub-based workflow's technical issues. During the period, we detected only a few-minutes-long shortages in the APIs. However—due to our technical issues—nine days could not be tracked. On 346 out of 366 days (94.54%) we could track at least 90% of the scheduled trips. The technical problems affected 9 days in an acceptable amount, making the majority (over 70%) of the trips trackable. However, the remaining 10 days cannot be evaluated since the majority of the scheduled trips were untracked.

4.1.2. Ratio and Age of Tracked Trips

We analyzed the features of the tracked trips before and after the execution of our reconstruction stage. Two important questions can be answered by applying the weak and strong validation techniques on the states:

1. What is the age (the elapsed time between the timestamps of recording and fetching) of the collected vehicle positions?
2. Could the age be enhanced by increasing the frequency of the requests over the periods?

We created Figure 2 to answer both the questions by adding four (4) data series, having the following features:

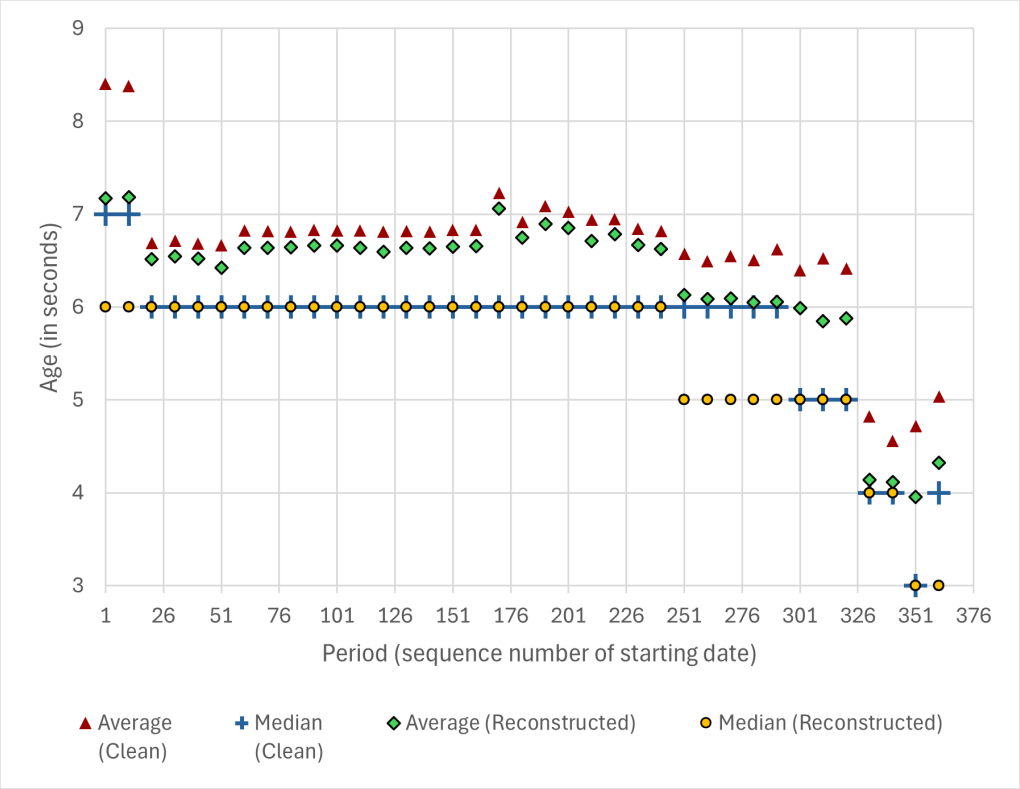

Figure 2. Average and median of the tracked vehicle positions' age over the service dates. Each marker represents the median for the given data series in the period of 10 days, starting on the marked day.

- Average (Clean)—This series contains the average age of the clean dataset per service date. The period `P1` had an average of 8.34 s, while periods `P2` and `P3` had an average of 6.79 s; finally, period `P4` had an average of 4.89 s. Consequently, the age of the cleaned dataset decreased by increasing the query frequency through the periods.
- Median (Clean)—The pattern that can be realized in the medians is similar to the averages' one: in period `P1`, we could only reach a median of 7, but period `P3` could reach a median of 3, which guarantees extremely accurate and fresh data.
- Average (Reconstructed)—An interesting feature can be detected in the dataset: the average delta decreases if the reconstruction stage is applied. As a speculation, BKK's system determines the analyzed properties such as *stop sequences* and *stop ID* more accurately if the vehicle positions are fresh; moreover, vehicles that do not sign off from their trips generate greater deltas while standing in the terminuses or in depots.
- Median (Reconstructed)—The clean and reconstructed medians meet each other. The median was always less than the average except for period `P4`.

4.1.3. Validating the Raw and Cleaned Scheduled Trips

Using our methods `ANNOTATE()` and `VALIDATE()`, we can perform both a strong and weak validation of our raw and pre-processed (reconstructed) versions of the dataset:

- Applying the strong validation before performing the reconstruction method on the tracked trips, we can detect only 29 days providing more than 20% valid trips, while the median of the non-tracked trips' ratio is 1.97%. Finally, the median of the invalid trips' ratio is 82.45%.

- Applying the strong validation after performing the reconstruction method on the tracked trips, the median of the valid trips' ratio is 78.13%, while the median of the invalid trips' ratio is 19.28%.
- Applying the weak validation after performing the reconstruction method on the tracked trips, accepting trips annotated with the labels `HEAD_1`, `TAIL_1`, and `GAP_1` increases the median of the valid trips' ratio to 94.46%, decreasing the median of the invalid trips' ratio to 3.36%.

