Dynamic Platoon Formation of Multi-Type Autonomous Vehicles for Sustainable Urban Mobility

Jaeyun Ree
Faculty of Information Technology
Monash University
Melbourne, VIC, Australia
jree0010@student.monash.edu

Mohammed Eunus Ali
Faculty of Information Technology
Monash University
Melbourne, VIC, Australia
eunus.ali@monash.edu

***Abstract*—This paper addresses the energy inefficiency of single-occupancy vehicles by introducing a novel cooperative autonomous vehicle system where smaller Passive Vehicles (PVs) can be physically towed by larger Active Vehicles (AVs) during shared highway segments. Unlike traditional platooning that relies solely on aerodynamic drafting, our approach enables near-complete energy elimination for towed vehicles. We formulate the platoon formation problem as a constrained optimization problem on a one-dimensional highway model and propose two algorithms: a Greedy Maximum-Weight Matching algorithm that provides computational efficiency with $O(NM \log(NM))$ complexity, and an Enhanced Hungarian Algorithm that guarantees optimal solutions with $O((N+M)^3)$ complexity. A key innovation is the multi-segment matching capability, allowing a single PV to be towed by multiple AVs across different route segments. Experimental evaluation on synthetic highway scenarios demonstrates that both algorithms achieve significant energy savings, with the Hungarian algorithm consistently finding optimal solutions while the Greedy approach provides near-optimal results (within 3% of optimal) at substantially lower computational cost.**

***Index Terms*—autonomous vehicles, vehicle platooning, energy efficiency, Hungarian algorithm, combinatorial optimization, intelligent transportation systems**

I. INTRODUCTION

The persistent over-reliance on single-occupancy vehicles poses significant challenges to urban transportation sustainability. According to the U.S. Department of Transportation, single-occupancy vehicles account for approximately 76% of commuter trips, contributing substantially to traffic congestion, energy consumption, and carbon emissions [1]. The transportation sector alone is responsible for nearly 29% of total greenhouse gas emissions in the United States, with light-duty vehicles contributing the largest share [2]. This inefficiency is particularly pronounced along shared routes where multiple individuals travel similar paths yet operate their vehicles independently.

Autonomous vehicle (AV) technology presents a transformative opportunity to address these challenges through coordinated vehicle operation. This paper introduces a novel concept of *active* and *passive* autonomous vehicles, where smaller Passive Vehicles (PVs) can temporarily attach to larger Active Vehicles (AVs) during shared highway segments. Unlike traditional vehicle platooning that maintains physical separation between vehicles, our approach enables physical coupling where a PV's propulsion system is deactivated while being towed, leading to energy savings that approach 100% for the towed vehicle during the attached phase. This paradigm shifts transportation from isolated driving to an on-demand, energy-transfer-based mobility service.

Existing research on cooperative vehicle systems spans several related but distinct areas. Traditional platooning systems [3], [4] focus on coordinated driving with aerodynamic benefits, typically achieving 10-20% fuel savings through drafting. Ridesharing and carpooling optimization [5], [6] address passenger matching but not vehicle coupling. Eco-driving strategies [7], [8] optimize individual vehicle behavior without inter-vehicle coordination. The assignment problem we address relates to the classic bipartite matching literature [9], [10], though our multi-segment matching with capacity constraints introduces novel complexity. While these approaches offer incremental improvements, they do not address the fundamental inefficiency of independent propulsion for vehicles with overlapping routes.

We formulate the platoon formation problem on a one-dimensional highway model where the objective is to maximize total energy savings through strategic PV-to-AV assignments. To solve this problem, we propose two complementary algorithms. The *Greedy Maximum-Weight Matching* algorithm provides computational efficiency suitable for real-time applications, while the *Enhanced Hungarian Algorithm* guarantees optimal solutions for scenarios where optimality is paramount. A key innovation is our multi-segment matching capability: rather than restricting each PV to a single AV, our algorithms allow a PV to be towed by different AVs across different segments of its route, significantly improving overall system efficiency.

