ManeuverGPT
Agentic Control for Safe Autonomous Stunt Maneuvers

Shawn Azdam[2,4], Pranav Doma[1,2], and Aliasghar Moj Arab[1,2,3]

***Abstract*—The next generation of active safety features in autonomous vehicles should be capable of safely executing evasive hazard-avoidance maneuvers to achieve rapid motion at the limits of vehicle handling.**

This paper presents a novel framework, *ManeuverGPT*, for generating and executing highly dynamic stunt maneuvers in autonomous vehicles using large language model (LLM)-based agents as controllers. We target aggressive maneuvers, such as J-turns, within the CARLA simulation environment and demonstrate an iterative, prompt-based approach to refine vehicle control parameters, starting *tabula rasa* without retraining model weights. We propose an *agentic* architecture composed of three specialized agents: (1) a *Query Enricher Agent* for contextualizing user commands, (2) a *Driver Agent* for generating maneuver parameters, and (3) a *Parameter Validator Agent* that enforces physics-based and safety constraints. Experimental results demonstrate successful J-turn execution across multiple vehicle models through textual prompts that adapt to differing vehicle dynamics. We evaluate performance via established success criteria and discuss limitations regarding numeric precision and scenario complexity. Our findings underscore the potential of LLM-driven control for high-agility maneuvers, while highlighting the importance of hybrid approaches that combine language-based reasoning with algorithmic validation. We provide an open-source implementation at `https://github.com/SHi-ON/ManeuverGPT` to foster further research within the broader community.

I. Introduction

Autonomous vehicles (AVs) are advancing swiftly, offering potential benefits such as reduced traffic congestion, lower accident rates, and enhanced mobility. Integrating human-inspired active safety features derived from evasive hazard avoidance maneuvers, enables agile motion at the edge of handling limits to support the development of next-generation "accident-free" vehicles. Despite these advantages, automated driving still faces critical challenges in executing extreme maneuvers under uncertain and varying conditions.

One representative example is the J-turn—a rapid 180° rotation of the vehicle at speed, as shown in Figure 1. Safely executing this maneuver is especially valuable in emergency scenarios where blockage of the planned trajectory necessitates immediate reversal of direction. Such situations may arise in accident avoidance, emergency evacuation, or sudden road closures [1], [2]. However, designing a reliable controller to perform such a maneuver is challenging due to the complex dynamics involved and the narrow margin for error. To address these challenges, we developed an agentic architecture that incorporates foundation models [3] to assist in controllers for stunt maneuvers, which we call ManeuverGPT.

Traditional approaches to such maneuvers often rely on carefully tuned controllers or reinforcement learning (RL) methods that require extensive training data, environment interactions, and domain-specific engineering [4], [5]. Moreover, adapting to new vehicle dynamics can demand significant retraining or model redesign [6], [7]. Recent advances in Large Language Models (LLMs) offer an alternative strategy for finding control policies for complex maneuvers without extensive retraining. Pre-trained LLMs have shown promising results in planning, code synthesis, and robotic instruction following [8], [9].

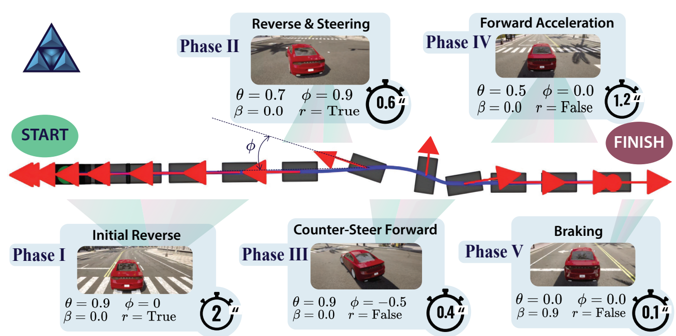

Fig. 1. The five-phase J-turn maneuver executed by our ManeuverGPT controller. Phase I begins with an initial reverse, followed by reverse steering (Phase II), counter-steering forward (Phase III), forward acceleration (Phase IV), and concludes with braking (Phase V). Throttle (θ), steering (ϕ), brake (β), and reverse (r) control parameters are annotated throughout, culminating in a 180° turnaround.

In this work, we explore whether LLM agents can generate and refine *control commands* for stunt driving maneuvers by iteratively adjusting key vehicle parameters and textual prompts to account for variations in dynamics and environmental conditions, starting tabula rasa. We propose a novel framework, ManeuverGPT, that incorporates three specialized LLM-driven agents in a closed-loop architecture.