4.2. Interactive Visualization of the Dataset

To support the presentation and the analysis of the dataset, we have developed a web application called DEPOT that is capable of visualizing the most important geospatial features of the GTFS schedules, enhanced by the data of GTFS-RT feed *Vehicle Positions*. The application is hosted on the local server of the University of Debrecen, Faculty of Informatics.

The application was developed using Leaflet, and a self-hosted OpenStreetMap TileServer powered by the `tileserver-gl` docker image. The TileServer serves the data obtained from the Geofabrik. Our primary goal was to focus primarily on the `shapes.txt` and `routes.txt` files. First, the user selects the service date, followed by the investigated period. The application displays the list of routes that have scheduled trips in the given time period.

The application provides four layers for the interactive visualization of stops and shapes of the GTFS files and the aggregated entries of the GTFS-RT feed *Vehicle Positions*:

- Layer `map` covers Hungary to support suburban routes leaving the boundary of Budapest.
- Layer `shapes` contains those shapes that had scheduled trips for the selected time period for the given routes.
- Layer `stops` contains markers for the available stops of the displayed shapes.
- Layer `vehicles` contains markers for those positions that have recorded vehicles in the GTFS-RT feed *Vehicle Positions*.

The motivation was to create an interactive environment for the customized visualization; thus, the z-indices of the `shapes`, `stops`, and `vehicles` layers can be reorganized using a drag and drop method. Moreover, each non-map layer can be toggled, as well as their interaction with the mouse. All the visualization features can be edited in the left-side panel of the desktop-optimized UI; the Leaflet-based visualization is displayed as the right-side panel. Another priority was to provide the opportunity to provide the visualization's state for further reference and sharing; thus, the current setup is stored as the URLs' query parameters. The application supports zoom levels between the range of 10 and 21, providing smooth adjustment based on the boundaries and zoom level.

4.2.1. Layer `Shapes`

This layer displays the scheduled routes with all their shapes, constructing `PolyLine` objects from the waypoints specified by the `shapes.txt` file. When the cursor moves over a segment, a tooltip displays all corresponding routes and shapes. Moreover, a right-aligned `Circle` marker is added for the midpoint of each segment. Segments can be displayed in unified or random-assigned colors. The latter option supports the detection of duplicated and overlapping segments of multiple shapes (see Figure 3).

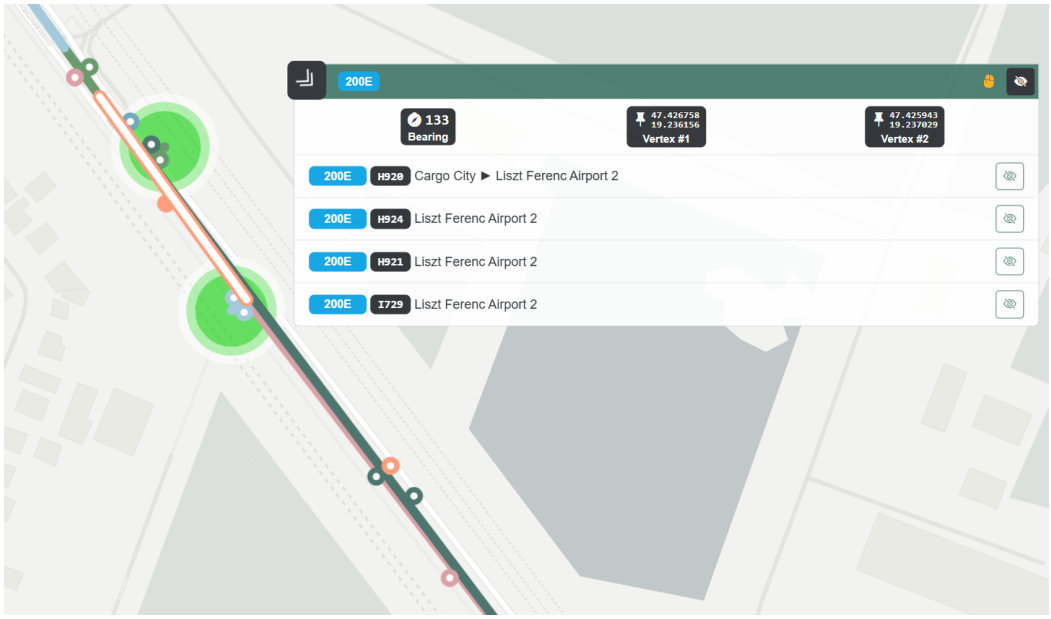

Figure 3. Overlapping edges of routes `100E` and `200E` around the stop "Repülőtér, D porta", serving only route `200E`. The number and the alignment of the `Circle` markers show that multiple segments overlap since the shapes do not share the same waypoints. In the southern section of the screenshot, an orange and green edge overlap, having the bearing of `311` degrees; in the other direction, another green and pink edge overlap each other, with the bearing of `131` degrees. The selected orange segment between the two stops is shared by shapes `H920`, `H924`, `H921`, and `I729` of route `200E` on service date 16 September 2023. The interactive visualization can be accessed here.