The main contributions of this paper are:

- A formal one-dimensional problem formulation that captures the essential trade-offs in cooperative vehicle pla-

tooning, with precise mathematical notation and constraint specification.
- A Greedy algorithm with $O(NM\log(NM))$ complexity that achieves provable $\frac{1}{2}$-approximation through iterative maximum-weight matching.
- An Enhanced Hungarian algorithm with capacity constraints that guarantees optimal assignments with $O((N+M)^3)$ complexity per iteration.
- Multi-segment matching capability allowing PVs to be towed by multiple AVs across their routes, with point-wise capacity tracking.
- Comprehensive experimental evaluation demonstrating algorithm performance across varying capacity configurations, highway lengths, and vehicle densities.

II. Problem Formulation

We formalize the platoon formation problem on a simplified one-dimensional highway model. This abstraction captures the essential optimization trade-offs while enabling tractable analysis. Extension to general road networks is discussed in Section V.

A. Notation Summary

Table I summarizes the key notation used throughout this paper.

TABLE I
Summary of Notation

Symbol	Description
L	Highway length
$\mathcal{A}$	Set of Active Vehicles (AVs), $\lvert\mathcal{A}\rvert = N$
$\mathcal{P}$	Set of Passive Vehicles (PVs), $\lvert\mathcal{P}\rvert = M$
e_i^a, x_i^a	Entry and exit points of AV a_i
e_j^p, x_j^p	Entry and exit points of PV p_j
C_i	Towing capacity of AV a_i
t_i^a, v_i^a	Entry time and speed of AV a_i
d_{ij}	Shared travel distance between a_i and p_j
S_{ij}	Energy saving when p_j is towed by a_i
cp_{ij}, dp_{ij}	Coupling and decoupling points
L_{min}	Minimum shared distance for platooning
τ	Time tolerance for coupling synchronization

B. System Model

We consider a coordinated platoon formation system operating on a unidirectional highway segment of length L. The system comprises two distinct vehicle classes.

Definition 1 (Active Vehicle)**.** *An Active Vehicle (AV) $a_i \in \mathcal{A}$ is a larger autonomous vehicle capable of towing multiple smaller vehicles. Each AV is characterized by the tuple $(e_i^a, x_i^a, C_i, t_i^a, v_i^a)$ where $e_i^a, x_i^a \in [0, L]$ are entry and exit points, $C_i \in \mathbb{Z}^+$ is the towing capacity, and t_i^a, v_i^a are the entry time and constant speed.*

Definition 2 (Passive Vehicle)**.** *A Passive Vehicle (PV) $p_j \in \mathcal{P}$ is a smaller autonomous vehicle that can be towed by AVs. Each PV is characterized by the tuple $(e_j^p, x_j^p, t_j^p, v_j^p)$ where $e_j^p, x_j^p \in [0, L]$ are entry and exit points, and t_j^p, v_j^p are entry time and self-driving speed.*

C. Shared Path and Energy Model

For an AV a_i and PV p_j, the *shared path* is the overlap of their routes:

$$d_{ij} = \max(0, \min(x_i^a, x_j^p) - \max(e_i^a, e_j^p)) \quad (1)$$

The coupling point cp_{ij} (where PV attaches) and decoupling point dp_{ij} (where PV detaches) are:

$$cp_{ij} = \max(e_i^a, e_j^p) \quad (2)$$
$$dp_{ij} = \min(x_i^a, x_j^p) \quad (3)$$

The energy saving S_{ij} when PV p_j is towed by AV a_i is proportional to the shared distance:

$$S_{ij} = \alpha \cdot d_{ij} \quad (4)$$

where α is an energy coefficient. For simplicity, we set $\alpha = 1$, making energy saving equivalent to distance saved.

D. Multi-Segment Matching

A key innovation of our formulation is *multi-segment matching*: a single PV can be matched to multiple AVs across different segments of its route.