The main contributions of this work are threefold: (1) a novel multi-agent LLM-driven framework capable of generating constraint-satisfying control parameters without reliance on gradient-based optimization; (2) an adaptive cross-vehicle generalization mechanism that consistently achieves high suc-

[1]Department of Mechanical and Aerospace Engineering, New York University, New York, NY 10012, USA.

[2]Agile Safe Autonomous Systems (ASAS) Lab, Tandon School of Engineering, New York University, Brooklyn, NY 11201, USA.

[3]General Autonomy Inc., 201 Centennial Ave. Piscataway, NJ, 08854, USA.

[4]Azdam AI, New York, NY, USA.

Emails: `shawn@azdam.ai`, `pd2365@nyu.edu`, `aliasghar.arab@nyu.edu`, `mojarab@genauto.ai`.

cess rates across diverse vehicle dynamics using only textual parameter adjustment; and (3) a comprehensive performance evaluation based on established metrics including final orientation accuracy, collision checks, and time constraints.

We also discuss limitations related to numeric precision and scenario complexity, which point to the need for hybrid methods that combine language-based reasoning with conventional control theory.

The remainder of this paper is organized as follows. We review related work on LLMs in robotics and agentic architectures for autonomous vehicles. We then formally describe our framework, including its theoretical underpinnings and an iterative validation pipeline. Experimental results in CARLA demonstrate how prompt-based adjustments suffice to achieve feasible execution across different vehicle models. We conclude by highlighting open research directions, particularly in bridging simulation-to-reality gaps and ensuring robust safety guarantees for high-speed maneuvers.

We focus on the J-turn maneuver [10], [11] in the CARLA simulation environment [12] and evaluate how textual refinements alone (as opposed to gradient-based learning) can produce successful 180° reorientations within specified time limits and with collision-free execution.

II. Related Work

Recent advances in large language models (LLMs) have spurred research into their application for robotics control [13]. Several works have demonstrated that LLMs can extract actionable knowledge for embodied agents, enabling zero-shot planning and reasoning in complex scenarios [14], [15].

LanguageMPC [16] showed direct translation of linguistic decisions to model predictive control (MPC) parameters, reducing navigation costs compared to conventional controllers. The LLM4AD architecture [17], showed how natural language commands can be transformed into vehicle control parameters through structured reasoning processes—a conceptual approach our work extends to the domain of highly dynamic maneuvers.

Traditional MPC techniques, as applied in autonomous driving systems [18], offer strong guarantees in terms of constraint handling and optimality, but they rely heavily on precise dynamic models that may not capture complex driving behaviors. Arab [19] demonstrated MPC's capability for extreme maneuvers through a sparse stable-trees algorithm, achieving high-agility maneuvers in 1/7-scale vehicle tests while maintaining safety via augmented stability regions. In contrast, deep RL methods have shown promising results for aggressive maneuvers such as drifting and J-turns [6], yet they often lack formal safety guarantees.

Emerging hybrid approaches combine linguistic reasoning with physical constraints. Chen et al. [20] developed AsyncDriver with decoupled 2 Hz LLM/20 Hz planner operation, reducing the computational overhead by 63%. Long et al. [21] integrated VLMs with MPC to improve safety margins by 38% in adverse conditions through visual-language parameter generation.

In summary, while each of these paradigms—LLM-based control, MPC, and RL—has its own advantages, our work combines the adaptability of LLMs with rigorous safety checks through phase-optimized parameter generation and multi-stage validation, advancing beyond existing hybrid approaches [16], [20].

III. Methodology

Our framework, illustrated in Fig. 2, comprises three collaborative LLM-driven agents coordinated by a central unit. User input is enriched and can be serialized as a JSON dictionary:


```
{"maneuver_phase":1, "vehicle_spec":
  {"mass":1500,"drive":"RWD"}, "safety_bound":
   {"max_steer":0.9},"env":"dry asphalt"}}
```


The **Query Enricher Agent** ($\mathcal{A}_E$) interprets the initial user input and augments it by referencing a curated knowledge base containing historical runs and manufacturer specifications, yielding an enriched query $E(Q)$. This explicit structure enables the **Driver Agent** ($\mathcal{A}_D$) to perform key–value reasoning to synthesize a sequence of maneuver parameters P (e.g., throttle, steer, brake, etc.). This process generates candidate plans that are aligned with the intended stunt maneuver while also being calibrated for precision and adaptability. Subsequently, the candidate parameter set is subjected to evaluation by the **Parameter Validator Agent** ($\mathcal{A}_V$), which enforces all relevant safety requirements and operational constraints—including permissible parameter ranges and tire force envelope adherence—formally represented as the constraint set $\mathcal{C} = \{C_s, C_o\}$.

