A Survey on Trajectory-Based Carpool Ride Matching System using Map APIs

Dr. Kethineni Vinod Kumar[1], D. Madhusudhan[2], C. Tejavati[3], CH. Ashok Kumar[4], E. Shiva Prajval[5], J. Bindu[6]

Associate Professor[1], Under graduate[2,3,4,5,6], Department of Computer Science and Engineering
Sanskrithi School of Engineering, Puttaparthi, Andhra Pradesh, India.
kethineni.vinod@gmail.com, madhusudand7017@gmail.com, tejareddychalimamidi@gmail.com, ashokkumarchigurusetti@gmail.com, shivaprajval10@gmail.com, bindujaripiti@gmail.com

Abstract

Urban mobility systems are increasingly burdened by traffic congestion, excessive fuel consumption, and environmental degradation due to inefficient vehicle usage. Conventional ride-sharing platforms rely primarily on proximity-based matching, which often results in suboptimal ride allocation and increased travel time.

This paper proposes a ***Trajectory-Based Carpool Ride Matching System*** *that leverages real-time map services and route alignment techniques to enhance ride- sharing efficiency. The system matches passengers with drivers based on route similarity rather than geographic proximity, thereby minimizing detours and improving travel efficiency.*

A multi-criteria Quality of Experience (QoE) model is incorporated to evaluate ride options based on factors such as cost, safety, travel time, and environmental impact. Additionally, a modified route optimization approach inspired by the Traveling Salesman Problem (TSP) is applied to optimize pickup and drop-off sequences.

The system is implemented using a full-stack architecture with modern web technologies and integrates real-time communication mechanisms.

Keywords Carpooling, Trajectory Matching, Ride Sharing, Map APIs, QoE, Route Optimization, Smart Transportation.

I. Introduction

Rapid urban expansion has significantly increased the demand for transportation, leading to congestion, longer travel durations, and environmental concerns. A major inefficiency in current transportation systems is the underutilization of vehicles, as many operate with a single occupant.

Ride-sharing systems have been introduced to address these issues by enabling multiple users to share rides. However, traditional systems predominantly depend on proximity- based matching, which connects users based on geographic closeness rather than route compatibility. This approach often results in unnecessary detours and inefficient ride assignments.

To overcome these limitations, this research proposes a **trajectory-based ride matching approach**, where passenger requests are aligned with driver routes using geospatial data. By considering route similarity, the system ensures efficient ride allocation and minimizes travel deviations.

The proposed system also integrates user- centric parameters through a QoE model, enabling better decision-making and personalized recommendations. This research aims to provide an efficient, scalable, and environmentally sustainable solution for modern transportation challenges.

II. Literature Review

Several researchers have explored different approaches to improve ride-sharing systems. Early systems focused on **proximity-based matching**, where passengers are matched with nearby drivers. While these systems reduce waiting time, they often lead to inefficient routing and increased travel distance.

Some studies introduced **trajectory-based approaches**, where matching is performed based on route similarity. These systems improve efficiency but often lack user preference consideration.

Other research focused on **multi-parameter recommendation systems**, which evaluate rides based on factors such as cost, time, and user ratings. However, these models are not always integrated with real-time route analysis.

Optimization techniques such as shortest path algorithms and heuristic methods have also been used to improve ride allocation. However, many systems struggle to handle real-time dynamic data effectively.

The proposed system addresses these limitations by integrating trajectory-based matching, QoE evaluation, and real-time optimization into a unified framework.

The integration of **Map APIs** has further improved ride-sharing systems by providing real-time data such as distance, travel time, and traffic conditions. These APIs enable accurate route calculations and enhance navigation capabilities. However, in many existing systems, Map APIs are used only for visualization and navigation, rather than being fully utilized for intelligent ride matching and decision-making.

Another important aspect considered in recent research is **security and privacy**. Ride-sharing systems involve sensitive user data, including personal information and location details. Some studies have proposed secure authentication mechanisms and encrypted communication to protect user data. While these approaches improve security, they may introduce additional complexity and affect system performance.

Furthermore, recent advancements in intelligent systems have introduced the concept of **AI-based ride recommendation and prediction models**. These models analyze historical data, user behavior, and travel patterns to provide smarter ride suggestions. Although promising, these approaches are still evolving and require significant computational resources.

III. System Architecture

3.1 Overview of the System Architecture

The proposed **Trajectory-Based Carpool Ride Matching System** is designed using a modular client–server architecture that ensures efficient communication between system components. The architecture integrates frontend, backend, database, and real-time communication modules to provide a scalable and responsive platform.

Figure 1 – System Architecture of Trajectory-Based Carpool Ride Matching System

The frontend allows users to interact with the system, while the backend processes requests and communicates with the database. Real- time updates are handled using Server-Sent Events (SSE), ensuring smooth interaction between users.

3.2 Web Application (Client Layer)

The web application serves as the user interface for both passengers and drivers. It is developed using modern frontend technologies and provides a responsive and user-friendly experience.

3.3 Backend API (Application Server Layer)

The backend API is responsible for handling all application logic and processing user requests. It acts as an intermediary between the frontend and the database.

