CoopReflect: Towards Natural Language Communication for Cooperative Autonomous Driving via Multi-Agent Learning

Jiaxun Cui
The University of Texas at Austin
cuijiaxun@utexas.edu

Chen Tang
University of California, Los Angeles
ctangac@ucla.edu

Jarrett Holtz
Robert Bosch LLC
jarrett.holtz@us.bosch.com

Janice Nguyen
University of California, Riverside
jnguy172@ucr.edu

Alessandro G. Allievi
Robert Bosch LLC
alessandro.allievi@us.bosch.com

Hang Qiu
University of California, Riverside
hangq@ucr.edu

Peter Stone
The University of Texas at Austin &
Sony AI
pstone@cs.utexas.edu

ABSTRACT

Past work has demonstrated that autonomous vehicles can drive more safely if they communicate with each other. However, this communication is usually not human-understandable. Using natural language as a vehicle-to-vehicle (V2V) communication protocol offers the potential for autonomous vehicles to drive cooperatively not only with each other but also with human drivers. To explore the potential use of natural language for V2V communication, we develop LLM-based driving agents and study their interactions in a new simulation environment, TalkingVehiclesGym, which features traffic scenarios where communication can potentially help avoid imminent collisions and/or support efficient traffic flow. While LLM agents relying solely on chain-of-thought reasoning struggle to coordinate effectively, we introduce CoopReflect, a multi-agent learning framework that equips agents with knowledge for both natural language message generation and high-level decision-making through trial and error and multi-agent debriefing. Experiments show that CoopReflect produces more meaningful and human-understandable messages than existing baselines, enabling stronger cooperation. Finally, we distill scenario-specific knowledge into a unified language model policy, achieving cross-scenario generalization and substantially reducing decision-making latency. Our code and demo videos are available at https://talking-vehicles.github.io/

KEYWORDS

V2V Communication, Learning LLM Agents, Autonomous Driving

ACM Reference Format:
Jiaxun Cui, Chen Tang, Jarrett Holtz, Janice Nguyen, Alessandro G. Allievi, Hang Qiu, and Peter Stone. 2026. CoopReflect: Towards Natural Language Communication for Cooperative Autonomous Driving via Multi-Agent Learning. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 18 pages. https://doi.org/10.65109/MOAV6406

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/MOAV6406

1 INTRODUCTION

Driving is inherently a multi-agent problem [11, 28], in which each driver makes independent decisions based on their own perceptions while interacting with others on the road. As we transition towards (semi-)autonomous vehicles, centralized control [2] of all cars may appear efficient, but it is impractical and unlikely to gain widespread adoption. On the other hand, cooperative driving through communication channels is feasible and can offer significant benefits even when implemented in a limited capacity. Past research has demonstrated the advantages of cooperation among autonomous cars for perception [34, 40], prediction [35], and planning [8]. However, these benefits are limited to vehicles that use the same learned environmental representation and communication language, limiting broader participation from those with different representations or language and leaving human drivers reliant solely on their local perceptions without being privy to the collaboration efforts.

As vision-language models become increasingly prevalent for language-conditioned reasoning and high-level planning in complex traffic environments, the use of natural language as a complementary, general-purpose communication interface offers significant potential for both vehicle–human and vehicle–vehicle coordination. Prior work has made progress toward this vision by training driving agents to generate and explain driving decisions in natural language [18, 36] or to coordinate with human drivers within a single vehicle [9], often leveraging large-scale datasets [16, 17, 24, 31]. However, **inter-vehicle** communication using natural language is relatively underexplored, particularly with respect to its feasibility and design considerations in cooperative driving scenarios, despite a few contemporaneous efforts [13, 14]. These efforts demonstrate a growing interest in the problem but remain preliminary in scope and evaluation. Complementing and extending this contemporaneous work, we introduce **TalkingVehiclesGym**, a multi-agent simulation framework that models vehicle-to-vehicle communication and enables closed-loop evaluations of natural language interactions across a suite of accident-prone traffic scenarios.

Recent advances in Large Language Models (LLMs) present new opportunities for agents to learn to speak and understand natural language messages in cooperative driving scenarios. In this work, we study how LLM agents can interact using natural language and optimize communication strategies through trial-and-error multi-agent interactions. Our initial experiments show that LLM agents relying only on chain-of-thought reasoning struggle to perform well, and single-agent reflection methods provide only modest improvements in coordination. Hence, we introduce **CoopReflect**, a multi-agent learning method enabling LLM agents to engage in centralized reflection to refine their cooperation strategies. The resulting reflections are later incorporated as memories into decentralized agent execution. Our experimental results in simulation indicate that, when LLM agents initially fail to collaborate effectively, our proposed learning method helps them both learn what to communicate and how to respond to messages through interactions. Finally, we distill the learned behaviors of large models in each scenario into a compact language model, achieving scenario and role generalization, as well as near-real-time inference with decision latency under 500 ms, compared to the 10 s required by large models in wall-clock time.

