IJARSCT
ISSN: 2581-9429
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Volume 5, Issue 11, April 2025
Impact Factor: 7.67

Route Optimization Solution for Courier Services

Dr. Amol Pande[1], Atul Mahajan[2], Vitthal kudav[3], Rushikesh Manje[4], Sangaram Lad[5]
Professor & HOD, Department of Computer Engineering[1]
Student, Department of Computer Engineering[2 3 4 5]
Datta Meghe College of Engineering, Navi Mumbai, India

Abstract: *Route optimization is a key component in modern-day logistics, delivery services, and transportation planning, where the primary objective is to identify the most time-efficient and resource-effective paths between multiple destinations. The conventional approach to route optimization often relies on pre-defined or static road network data, which fails to account for the constantly changing nature of urban traffic conditions. The present work proposes a dynamic and data-driven approach for route optimization by integrating graph theory with real-time traffic data. A specific geographical area is selected and modelled as a weighted graph, where intersections or coordinates act as nodes and the roads connecting them are treated as edges. Initially, edge weights are assigned based on estimated travel times or distances. However, to enhance accuracy, these static weights are dynamically updated using live traffic data obtained from the Google Maps API. These updates reflect real-world variations such as congestion, road closures, and delays, which significantly impact the travel time along different paths. Dijkstra's algorithm, a well-established shortest path algorithm, is employed to compute the minimum time routes between any two nodes in the graph. It is particularly effective due to its ability to find optimal paths in graphs with non-negative edge weights and its relatively low computational complexity for sparse graphs. The algorithm is adapted to work with the dynamically updated weights, ensuring that every routing decision is made based on the most recent and accurate traffic conditions. The approach is further extended to support routing for multiple destinations in a single trip, commonly encountered in delivery and courier services. By iteratively applying Dijkstra's algorithm and updating travel times in real-time, the system can generate optimized multi-stop routes that minimize total delivery time and operational costs.*

Keywords: Route optimization, last-mile delivery, adaptive routing, dynamic adjustments, real-time traffic data, fuel efficiency, cost reduction, optimization algorithms, travel time minimization, OR-Tools, graph-based routing, logistics efficiency, scalability, operational savings, intelligent transportation systems, delivery prioritization, sustainable logistics, data-driven decision making, smart mobility solutions

I. INTRODUCTION

In today's world of rapidly evolving logistics and transportation systems, efficient route planning has become a critical component in ensuring timely deliveries, cost savings, and optimal resource utilization. Traditional routing techniques rely heavily on static maps and predefined conditions that do not adapt well to dynamic, real-time changes such as traffic congestion, road closures, or construction work. This lack of adaptability can lead to increased delivery times, fuel consumption, and operational inefficiencies.

Dynamic Route Optimization addresses these challenges by incorporating real-time data into the route planning process. By leveraging advanced algorithms like Dijkstra's algorithm in combination with live traffic data sourced through the Google Maps API, our system intelligently adjusts routes based on current road conditions. This enables the calculation of the most efficient paths between multiple destinations, optimizing the overall delivery or travel experience.

Our approach involves modeling a particular area as a weighted graph where nodes represent locations and edges denote roads. Edge weights reflect travel times and are dynamically updated based on real-time traffic data. Using

Dijkstra's algorithm, the shortest path between various waypoints is computed, ensuring that the selected routes reflect current traffic scenarios. Furthermore, the system supports multi-stop routing, providing optimized paths for delivery services covering multiple destinations.

The proposed solution provides a scalable and adaptable framework that can be applied across diverse sectors such as logistics, ridesharing, public transportation, and emergency services. It also enables integration with historical data to further enhance prediction accuracy and routing efficiency.