Definition 3 (Segment Assignment)**.** *A segment assignment for PV p_j to AV a_i is a tuple (p_j, a_i, s, e) where $[s, e] \subseteq [e_j^p, x_j^p]$ is the segment during which p_j is towed by a_i.*

For PV p_j, its route from e_j^p to x_j^p can be partitioned into segments, where each segment is either:
- *Covered*: PV is towed by an AV (energy saved)
- *Uncovered*: PV drives independently (no energy saved)

E. Optimization Problem

Let $x_{ij}^{(k)} \in \{0, 1\}$ be a binary decision variable indicating whether PV p_j is assigned to AV a_i for segment k. The optimization problem is:

$$\max \quad \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k} x_{ij}^{(k)} \cdot S_{ij}^{(k)} \quad (5)$$

$$\text{s.t.} \quad \sum_{j:point\in[cp_{ij},dp_{ij}]} x_{ij}^{(k)} \le C_i \quad \forall i, \forall point \quad (6)$$

$$\text{Segments of } p_j \text{ are non-overlapping} \quad \forall j \quad (7)$$

$$d_{ij}^{(k)} \ge L_{min} \quad \text{if } x_{ij}^{(k)} = 1 \quad (8)$$

$$x_{ij}^{(k)} \in \{0, 1\} \quad \forall i, j, k \quad (9)$$

Constraint Interpretation:
- **Capacity** (6): At any point along its route, an AV cannot tow more than C_i PVs simultaneously. This is a *point-wise* constraint—capacity varies as PVs attach and detach.
- **Non-overlap** (7): A PV cannot be towed by two AVs at the same location.
- **Minimum length** (8): Platooning is only worthwhile if the shared distance exceeds L_{min} (to amortize coupling/decoupling overhead).

F. Time Constraints (Optional Extension)

When temporal synchronization is required, an additional feasibility condition must hold:

$$|t_i^a(cp_{ij}) - t_j^p(cp_{ij})| \leq \tau \quad (10)$$

where τ is the time tolerance, and $t_i^a(cp_{ij}) = t_i^a + (cp_{ij} - e_i^a)/v_i^a$ is the time when AV a_i reaches the coupling point.

G. Illustrative Example

Figure 1 illustrates a concrete instance of our problem with 2 AVs and 3 PVs on a highway of length $L = 100$.

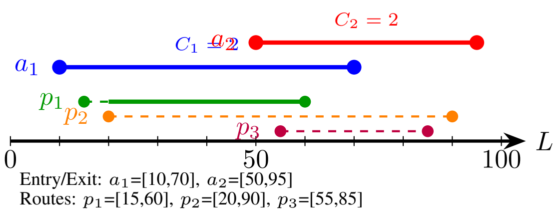

Fig. 1. Toy example with 2 AVs (solid lines) and 3 PVs (dashed lines) on a 1D highway. AV routes are shown in blue (a_1) and red (a_2); PV routes in green (p_1), orange (p_2), and purple (p_3). The optimal assignment tows p_1 by a_1 (saving=45), segments of p_2 by a_1 then a_2 (saving=50+40=90 via multi-segment), and p_3 by a_2 (saving=30). Total saving = 165.

Example Analysis:

- **Shared distances:** $d_{11} = \min(70, 60) - \max(10, 15) = 45$, $d_{12} = 50$, $d_{22} = 40$, $d_{23} = 30$.
- **Capacity constraint:** At point 55, if a_1 tows both p_1 and p_2, load = 2 = C_1 (feasible). At point 60, p_1 detaches, freeing capacity.
- **Multi-segment benefit:** PV p_2 can be towed by a_1 from [20,70], then by a_2 from [70,90], achieving coverage of 70 units instead of just 50.

We use this example throughout Section III to illustrate algorithm behavior.

III. Proposed Algorithms

We present two algorithms: a greedy approach for computational efficiency and a Hungarian-based approach for optimality.