In summary, our contributions are threefold:

(1) We introduce TalkingVehiclesGym, a multi-agent simulation environment that enables closed-loop evaluation of natural language communication in cooperative driving scenarios;
(2) We propose CoopReflect, a multi-agent learning framework that enables LLM agents to refine cooperation strategies through centralized reflection over multi-agent interactions;
(3) We demonstrate that the cooperative strategies learned by CoopReflect can be distilled into a single compact language model, achieving efficient inference and generalization.

While this exploratory work does not attempt to address the challenges required to make it fully human-usable — e.g., by enforcing short, real-time messaging — this paper takes a crucial step in that direction by restricting all messages to be in natural language and providing a testbed that allows closed-loop improvements.

2 PROBLEM DEFINITION

In this paper, we focus on the subset of agents that actively participate in the cooperation. We assume that these cooperative vehicles implicitly aim to help each other, treating all other (referred to as "background") vehicles as uncontrollable elements of the environment. Therefore, we frame the problem of ***Talking Vehicles*** as a partially observable stochastic game (POSG), focusing on optimizing the social welfare of a *focal population* ($\mathcal{F}$) [1] — defined as the joint reward of all participating agents — as the primary objective. The reward functions associated with each agent's individual tasks may or may not fully align, necessitating coordination among agents to achieve high joint rewards. Each agent's observation space is limited to a partial view of the full state, and agents make decisions in a decentralized manner based on their own partial observations and messages received from other agents. In this problem, each agent's action space comprises two main components: **(1) message generation** and **(2) vehicle control**. In this work, the message generation space is *open-vocabulary* over natural language (English), instead of being restricted to predefined templates.

We define a POSG with a deterministic observation function and undiscounted rewards as the tuple

$$\langle \mathcal{I}, \mathcal{S}, \{O_i\}, \{\mathcal{A}_i\}, \mathcal{P}, \{\mathcal{R}_i\} \rangle,$$

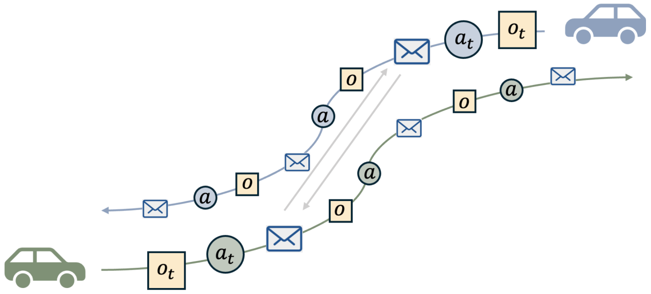


Figure 1: Overview of the In-Episode Communication Mechanism. At each time step, every agent simultaneously generates control actions and messages (a_t) conditioned on its observation (o_t). The exchanged messages are then passed among agents, and the resulting dialogue is incorporated into decision-making at the next time step.

where $\mathcal{I} = \{1, 2, ..., N\}$ refers to the identities of all agents in a scenario; $\mathcal{S}$ is the state space comprehensively describing the environment; O_i is the observation space describing agent i's view of the state; $\mathcal{A}_i$ is the action space of agent i; $\mathcal{P}$ is the state transition function $\mathcal{S} \times \mathcal{A}_1 \times \mathcal{A}_2 \times ... \times \mathcal{A}_N \rightarrow \mathcal{S}$; $\mathcal{R}_i$ is the reward function of agent i. The focal group of agents is denoted by $\mathcal{F} \subseteq \mathcal{I}$, representing a subset of all agents $\mathcal{I}$. Here, an agent refers to an entity in the POSG (*e.g.*, a vehicle), controlled by a policy π_i that specifies how agent i selects actions based on its available information. The goal of each agent $i \in \mathcal{F}$ is to learn a policy π_i to maximize the expected cumulative task returns of all agents in $\mathcal{F}$, given background agent policies outside the focal group: $\max_{\{\pi_i\}_{i \in \mathcal{F}}} \mathbb{E}\left[\sum_{i \in \mathcal{F}} \sum_{t=0}^{t=T} R_i(s_t, \mathbf{a}_t) \middle| \{\pi_j\}_{j \notin \mathcal{F}, j \in \mathcal{I}} \right]$, where s_t is the state at time t, and $\mathbf{a_t} = (a_1^t, a_2^t, ..., a_N^t)$ is the joint action of all agents at time t.