As Figure 3 shows, waypoints provided in the `shapes.txt` are not necessarily creating a unique subset for those shapes that share sections. This feature results in an anomaly that makes the automatized processing of the shapes impossible. We have detected various scenarios in the investigated GTFS files:

- Segment (P_1, P_2), belonging to direction `0` of a route usually appears as segment (P_2, P_1), describing the same route in direction `1`. If the streets are narrow, this representation is the most suitable for processing the data and visualizing the routes.
- Parallel shapes—which are heading in the same direction on the same street—are built from different waypoints, resulting in a disjunct but overlapping set of edges. For creating a network (e.g., for visualizing traffic-related congestion, delay times, and throughput), a post-processing algorithm would be required to simplify the shapes and rebuild them from common vertices and shared edges.
- Waypoints are added with precision errors; this feature can be recognized by having local knowledge, but the visualization of overlapping edges can simplify their detection. For example, the "Liszt Ferenc Airport 2" is served by the already visualized routes `100E` and `200E`; the former one is an express route from the downtown, and the second one provides regular service from the Eastern districts of Budapest. Their shapes provide many anomalies, such as crossing each other and being misplaced on the wrong side of the expressway around the terminals. This leads to placing vertices further away from their desired location than the real distance of them in those sections where the `100E` route uses a rapid highway while the `200E` route uses the parallel street. Thus, fixing these shapes by applying a constant threshold and merging the parallel edges would be insufficient.

To make the investigation of the edges easier, users can apply a bearing (direction) filter on the dataset, choosing from two different threshold values. This is similar to a classic edge detection filter for image processing, hiding all the directed edges that are not facing the selected direction. Users can filter the displayed shapes by dynamically toggling them.

4.2.2. Layer `Vehicles`

As we aggregated the entries of the GTFS-RT feed *Vehicle Positions*, the tracked GPS positions of the vehicles serving planned trips of the selected routes were recognized. Based on the zoom level of the map, we merged nearby locations to form groups with a diameter of 10 m for zoom levels of range 17–21, 100 m for zoom levels of range 14–16, and 500 m for zoom levels of range 10–13. When moving the cursor over a location, a tooltip displays those routes and their shapes that had recorded positions in the given GPS range. Moreover, the tracked number of vehicles appears by their bearing (direction). These features can help users to identify various anomalies:

- Tracked positions can be inaccurate, especially when vehicles are staying at stops and surrounded by buildings. As our hypothesis suggests, this feature can also lead to miscalculations of the environment and result in the representation of trip logs. This feature was detected near the terminus "Liszt Ferenc Airport 2", where the vehicles just under the facade of the terminal building and an elevated road are also partially covered (see Figure 4).
- Misaligned waypoints and edges can be detected by displaying the vehicles serving the corresponding trips. A reconstruction algorithm could adjust the shapes dynamically by checking the vehicle positions.
- For future visualizations, the throughput, delays, and all the features can be visualized for individual directions using the bearing (direction) filters. Busy junctions do not necessarily have the same features for all the crossing directions.

To open more perspectives for analysis, users can choose between two different methods of displaying data:

- *Colors mode* displays each position as a set of `Circle` markers aligned to the middle of the covered area. By default, the GTFS routes' official colors are being used; however, users can differentiate each route from one another, providing an opportunity to distinguish different kinds of services such as trams, trolleybuses, and buses. This feature was used in Figure 4 for the routes `100E` and `200E`.
- *Heatmap mode* displays the tracked positions as a heatmap; the opacity and the size of a marker are greater if more recorded vehicles are part of it in the displayed area. Similar to the previous mode, each color has its own component, making their parallel visualization possible. This feature is used in Section 4.2.3 to visualize the amount of noise around the terminuses.

To support the visualization of our reconstruction method introduced in Section 3.3, three sublayers are available:

- Sublayer `vehicles-filtered`—contains the filtered set of tracked positions;
- Sublayer `vehicles-fixed`—contains the reconstructed set of tracked positions;
- Sublayer `vehicles-diff`—contains those vehicle positions that were classified as noise by our validating and reconstruction algorithm.

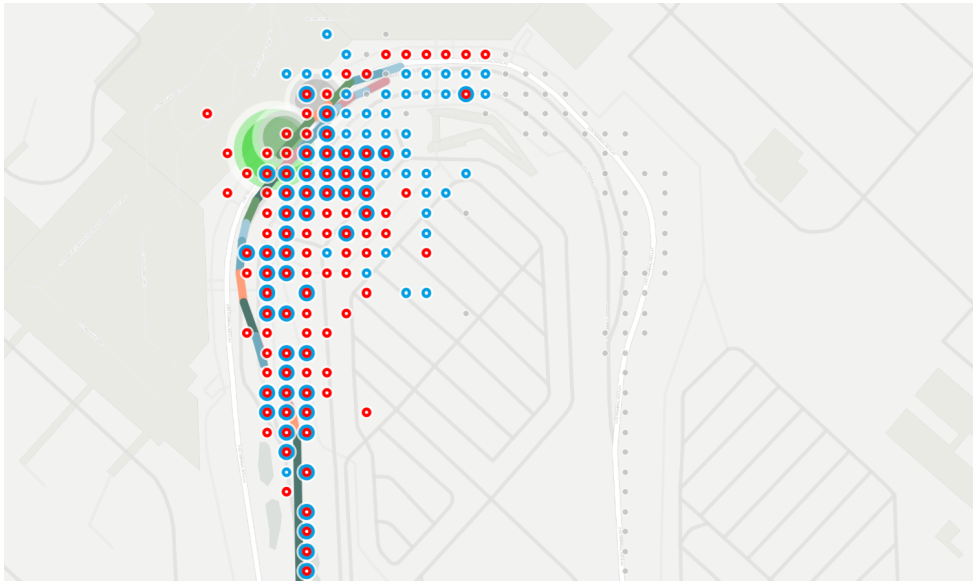

Figure 4. Tracked positions of routes `100E` and `200E`, staying at stop "Liszt Ferenc Airport 2". The stops of these routes are located under the facade of the terminal building, generating a significant amount of precision errors, exceeding the diameter of 100 m on the service date 7 September 2023. The interactive visualization can be accessed here.