A central orchestration unit manages agent interactions in a closed-loop process. If the Parameter Validator identifies issues, the system triggers re-evaluation through the Driver Agent or requests clarification from the Query Enricher. This feedback mechanism, informed by both validation and simulator results, ensures that all generated maneuvers consistently meet the safety and performance standards.

To formally capture the stunt maneuvers addressed by our agents, let the state of the autonomous vehicle at time t be denoted by $\boldsymbol{x}(t) \in \mathcal{X} \subseteq \mathbb{R}^n$, where $\boldsymbol{x}$ is the state-space vector of dimension n and $\mathcal{X}$ is the feasible state space. Vehicle maneuvers are controlled through the input vector $\boldsymbol{u}(t)$ as

$$\boldsymbol{u}(t) = [\theta(t), \phi(t), \beta(t), r(t)] \in \mathcal{U} \tag{1}$$

where θ is the normalized throttle rate, ϕ is the normalized steering rate, β is the normalized brake intensity, and r indicates reverse status. The vehicle dynamics are represented as the nonlinear system

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}\big(\boldsymbol{x}(t), \boldsymbol{u}(t)\big) \tag{2}$$

with $\boldsymbol{f}(\cdot)$ capturing the kinematic, dynamic, and tire-road interaction effects studied in our earlier work [19]. We assume $\boldsymbol{f}$ is continuous and locally Lipschitz in both $\boldsymbol{x}$ and $\boldsymbol{u}$.

The velocity components of state $\boldsymbol{x}(t)$ include longitudinal velocity $\boldsymbol{v}_x$, lateral velocity $\boldsymbol{v}_y$, and rotational velocity $\boldsymbol{\omega}$. These components follow coupled differential relationships

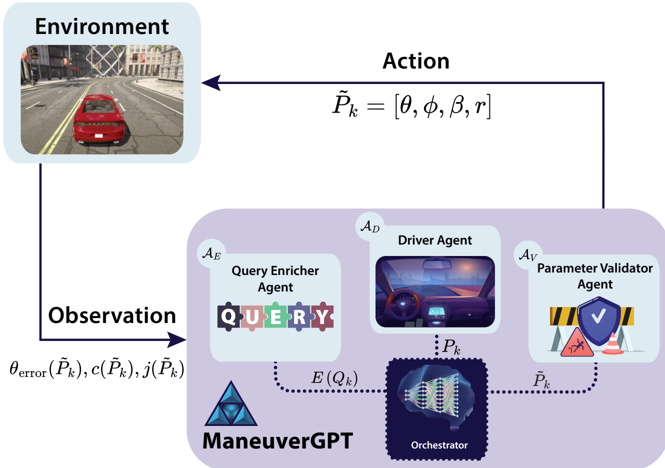

Fig. 2. Overview of the modular agentic framework comprising three agents and an orchestrator. The forward path represents the flow of action parameters $\tilde{P}_k$ to the environment, while the feedback path conveys observations—heading deviation θ_{error}, collision indicator c, and jerk j—which inform subsequent agent reasoning and control refinement.

that can be expressed as specific components of the general function $\boldsymbol{f}$ [22]:

$$\dot{\boldsymbol{v}}_x = \boldsymbol{a}_x = \boldsymbol{f}_v(\theta, \beta, r) - \boldsymbol{\omega}\boldsymbol{v}_y \tag{3}$$
$$\dot{\boldsymbol{v}}_y = \boldsymbol{a}_y = \boldsymbol{f}_l(\phi, \boldsymbol{v}_x) + \boldsymbol{\omega}\boldsymbol{v}_x \tag{4}$$
$$\dot{\boldsymbol{\omega}} = \boldsymbol{\alpha} = \boldsymbol{f}_r(\phi, \boldsymbol{v}_x, \boldsymbol{v}_y) \tag{5}$$

where $\boldsymbol{f}_v$, $\boldsymbol{f}_l$, and $\boldsymbol{f}_r$ are the component functions of $\boldsymbol{f}$ that govern longitudinal, lateral, and rotational accelerations, respectively.

A stunt maneuver is *feasible* if it satisfies:

- **Physical Constraints:** Steering angles, throttle, brake, and reverse commands remain within manufacturer or simulation limits.
- **Safety Constraints:** No collisions occur during execution.
- **Performance Criteria:** The final state satisfies goal conditions (e.g., finishing a J-turn at $180° \pm \Delta_\theta$).

We define a discrete iteration $k \in \{1, 2, \dots\}$ as follows:

1) Generate parameters $P_k = \mathcal{A}_D\big(E(Q_k)\big)$.
2) Validate P_k to obtain $\tilde{P}_k$ via $\mathcal{A}_V$, ensuring that constraints are met.
3) Run a simulation with $\tilde{P}_k$ to measure performance metrics (heading angle error, collisions, etc.).
4) Produce feedback Q_{k+1} based on the measured performance.