The agent's policy is structured to output both control and communication commands. Specifically, $\pi_i(O_i, \{M_j\}_{j \in \mathcal{F}}) \rightarrow \mathcal{A}_i$ maps the observation of agent i and the received messages $\{M_j\}_{j \in \mathcal{F}}$ to its action space $\mathcal{A}_i = \langle \mathcal{M}_i, C_i \rangle$, where $\mathcal{M}_i$ represents the message generation space, which is constrained to natural language, and C_i denotes the vehicle control space with dimensions for throttle, brake, and steering inputs. At time step t, the message M_i generated by agent i is broadcast to all connected agents within a certain communication radius at the next time step $t + 1$ (Figure 1).

This problem presents the following technical challenges:

(1) How can agents understand the situation and **generate** meaningful messages to collaboratively perceive the environment or negotiate in natural language;
(2) How can agents **comprehend** incoming natural language messages and **incorporate** them into driving decision-making?

3 ENVIRONMENT

To provide concrete and typical driving scenarios that expose the *talking vehicles* challenge, we have developed a simulation environment, **TalkingVehiclesGym**, which is a multi-agent gymnasium environment [5, 33] for the closed-loop evaluation of urban driving policies. TalkingVehiclesGym supports a flexible configuration of multi-agent scenarios, incorporating heterogeneous agents such as language agents, sensory agents, human agents, heuristic behavior agents, etc. It also enables **in-episode** communication between

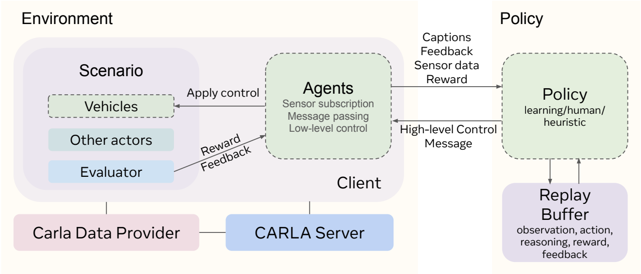


Figure 2: *TalkingVehiclesGym* Simulation framework. An agent is defined within the scenario and has a specific sensor registration and action space. A policy takes observations from an agent, computes actions, and learns from the experience replay buffer.

agents using a realistic simulated communication protocol based on MQTT, a lightweight publish–subscribe protocol widely used in distributed systems. The simulation dynamics are implemented in CARLA [12], which provides realistic vehicle physics and complex urban traffic layouts.

3.1 Scenarios ($\mathcal{P}$) and Rewards ($\mathcal{R}$)

TalkingVehiclesGym has been set up with several accident-prone scenarios where multi-agent communication could be beneficial, as shown in Figure 3. Scenarios labeled with `Cooperative Perception` are cases where agents can benefit from receiving information about regions beyond their own line of sight, and scenarios labeled with `Negotiation` are cases where agents must communicate to resolve conflicts in their intended plans. Each scenario features a focal group ($\mathcal{F}$) of agents operating alongside background agents with pre-scripted behaviors. Each focal agent is assigned a *task* described in natural language, with success defined as reaching its target location within a time limit without collisions. Agents without motion targets, such as a stationary truck in cooperative perception tasks, do not earn rewards directly for themselves. However, the optimization objective encourages these agents to send messages that assist others to achieve their tasks. More scenario descriptions and reward setups are detailed in Appendix C.

3.2 Observation Space ($\mathcal{O}$)

Our environment integrates a diverse range of sensor and simulator inputs inherited from CARLA. To focus on reasoning and multi-agent learning, we simplify environmental perception for **text-based agents** by introducing a rule-based, **partially observable captioner**. This module abstracts away the perception task, which would otherwise require object detection or vision-language models, by directly converting scenario information — such as the states of the ego vehicle and others, lane details, and road conditions — into natural language descriptions that convey *factual* information while maintaining the partial observability imposed by the agent's line-of-sight sensors. For agents equipped with a transmitter/receiver device (**transceiver**), real-time communication is enabled during episodes, and the message dialog is included as part of their observations. An example of a text-based observation is provided in Appendix D.

3.3 Action Space ($\mathcal{A}$)

The action space for each agent encompasses both vehicle control and message generation. The vehicle control space C is three-dimensional, consisting of throttle, brake, and steering. To reduce the decision-making frequency, agents execute high-level vehicle motion commands represented as temporal sequences of low-level vehicle controls $(C_t, C_{t+1}, ..., C_{t+k})$, where each command spans k time steps. These high-level commands are atomic actions such as `go (adapt to a target speed)`, `stop`, `slow down`, `speed up`, and `change to the left lane`. They are composed through a combination of a global route planner and a local PID controller. The message generation space $\mathcal{M}$ is open-vocabulary, restricted to natural language tokens in this work, but the communication system is flexible enough to support other communication modes. In this work, messages are generated alongside the high-level control commands every 0.5 seconds ($k = 10$ simulation steps).