4.2.3. Analysis

Figure 5 shows the nearby streets of Keleti Railway station, which is one of the busiest locations in Budapest, serving trolleybus, bus, tram, underground, and night bus routes. The vehicle positions of vehicles for bus routes `7` and `7E`, tram routes `23` and `24`, trolleybus routes `73`, `76`, `78`, `79`, and `80`, and night bus routes `931` and `931A` are displayed using the `Circle` markers. Even though the displayed routes have four different colors, it can be easily identified that positions and directions belong to each of them. All routes but the buses have terminuses around the square; the many trolleybus positions on the eastern side belong to the vehicles waiting for their trips. Figure 6 shows the same area, but contains the `vehicles-diff` sublayer. Putting the two figures near each other clearly shows that the bus routes are not affected by the reconstruction algorithm, while the tram and trolley bus routes had a significant number of noise positions along the square and the Keleti Railway station's building.

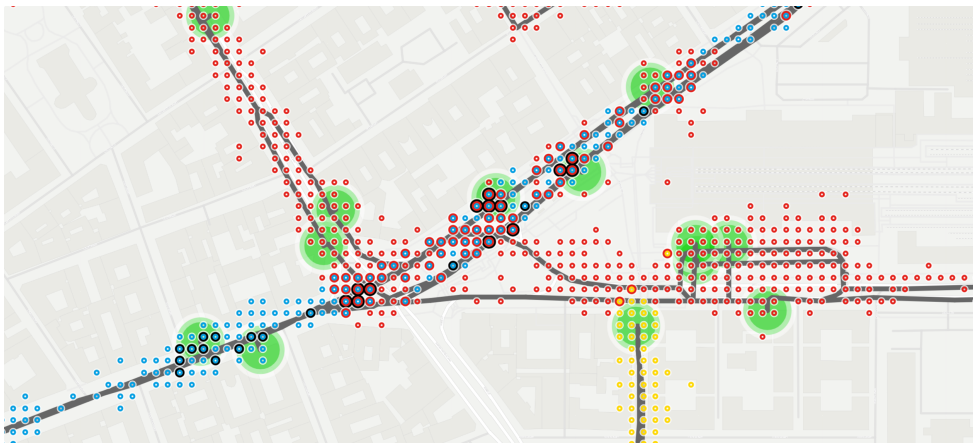

Figure 5. Keleti Railway Station and its surroundings on service date 16 September 2023, using the `vehicles-fixed` sublayer. Trolleybus routes are displayed in red, bus routes in blue, tram routes in yellow, and night bus routes in black. The interactive visualization can be accessed here.

Figure 6. Keleti Railway Station and its surroundings on service date 16 September 2023, using the `vehicles-diff` sublayer. Trolleybus routes are displayed in red, bus routes in blue, tram routes in yellow, and night bus routes in black. The interactive visualization can be accessed here.

Figures 7 and 8 show how the bearing (direction) filter is applied to the layers `vehicles` and `shapes`, displaying the roundabout near Budapest Airport. Here, the shapes of routes `100E` and `200E` contain an inconsistency since both the routes enter and leave the roundabout the same. Figure 7 displays the layers without filtering, while Figure 8 uses the small version of the filter with the bearing (direction) of 160.

Using routes `100E` and `200E`, our reconstruction method can also be well-demonstrated as the characteristics of the heatmap change significantly when switching from the `vehicles-filtered` sublayer to the `vehicles-fixed` sublayer (see Figures 9 and 10) since the original gap between the lower and higher intensities (highlighting the terminuses) totally disappear. having clear high-intensity points around the terminuses "Liszt Ferenc Airport 2" (shared by both the routes), "Határ út"/"Kőbánya-Kispest" (the terminus of route `200E`, where route `100E` passes through), and "Deák Ferenc tér" (the terminus of route `100E`). To help distinguish the routes, the original color of route `100E` was replaced with red.

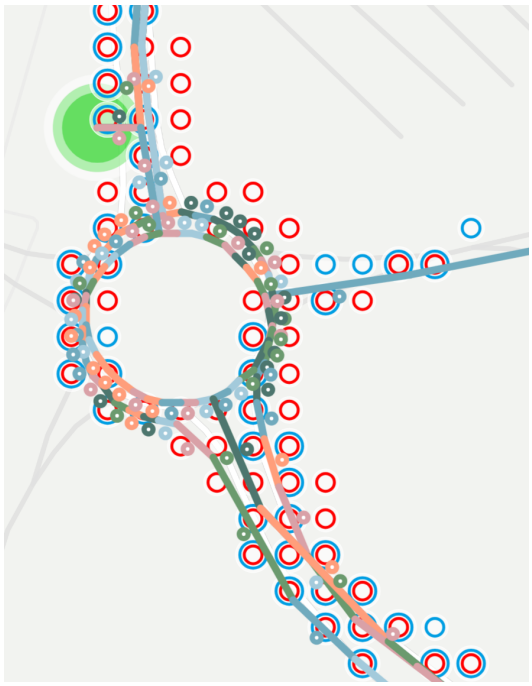

Figure 7. Filtering layers based on bearing (direction). The interactive visualization can be accessed here.