II. LITERATURE SURVEY

Existing Research:

Predictive Analytics for Last-Mile Delivery Optimization

A study published in the ***European Journal of Logistics, Purchasing and Supply Chain Management*** examines the use of predictive analytics in last-mile delivery optimization. It highlights how real-time traffic data, customer preferences, and delivery constraints can be integrated into dynamic routing models to improve efficiency. The research discusses traditional Vehicle Routing Problem (VRP) models and their extensions, such as VRP with Time Windows (VRPTW) and Dynamic VRP (DVRP), and introduces machine learning techniques for better route prediction.

AI-Driven Optimization in Last-Mile Logistics

Research from ***MIT*** explores how artificial intelligence and machine learning technologies enhance last-mile delivery operations. The study demonstrates that AI-driven route optimization significantly reduces delivery times and fuel consumption. It also discusses neural network architectures optimized for urban delivery scenarios, showing improvements in estimated delivery windows and fleet management.

Hybrid Delivery Networks and Sustainability

A review published in ***MDPI*** focuses on hybrid delivery networks integrating drones, ground robots, and conventional vehicles. The study highlights how AI and IoT improve predictive analytics, dynamic routing, and fleet management. It also discusses sustainable logistics approaches, such as electric vehicle fleets and shared delivery infrastructures, which help minimize environmental impact.

III. PROPOSED SYSTEM

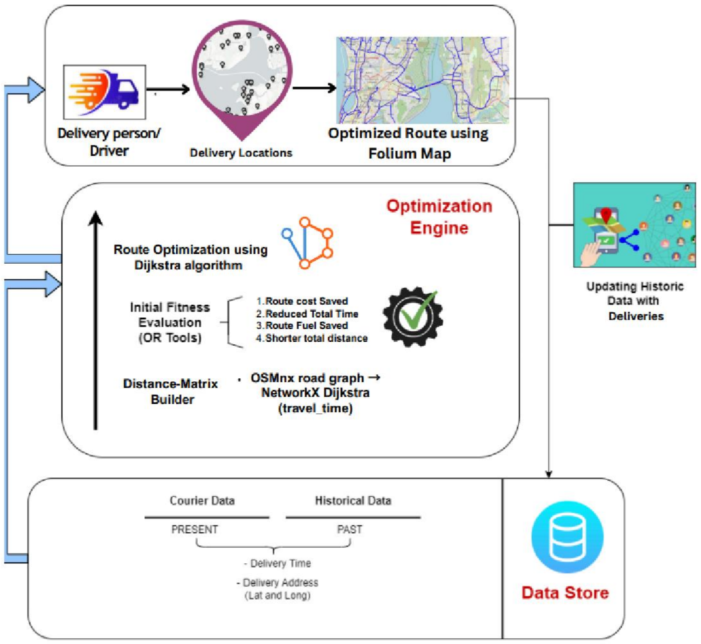
Delivery person/ Driver
Delivery Locations
Optimized Route using Folium Map
Optimization Engine
Route Optimization using Dijkstra algorithm
Initial Fitness Evaluation (OR Tools)
1.Route cost Saved
2.Reduced Total Time
3.Route Fuel Saved
4.Shorter total distance
Distance-Matrix Builder
OSMnx road graph → NetworkX Dijkstra (travel_time)
Updating Historic Data with Deliveries
Courier Data
PRESENT
Historical Data
PAST
- Delivery Time
- Delivery Address (Lat and Long)
Data Store

Fig. 1. System Overview

The proposed system is designed to optimize last-mile delivery routing using graph-based mapping, heuristic optimization, and interactive visualizations. It reduces travel distance, time, fuel consumption, and costs while dynamically adjusting to real-world constraints like traffic conditions and road accessibility.
This system is only a model, developed as a proof of concept for delivery route optimization. It is not a final user product but serves as a framework for testing and validating efficiency improvements in routing strategies.

System Architecture
1. Data Collection and Preprocessing
Location data, including latitude, longitude, and addresses, is extracted from a regional dataset covering Mumbai, Navi Mumbai, and Thane.
Delivery points are filtered and structured for routing analysis. Coordinates are mapped to nearest road network nodes to ensure real-world alignment.