4 METHOD

The core technical challenge of the *talking vehicles* problem is to enable embodied agents to *communicate in natural language for cooperative purposes and to adjust their actions dynamically according to the conversation.* While prior works relied on extensive imitation learning data from human play to train agent policies that can speak or make decisions in natural language contexts [3], we instead explore leveraging pre-trained large language models to endow agents with such communication and reasoning capabilities. To establish an initial solution, we adopt an **LLM agent framework** (Figure 4) that employs LLMs as a foundational prior for autonomous agents to engage in human-like communication, structuring messages within natural language space, and allowing agents to interpret messages to make informed driving decisions. However, LLMs are typically not trained for cooperative driving tasks. To address this limitation, we introduce **CoopReflect**, a **novel multi-agent learning method for LLM agents** built upon feedback loops that allow LLM agents to iteratively refine their communication and control policies through trial-and-error interactions with confederate agents. Inspired by how humans reflect and debrief after a cooperative game such as Hanabi, we enable agents to discuss cooperative strategies after each interaction episode.

4.1 Agent Policy

Each agent i follows a policy: $\pi_i(O_i, \{M_j\}_{j \in \mathcal{F}}) \rightarrow \langle \mathcal{M}_i, C_i \rangle$, where the distribution over actions follows the LLM used by the agent. Here, O_i represents agent i's comprehensive observation encompassing task and goal descriptions, environment details, and common traffic rules, expressed as a text or token sequence $\{t_i^o\}$. A received message $M_j = \{t_j^m\}$ from agent j and a message to send $M_i = \{t_i^m\}$ are also text sequences generated by language agents. $C_i = \{t_i^c\}$ represents a text sequence for high-level control commands. The joint probability of selecting a command and generating a message is expressed as $P_i(\{t_i^m\}; \{t_i^c\} | \{t_i^o\}; \{\{t_j^m\}\}_{j \in \mathcal{F}})$ where "**;**" indicates text concatenation and a large language model serves as the oracle to determine the probabilities.

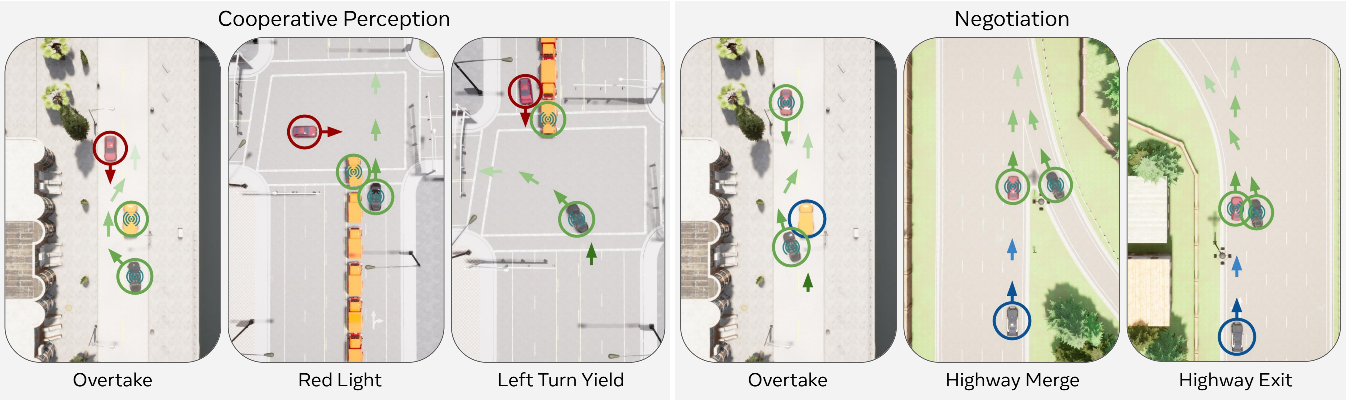


Figure 3: ***Overview of Scenarios and Agent Roles.*** **Green circles: Focal agents, agents aim at establishing coordination through communication; Red circles: Potential colliders; Blue circles: Background agents.**