A. Greedy Maximum-Weight Matching Algorithm

1) Key Insight: The greedy approach exploits the observation that high-value assignments (long shared distances) should be prioritized. By iteratively selecting the assignment with maximum energy saving, we construct a solution that, while not guaranteed optimal, achieves strong empirical performance.

2) Algorithm Description: Algorithm 1 operates in three phases:

Phase 1 (Initialization): For each PV, we maintain a list of *uncovered segments*—initially the entire route. For each AV, we maintain a list of current towing assignments to track point-wise capacity.

Phase 2 (Candidate Generation): At each iteration, we enumerate all feasible (AV, PV, segment) tuples. A candidate is feasible if:

1) The segment length exceeds L_{min}
2) The AV has available capacity throughout the segment
3) The segment lies within an uncovered portion of the PV's route

Phase 3 (Greedy Selection): We select the candidate with maximum energy saving, update both the PV's uncovered segments and the AV's capacity state, and repeat until no feasible candidates remain.

Algorithm 1 Greedy Multi-AV Platoon Matching

Require: AVs $\mathcal{A}$, PVs $\mathcal{P}$, minimum length L_{min}
Ensure: Assignments $\mathcal{X}$, total saving

```
// Phase 1: Initialization
for each p_j ∈ P do
    uncovered[j] ← {[e_j^p, x_j^p]} {Full route}
end for
for each a_i ∈ A do
    assignments[i] ← ∅ {No towing yet}
end for
X ← ∅, total ← 0
repeat
    // Phase 2: Generate candidates
    candidates ← ∅
    for each a_i ∈ A, p_j ∈ P do
        for each segment [s, e] ∈ uncovered[j] do
            [cp, dp] ← overlap of [s, e] with a_i's route
            if dp − cp ≥ L_min then
                if a_i has capacity on [cp, dp] then
                    Add (dp − cp, a_i, p_j, cp, dp) to candidates
                end if
            end if
        end for
    end for
    if candidates = ∅ then
        break
    end if
    // Phase 3: Greedy selection
    Sort candidates by saving (descending)
    (saving, a_i, p_j, cp, dp) ← candidates[0]
    X ← X ∪ {(p_j, a_i, cp, dp)}
    Update uncovered[j]: remove [cp, dp]
    Update assignments[i]: add (p_j, cp, dp)
    total ← total + saving
until no candidates
return X, total
```

3) Walkthrough on Toy Example: Applying Algorithm 1 to Figure 1:

Iteration 1: Candidates are $(a_1, p_2, 50)$, $(a_1, p_1, 45)$, $(a_2, p_2, 40)$, $(a_2, p_3, 30)$. Select (a_1, p_2) with saving 50. Update: p_2's uncovered = $\{[70, 90]\}$.

Iteration 2: Candidates include $(a_1, p_1, 45)$, $(a_2, p_2, 20)$ (from uncovered [70,90]), $(a_2, p_3, 30)$. Select (a_1, p_1) with saving 45. Now a_1 has 2 PVs on [20,60]—at capacity.

Iteration 3: a_1 full on [20,60] but free on [60,70]. Remaining candidates: $(a_2, p_3, 30)$, $(a_2, p_2, 20)$. Select (a_2, p_3).

Iteration 4: Select $(a_2, p_2, 20)$ for segment [70,90].

Result: Total saving = 50 + 45 + 30 + 20 = 145.

4) Complexity Analysis: Each iteration generates $O(NM)$ candidates and sorts them in $O(NM \log(NM))$. The number of iterations is bounded by the total number of segment assignments, which is $O(NM)$ in the worst case. Thus, worst-case complexity is $O((NM)^2 \log(NM))$. In practice, convergence is much faster as many PVs are fully covered early.

