Vehicle Routing

Leo Jaeeun Kim
University of Michigan
General Motors

ABSTRACT

Vehicle routing is a critical component of the hierarchical planning architecture in autonomous driving systems. Positioned at the highest level of decision-making, vehicle routing determines the optimal sequence of locations a vehicle—or a fleet—should visit, guiding long-horizon navigation across complex road networks. Unlike traditional offline routing, autonomous systems must perform this task under real-time constraints, integrating sensor-derived perception, environmental uncertainty, and dynamic mission requirements. This paper investigates the role of vehicle routing within the full autonomy stack, distinguishing it from mid-level behavior planning and low-level path control. We formalize the vehicle routing problem for autonomous systems, review both classical graph-based approaches and modern learning-based techniques, and propose system-level integrations that enhance adaptability and mission efficiency. By reinforcing the routing layer within the hierarchical planning pipeline, autonomous systems can better align high-level objectives.

INTRODUCTION

Autonomous vehicles make decisions through a hierarchical planning architecture that partitions the problem space into three main levels: vehicle routing (high-level), behavior planning (mid-level), and path planning and control (low-level). Each layer is responsible for decisions at different spatial and temporal resolutions, yet all are interconnected to ensure safe, efficient, and goal-driven navigation.

At the top of this hierarchy, vehicle routing governs long-term planning by determining *where* the vehicle should go and *in what order* destinations should be visited. This includes trip planning for a single vehicle as well as coordination for multiple agents in fleet applications such as autonomous ride-sharing, delivery logistics, or emergency response. The output of routing provides global waypoints or sub-goals to mid- and low-level planners that execute real-time maneuvers and ensure dynamic feasibility.

While traditional routing problems are well studied in operations research under formulations like the Vehicle Routing Problem (VRP) or Traveling Salesman Problem (TSP), their adaptation to autonomous driving introduces several new dimensions:

- Real-time environmental changes, perceived via onboard sensors
- Integration with traffic dynamics and road constraints
- Compatibility with downstream behavior and path planners
- Uncertainty in localization and object detection
- Multi-agent coordination with shared resources

Importantly, vehicle routing in autonomous systems must operate in synergy with perception. High-level decisions are informed by semantic maps, traffic conditions, and even predictions of future road states. In advanced scenarios, vehicles may engage in active perception, adjusting routes to improve situational awareness or sensor coverage—blurring the boundary between planning and perception.

This paper focuses on the vehicle routing layer within the hierarchical autonomy stack. We begin by formalizing the problem under constraints typical to autonomous vehicles, then explore solution strategies ranging from classical shortest-path algorithms to learning-based and hybrid methods. We also present system-level designs that ensure consistent information flow across planning layers, enabling scalable and robust routing decisions under uncertainty.

Through this exploration, we aim to establish a principled foundation for vehicle routing as a cornerstone of autonomous mobility systems, tightly coupled with downstream planners and informed by upstream perception.

METHOD

The vehicle routing problem (VRP) in autonomous systems is commonly modeled as a graph-theoretic optimization task. Let $G = (V, E, w)$ be a complete, undirected, weighted graph, where the vertex set $V = \{v_0, v_1 \dots, v_n\}$ includes the depot v_0 and the locations to be visited v_1 through v_n. The edge set E represents all possible direct routes between locations, and each edge $e \in$ E is associated with a non-negative cost *w(e)* reflecting distance, time, energy, or any application-specific metric. The routing objective is to compute a minimum-weight Hamiltonian cycle that starts and ends at the depot and visits every node exactly once.

To formalize this as an optimization problem, a widely adopted approach is to use a Mixed-Integer Linear Programming (MILP) formulation. Here, binary decision variables $x_e \in \{0,1\}$ are used to represent whether a given edge $e \in$ E is included in the final solution. The objective function seeks to minimize total cost across selected edges:

$$\min \sum w(e) * x_e$$

Subject to the constraints:

$$\sum_{e \in \delta(v)} x_e = 2\,, \qquad \forall v \in V$$

Each node must be entered and exited exactly once

Binary edge selection (or convex relaxation):

$$0 \le x_e \le 1, \qquad \forall e \in E$$

The latter turns the problem into a convex relaxation for easier solving but loses exactness.

However, the convex relaxation often results in subtours—small disconnected loops that satisfy the local constraints but fail to visit all nodes in one connected cycle. This issue can be mitigated using subtour elimination constraints, which prevent invalid cycle configurations. Several approaches exist for subtour elimination, including:

- Miller-Tucker-Zemlin (MTZ) constraints: Introduce auxiliary variables to enforce node ordering and eliminate disjoint loops.

- Cutting-plane methods: Dynamically detect and add violated subtour constraints during solving.

- Lazy constraints: Leveraged in modern MILP solvers to conditionally add subtour rules only when needed.

