

Bus Navigation System: A GPS and Real-Time Transit Route Guidance Framework

Ketki Lokhande ¹, Sanika Gadbaile ², Avantika Rashatwar ³, Sayali Narkhede ⁴

¹Department of Computer Science and Engineering, [Jagadambha College Of Engineering & Technology, Yavatmal]

²Department of Computer Science and Engineering, [Jagadambha College Of Engineering & Technology, Yavatmal]

³Department of Computer Science and Engineering, [Jagadambha College Of Engineering & Technology, Yavatmal]

Abstract - Urban bus passengers often face difficulty in identifying suitable routes, locating the correct stop, estimating arrival time, and reacting to delays or service changes. This paper proposes a bus navigation system that combines route and stop data, geographic positioning, shortest-path routing, and real-time transit information to provide passenger-oriented guidance. The proposed framework accepts a source and destination, identifies candidate bus services, calculates feasible routes, tracks vehicle position when real-time data are available, and presents estimated arrival information and service alerts. A standardized transit data model is used to represent agencies, routes, trips, stops, and stop times, while a real-time layer can represent vehicle positions, trip updates, and alerts. The system is designed as a modular architecture so that it can operate with static timetable data when live feeds are unavailable and progressively add real-time capabilities as data quality improves. The paper describes the system architecture, route-selection method, ETA estimation approach, implementation considerations, and an evaluation methodology. The proposed system is intended to reduce passenger uncertainty, simplify public-bus navigation, and provide a practical foundation for smart-transit applications.

Key Words: *Bus navigation, public transportation, GPS, route planning, real-time transit, ETA.*

1. INTRODUCTION

Public bus transportation is an important component of urban mobility, but passengers may still experience uncertainty when routes are complex, timetables are difficult to interpret, or vehicles operate with variable travel times. A passenger travelling to an unfamiliar location may need to determine which bus to board, where to board it, where to transfer, and when to get off. These difficulties become more significant when a service is delayed or diverted.

A bus navigation system can address these problems by integrating transport-network information with geographic location and route-planning algorithms. The system proposed in this paper treats the transit network as a graph in which stops and relevant locations are represented as nodes and travel connections are represented as edges. Static service information describes routes, trips, stops, and scheduled stop times, while real-time information can provide vehicle positions, trip updates, and service alerts. The General Transit Feed Specification (GTFS) provides a standardized structure for these types of transit information [1], [2].

The objective of this work is to design a practical and extensible framework for passenger bus navigation. Rather than depending entirely on a proprietary mapping service, the architecture separates data ingestion, route computation, real-time processing, and user guidance. This makes it suitable for academic prototypes and for future deployment by a transport operator that can provide structured transit data.

2. RELATED WORK

Transit information systems have evolved from static timetable displays to systems that combine scheduled and real-time information. GTFS is an open standard for distributing transit information such as agencies, routes, trips, stops, schedules, and related service information. GTFS Realtime extends this model with information such as trip updates, vehicle positions, and service alerts [1], [2]. These standards make it possible for different applications to consume transit information using a consistent data structure.

Research on bus arrival prediction has shown that accurate passenger information depends on current vehicle location, network structure, and the variability of travel time. One study developed a real-time bus arrival prediction model using the current location of the bus, historical information, and a bus-network data model, and reported that the network

representation and timeperiod classification influence prediction accuracy [3]. More recent research has investigated nonlinear spatio-temporal models for public-transit arrival prediction, reflecting the effects of dwell time, traffic signals, demand fluctuations, and other operating conditions [4].

The proposed system uses these findings as design guidance but focuses on the broader passenger-navigation problem rather than on a single prediction model. The architecture therefore permits a simple schedule-based ETA method initially and allows a more advanced predictive model to be introduced later.

3. PROPOSED SYSTEM

The proposed Bus Navigation System consists of five principal layers: user interaction, transit-data management, location and real-time processing, route computation, and passenger guidance. The user enters a source and destination, and the system identifies nearby stops and candidate services. The route engine evaluates feasible journeys and selects a suitable route according to travel time, transfers, walking distance, and service availability.