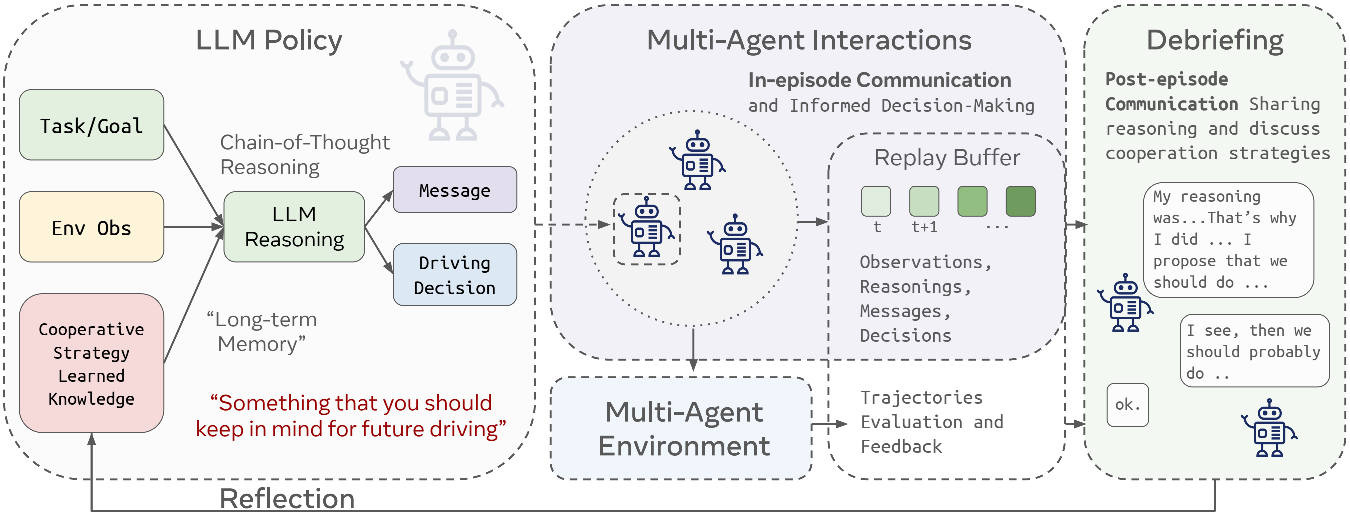

LLM Policy
Task/Goal
Env Obs
Cooperative Strategy Learned Knowledge
Chain-of-Thought Reasoning
LLM Reasoning
Message
Driving Decision
"Long-term Memory"
"Something that you should keep in mind for future driving"
Multi-Agent Interactions
In-episode Communication and Informed Decision-Making
Replay Buffer
t
t+1
...
Observations, Reasonings, Messages, Decisions
Multi-Agent Environment
Trajectories Evaluation and Feedback
Debriefing
Post-episode Communication Sharing reasoning and discuss cooperation strategies
My reasoning was...That's why I did ... I propose that we should do ...
I see, then we should probably do ..
ok.
Reflection

Figure 4: ***CoopReflect Agent Framework and Agent Learning Pipeline.***

In-Context Knowledge. Rather than fine-tuning LLM weights through gradient-based updates, CoopReflect adapts agent policies by modifying the *context*, leveraging the auto-regressive nature of LLMs. Define $K_i = \{t_i^k\}$ as agent i's accumulated knowledge and $S_i = \{t_i^s\}$ as its cooperative strategy. The joint probability of generating commands and messages is then influenced by these additional prompt tokens:

$$P_i(\{t_i^m\};\{t_i^c\}|\{t_i^k\};\{t_i^s\};\{t_i^o\};\{\{t_j^m\}\}_{j\in\mathcal{F}}).$$

Chain-of-Thought (CoT) Reasoning. LLM agents often generate better decisions by producing intermediate reasoning traces that structure their understanding of the situation [37]. To leverage this observation, we prompt LLMs to reason step-by-step about the environment, incorporating observations, received messages, and in-context knowledge. The reasoning process generates an intermediate text sequence, denoted as $R_i = \{t_i^r\}$. The LLM agent then sample action token sequences $\{t_i^m\}$ and $\{t_i^c\}$ according to the probability distribution:

$$P_i(\{t_i^m\};\{t_i^c\}|\{t_i^k\};\{t_i^s\};\{t_i^o\};\{\{t_j^m\}\}_{j\in\mathcal{F}};\{t_i^r\}).$$

The resulting output is formatted in JSON with two keys: "command" and "message".

4.2 Agent Learning: Post-Episode Debriefing

The learning process is depicted in Figure 4. Initially, the LLM agents interact with each other in the scenarios and accumulate experience, which is stored in a replay buffer. Following the interaction phase, the agents engage in a debriefing session where they utilize past experiences as context to collaboratively refine a cooperative strategy. The outcomes of these discussions are summarized into two critical components: *knowledge* ($K_i = \{t_i^k\}$) and *cooperative strategies* ($S_i = \{t_i^s\}$). These components are subsequently integrated as in-context knowledge for future interactions, playing a pivotal role in shaping and improving the policy.