2. Road Network Extraction and Distance Matrix Calculation
Using OSMnx, the system fetches the road network graph and computes travel speeds and times for each road segment.
A distance matrix is created using Dijkstra's shortest path algorithm, storing time-based travel costs between nodes.

3. Route Optimization Using OR-Tools
A Vehicle Routing Problem (VRP) model is constructed to determine the most efficient delivery sequence.
Optimization is performed using the PATH CHEAPEST ARC method, which selects the most cost-effective travel path iteratively.
The system dynamically adjusts routes to minimize delivery time and fuel consumption.

4. Animated Route Visualization and Performance Metrics
Folium maps visualize optimized routes interactively, allowing users to inspect delivery locations and paths

IV. IMPLEMENTATION

1. Data Collection and Preprocessing
Load a CSV dataset containing latitude, longitude, city, and address information for delivery locations across Mumbai, Navi Mumbai, and Thane.
Filter locations specific to Thane (first test case) and extract essential attributes.
Assign unique IDs to each delivery point and categorize them based on color codes for visualization (red for start point, black for delivery locations).
Code Snippet:

```
python
import pandas as pd
df = pd.read_csv("/content/drive/My Drive/mumbai_navi_thane_data.csv")
city = "Thane"
dtf = df[df["City"] == city][["City", "Street Address", "Latitude", "Longitude"]].reset_index(drop=True)
dtf = dtf.reset_index().rename(columns={"index": "id", "Latitude": "y", "Longitude": "x"})
dtf["color"] = "black"
dtf.loc[0, "color"] = "red"
start = dtf.iloc[0][["y", "x"]].values
```

2. Road Network Extraction Using OSMnx
Retrieve the road network graph for Thane using OSMnx, ensuring routing follows actual roads.
Extract travel speeds and estimated travel times for each street segment.
Visualize the road network graph to confirm connectivity.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 11, April 2025
Impact Factor: 7.67

Code Snippet:

python

```
import osmnx as ox
G = ox.graph_from_point(start, dist=10000, network_type="drive")
G = ox.add_edge_speeds(G)
G = ox.add_edge_travel_times(G)
fig, ax = ox.plot_graph(G, bgcolor="black", node_size=5, node_color="white", figsize=(16, 8))
```

3. Snapping Delivery Points to Nearest Road Nodes

Find the nearest road network node for each delivery location using OSMnx.
Replace location coordinates with those of the nearest snapped node to ensure valid routing paths.

Code Snippet:

python

```
import networkx as nx
snapped_nodes = []
snapped_coords = []
for idx, row in dtf.iterrows():
    nearest_node = ox.distance.nearest_nodes(G, X=row['x'], Y=row['y'])
    snapped_nodes.append(nearest_node)
    node_data = G.nodes[nearest_node]
    snapped_coords.append((node_data['y'], node_data['x']))
dtf['node'] = snapped_nodes
dtf['y'], dtf['x'] = zip(*snapped_coords)
dtf = dtf.drop_duplicates("node", keep='first')
```

Building the Distance Matrix Using NetworkX

Construct a distance matrix using Dijkstra's shortest path algorithm to calculate shortest travel times between delivery points.
Store results in a structured matrix, which is used for optimizing the delivery route sequence.

Code Snippet:

python

```
import numpy as np
def f(a, b):
    try:
        return nx.shortest_path_length(G, source=a, target=b, method='dijkstra', weight='travel_time')
    except:
        return np.nan
distance_matrix = np.asarray([[f(a, b) for b in dtf["node"].tolist()] for a in dtf["node"].tolist()])
distance_matrix = pd.DataFrame(distance_matrix, columns=dtf["node"].values, index=dtf["node"].values)
distance_matrix = distance_matrix.fillna(distance_matrix.max().max()).round().astype(int)
```

5. Route Optimization Using OR-Tools

Define a Vehicle Routing Problem (VRP) model.
Use OR-Tools to apply the PATH CHEAPEST ARC algorithm, which selects the shortest travel path iteratively.
Extract the optimized route sequence and compute total travel time savings.