The backend includes several modules:

- Authentication and Access Control: Implements secure login using token- based authentication and role-based access control
- Ride Management Module: Handles creation, updating, and retrieval of ride information
- Booking and Safety Module: Manages booking operations, seat allocation, and safety-related data
- Chat and Share Module: Enables communication between passengers and drivers and supports trip-sharing links

The backend ensures efficient data processing and maintains system reliability.

3.4 Database Layer (MongoDB)

The database layer stores all system-related data in a structured format. It is designed to handle large volumes of data efficiently.

The database stores:

- User profiles (passengers and drivers)
- Ride details (source, destination, time, price)
- Booking records
- Chat messages and ratings

The backend communicates with the database to retrieve and update information as required. This ensures data consistency and smooth system operation.

3.5 System Workflow

The system workflow describes the sequence of operations involved in the ride-sharing process. Initially, the user opens the web application and logs in or registers. After authentication, the user selects a role as either a driver or a passenger.

If the user is a driver, they create a ride by entering details such as source, destination, date, and price. If the user is a passenger, they search for available rides and view ride details.

Once a suitable ride is found, the passenger proceeds with booking, and the system updates seat availability. The driver can view the list of booked passengers. Communication between the driver and passenger is enabled through chat or a shared link.

After the trip, the ride is marked as completed, and both users provide ratings and feedback. This workflow ensures efficient coordination and smooth operation of the system.

This workflow ensures smooth coordination between drivers and passengers and enables efficient ride-sharing operations.

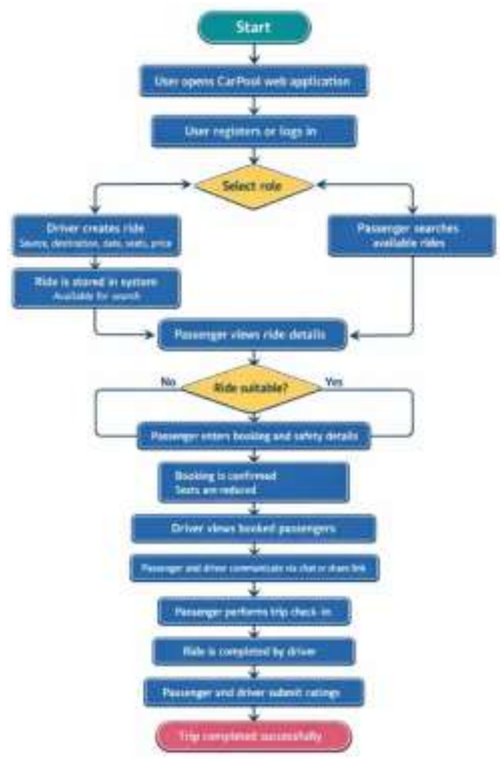


Figure 2 – System Workflow Diagram

IV. System Implementation

The implementation phase transforms the system design into a functional web application. The proposed system is developed as a full-stack application that enables users to search, create, and manage rides efficiently. It integrates frontend, backend, and database components to provide a seamless user experience.

4.1 Home Page Interface

The system is developed using a modern web development environment that supports scalable application design. The frontend is built using responsive technologies, while the backend handles request processing and business logic. A NoSQL database is used to store system data efficiently.

4.2 User Dashboard Implementation

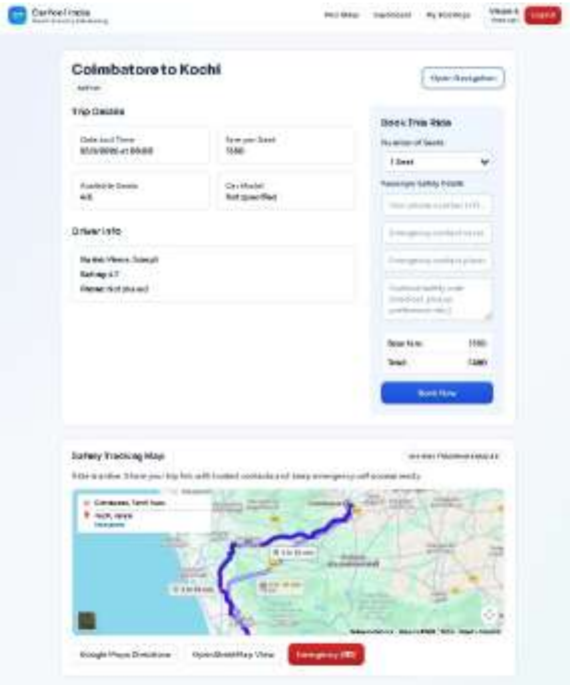
The user dashboard provides a personalized interface for both passengers and drivers. It allows users to access key functionalities such as ride search, booking management, and ride creation in a structured manner.

4.3 Ride Search and Selection

The ride search feature enables passengers to find rides based on source, destination, and time. The system displays available rides along with relevant details, allowing users to select suitable options.

4.4 Booking and Confirmation

The booking module allows passengers to reserve seats in selected rides. Once the booking is confirmed, the system updates seat availability and stores booking details in the database.