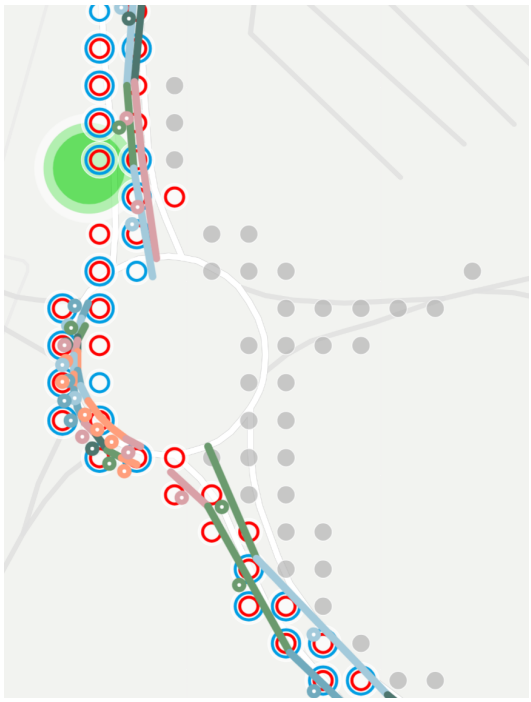

Figure 8. Filtering layers based on bearing (direction). The interactive visualization can be accessed here.

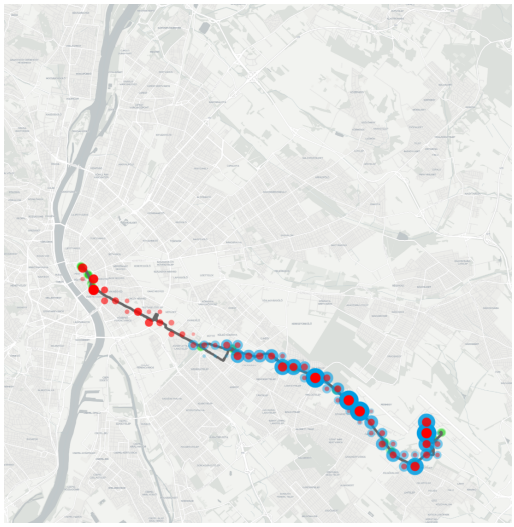

Figure 9. The `vehicles-fixed` sublayer using the *Heatmap mode* for routes `100E` and `200E`, on 16 September 2023. The interactive visualization can be accessed here.

To understand this behavior, we must check the `vehicles-diff` sublayer both in *Color mode* as seen in Figure 11 and *Heatmap mode* as seen in Figure 12. The original vehicle positions contain two anomalies:

- Many noises were recorded around the terminuses, as Figure 12 shows.
- As Figure 11 shows, various locations were tracked in the Northern and Southern districts. The vehicles left the shapes belonging to routes `100E` and `200E`. Manually checking the extra positions, an interesting exploration can be made: the red markers head to the depot of the company providing the service for route `100E`, while the blue markers head to the depot of the company providing the service for route `200E`.

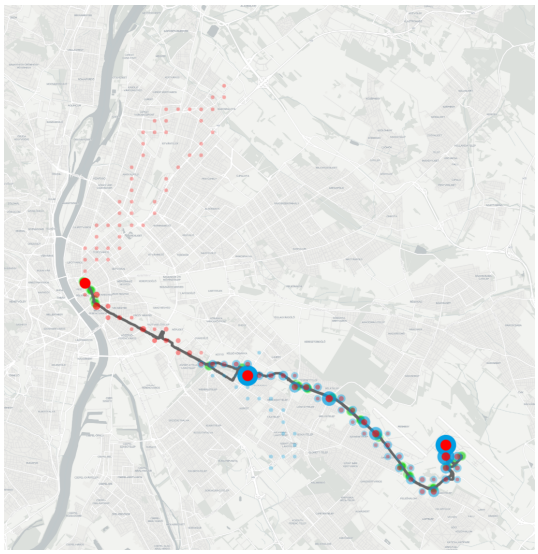

Figure 10. The `vehicles-filtered` sublayer using the *Heatmap mode* for routes `100E` and `200E`, on 16 September 2023. The interactive visualization can be accessed here.

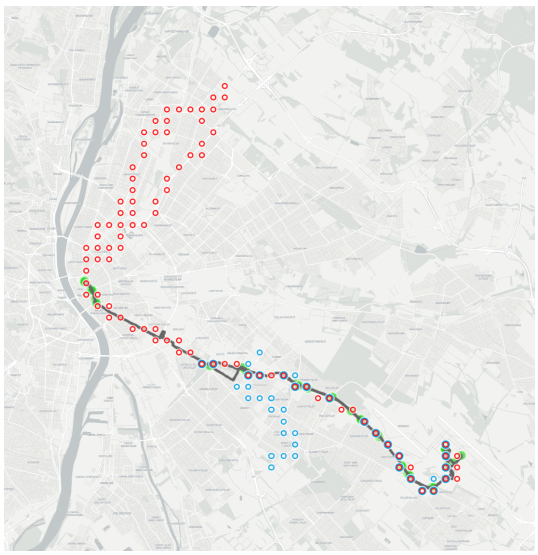

Figure 11. The `vehicles-diff` sublayer using the *Color mode* for routes `100E` and `200E`, on 16 September 2023. The interactive visualization can be accessed here.

This feature is not exclusive for the investigated routes `100E` and `200E`: the same behavior can be discovered on all service dates within the one-year period; moreover, tram, trolleybus, and bus routes were affected as well.

In addition to the visualization-based analysis, the removal of the vehicle positions can also be evaluated by checking the relationship between the positions and their nearest stops. As our hypothesis and the visualization suggest, most of the noises are detected around the terminuses and positions that are not related to the routes' official shapes. To prove these expectations via numbers, we yielded the set of strongly and weakly valid trips after applying our reconstruction method. We calculated the distance of each position to the nearest stop, creating three categories for each threshold value: the set of positions whose nearest stop is the origin of the trip (P_O); the set of positions whose nearest stop is the destination of the trip (P_D); and the set of positions whose nearest stop is any regular stop of the trip (P_R).