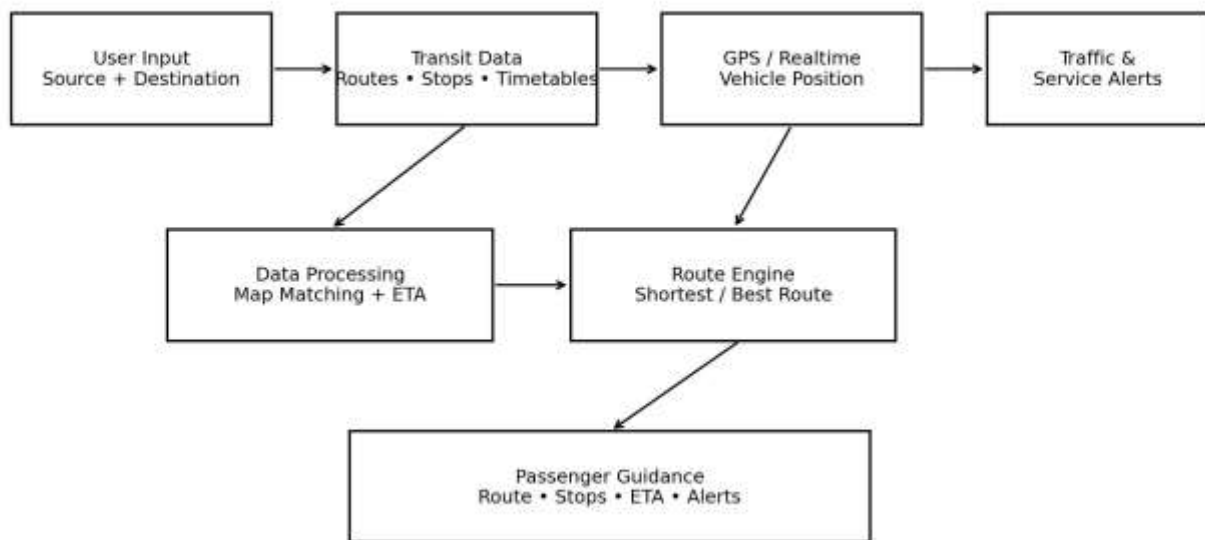


Fig -1: High-level architecture of the proposed bus navigation system

3.1 Transit Data Layer

The static data layer stores transport-agency information, routes, trips, stops, stop times, calendars, and optional transfer or accessibility information. A GTFS Schedule feed provides a practical representation because its core files describe the agency, routes, trips, stops, stop times, and service dates [1]. The system can import these data into a relational database or a document-oriented store depending on implementation requirements.

3.2 Location and Real-Time Layer

The location layer receives the user's location from a device positioning service and, where available, the position of buses from a vehicle tracking source. GTFS Realtime defines vehicle-position information as a way to communicate the location of a vehicle, commonly obtained from an onboard GPS device [2]. Trip updates and service alerts can be used to modify expected arrival information or warn passengers about cancellations, detours, and stop changes.

3.3 Route Computation

The transit network is modeled as a weighted graph. A node represents a bus stop or another relevant transfer point. An edge represents a ride segment, walking connection, or transfer. Each edge can be assigned a dynamic cost that combines travel time, waiting time, transfer penalties, and walking distance. A shortest-path algorithm can then identify a candidate route. For a basic prototype, Dijkstra's algorithm is appropriate when edge costs are non-negative; an A* strategy can be considered when geographic heuristics are available.

A generalized route cost can be represented as: $C = \alpha T + \beta W + \gamma R + \delta D$, where T is estimated in-vehicle travel time, W is waiting time, R is transfer penalty, D is walking distance, and α , β , γ , and δ are configurable weights. The weights allow the system to prioritize faster journeys, fewer transfers, or shorter walking distances depending on passenger preference.

3.4 ETA Estimation

When only timetable data are available, the estimated arrival time can be obtained from scheduled stop times and the passenger's selected trip. When live vehicle positions are available, the system can estimate the remaining travel time using the current position, route segment, historical travel-time statistics, and current service conditions. A simple implementation may use a weighted average of scheduled and observed travel time, while an advanced implementation may replace this component with a statistical or machine-learning predictor.

3.5 Passenger Guidance

The passenger interface should present only the information required to complete the journey: selected bus route, boarding stop, walking direction to the stop, expected arrival, intermediate stops, transfer instructions, destination stop, and alerts. The interface should also clearly distinguish scheduled times from real-time estimates so that users understand the confidence level of the information.

4. SYSTEM METHODOLOGY

The system workflow begins when the passenger enters a destination. First, the application obtains the current location or accepts a manually selected origin. Second, nearby stops are identified using geographic distance. Third, the system searches the transit database for routes serving those stops. Fourth, feasible trips are generated using service calendars and stop-time information. Fifth, the route engine assigns costs to candidate journeys and ranks them. Finally, the selected journey is displayed and updated when new vehicle or service information becomes available.

- Input: current location, destination, travel date/time, and optional preferences.
- Stop discovery: identify boarding and destination stops within a configurable walking radius.
- Trip generation: retrieve routes and trips serving the selected stops.
- Route ranking: calculate journey cost using time, waiting, transfers, and walking.
- Real-time update: replace or adjust scheduled estimates when reliable live information arrives.
- Guidance: display the route as sequential instructions and notify the user of relevant changes.

5. IMPLEMENTATION DESIGN

A prototype can be implemented using a web or mobile front end connected to a backend service. The backend is responsible for data ingestion, route search, ETA calculation, and real-time updates. A relational database such as PostgreSQL can store structured transit data, while a geospatial extension can support proximity queries. A REST API can expose endpoints for stops, routes, journeys, vehicle positions, and alerts.