Code Snippet:

python

```
from ortools.constraint_solver import pywrapcp, routing_enums_pb2
```

```
drivers = 1
lst_nodes = dtf["node"].tolist()
start_node = dtf.iloc[0]["node"]
manager = pywrapcp.RoutingIndexManager(len(lst_nodes), drivers, lst_nodes.index(start_node))
model = pywrapcp.RoutingModel(manager)
def get_distance(from_index, to_index):
   return distance_matrix.iloc[from_index, to_index]
distance = model.RegisterTransitCallback(get_distance)
model.SetArcCostEvaluatorOfAllVehicles(distance)
parameters = pywrapcp.DefaultRoutingSearchParameters()
parameters.first_solution_strategy = routing_enums_pb2.FirstSolutionStrategy.PATH_CHEAPEST_ARC
solution = model.SolveWithParameters(parameters)
index = model.Start(0)
route_idx = []
route_distance = 0
while not model.IsEnd(index):
   route_idx.append(manager.IndexToNode(index))
   previous_index = index
   index = solution.Value(model.NextVar(index))
   route_distance += get_distance(previous_index, index)
lst_route = [lst_nodes[i] for i in route_idx]
print(f'Total distance: {round(route_distance / 1000, 2)} km')
```

6. Visualization Using Folium and Plotly

Display optimized delivery locations on a Folium map.

Animate real-time delivery tracking using Plotly.

Code Snippet:

```
import folium
route_map = folium.Map(location=[G.nodes[lst_route[0]]['y'], G.nodes[lst_route[0]]['x']], zoom_start=14)
folium.PolyLine([(G.nodes[n]['y'], G.nodes[n]['x']) for n in lst_route], color="blue",
weight=2.5).add_to(route_map)
route_map
```

7. Comparison with Unoptimized Route

Simulate greedy nearest-neighbor delivery to compare performance.

Compute distance saved, time saved, fuel efficiency, and cost reduction.

Code Snippet:

```
python
unvisited = set(lst_nodes)
current = start_node
unoptimized_route = [current]
unvisited.remove(current)
while unvisited:
   next_node = min(unvisited, key=lambda node: distance_matrix.loc[current, node])
   unoptimized_route.append(next_node)
   unvisited.remove(next_node)
   current = next_node
unoptimized_time = sum(distance_matrix.loc[a, b] for a, b in zip(unoptimized_route[:-1],
unoptimized_route[1:]))
optimized_time = route_distance
optimized_distance_km = optimized_time / 3600 * 30
unoptimized_distance_km = unoptimized_time / 3600 * 30
fuel_saved = (unoptimized_distance_km - optimized_distance_km) / 12
cost_saved = fuel_saved * 100
```

8. Final Output

- Maps displaying optimized delivery routes.
- Animated tracking of package distribution.
- Performance comparisons highlighting savings in distance, time, fuel, and cost.

V. RESULTS AND DISCUSSION

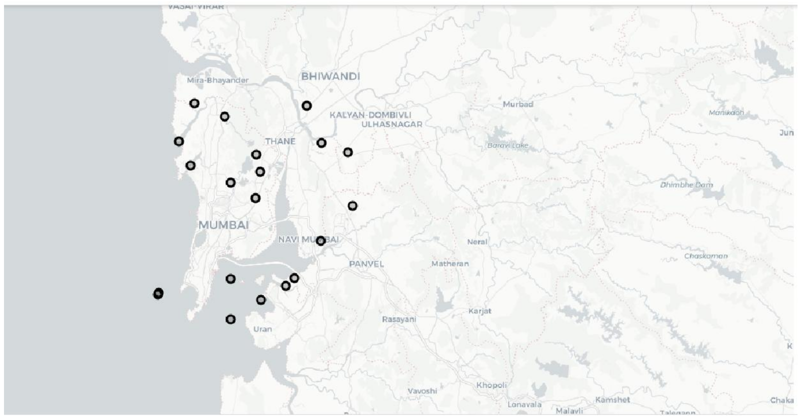

Fig. 2. Delivery Locations

Map showing delivery locations marked with black dots, representing the destinations to be visited during the delivery route.