This yields a feedback-driven sequence $\{Q_k\}$ and $\{P_k\}$.

In each iteration, we evaluate how well $\tilde{P}_k$ meets stunt objectives via the cost function

$$L(\tilde{P}_k) \;=\; \alpha_1 \left|\theta_{\text{error}}(\tilde{P}_k)\right| \;+\; \alpha_2\, c(\tilde{P}_k) \;+\; \alpha_3\, j(\tilde{P}_k) \tag{6}$$

where $\theta_{\text{error}}(\tilde{P}_k)$ is the final heading deviation, $c(\tilde{P}_k) \in \{0, 1\}$ indicates whether a collision occurred, $j(\tilde{P}_k)$ is a measure of jerk (i.e., derivative of acceleration), and $\alpha_{1,2,3} \geq 0$ are weighting coefficients. Minimizing $L(\tilde{P}_k)$ aims to achieve precise reorientation, avoid collisions, and maintain smooth maneuvers.

Some reinforcement learning approaches define a reward $R(\tilde{P}_k)$ to be maximized rather than a cost to be minimized. One possible definition is:

$$R(\tilde{P}_k) = \alpha_1 \left(180° - |\theta_{\text{error}}(\tilde{P}_k)|\right) + \alpha_2 \left(1 - c(\tilde{P}_k)\right) - \alpha_3\, j(\tilde{P}_k) \tag{7}$$

using the same weighting coefficients $\alpha_{1,2,3}$ as in the cost function. In this formulation, increasing R is effectively equivalent to decreasing L. Our iterative algorithm remains cost-based but could be adapted to reward-based methods by taking $L \propto -R$.

Algorithm 1: ManeuverGPT

Input : - User command Q_1
- Constraints $\mathcal{C} = \{C_s, C_o\}$
- Maximum iterations $k_{\max}$
- Cost threshold ε

Output: Feasible parameter set $\tilde{P}_k$ (or best-effort parameters)

$k \leftarrow 1$
$\tilde{P}_{\text{best}} \leftarrow \emptyset$; $L_{\text{best}} \leftarrow \infty$
while $k \leq k_{\max}$ **do**
  $E(Q_k) \leftarrow \mathcal{A}_E(Q_k)$
  $P_k \leftarrow \mathcal{A}_D\big(E(Q_k)\big)$
  $\tilde{P}_k \leftarrow \mathcal{A}_V(P_k)$
  if $L(\tilde{P}_k) \leq L_{best}$ **then**
    $\tilde{P}_{\text{best}} \leftarrow \tilde{P}_k$; $L_{\text{best}} \leftarrow L(\tilde{P}_k)$
  end
  if $L(\tilde{P}_k) \leq \varepsilon$ **then**
    return $\tilde{P}_k$ `// Satisfactory solution`
  end
  else
    $Q_{k+1} \leftarrow \text{Feedback}(Q_k, \theta_{\text{error}}(\tilde{P}_k), c(\tilde{P}_k), j(\tilde{P}_k))$
    $k \leftarrow k + 1$
  end
end
return $\tilde{P}_{\text{best}}$ `// Best-effort solution`

Theorem 1 (Finite-Time Feasibility)**.** *Suppose that:*

1) *The validation operation $\mathcal{A}_V$ ensures $\tilde{P}_k$ remains in a compact set $\mathcal{U}_{\text{safe}}$.*
2) *The feedback reduces $L(\tilde{P}_k)$ by at least a constant $\delta > 0$ whenever $L(\tilde{P}_k) > \varepsilon > 0$.*
3) *The environment simulation is deterministic with respect to $(x_0, \tilde{P}_k)$.*

Then, there exists an integer K such that, for all $k \geq K$, the parameter set $\tilde{P}_k$ is feasible (i.e., satisfies stunt requirements), or the user terminates the procedure after a finite number of iterations.

Proof. Let $L(\tilde{P}_k)$ be nonnegative. Each iteration reduces $L(\tilde{P}_k)$ by at least δ when $L(\tilde{P}_k) > \varepsilon$. Since L is bounded below by zero, a simple monotonicity argument shows it must reach feasibility (i.e., drop below the ε threshold) within a finite number of steps, or else the user halts after some finite k. □

In our framework, the cost reduction guarantee in assumption 2 is realized through structured prompt refinement. When $L(\tilde{P}_k) > \varepsilon$, the Driver Agent ($\mathcal{A}_D$) receives specific feedback about performance gaps (e.g., "reduce steering angle by 5-10% to minimize overshoot"). This targeted feedback, combined with the Parameter Validator's constraints, ensures progressive improvement in subsequent iterations. For example, prompt refinements might include:

> *Initial prompt:* "Execute a J-turn maneuver."
> *Refined prompt after feedback:* "Execute a J-turn maneuver with gentler steering during phase 2 (currently overshooting by 12°) and increase brake intensity to 0.7 during final stabilization."