Module	Input	Output
User module	Origin, destination, preferences	Journey request
Transit database	Routes, stops, trips, stop times	Candidate services
Location module	GPS position	Nearby stops / vehicle location
Route engine	Candidate services + costs	Ranked journeys
Realtime module	Vehicle positions, updates, alerts	Updated ETA / warnings
Guidance module	Selected journey	Step-by-step passenger instructions

Table -1: Functional modules of the proposed system

6. ALGORITHM

The following high-level procedure can be used for journey selection. The system first determines candidate origin and destination stops, then constructs a time-dependent transit graph. Each edge is assigned an estimated cost based on waiting, riding, transfer, and walking components. The route engine returns the lowest-cost feasible journey. If real-time information changes the expected cost beyond a predefined threshold, the journey is recalculated.

- Step 1: Read origin, destination, requested departure/arrival time, and passenger preferences.
- Step 2: Find nearby origin and destination stops using geographic distance.
- Step 3: Retrieve active routes and trips serving the candidate stops.
- Step 4: Build a time-dependent graph using scheduled and, when available, live travel information.
- Step 5: Apply the selected route-search algorithm and rank feasible journeys.
- Step 6: Display the best journey and monitor real-time changes.
- Step 7: Recompute the journey when a significant delay, cancellation, or detour affects the selected service.

7. EVALUATION METHODOLOGY

Because the proposed work is a system framework, evaluation should be performed on an actual transit dataset before reporting numerical performance claims. A suitable experimental study would compare the proposed route-selection and ETA modules against timetable-only navigation. The evaluation should use several routes, different times of day, and representative traffic or delay conditions.

Key metrics should include route-planning response time, ETA error, successful journey completion, number of unnecessary transfers, walking distance, and user-perceived usefulness. For ETA prediction, Mean Absolute Error (MAE) can be calculated as $MAE = (1/n) \sum |y_i - \hat{y}_i|$, where y_i is the observed arrival time and $\hat{y}_i$ is the predicted arrival time. For route selection, average travel time and transfer count can be compared across candidate methods.

The evaluation should also test degraded-data conditions. If live vehicle information is unavailable, the application should fall back to scheduled information and clearly label the result as schedule-based. If a route is cancelled or a stop is temporarily unavailable, the system should remove or penalize affected journey options and search for alternatives.

8. EXPECTED BENEFITS AND LIMITATIONS

The proposed system can provide a single passenger-facing workflow for discovering routes, locating stops, tracking buses, and receiving service information. Its modular design can support both small-scale college or city prototypes and larger deployments. Standardized transit data also improve interoperability with external data providers and applications.

The principal limitation is dependence on data quality. Incorrect stop coordinates, outdated schedules, missing vehicle positions, or delayed service feeds can reduce navigation accuracy. GPS measurements can also be noisy, and traffic conditions may change faster than the system can update its estimates. Therefore, the application should communicate uncertainty rather than present every ETA as exact.

9. FUTURE SCOPE

Future development can incorporate machine-learning-based ETA prediction, congestion-aware routing, multilingual voice guidance, accessibility-aware routing, crowding information, and personalized route preferences. Additional research can investigate uncertainty-aware ETA displays, multimodal journeys that combine buses with walking or other public transport, and privacy-preserving location processing. A field study with a real transport operator would provide the evidence required to quantify improvements in travel time, ETA accuracy, and passenger satisfaction.

10. CONCLUSIONS

This paper presented a proposed bus navigation system that integrates static transit information, geographic positioning, route computation, and real-time transit updates. The architecture separates data management, route planning, ETA estimation, and passenger guidance so that the system can begin with timetable-based information and progressively

adopt real-time capabilities. Standardized transit representations such as GTFS and GTFS Realtime provide a practical foundation for interoperable transit applications. The proposed evaluation methodology emphasizes measurable route quality, ETA accuracy, response time, and robustness under incomplete data. A future implementation and field evaluation can validate the framework using real busnetwork and vehicle-tracking data.

ACKNOWLEDGEMENT

The authors acknowledge the open transit-data community and the maintainers of the General Transit Feed Specification documentation, which provides a standardized foundation for representing public transportation information.

REFERENCES

- [1] General Transit Feed Specification, “GTFS Schedule Reference,” MobilityData, revised Apr. 27, 2026. Available: <https://gtfs.org/documentation/schedule/reference/>
- [2] General Transit Feed Specification, “GTFS Realtime Reference,” MobilityData. Available: <https://gtfs.org/documentation/realtime/reference/>
- [3] M. D. et al., “Bus arrival time prediction based on network model,” *Procedia Computer Science*, vol. 113, pp. 138–145, 2017, doi: 10.1016/j.procs.2017.08.331.
- [4] A. Achar, A. Natarajan, R. Regikumar, and B. Anil Kumar, “Predicting public transit arrival: A nonlinear approach,” *Transportation Research Part C: Emerging Technologies*, vol. 144, art. 103875, 2022, doi: 10.1016/j.trc.2022.103875.
- [5] E. W. Dijkstra, “A note on two problems in connexion with graphs,” *Numerische Mathematik*, vol. 1, pp. 269–271, 1959, doi: 10.1007/BF01386390.

BIOGRAPHIES (Optional not mandatory)

Ketki is a student researcher interested in software development, intelligent transportation systems, and practical applications of web and location-based technologies.