Replay Buffer. Each agent i stores their trajectories locally. Each trajectory is a temporal sequence of transition data $T_i = \langle o_{i,t}, a_{i,t}, o_{i,t+1} \rangle$, which includes current and next local observations, commands, messages, and reasoning in a **replay buffer**, serving as a repository for further learning and iterative refinement. When an episode concludes, the environment evaluates each agent's performance and provides scalar rewards along with **verbal feedback**, such as "`Vehicle 109 collided with Vehicle 110 after 2 seconds.`" or "`Vehicle 111 stagnated for too long to complete its task.`" Each transition in the replay buffer is subsequently **retrospectively labeled** with enriched metadata, including responses from other agents, collision details (e.g., time to collision), stagnation specifics, and final rewards and outcomes.

Batch Context Sampling. Before engaging in the post-episode discussion (debriefing), each learning agent reflects on its individual experience from their perspective first. While analyzing the entire trajectory would provide a comprehensive understanding of failure cases, computational and context window constraints necessitate sampling a subset (**batch**) of key frames from its replay buffer. To prioritize relevant data, the sampling process heuristically assigns (following Equation 1) higher probabilities to transitions that occur immediately before collisions, involve actions contributing to collisions, or lead to stagnation due to agents slowing down. Additionally, transitions that feature more intensive multi-agent interactions are given more weight. These selected samples serve as the context for subsequent analysis and strategy formulation, allowing the agent to focus on critical timesteps for improving performance.

Debriefing. When an episode ends due to the fault of a single agent, only that agent performs individual reflection. A debriefing (cooperative reflection) session is initiated when an episode ends in failure (either due to a collision or stagnation) arising from poor cooperation among agents. The debriefing session proceeds in a **turn-based** manner over N rounds, with the aim of improving cooperative behavior in future interactions. The speaking order is deterministic in this work for each session, and agents take turns speaking in a round-robin format. The agent chosen to speak first is responsible for proposing a **joint** cooperative strategy ($S_1, S_2, ...S_{i\in\mathcal{F}}$) for everyone participating in the debriefing (the focal group). This agent begins by reasoning through its local transition data batch, analyzing the consequences of its actions, their influence on others and vice versa, and formulating a proposed strategy. Subsequently, the other agents take turns sharing their perspectives, providing feedback, or offering alternative insights based on their analysis of their own experience batch. After the discussion, each agent summarizes the discussion to develop **individual** cooperative strategies (S_i) and knowledge (K_i). These outcomes will later serve as in-context guidelines for future driving tasks. This joint discussion for future individual decision-making structure mirrors the principles of the Centralized Training Decentralized Execution (CTDE) framework [4], a widely used approach in multi-agent learning. Our implementation details are available in Appendix A.1.

5 EXPERIMENTS

This section presents an empirical evaluation of CoopReflect and baseline approaches across different cooperative driving scenarios. We investigate the following research questions:

(1) Can LLM agents establish collaboration through chain-of-thought reasoning without prior interactions? (The evaluated LLMs in this work can not.)
(2) Does decentralized reflection enable LLM agents to improve their collaborative ability as they gain more interaction experiences? (Yes.)
(3) Does centralized discussion among LLM agents provide additional improvements in collaboration and communication compared to decentralized reflection? (Yes.)
(4) Can natural language communication enhance the performance and coordination of LLM agents compared to those without communication? (Only if well trained.)

Metrics. Evaluation metrics are established based on the outcomes of agents who can incur reward (reward-eligible) for their tasks in the focal group, which is scenario-specific. For a scenario with N reward-eligible agents in the focal group, evaluated over M episodes, we utilize two key metrics:

(1) the **average collision rate** (**CR**), normalized by the group size, is $\frac{1}{N} \cdot \frac{1}{M} \sum_{m=1}^{M} \sum_{i\in\mathcal{F}} \mathbb{1}(\text{agent } i \text{ involved in a collision})$, where collisions may involve both focal and background agents;
(2) the **average success rate** (**SR**), also normalized by the group size, is $\frac{1}{N} \cdot \frac{1}{M} \sum_{m=1}^{M} \sum_{i\in\mathcal{F}} \mathbb{1}(\text{agent } i \text{ succeeded})$.

Here, $\mathbb{1}$ is the indicator function, equal to 1 if the event occurs and 0 otherwise. The remaining failure cases, where agents exceed the time limit, heuristically determined to represent the upper bound for efficient task completion, without success or collision, are captured by the **average time out rate**, which can be derived as $TR = 1 - SR - CR$.

Experimental Setup. For each baseline[1], we consider two settings labeled as *Silent* and *Comm*. In the ***Silent*** setting, LLM agents focus solely on controlling the vehicle based on their individual perception and reasoning without communication. The ***Comm*** setting allows a method to generate either only messages or both messages and driving commands. For each LLM-based learning method, we allow agents to interact for up to 60 episodes per scenario, which is a random sequence alternating between safe (or randomized agent positions for highway negotiation settings) and accident-prone configurations with equal percentages. We define a *"solved"* criterion for learning success in a scenario as 20 consecutive successful episodes. Due to the uncontrollable randomness in the OpenAI models, we give each learning method 3 knowledge reset opportunities [2] to either report the *"solved"* result, or otherwise, the last run for each seed. After learning, each method is evaluated for 30 episodes per scenario configuration per seed. We report experimental results aggregated with 3 seeds.