This structure guarantees that each iteration addresses specific performance deficits, ensuring the δ improvement.

Remark: Theorem 1 guarantees feasibility but not *optimality* in the control-theoretic sense (e.g., minimal torque or minimal time). It establishes that our iterative prompting approach can identify constraint-satisfying solutions under mild assumptions, which aligns with our goal of achieving safe executable maneuvers rather than mathematically optimal trajectories. This synergy between language-based generation and physically motivated cost functions is validated empirically in the next section.

IV. Experimental Setup

Experiments were conducted in the CARLA simulator (v0.9.14), which provides high-fidelity vehicle dynamics and sensor modeling for testing complex maneuvers like J-turns.

Performance was quantified using the following metrics: **Angle Error** is defined as the difference between the achieved turn angle and the ideal 180° (Optimal: $\leq 3°$, Acceptable: $\leq 10°$); **Success Rate** denotes the percentage of trials with an angle error below 10°; **Yaw Rate** represents the angular velocity around the vertical axis (degrees per second); **Jerk** quantifies the rate of change of acceleration as a proxy measure of smoothness; **Steering Smoothness** is given by the inverse of the mean absolute yaw changes; **Execution Time** is the duration required to complete the maneuver; and **Collision Detection** is a binary metric assessing collision-free operation. These metrics provide a multifaceted evaluation of both technical performance and real-world applicability.

Our system architecture integrates a GPT-family model as the core of the language agents. The agents exchange and share message context through an orchestrator component built on top of an in-memory database for asynchronous processing. We adopt a phase-based control protocol that plans complete trajectories rather than making frame-by-frame decisions. This approach enables high-level maneuver planning while maintaining computational efficiency and responsiveness.

To ensure reliable and safe operation, we implemented multiple protective mechanisms throughout the experimental framework. Control parameter validation constrained all inputs within physical limits (throttle/brake: [0,1], steering: [-1,1]). Prompt-based safety constraints provided explicit instructions for vehicle stability during maneuvers. We implemented collision detection that terminates trials upon impact and provides negative feedback to the model. We also repeat the experiment from identical initial states across multiple runs to test iterative improvement.

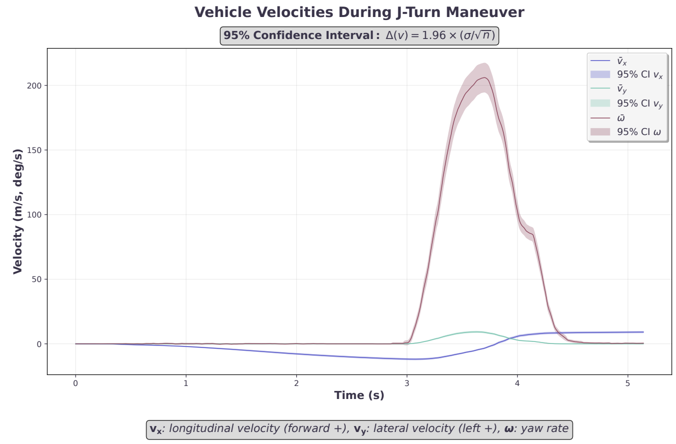

Fig. 3. Time series of vehicle velocities during a J-turn maneuver, showing longitudinal (v_x in m/s), lateral (v_y in m/s), and rotational (ω in deg/s) velocity components. The confidence interval (CI) range is computed as $\Delta(v) = 1.96 \times (\sigma/\sqrt{n})$, where σ is the standard deviation and n is the number of trials.

V. Results and Discussion

Our investigation of LLM-based controllers for J-turn maneuvers revealed insights across vehicle dynamics and architecture adaptability. The controller achieved an overall success rate of 86% across 100 runs, improving from 83.3% in early trials to 90% in later trials (Table I).

TABLE I
Parameter Execution Performance (Averaged over 100 Runs)

Batch	Total Parameters	Implemented	Rejected	Success (%)
Overall	100	86	14	86.0%
Early (first 60)	60	50	10	83.3%
Later (last 40)	40	36	4	90.0%

The velocity profile in Figure 3 shows that, during acceleration (0–1.5 s), the controller increases longitudinal velocity to about 15 m/s with minimal lateral or rotational velocity. Turn initiation (1.5–3 s) brings in lateral velocity, while the maximum rotation phase (3–4 s) shows peak rotational velocity, maximum lateral velocity, and a drop in longitudinal velocity marking the main directional change. Stabilization (4–5 s) exhibits damping oscillations as the controller returns longitudinal velocity to around 12 m/s. Confidence intervals for velocities indicate consistent performance, except for higher rotational velocity variability due to traction conditions and control timing differences.