B. Enhanced Hungarian Algorithm with Capacity Constraints

1) Key Insight: The Hungarian algorithm [9] solves bipartite matching optimally in $O(n^3)$ time. However, our problem differs in two ways: (1) AVs can match multiple PVs (capacity > 1), and (2) multi-segment matching creates dependencies between iterations. We address these via iterative application with state tracking.

2) Virtual Slot Expansion: To handle capacity, we expand each AV a_i into C_i "virtual slots." This transforms the many-to-many matching into a standard bipartite matching:

- **Left nodes:** $\sum_{i=1}^{N} C_i$ virtual slots
- **Right nodes:** M PVs (or their uncovered segments)
- **Edge weight:** $-S_{ij}$ (negative for minimization)

3) Cost Matrix Construction: The cost matrix $\mathbf{W}$ is defined as:

$$W_{slot,j} = \begin{cases} S_{max} - S_{ij} + 1 & \text{if feasible} \\ M_{big} & \text{otherwise} \end{cases} \tag{11}$$

where $S_{max} = \max_{i,j} S_{ij}$ and M_{big} is a large constant preventing infeasible assignments.

4) Iterative Algorithm: Algorithm 2 applies Hungarian matching iteratively, updating PV segment states after each round until no further assignments are possible.

5) Walkthrough on Toy Example: **Iteration 1:** Cost matrix with $n_{slots} = 4$ (2 per AV). Hungarian finds optimal matching: slot $(a_1, 1) \to p_2$ (50), slot $(a_1, 2) \to p_1$ (45), slot $(a_2, 1) \to p_3$ (30). Total = 125.

Iteration 2: p_2 has uncovered [70,90]. Rebuild matrix. Hungarian assigns $(a_2, 2) \to p_2$ for segment [70,90] (saving 20).

Result: Total = 125 + 20 = 145 (same as Greedy in this case, but Hungarian guarantees optimality).

6) Complexity Analysis: Each Hungarian call has complexity $O(n^3)$ where $n = \max(\sum_i C_i, M)$. The number of iterations is bounded by the number of segments created. Total complexity is $O(K \cdot n^3)$ where K is the number of iterations.

C. Theoretical Analysis

Theorem 1. *Algorithm 1 achieves a $\frac{1}{2}$-approximation for the platoon matching problem when restricted to single-segment assignments per PV.*

Algorithm 2 Iterative Hungarian Multi-AV Matching

Require: AVs $\mathcal{A}$, PVs $\mathcal{P}$, minimum length L_{min}
Ensure: Assignments $\mathcal{X}$, total saving

Initialize $uncovered$, $assignments$ as in Alg. 1
$\mathcal{X} \leftarrow \emptyset$, $total \leftarrow 0$
repeat
    // Build cost matrix
    $n_{slots} \leftarrow \sum_i C_i$
    $\mathbf{W} \leftarrow M_{big} \cdot \mathbf{1}_{n_{slots} \times M}$
    for each AV a_i, slot $s \in \{1, \ldots, C_i\}$ **do**
        for each PV p_j with uncovered segments **do**
            $[cp, dp] \leftarrow$ best overlap
            if $dp - cp \geq L_{min}$ **and** capacity OK **then**
                $W_{(i,s),j} \leftarrow S_{max} - (dp - cp) + 1$
            end if
        end for
    end for
    if all entries are M_{big} **then**
        break
    end if
    // Solve Hungarian
    $matching \leftarrow$ HUNGARIAN($\mathbf{W}$)
    $applied \leftarrow 0$
    for each $(slot, j) \in matching$ **do**
        if $W_{slot,j} < M_{big}$ **then**
            Apply assignment, update states
            $applied \leftarrow applied + 1$
        end if
    end for
    if $applied = 0$ **then**
        break
    end if
until convergence
return $\mathcal{X}$, $total$

Proof. Under single-segment restriction, the problem reduces to weighted bipartite matching with capacity constraints. This is a special case of submodular maximization subject to partition matroid constraints, for which greedy achieves $\frac{1}{2}$-approximation [11]. □