Despite these improvements, MILP-based methods are computationally infeasible for large-scale VRPs due to their exponential worst-case complexity, typically $O(n^2 * 2^n)$. As such, autonomous vehicle systems cannot rely on exact solvers for real-time deployment.

To address scalability, various approximate and heuristic methods have been developed. Among the most successful is the Lin-Kernighan Heuristic (LKH), which employs local search with variable-length segment swaps (k-opt). It delivers near-optimal results efficiently, even for problems with thousands of nodes. The Christofides algorithm is another classical technique, offering a 1.5-approximation guarantee but with higher polynomial complexity $O(n^4)$. and less practical applicability for large inputs.

Another tractable family of methods comes from constraint programming (CP). Unlike MILP, CP frameworks like Google's OR-Tools CP-SAT solver can flexibly model non-linear and domain-specific constraints such as: Vehicle capacity limits, Time window constraints, Multi-depot scenarios, and Energy or charging constraints. These solvers use constraint propagation and branching strategies to navigate the search space efficiently, often outperforming MILP in real-world logistics and autonomous delivery contexts. In parallel, the recent wave of learning-based methods introduces neural and data-driven architectures for routing. These include pointer networks, graph neural networks, and transformer-based models that learn to predict visit sequences directly from graph inputs. For example, models such as the Attention Model by Kool et al. or reinforcement learning agents trained on synthetic routing distributions can produce valid routes with minimal computation time at inference. The key strengths of learning-based approaches are:

- Speed: Once trained, inference is near-instantaneous.

- Adaptability: Can generalize across varying problem sizes and distributions.

- Data efficiency: Learning from real-world routing logs can encode patterns missed by rule-based methods.

However, learning methods come with drawbacks. They lack hard feasibility guarantees and may require post-processing to enforce constraints. Moreover, they often need large-scale labeled datasets or strong simulation environments for training, which can be resource-intensive.

Ultimately, no single method suffices for all autonomous vehicle routing tasks. A practical routing architecture often combines strengths of each approach. For instance, a hybrid solution might apply an exact MILP solver for core sub-routing among critical nodes, use heuristics like LKH for full route estimation, and incorporate a learning-based model to initialize or guide the search.

In summary, vehicle routing in autonomous vehicles is a rich and evolving domain that intersects classical optimization, constraint reasoning, and modern machine learning. The choice of method depends on task complexity, real-time requirements, constraint structure, and available computational resources. A well-designed routing stack balances optimality with responsiveness and robustness—key attributes for safe and efficient autonomous operation.

IMPLEMENTATION

The vehicle routing problem in last-mile package delivery—such as that faced by e-commerce logistics platforms like Amazon—requires an implementation that efficiently handles delivery demand, vehicle capacity, and real-time operational constraints. A typical system must assign packages to a fleet of vehicles and determine delivery routes that minimize total distance or time while respecting truck capacities and ensuring all delivery locations are visited.

Environment and Data Simulation

To simulate a realistic urban delivery environment, a two-dimensional coordinate space is populated with randomly distributed nodes representing delivery locations. Node 0 is designated as the central depot, from which all delivery vehicles depart and to which they return. Each customer node is assigned a randomly generated demand (e.g., package volume or weight), and vehicles have uniform capacity constraints. The Euclidean distance between each pair of nodes is computed to serve as the cost metric for routing.

Mathematical Model Construction

The Capacitated Vehicle Routing Problem (CVRP) is modeled as a Mixed-Integer Linear Program (MILP). The graph is defined as $G = (V, E, w)$, where V represents nodes, E denotes arcs connecting pairs of nodes, and $w(i, j)$ represents the cost associated with traversing edge (i, j) . Decision variables $x_{ijv} \in \{0,1\}$ indicate whether vehicle *v* travels directly from node *i* to node *j*, and $y_{iv} \in \{0,1\}$ denote whether vehicle *v* visits node *i*.

The objective function minimizes total route cost:

$$min \sum_{v \in V} \sum_{(i,j) \in E} w(i,j) * x_{ijv}$$

Subject to the following constraints:
- Each customer node is visited exactly once.
- Flow conservation ensures vehicles entering a node also exit it.
- Each vehicle must start and end its route at the depot.
- The sum of demands assigned to a vehicle must not exceed its capacity.