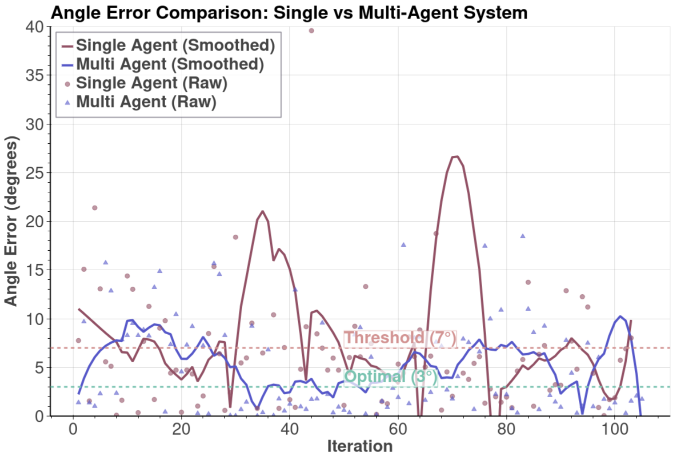

Fig. 4. Angle error comparison between single-agent and multi-agent systems. The raw and smoothed angle errors are plotted for both systems across multiple trials. The 7° threshold (dashed purple line) represents the acceptable error limit, while the 3° optimal error (dashed green line) indicates the desired accuracy.

Our multi-agent architecture substantially outperformed a single-agent implementation, where a single LLM agent performed all three functions (query enrichment, parameter generation, and validation) without specialized role separation. For this architectural comparison, we used a more stringent intermediate threshold of 7° (rather than the 10° used in vehicle comparisons), as both implementations tested the same vehicle type (sedan). The multi-agent system maintained the error below 7° for 76% of the trials, compared to only 52% with the single-agent system. Furthermore, the multi-agent system achieved optimal performance (less than 3°) in 31% of the trials, versus 18% for the single-agent approach.

As shown in Figure 4, appropriate task decomposition enhances control precision and reliability. The smoothed trend lines indicate that the multi-agent system maintains more stable performance over time, whereas the single-agent system exhibits greater variability and some regression in later trials. The Validator imposes guardrails that reduce the single-agent's tendency to drift in parameter selection. For instance, we observed that, without the Validator, the single-agent sometimes selects steering angles > 1.0, leading to immediate collisions.

The controller also demonstrated varying effectiveness across different vehicle dynamics. Table II shows that the sedan consistently outperformed the sports coupe across nearly all metrics, with 62.7% lower mean angle error (9.09° vs. 21.39°) and 29.5% higher success rate (90.11% vs. 70.57%).

Figure 5 shows that both vehicles achieved near-perfect turns in their best trials, but the sports coupe showed substantially higher variability (SD: 44.24 vs. 24.04), although it also approached optimal thresholds in later iterations. The sedan maintained performance below the 10° threshold more consistently, while the sports coupe exhibited more frequent excursions beyond acceptable limits.

Figure 6 shows the learning progress over 5 batches (epochs) of 20 iterations. The controllers' performance gradually converges toward the optimal error level (3°) and remains below the 10° threshold as training progresses. Each iteration takes roughly five seconds using API calls, making the learning process practical for online use.

TABLE II
PERFORMANCE COMPARISON BETWEEN SEDAN AND SPORTS COUPE MODELS

Metric	Sedan	Sports Coupe
Mean Angle Error (°)	9.09	21.39
Median Angle Error (°)	5.35	6.99
Min Angle Error (°)	0.00	0.12
Max Angle Error (°)	179.98	178.78
Standard Deviation	24.04	44.24
Success Rate (%)	90.11	70.57
Mean Jerk (m/s^3)	0.81	1.26
Avg Max Jerk (m/s^3)	48.31	55.32
Mean Yaw Rate (°/s)	16.14	13.82
Steering Smoothness	0.39	0.44
Avg Execution Time (s)	264.00	292.17