[1] Except for CoopReflect, which is only tested under the *Comm* setting since it is particularly designed for improving multi-agent communication.

[2] Knowledge reset is done by clearing the learned knowledge before reaching a *solved state indicator*, defined by 20 consecutive successful training episodes in this work.

Baselines. We established several baselines and scenarios to answer the research questions:

1. **Zero-shot**: a base LLM agent using Chain-of-Thought (CoT) reasoning only,
2. **Reflection**: an LLM agent with CoT reasoning contextualized with knowledge from self-reflection,
3. **Correction+RAG (Silent)**: an LLM agent that corrects past actions via self-reflection, storing these corrections in a vector-based, retrievable memory, and uses few-shot retrieved example augmented generation (Correction+RAG). The retrieval augmented method without communication adapts DiLU [38], a non-communicating single-agent LLM-based approach that drives via reflection, to our environment.
4. **Correction+RAG (Comm)**: an LLM agent that resembles AgentsCoDriver [14], the multi-agent communication extension of DiLU, but they do not actively optimize the messages.

For a fair comparison across baseline LLM agents, we do not initialize the knowledge with human data, nor is there human involvement during the learning process. Moreover, we apply the same batch context sampling method for reflection or correction for all LLM agent baselines as our method. Additionally, we include **Coopernaut** [8], a LiDAR-based cooperative driving method, as an aspirational reference point for cooperative perception. Note that Coopernaut is not directly comparable because it processes sensory data and communicates intermediate neural representations rather than natural languages. We do not compare with other multi-agent communication baselines for the same reasons.

5.1 Quantitative and Qualitative Results

Table 1 presents the quantitative evaluation of all methods across tasks. Notably, in this proof of concept, none of the LLM methods compared operate in real-time, requiring approximately 10 real-world seconds per decision step (0.5 seconds equivalent in simulation) using `gpt-4o-mini`. The inference latency primarily depends on reasoning, but we demonstrate an approach towards real-time inference in Section 5.2. On average, the natural language message bandwidth remains below 300 bytes per decision step, requiring less than 0.01 Mbps communication bandwidth. Based on these results, we provide responses to the research questions posed at the start of the section.

R1: LLM agents with CoT examined in this paper do not establish collaboration through communication in zero-shot interactions. Our experiments show that Zero-Shot agents (`gpt-4o-mini`), even with communication enabled, fail to coordinate effectively. The failure modes are (1) agents do not communicate effectively to understand each other's needs in perception or achieve agreement in negotiation, or (2) even when the messages make sense to humans, agents do not respond with appropriate driving commands. This result suggests that without prior training or explicit strategies, chain-of-thought reasoning alone is insufficient to foster effective coordination. At the time of writing, preliminary experiments with other LLMs such as `Llama 3` and `gpt-4o` follow a similar pattern. Future work could systematically examine whether large reasoning models like `gpt-o4` can mitigate these limitations.

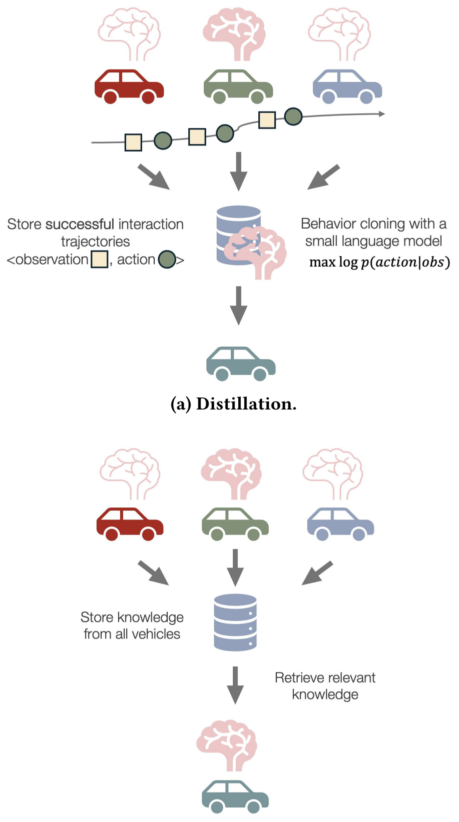
R2: Decentralized learning can enable LLM agents to improve their collaborative ability as they gain more interaction experiences. The decentralized learning methods, Reflection and Correction+RAG, show significant improvement in reducing collision rates from Zero-Shot across tasks. Reflection allows agents to individually analyze their experience to generate knowledge, but the knowledge is often more reactive than proactive (see Appendix E.7 for example). The Correction+RAG method records successful episodes to preserve successful coordination patterns and correct commands and messages at key frames selected through a heuristic batch sampling. However, although the method improves the control response strategy, we find that it qualitatively does not always produce messages that are consistent with the actions, likely due to its open-loop message revision process. Both methods show promise, but have room for improvement.