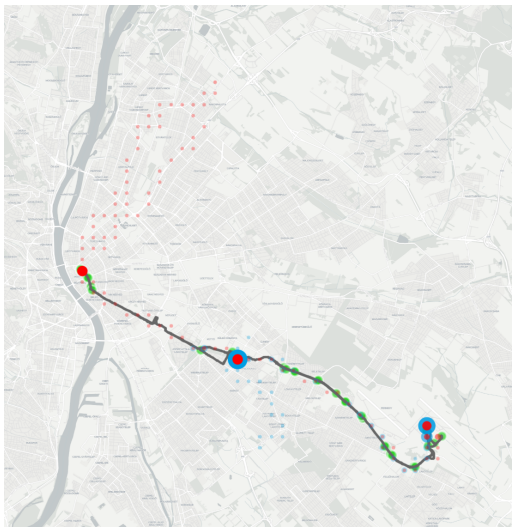

Figure 12. The `vehicles-diff` sublayer using the *Heatmap mode* for routes `100E` and `200E`, on 16 September 2023. The interactive visualization can be accessed here.

The remaining tracked vehicle positions are located around the routes' shapes, as Figure 13 shows, eliminating all the noises.

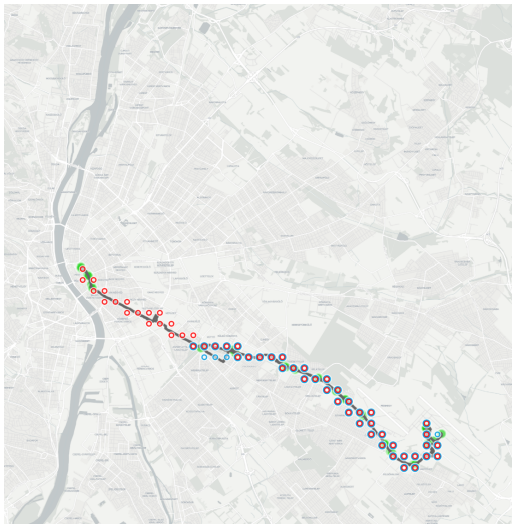

Figure 13. The `vehicles-fixed` sublayer using the *Color mode* for routes `100E` and `200E`, on 16 September 2023. The interactive visualization can be accessed here.

Table 2 shows that 261,304,720 positions were removed from trips that are classified as strongly valid ones; P_O contains more than half of them since 145,232,236 positions are matched with the origin of the routes. An additional 97,753,542 positions belong to P_D, resulting in only 18,318,942 positions belonging to P_R. In other words, only 7.0% of the removed entries belong to any intermediate stops of the routes, and 93.0% of the noises were detected close to the destinations. The set of weakly valid trips has a similar ratio for P_R: 11.2%; however, the set P_D shows a significant amount of 70.3% of the removed vertices. While checking the union of strongly and weakly valid trips, 152,624,926 positions are classified as noise around the origins, 125,817,373 positions around the destinations, and only 22,809,175 positions around the intermediate stops of the trips. Thus, 92.4% of the detected and removed noises appeared around the terminuses.

Table 2. The number of removed vehicle positions by the distance to the nearest stop along the route and the type of the stop. The table contains only valid trips, and the subsets *strongly valid* and *weakly valid* trips are disjoint.

Range (m)	Strongly Valid Trips			Weakly Valid Trips		
	P_O	P_D	P_R	P_O	P_D	P_R
0–5	31,087,151	4,892,666	734,253	1,585,362	1,380,101	184,481
5–15	61,375,162	14,279,612	3,735,756	3,142,669	3,675,030	800,337
15–25	12,416,218	11,746,503	4,685,812	548,811	3,214,766	917,629
25–50	20,092,005	23,631,221	2,425,938	937,774	6,369,867	541,066
50–100	11,244,705	23,393,471	1,305,257	527,450	6,587,017	397,407
100–250	7,441,194	14,792,192	1,198,658	380,941	4,837,538	409,886
250–	1,575,801	5,017,877	4,233,268	269,683	1,999,512	1,239,427
Total	145,232,236	97,753,542	18,318,942	7,392,690	28,063,831	4,490,233

5. Conclusions

In this study, we introduced our method to fetch useful information from the standard formats of GTFS and GTFS-RT. After this initial step, we performed analysis on the dataset to realize possible noises that a dataset can contain for various reasons, such as inaccurate GPS positions and rare updates of vehicles. We introduced an annotation method that can be used to categorize trips based on the characteristic features of their data; moreover, we designed a reconstruction method to eliminate the problematic entries of each tracked trip. The research was carried out by collecting the GTFS planned timetables daily and retrieving the messages of the GTFS-RT feeds published by the BKK Centre for Budapest Transport. The available GTFS and GTFS-RT data covers all the types of public transport services operated in Budapest, such as bus, trolleybus, tram, suburban railway, and metro (subway) services. During the one-year data collection period, we continuously improved our methods to increase the frequency of API queries, retrieving more accurate data based on the live vehicle positions and the static (planned) timetable. Our visualization application, DEPOT, is available online and provides an interactive environment to represent the most important geospatial features of the GTFS files and GTFS-RT feed *Vehicle Positions*. This powerful web application provides a platform for interested audiences to browse our cleaned historical database.