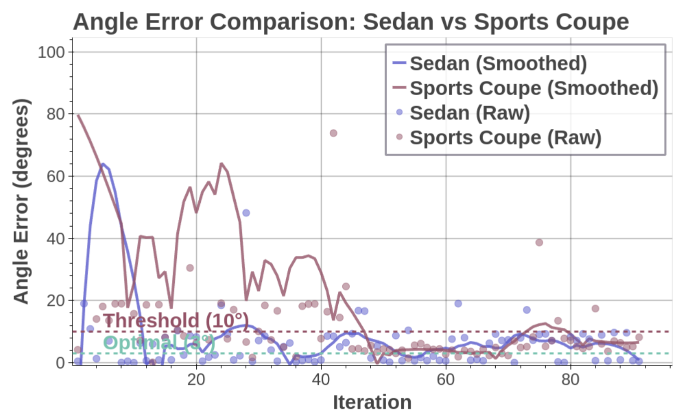

Fig. 5. Comparison of angle error between sedan and sports coupe vehicle models during J-turn maneuvers. The angle error is calculated as the absolute difference between the actual turn angle and the ideal 180-degree turn. Lower values indicate better performance, with 3° considered optimal (green line) and 10° as the maximum acceptable threshold (red line). The smoothed lines represent the trend using a Savitzky-Golay filter, while individual data points show raw measurements. Angle error $\theta_e = |\theta_{\text{final}} - \theta_{\text{initial}} - 180°|$ consistently achieves sub-10° precision in the simulation environment.

These differences highlight a key insight: while our multi-agent architecture can adapt to different vehicle dynamics, its effectiveness varies based on inherent stability characteristics. The sedan's more forgiving dynamics allow greater error margins in control parameters, while the sports coupe's higher responsiveness amplifies small control errors into larger outcome differences.

The sports coupe's shorter wheelbase, higher power-to-weight ratio, and rear-biased weight distribution make it more responsive but also more challenging to control precisely during highly dynamic maneuvers, explaining the performance disparity despite the controller's adaptive capabilities.

VI. Conclusion

We have presented ManeuverGPT, a multi-agent framework that utilizes LLMs for generating and refining aggressive

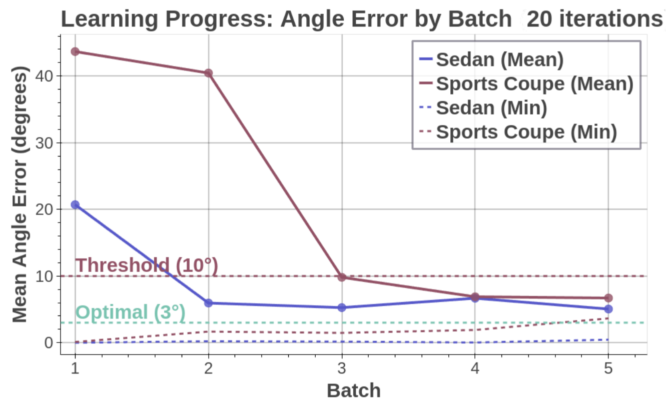

Fig. 6. Learning progress of the steering controller for sedan and sports coupe vehicles across batches of 20 iterations. The plot shows the mean angle error (solid lines) and minimum angle error (dashed lines) for each batch, demonstrating how controller performance improves over time. The horizontal dashed lines at 3° and 10° represent the optimal and maximum acceptable error thresholds, respectively.

stunt maneuvers such as J-turns in the CARLA simulator. Our research demonstrates that LLM-based controllers can effectively plan and execute complex vehicle maneuvers through iterative feedback without requiring modification of internal parameters. Our findings reveal that: (1) multi-agent architectures outperform single-agent implementations by 46% in optimal execution rate; (2) the controller adapts to different vehicle dynamics, achieving 90.11% success with sedans and 70.57% with more challenging sports coupes; and (3) performance improves through structured feedback, with implementation success increasing from 83.3% to 90% over successive iterations.

While our prompt-based approach enables the execution of complex maneuvers tabula rasa and avoids retraining model weights, several challenges remain. The current implementation lacks formal safety guarantees that conventional MPC methods provide, and precise numeric control would benefit from hybrid approaches combining LLM reasoning with algorithmic optimization. Additionally, scaling to more complex traffic scenarios and bridging the simulation-to-reality gap both present significant hurdles.

Future work should integrate explicit safety constraints into the LLM prompt context, develop automated re-prompting based on sensor data, and explore synergies between language-based high-level planning and specialized models for reactive control. This research establishes LLM-driven control as a promising approach for rapid prototyping and novel maneuver development when paired with appropriate validation frameworks, potentially expanding the envelope of autonomous vehicle capabilities for safety-critical evasive maneuvers.

Acknowledgments

The authors would like to thank Dr. Bahram Behzadian for his invaluable insights and critical feedback on this manuscript. His thorough review and thoughtful suggestions greatly contributed to the quality of this work.