Fig. 3. Distance Graph

Visualizes the pairwise travel times between delivery nodes using a heatmap, where lighter shades indicate shorter travel times and darker shades represent longer or inaccessible paths

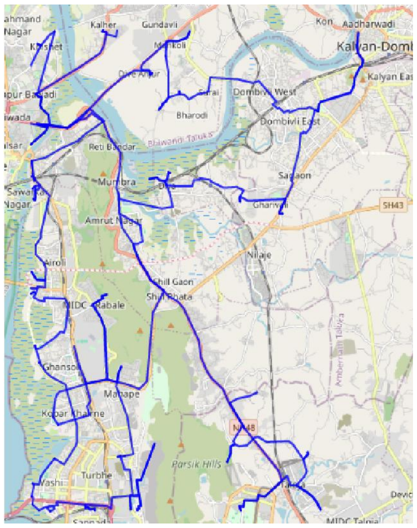
Fig. 4. Optimised Route Generation

Displays the computed shortest and most efficient delivery route starting from the warehouse, covering all selected delivery points based on travel time optimization.

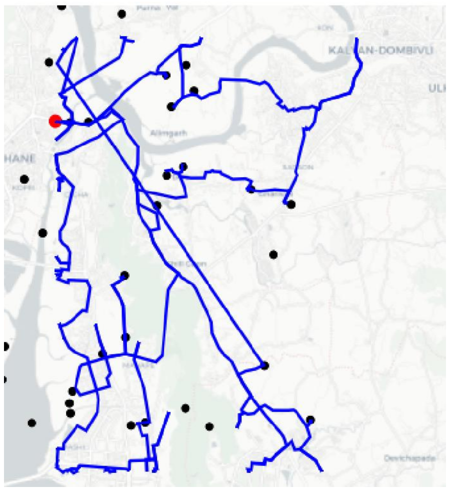
Fig. 5. Simulated Final Route

Visualizes the animated progression of the delivery vehicle along the optimized route, simulating package drop-offs at each location in real-time.

VI. FUTURE SCOPE

Integration of the Model into a Working Product:

Overview of the Product

The model can be developed into a smart last-mile delivery optimization platform, used by logistics companies and e-commerce businesses. This product would serve as a real-time delivery route planner, dynamically adjusting delivery sequences based on live traffic conditions, order changes, and fleet availability.

Automated Route Optimization

The system would integrate with delivery management software, automatically computing the most efficient paths. It would use real-time traffic data to dynamically update routes and prevent delays.

Multi-Vehicle Fleet Coordination
The product could manage multiple delivery vehicles rather than just one.
The system would assign delivery tasks intelligently across available drivers based on location proximity.
Integration with GPS and Tracking Systems
It would connect to GPS devices inside delivery vehicles for real-time navigation.
Customers could track their deliveries in real time, enhancing transparency.
Mobile and Web Dashboard
A mobile app and web interface would allow fleet managers to monitor deliveries.
Drivers would receive optimized routes directly on their mobile devices.
AI-Based Demand Prediction
The system could analyse past delivery trends and predict peak demand periods.
Businesses could pre-adjust routes and fleet availability based on expected order volumes.

VII. CONCLUSION

- Successful optimization ensures that delivery operations run with maximum efficiency.
- By continuously refining travel paths, deliveries are completed in less time, avoiding unnecessary delays.
- This adaptability directly leads to lower fuel consumption, as vehicles take the most efficient routes, minimizing idle time and distance travelled.
- Reduced fuel use not only cuts operational costs but also contributes to sustainability efforts by lowering emissions.
- The ability to optimize delivery routes also results in significant cost savings, as companies can achieve higher delivery
- throughput with fewer resources, maximizing efficiency and profitability.