The current dataset and the application DEPOT is only the first step of a large project, but it already proves that the data collecting and cleaning methods result in a historical dataset with an accuracy of over 90%. We have already started to collect GTFS and GTFS-RT data from two additional European cities, and we would like to push our research in a multimodal direction by involving more factors that can impact transportation. For example, weather conditions could be collected and processed to find answers to more complex questions. For example, if the weather conditions for tomorrow are expected to be similar to the one on 2 March 2025, what issues may occur? As the web application is publicly available, other researchers can use it for further investigation and may inspire other research work.

Author Contributions: Conceptualization, Róbert Tóth; methodology, Róbert Tóth; software, Róbert Tóth; validation, Róbert Tóth, Márton Ispány, and Marianna Zichar; formal analysis, Róbert Tóth, Márton Ispány, and Marianna Zichar; data curation, Róbert Tóth; writing—original draft preparation, Róbert Tóth and Marianna Zichar; writing—review and editing, Róbert Tóth, Márton Ispány, and Marianna Zichar; visualization, Róbert Tóth; supervision, Márton Ispány and Marianna Zichar; project administration, Róbert Tóth. All authors have read and agreed to the published version of the manuscript.

Funding: Róbert Tóth was supported by the EKÖP-24-4 University Research Scholarship Program of the Ministry for Culture and Innovation from the source of the National Research, Development and Innovation Fund.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Acknowledgments: We would like to thank BKK Centre for Budapest Transport for publishing and maintaining their OpenData Portal (https://opendata.bkk.hu) and also for making this research possible by providing access and rights to their data.

Conflicts of Interest: The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

API	Application Programming Interface
BKK	The Centre for Budapest Transport
CSV	Comma-separated values
GPS	Global Positioning System
GTFS	General Transit Feed Specification
GTFS-RT	General Transit Feed Specification Realtime
JSON	JavaScript Object Notation
MÁV	MÁV-START Railway Passenger Transport Company
MÁV-HÉV	MÁV-HÉV Suburban Railway Company
OSM	OpenStreetMap
PB	Protocol Buffers

References

1. Gonzalez, M.C.; Hidalgo, C.A.; Barabasi, A.L. Understanding individual human mobility patterns. *Nature* **2008**, *453*, 779–782. [CrossRef]
2. Walker, R.A. A theory of suburbanization: Capitalism and the construction of urban space in the United States. In *Urbanization and Urban Planning in Capitalist Society*; Routledge: Abingdon-on-Thames, UK, 2018; pp. 383–429.
3. Pieretti, G. Suburbanization. In *Encyclopedia of Quality of Life and Well-Being Research*; Springer: Berlin/Heidelberg, Germany, 2024; pp. 7009–7011. [CrossRef]
4. Buchanan, M. The benefits of public transport. *Nat. Phys.* **2019**, *15*, 876. [CrossRef]
5. Gase, L.N.; Kuo, T.; Teutsch, S.; Fielding, J.E. Estimating the costs and benefits of providing free public transit passes to students in Los Angeles County: Lessons learned in applying a health lens to decision-making. *Int. J. Environ. Res. Public Health* **2014**, *11*, 11384–11397. [CrossRef]
6. Zhao, F.; Fashola, O.I.; Olarewaju, T.I.; Onwumere, I. Smart city research: A holistic and state-of-the-art literature review. *Cities* **2021**, *119*, 103406. [CrossRef]
7. Camero, A.; Alba, E. Smart City and information technology: A review. *Cities* **2019**, *93*, 84–94. [CrossRef]
8. Meijer, A.; Bolívar, M.P.R. Governing the smart city: A review of the literature on smart urban governance. *Int. Rev. Adm. Sci.* **2016**, *82*, 392–408. [CrossRef]
9. Besenczi, R.; Bátfai, N.; Jeszenszky, P.; Major, R.; Monori, F.; Ispány, M. Large-scale simulation of traffic flow using Markov model. *PLoS ONE* **2021**, *16*, e0246062. [CrossRef]
10. Trencher, G. Towards the smart city 2.0: Empirical evidence of using smartness as a tool for tackling social challenges. *Technol. Forecast. Soc. Change* **2019**, *142*, 117–128. [CrossRef]
11. Bris, A.; Cabolis, C.; Lanvin, B.; Milner, W.; Madureira, O.; Caballero, J.; Zargari, M.; Sharma, C.; Grimm, F.; Tozer, A. *Smart City Index Report 2024*; Technical Report; IMD World Competitiveness Center: Lausanne, Switzerland, 2024.