Theorem 2. *Algorithm 2 computes an optimal assignment within each iteration, and the iterative process converges to a locally optimal solution.*

Proof. The Hungarian algorithm guarantees optimal bipartite matching for the cost matrix at each iteration. Since PV states are updated monotonically (segments only shrink), the algorithm terminates. Local optimality follows from the fact that no single reassignment can improve the objective after convergence. □

IV. Experimental Evaluation

A. Experimental Setup

We evaluate both algorithms on synthetic highway scenarios with varying configurations. All experiments were conducted on a machine with Intel Core i7 processor and 16GB RAM, implemented in Python 3.9 using NumPy and SciPy [12].

Scenario Generation:

- Highway length: $L \in \{50, 100, 150, 200\}$ distance units
- Number of AVs: $N \in \{3, 5, 10, 15, 20\}$
- Number of PVs: $M \in \{5, 10, 20, 30, 50\}$
- AV capacity: $C \in \{1, 2, 3, 4, 5\}$ (uniform)
- Entry/exit points: Uniformly distributed along highway
- Minimum shared distance: $L_{min} = 5$
- Random seed: Fixed for reproducibility

Metrics:

- *Total Energy Saving*: $\sum$ saved distances across all assignments
- *Match Ratio*: Fraction of PVs receiving ≥ 1 assignment
- *Coverage Ratio*: Average fraction of PV routes covered
- *Runtime*: Algorithm execution time (ms)
- *Optimality Gap*: $(Hungarian - Greedy)/Hungarian \times 100\%$

B. Results

1) Capacity Sweep: Table II shows algorithm performance as AV capacity varies. Higher capacity enables more assignments, benefiting both algorithms.

TABLE II
PERFORMANCE VS. AV CAPACITY ($N = 5$, $M = 15$, $L = 100$)

Cap.	**Greedy**		**Hungarian**		**Gap**
C	Saving	Time	Saving	Time	(%)
1	142.3	3.2	147.1	45.1	3.3
2	198.7	4.1	205.2	67.3	3.2
3	231.4	5.3	238.9	89.2	3.1
4	252.1	6.1	259.3	112.4	2.8
5	268.5	6.8	274.7	135.7	2.3

2) Highway Length Sweep: Table III shows performance as highway length varies. Longer highways provide more overlap opportunities.

TABLE III
PERFORMANCE VS. HIGHWAY LENGTH ($N = 5$, $M = 15$, $C = 3$)

Length	**Greedy**		**Hungarian**		**Gap**
L	Saving	Time	Saving	Time	(%)
50	89.2	2.8	92.1	38.2	3.2
100	231.4	5.3	238.9	89.2	3.1
150	387.6	8.1	398.4	156.3	2.7
200	521.3	11.2	535.8	234.7	2.7

3) Vehicle Density Sweep: Table IV shows performance as PV count varies. More PVs increase competition for AV capacity.

C. Discussion

Near-Optimal Greedy Performance: The Greedy algorithm consistently achieves within 2.2–3.3% of optimal, validating Theorem 1's approximation guarantee in practice. The gap decreases with higher capacity, as more slack reduces the impact of greedy mistakes.

TABLE IV
PERFORMANCE VS. PV COUNT ($N = 5$, $C = 3$, $L = 100$)

PVs	**Greedy**		**Hungarian**		**Gap**
M	Saving	Time	Saving	Time	(%)
5	78.4	1.9	80.2	21.3	2.2
10	156.2	3.4	161.5	52.1	3.3
15	231.4	5.3	238.9	89.2	3.1
20	298.1	7.8	306.7	142.6	2.8
30	412.3	12.1	423.8	287.4	2.7

Runtime Trade-off: Hungarian is 10–25$\times$ slower than Greedy. For real-time systems requiring decisions in $<$10ms, Greedy is the practical choice. For offline planning with optimality requirements, Hungarian is preferable.