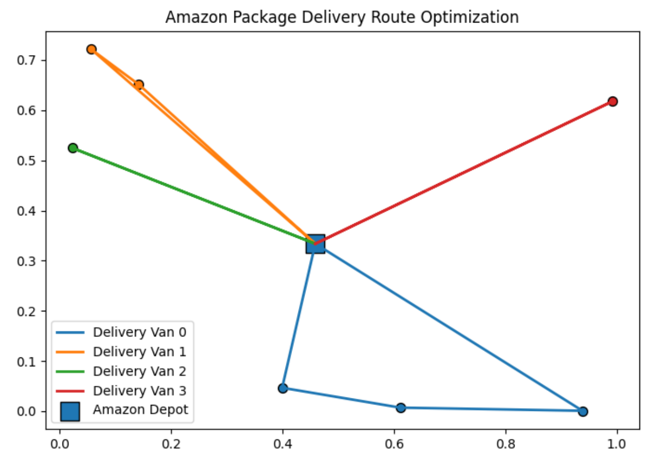
These constraints ensure feasible delivery plans that satisfy operational requirements.

Subtour Elimination Mechanism
To enforce route connectivity and prevent invalid subtours—routes that form closed loops without returning to the depot—a constraint generation method is employed. After solving the MILP, the current solution is analyzed to identify disconnected strongly connected components (SCCs) that do not include the depot. If such components are found, additional constraints are introduced to force at least one route connection between the subtour and the remaining network. The optimization problem is then re-solved with the new constraints. This iterative process continues until all subtours are eliminated, ensuring that all vehicle routes begin and end at the depot.

Optimization and Solver Configuration
Gurobi is used as the optimization solver due to its performance in solving large-scale MILPs. Solver parameters are configured to enforce high-accuracy solutions, with a specified MIP gap of 10^6, and search heuristics tuned for exploration. The Pyomo optimization framework is utilized for model definition and constraint management, enabling flexible expression of both network structure and vehicle constraints.

Route Extraction and Visualization
After a feasible solution is obtained, vehicle-wise tours are reconstructed by traversing active arcs in the solution graph. Each vehicle's route starts at the depot, visits a subset of customer nodes based on capacity, and returns to the depot. Final results are visualized by plotting each route in a distinct color over the spatial coordinate map. The depot is visually emphasized to highlight route closure, and the distribution of deliveries across vehicles is examined for load balancing.

Practical Implications
This implementation structure aligns with operational needs in autonomous last-mile delivery. Real-world applications involve more complex constraints such as time windows, traffic conditions, and dynamic package insertion. However, the core methodology described here serves as a scalable foundation for decision-support systems that power logistics platforms. The MILP-based approach, combined with iterative subtour elimination and visualization, provides interpretable and tractable routing solutions suitable for embedded AV logistics systems or cloud-based delivery orchestration tools.



The example code for Amazon Package Delivery Route Optimization can be found in https://github.com/woa0425/Leo_ADAS_script/blob/main/Vehicle_Routing/VehicleRouting_python/DeliveryRouting_Optimization.py

A simplified Capacitated Vehicle Routing Problem (CVRP) was implemented using Pyomo and solved with Gurobi to simulate Amazon-style last-mile delivery. The model generates a synthetic instance with one depot and seven customer nodes, each assigned random demand values. Euclidean distances define edge weights for the complete graph. A fleet of four delivery vehicles, each with a fixed capacity, is tasked with servicing all customers while minimizing total route distance. The MILP model defines binary decision variables to determine whether a vehicle travels between two nodes and whether it visits a node. Core constraints enforce depot flow balance, unique customer visits, flow conservation per vehicle, and vehicle capacity limits. A subtour elimination loop is integrated by detecting strongly connected components (SCCs) in the current solution and adding constraints iteratively to prevent disconnected cycles.

CONCLUSION

This paper presented a structured examination of vehicle routing within the hierarchical planning architecture of autonomous driving systems, emphasizing its critical role in long-horizon decision-making. By formulating the routing task as a Capacitated Vehicle Routing Problem (CVRP), the study demonstrated how mixed-integer optimization techniques can be applied to model real-world delivery operations, such as Amazon-style last-mile logistics. The proposed MILP-based framework enforces essential constraints such as flow conservation, capacity limits, and unique customer visitation while addressing route connectivity through iterative subtour elimination.

The implementation using Pyomo and Gurobi showcased a scalable method for generating feasible, interpretable, and costeffective delivery routes in a simulated urban setting. Although exact optimization approaches provide high-quality solutions, their computational demands limit applicability in large-scale or real-time deployments. To overcome these challenges, the discussion included heuristic, constraint-programming, and learning-based alternatives that can complement or replace MILP in complex environments.

Vehicle routing remains a key enabler for intelligent fleet coordination and autonomous mobility. As real-world constraints grow in complexity—through time windows, dynamic orders, and shared resources—future research will benefit from hybrid systems that fuse optimization, perception, and machine learning. The presented methodology offers a foundational baseline upon which such adaptive, efficient, and scalable routing systems can be built.

REFERENCES

[1] University of Michigan, NAVARCH 565 – Lecture 13: Planning & Vehicle Routing, Fall 2023.