4.5 Driver Dashboard Implementation

The driver dashboard allows drivers to create and manage ride listings. Drivers can view booking requests, update ride details, and monitor passenger information.

4.6 Ride Management

The ride management feature enables drivers to edit or cancel rides. It provides a structured interface for managing all ride-related activities efficiently.

V. Results and Discussion

The implementation of the Trajectory-Based Carpool Ride Matching System demonstrates improved efficiency compared to traditional ride-sharing approaches. The system successfully enables users to search and book rides through a digital platform, reducing manual effort and waiting time.

The trajectory-based matching mechanism ensures that passengers are aligned with drivers whose routes closely match their travel paths. This reduces unnecessary detours and improves overall travel efficiency. As a result, both drivers and passengers experience reduced travel time and better route optimization.

The booking process is handled efficiently, with real-time updates ensuring accurate seat availability and confirmation. The integration of dashboards allows users to manage rides and bookings effectively. Drivers can easily monitor passenger requests, while passengers can track their bookings and ride status.

The system also enhances communication between users through chat and sharing features, improving coordination and reducing confusion during the ride. Additionally, the rating system allows users to provide feedback, which helps maintain service quality.

From a performance perspective, the system is capable of handling multiple user requests simultaneously without significant delays. The use of a structured backend and database ensures smooth data processing and reliable operation.

Overall, the proposed system improves ride-sharing efficiency, enhances user experience, and promotes better utilization of transportation resources. The results indicate that trajectory-based matching is more effective than traditional proximity-based methods.

VI. Conclusion

The Trajectory-Based Carpool Ride Matching System provides an effective and modern solution to the challenges faced in traditional ride-sharing systems. Conventional approaches that rely on proximity-based matching often lead to inefficient routing, increased travel time, and poor vehicle utilization. The proposed system overcomes these limitations by introducing a trajectory-based matching mechanism that aligns passenger requests with driver routes.

By integrating Map APIs, the system is capable of analyzing real-time route data and identifying optimal ride matches with minimal detours. This significantly improves travel efficiency and reduces unnecessary fuel consumption. The inclusion of a Quality of Experience (QoE)-based evaluation further enhances the system by considering multiple factors such as cost, safety, time flexibility, and user preferences while recommending rides.

The implementation of the system as a web- based application ensures accessibility and ease of use for both drivers and passengers. Features such as ride creation, ride search, booking management, and real-time updates provide a smooth and user-friendly experience. The use of dashboards enables users to manage their activities efficiently, while communication features improve coordination between participants.

The system also demonstrates strong performance in handling multiple user interactions simultaneously. The integration of frontend, backend, and database components ensures reliable data processing and consistent system operation. Additionally, the modular architecture allows for easy scalability and future enhancements.

Overall, the proposed system successfully improves ride-sharing efficiency, enhances user satisfaction, and promotes better utilization of transportation resources. It also contributes toward reducing traffic congestion and environmental impact by encouraging shared mobility.

VII. References

[1] S. Ma, Y. Zheng, and O. Wolfson, "T-Share: A large-scale dynamic taxi ridesharing service," IEEE Transactions on Knowledge and Data Engineering, vol. 26, no. 12, pp. 1–14, 2014.

[2] J. Yuan, Y. Zheng, X. Xie, and G. Sun, "Driving with knowledge from the physical world," in Proc. ACM SIGKDD, 2012, pp. 316–324.

[3] M. Furuhata, M. Dessouky, F. Ordóñez, M. Brunet, X. Wang, and S. Koenig, "Ridesharing: The state-of-the-art and future directions," Transportation Research Part B, vol. 57, pp. 28–46, 2013.

[4] N. Agatz, A. Erera, M. Savelsbergh, and X. Wang, "Optimization for dynamic ride-sharing: A review," European Journal of

Operational Research, vol. 223, no. 2, pp. 295–303, 2012.

[5] D. S. Johnson and L. A. McGeoch, "The traveling salesman problem: A case study in local optimization," Local Search in Combinatorial Optimization, pp. 215–310, 1997.

[6] Google, "Google Maps Platform Documentation," [Online]. Available: https://developers.google.com/maps

[7] MongoDB Inc., "MongoDB Documentation," [Online]. Available: https://www.mongodb.com/docs

[8] Node.js Foundation, "Node.js Documentation," [Online]. Available: https://nodejs.org

[9] React Developers, "React.js Documentation," [Online]. Available: https://react.dev

[10] M. Hasan and M. A. Hashem, "Online ride sharing system," International Journal of Computer Applications, vol. 182, no. 10, pp. 1–6, 2018.

[11] A. S. Patil and P. R. Jadhav, "Smart transportation system using web technologies," International Journal of Advanced Research in Computer Science, vol. 9, no. 2, pp. 45–50, 2019.

[12] R. Bagre and P. Kulkarni, "Smart transportation system using GPS and web services," International Journal of Engineering Research, vol. 8, no. 6, pp. 78–82, 2019.

[13] S. Tanpure, S. Patil, and P. Jadhav, "Automated ride sharing system using web

.