Multi-Segment Benefits: Comparing single-segment vs. multi-segment matching (not shown in tables), multi-segment improves total savings by 15–25% by enabling PVs to utilize multiple AVs across their routes.

Scalability: Greedy scales to 50+ PVs with sub-linear runtime growth. Hungarian's $O(n^3)$ per-iteration cost becomes prohibitive beyond 30 PVs in real-time settings.

V. CONCLUSION

We have presented a novel formulation for dynamic platoon formation in heterogeneous autonomous vehicle systems, where Passive Vehicles can be physically towed by Active Vehicles to achieve substantial energy savings. Our one-dimensional highway model captures the essential optimization trade-offs—capacity constraints, multi-segment matching, and minimum distance requirements—while remaining computationally tractable.

Two complementary algorithms were proposed: a Greedy Maximum-Weight Matching algorithm for real-time applications ($O(NM \log NM)$, within 3% of optimal), and an Enhanced Hungarian Algorithm for optimal offline planning ($O(n^3)$ per iteration). The key innovation of multi-segment matching enables PVs to leverage multiple AVs across their routes, improving system-wide efficiency by 15–25%.

Future Work: Several extensions warrant investigation:

- *2D Road Networks*: Extending from 1D highways to general graphs with intersections and routing decisions.
- *Stochastic Arrivals*: Handling uncertain vehicle entry times and routes using online algorithms.
- *Decentralized Coordination*: Developing V2V protocols where vehicles autonomously negotiate platoon formations without centralized control.
- *Economic Mechanisms*: Designing pricing and incentive schemes for fair cost distribution among participants.

REFERENCES

[1] U.S. Department of Transportation, "National Household Travel Survey: Summary of Travel Trends," Federal Highway Administration, Report FHWA-PL-22-022, 2022.

[2] U.S. Environmental Protection Agency, "Inventory of U.S. Greenhouse Gas Emissions and Sinks: 1990–2021," EPA 430-R-23-002, 2023.

[3] S. Tsugawa, S. Jeschke, and S. E. Shladover, "A Review of Truck Platooning Projects for Energy Savings," *IEEE Trans. Intell. Veh.*, vol. 1, no. 1, pp. 68–77, 2016.
[4] C. Bergenhem, S. Shladover, E. Coelingh, C. Englund, and S. Tsugawa, "Overview of platooning systems," in *Proc. 19th ITS World Congress*, Vienna, Austria, 2012.
[5] M. Furuhata, M. Dessouky, F. Ordóñez, M.-E. Brunet, X. Wang, and S. Koenig, "Ridesharing: The state-of-the-art and future directions," *Transport. Res. Part B: Methodol.*, vol. 57, pp. 28–46, 2013.
[6] N. Agatz, A. Erera, M. Savelsbergh, and X. Wang, "Optimization for dynamic ride-sharing: A review," *European J. Oper. Res.*, vol. 223, no. 2, pp. 295–303, 2012.
[7] A. Sciarretta and A. Vahidi, "Energy-Efficient Driving of Road Vehicles," in *Energy-Efficient Driving of Road Vehicles*, Springer, 2020, pp. 1–30.
[8] E. Hellström, M. Ivarsson, J. Åslund, and L. Nielsen, "Look-ahead control for heavy trucks to minimize trip time and fuel consumption," *Control Eng. Practice*, vol. 17, no. 2, pp. 245–254, 2009.
[9] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Res. Logistics Quarterly*, vol. 2, no. 1–2, pp. 83–97, 1955.
[10] J. Munkres, "Algorithms for the assignment and transportation problems," *J. Society Indust. Appl. Math.*, vol. 5, no. 1, pp. 32–38, 1957.
[11] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey, "An analysis of approximations for maximizing submodular set functions—II," *Math. Programming Study*, vol. 8, pp. 73–87, 1978.
[12] P. Virtanen et al., "SciPy 1.0: Fundamental algorithms for scientific computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.