VIII. ACKNOWLEDGMENT

We would like to express our sincere gratitude to Datta Meghe College of Engineering for providing us with the platform to prepare our project on "DYNAMIC ROUTE OPTIMIZATION SOLUTION FOR COURIER SERVICES". We are particularly thankful to our Principal, Dr. P. A. Dode, for instilling in us the importance of this research and for providing us with the necessary opportunities and time to conduct and present our findings.
We are immensely grateful to our guide and Head of the Computer Engineering Department, Dr. Amol Pande, for his invaluable support, encouragement, and insightful suggestions throughout our research process.
Furthermore, the successful completion of this research would not have been possible without the cooperation, suggestions, and help of our friends and family. Their encouragement and assistance have been invaluable, and we deeply appreciate their contributions.

REFERENCES

[1] Dantzig, G. B., & Ramser, J. H. (1959). The truck dispatching problem. Management Science, 6(1), 80–91. https://doi.org/10.1287/mnsc.6.1.80
[2] Laporte, G. (2009). Fifty years of vehicle routing. Transportation Science, 43(4), 408–416. https://doi.org/10.1287/trsc.1090.0301
[3] Golden, B., Raghavan, S., & Wasil, E. (2008). The Vehicle Routing Problem: Latest Advances and New Challenges. Springer Science & Business Media.
[4] Ghiani, G., Laporte, G., & Musmanno, R. (2013). Introduction to Logistics Systems Management (2nd ed.). Wiley. https://doi.org/10.1002/9781118555950
[5] Boysen, N., Briskorn, D., & Emde, S. (2017). Same-day delivery: Opportunities and challenges for route planning. OR Spectrum, 39(3), 713–735. https://doi.org/10.1007/s00291-017-0489-0
[6] Pillac, V., Gendreau, M., Guéret, C., & Medaglia, A. L. (2013). A review of dynamic vehicle routing problems. European Journal of Operational Research, 225(1), 1–11. https://doi.org/10.1016/j.ejor.2012.08.015

[7] Potvin, J. Y., & Bengio, S. (1996). The vehicle routing problem with time windows — Part II: Genetic search. INFORMS Journal on Computing, 8(2), 165–172. https://doi.org/10.1287/ijoc.8.2.165

[8] Google Developers. (n.d.). OR-Tools Routing Solver. Retrieved from https://developers.google.com/optimization/routing

[9] Wang, X., & Hu, H. (2020). A deep reinforcement learning framework for the vehicle routing problem. IEEE Access, 8, 158178–158190. https://doi.org/10.1109/ACCESS.2020.3018954

[10] IBM Research. (n.d.). AI for Supply Chain Optimization. Retrieved from https://research.ibm.com

[11] Kallehauge, B., Larsen, J., Madsen, O. B. G., & Solomon, M. M. (2005). Vehicle routing problem with time windows. In Column Generation (pp. 67–98). Springer. https://doi.org/10.1007/0-387-25486-2_3

[12] Yu, B., Yang, Z. Z., & Yao, B. (2011). An improved ant colony optimization for vehicle routing problem. European Journal of Operational Research, 196(1), 171–176. https://doi.org/10.1016/j.ejor.2008.02.028

[13] Jain, A., & Singh, P. (2021). A review on optimization techniques for vehicle routing problem with time windows. Materials Today: Proceedings. https://doi.org/10.1016/j.matpr.2021.07.473

[14] Zhang, R., Cheng, T. C. E., & Chen, Y. (2022). A hybrid deep learning-based dynamic route optimization method for intelligent logistics. Expert Systems with Applications, 198, 116828. https://doi.org/10.1016/j.eswa.2022.116828

[15] Suresh, K. P., & Rajendran, C. (2023). Cloud-based deployment automation tools: A comparative study. Journal of Cloud Computing, 12(1), 1–12. https://doi.org/10.1186/s13677-023-00412-7