References

[1] A. Arab, M. Khaleghi, A. Partovi, A. Abbaspour, C. Shinde, Y. Mousavi, V. Azimi, and A. Karimmoddini, "High-resolution safety verification for evasive obstacle avoidance in autonomous vehicles," *IEEE Open Journal of Vehicular Technology*, vol. 6, pp. 276 – 287, 2024.

[2] A. Arab and J. Yi, "Safe agile hazard avoidance system for autonomous vehicles," Jan. 4 2024, uS Patent App. 18/209,943.

[3] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill *et al.*, "On the opportunities and risks of foundation models," *arXiv preprint arXiv:2108.07258*, 2021.

[4] R. Zhao, K. Wang, W. Che, Y. Li, Y. Fan, and F. Gao, "Adaptive cruise control based on safe deep reinforcement learning," *Sensors*, vol. 24, no. 8, p. 2657, 2024.

[5] D. C. Selvaraj, S. Hegde, N. Amati, F. Deflorio, and C. F. Chiasserini, "An ml-aided reinforcement learning approach for challenging vehicle maneuvers," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 2, pp. 1686–1698, 2022.

[6] Y. Chen, C. Ji, Y. Cai, T. Yan, and B. Su, "Deep reinforcement learning in autonomous car path planning and control: A survey," *arXiv preprint arXiv:2404.00340*, 2024.

[7] P. Wang, C.-Y. Chan, and H. Li, "Automated driving maneuvers under interactive environment based on deep reinforcement learning," *arXiv preprint arXiv:1803.09200*, 2018.

[8] X. Jiang, Y. Dong, L. Wang, Z. Fang, Q. Shang, G. Li, Z. Jin, and W. Jiao, "Self-planning code generation with large language models," *ACM Transactions on Software Engineering and Methodology*, vol. 33, no. 7, pp. 1–30, 2024.

[9] B. Quartey, E. Rosen, S. Tellex, and G. Konidaris, "Verifiably following complex robot instructions with foundation models," *arXiv preprint arXiv:2402.11498*, 2024.

[10] A. Arab and J. Yi, "Instructed reinforcement learning control of safe autonomous j-turn vehicle maneuvers," in *2021 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*. IEEE, 2021, pp. 1058–1063.

[11] A. Shahabi, A. H. Kazemian, S. Farahat, and F. Sarhaddi, "Dynamic behavior of the full-car model in the j-turn maneuver by considering the engine gyroscopic effects," *Communications-Scientific letters of the University of Zilina*, vol. 23, no. 3, pp. B237–B249, 2021.

[12] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*. PMLR, 2017, pp. 1–16.

[13] F. Zeng, W. Gan, Y. Wang, N. Liu, and P. S. Yu, "Large language models for robotics: A survey," *arXiv preprint arXiv:2311.07226*, 2023.

[14] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch, "Language models as zero-shot planners: Extracting actionable knowledge for embodied agents," in *International conference on machine learning*. PMLR, 2022, pp. 9118–9147.

[15] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, "Large language models are zero-shot reasoners," *Advances in neural information processing systems*, vol. 35, pp. 22 199–22 213, 2022.

[16] H. Sha, Y. Mu, Y. Jiang, G. Zhan, L. Chen, C. Xu, P. Luo, S. E. Li, M. Tomizuka, W. Zhan, and M. Ding, "Languagempc: Large language models as decision makers for autonomous driving," *arXiv preprint arXiv:2310.03026*, 2023.

[17] C. Cui, Y. Ma, Z. Yang, Y. Zhou, P. Liu, J. Lu, L. Li, Y. Chen, J. H. Panchal, A. Abdelraouf *et al.*, "Large language models for autonomous driving (llm4ad): Concept, benchmark, simulation, and real-vehicle experiment," *arXiv preprint arXiv:2410.15281*, 2024.

[18] Z. Li, J. Jiang, W.-H. Chen, and L. Sun, "Autonomous lateral maneuvers for self-driving vehicles in complex traffic environment," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 2, pp. 1900–1910, 2023.

[19] A. Arab, K. Yu, J. Yu, and J. Yi, "Motion planning and control of autonomous aggressive vehicle maneuvers," *IEEE Transactions on Automation Science and Engineering*, vol. 21, pp. 1488–1500, 2024.

[20] Y. Chen, Z. han Ding, Z. Wang, Y. Wang, L. Zhang, and S. Liu, "Asynchronous large language model enhanced planner for autonomous driving," 2024. [Online]. Available: https://arxiv.org/abs/2406.14556

[21] K. Long, H. Shi, J. Liu, C. Xiao, and X. Li, "Vlm-mpc: Model predictive controller augmented vision language model for autonomous driving," *arXiv preprint*, 2024.

[22] R. Rajamani, *Vehicle dynamics and control*. Springer Science & Business Media, 2011.