12. Zini, A.; Roberto, R.; Corrias, P.; Felici, B.; Noussan, M. Accessibility Measures to Evaluate Public Transport Competitiveness: The Case of Rome and Turin. *Smart Cities* **2024**, *7*, 3334–3354. [CrossRef]
13. McHugh, B. Pioneering open data standards: The GTFS Story. In *Beyond Transparency: Open Data and the Future of Civic Innovation*; Code for America Press: Oakland, CA, USA, 2013; pp. 125–135.
14. Zhou, J.Q.; Jackson, J.; Stravitz, P.; Barlow, G.J.; Koonce, P. Addressing Data Latency in GTFS (General Transit Feed Specification) Realtime to Improve Transit Signal Priority. *Transp. Res. Rec.* **2024**, *2679*, 1329–1341. [CrossRef]
15. Newmark, G.L.; *Assessing GTFS Accuracy [Research Brief]*; Technical Report; San Jose State University. College of Business, Mineta Transportation Institute: San Jose, CA, USA, 2024. [CrossRef]
16. Kunz, N.; Gao, H.O. Global Geolocated Realtime Data of Interfleet Urban Transit Bus Idling. *arXiv* **2024**, arXiv:2403.03489. [CrossRef]
17. Wessel, N.; Allen, J.; Farber, S. Constructing a routable retrospective transit timetable from a real-time vehicle location feed and GTFS. *J. Transp. Geogr.* **2017**, *62*, 92–97. [CrossRef]
18. Bok, J.; Kwon, Y. Comparable Measures of Accessibility to Public Transport Using the General Transit Feed Specification. *Sustainability* **2016**, *8*, 224. [CrossRef]
19. Hadas, Y. Assessing public transport systems connectivity based on Google Transit data. *J. Transp. Geogr.* **2013**, *33*, 105–116. [CrossRef]
20. Braga, C.K.V.; Loureiro, C.F.G.; Pereira, R.H. Evaluating the impact of public transport travel time inaccuracy and variability on socio-spatial inequalities in accessibility. *J. Transp. Geogr.* **2023**, *109*, 103590. [CrossRef]
21. Vuurstaek, J.; Cich, G.; Knapen, L.; Ectors, W.; Yasar, A.U.H.; Bellemans, T.; Janssens, D. GTFS bus stop mapping to the OSM network. *Future Gener. Comput. Syst.* **2020**, *110*, 393–406. [CrossRef]
22. Kocsis, G.; Varga, I. gtfs2net: Extraction of General Transit Feed Specification Data Sets to Abstract Networks and Their Analysis. *Big Data* **2025**, *13*, 30–41. [CrossRef] [PubMed]
23. Vágner, A. Route planning on GTFS using Neo4j. *Ann. Math. Informaticae* **2021**, *54*, 163–179. [CrossRef]
24. Alzaidi, M.; Vágner, A. Application-Based Benchmarking on Redis and MongoDB for Trip Planning using GTFS Data. *TEM J. Technol. Educ. Manag. Inform.* **2023**, *12*, 2583–2592. [CrossRef]
25. Alzaidi, M.; Vágner, A. Trip Timing Algorithm for GTFS data With Redis Model to Improve the Performance. *J. Theor. Appl. Inf. Technol.* **2023**, *11*, 260–268. [CrossRef]
26. Fayyaz S, S.K.; Liu, X.C.; Zhang, G. An efficient General Transit Feed Specification (GTFS) enabled algorithm for dynamic transit accessibility analysis. *PLoS ONE* **2017**, *12*, e0185333. [CrossRef] [PubMed]
27. Chaves-Fraga, D.; Priyatna, F.; Cimmino Arriaga, A.; Toledo, J.; Ruckhaus, E.; Corcho, O. GTFS-Madrid-Bench: A Benchmark for Virtual Knowledge Graph Access in the Transport Domain. *J. Web Semant.* **2020**, *65*, 100596. [CrossRef]
28. Devunuri, S.; Lehe, L. A Survey of Errors in GTFS Static Feeds from the United States. *Findings* **2024.**
29. Kunama, N.; Worapan, M.; Phithakkitnukoon, S.; Demissie, M. GTFS-Viz: Tool for preprocessing and visualizing GTFS data. In Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 ACM International Symposium on Wearable Computers (UbiComp '17), Maui, HI, USA, 11–15 September 2017; pp. 388–396. [CrossRef]
30. Prommaharaj, P.; Phithakkitnukoon, S.; Demissie, M.G.; Kattan, L.; Ratti, C. Visualizing public transit system operation with GTFS data: A case study of Calgary, Canada. *Heliyon* **2020**, *6*, e03729. [CrossRef]
31. Para, S.; Wirotsasithon, T.; Jundee, T.; Demissie, M.G.; Sekimoto, Y.; Biljecki, F.; Phithakkitnukoon, S. G2Viz: An online tool for visualizing and analyzing a public transit system from GTFS data. *Public Transp.* **2024**, *16*, 893–928. [CrossRef]
32. Kocsis, G.; Varga, I. Extracting Mass Transportation Networks from General Transit Feed Specification Datasets. In Proceedings of the 7th International Conference on Complexity, Future Information Systems and Risk (COMPLEXIS 2022), Online, 22–23 April 2022; pp. 85–91. [CrossRef]
33. Wessel, N.; Widener, M.J. Discovering the space–time dimensions of schedule padding and delay from GTFS and real-time transit data. *J. Geogr. Syst.* **2017**, *19*, 93–107. [CrossRef]
34. Aemmer, Z.; Ranjbari, A.; MacKenzie, D. Measurement and classification of transit delays using GTFS-RT data. *Public Transp.* **2022**, *14*, 263–285. [CrossRef]
35. Xian, T.; Moylan, E.; Nelson, J. *Evidence from GTFS-R That Bus Priority Lanes Reduce Marginal Delay*; Working Paper; School of Civil Engineering, The University of Sydney: Camperdown, Australia, 2022.
36. Xian, T.; Chin, T.K.; Marks, B.; Nelson, J.; Moylan, E. Bus arrival and departure time updates in the Greater Sydney Area. *Sci. Data* **2024**, *11*, 1034. [CrossRef]
37. Devunuri, S.; Qiam, S.; Lehe, L.J. ChatGPT for GTFS: Benchmarking LLMs on GTFS semantics... and retrieval. *Public Transp.* **2024**, *16*, 333–357. [CrossRef]

38. BKK. Travel Options. Available online: https://bkk.hu/en/travel-information/travel-options-in-budapest/ (accessed on 21 November 2024).
39. Khademi-Vidra, A.; Nemecz, G.; Bakos, I.M. Satisfaction measurement in the sustainable public transport of Budapest. *Transp. Res. Interdiscip. Perspect.* **2024**, *23*, 100989. [CrossRef]