R3: Centralized debriefing enhances coordination more than decentralized reflection. The debriefing method, which focuses on generating explicit cooperation strategies, enables LLM agents to achieve more stable collaboration compared to decentralized reflection or zero-shot approaches, evidenced by higher success rates than baselines across tasks. Qualitatively, the resulting conversations are human-interpretable, paving the way for future human–AI communication in cooperative driving scenarios. Please refer to our supplementary videos for examples of the generated dialogues. The primary performance boost of CoopReflect stems from the formalized coordination strategy, which explicitly defines how each agent should communicate and respond within a dialogue across different scenarios. Examples of learned cooperative strategies for each scenario are illustrated in Appendix E. Interestingly, CoopReflect reveals that LLMs can struggle to understand complex AI-generated messages that resemble natural language, so agents eventually develop concise communication protocols (like "hold" and "go" in Appendix E.8) to ensure that their intentions are easily interpretable among themselves. However, open challenges still remain. For example, although the learned strategies are easy to verify, the debriefing process used to identify them can sometimes fail, where no agents can find issues with the cooperation strategies in harder and longer-horizon tasks, such as `negotiation-highway-exit`.

R4: Natural language communication in cooperative driving can be effective, but may pose safety risks without good communication strategies. Our method, which operates with natural language communication, provides a proof of concept for natural-language-based multi-agent coordination across scenarios. However, learning to communicate effectively remains challenging. In cooperative perception tasks, communication-enabled methods consistently outperform silent ones, highlighting the critical role of information sharing. In contrast, in negotiation scenarios such as `highway-merge` and `highway-exit`, agents generally perform better in silent mode. This result suggests that communication can add complexity and hinder coordination when not well-optimized. We speculate that the root cause lies in the suboptimal communication strategies learned under decentralized training, where messages may introduce noise rather than useful signals.

Table 1: Experimental results for per-scenario learning. Each method is trained and evaluated independently in each scenario, using three random seeds and 30 evaluation episodes per seed. Results are reported as `mean ± standard deviation`. *Zero-shot (Silent)* and *Zero-shot (Comm)* denote LLM agents using chain-of-thought reasoning without or with communication, respectively. *+Reflection* and *+Correction+RAG* indicate single-agent learning through reflection or correction with retrieval-augmented generation. *+Debrief* is our proposed multi-agent learning method CoopReflect. *Coopernaut* is a non-LLM communication baseline for cooperative perception only.

Scenario / Method	Cooperative Perception Scenarios						Negotiation Scenarios					
	Overtake (Perception)		**Red Light**		**Left Turn**		**Overtake (Negotiation)**		**Highway Merge**		**Highway Exit**	
	CR (%) ↓	SR (%) ↑	CR (%) ↓	SR (%) ↑	CR (%) ↓	SR (%) ↑	CR (%) ↓	SR (%) ↑	CR (%) ↓	SR (%) ↑	CR (%) ↓	SR (%) ↑
Zero-shot (Silent)	93.3 ± 3.4	0.0 ± 0.0	93.3 ± 6.7	6.7 ± 6.7	93.3 ± 5.8	6.7 ± 5.8	89.9 ± 2.8	7.2 ± 3.8	100.0 ± 0.0	0.0 ± 0.0	33.3 ± 9.3	66.1 ± 9.2
+Reflection	87.8 ± 3.4	0.0 ± 0.0	94.4 ± 6.9	5.6 ± 6.9	76.7 ± 20.8	23.3 ± 20.8	32.8 ± 29.4	36.7 ± 52.1	15.0 ± 23.1	84.4 ± 22.6	32.8 ± 13.4	67.2 ± 13.4
+Correction+RAG	62.0 ± 31.9	4.4 ± 7.7	93.3 ± 3.3	6.7 ± 3.3	64.4 ± 15.0	35.6 ± 15.0	46.7 ± 21.9	33.3 ± 28.0	35.6 ± 29.4	64.4 ± 29.4	33.9 ± 28.4	51.1 ± 14.2
Zero-shot (Comm)	91.1 ± 5.1	4.4 ± 5.1	60.0 ± 11.5	38.9 ± 10.7	85.6 ± 8.4	14.4 ± 8.4	87.8 ± 5.9	11.7 ± 6.7	67.2 ± 27.1	32.8 ± 27.1	53.3 ± 11.5	46.7 ± 11.5
+Reflection	63.3 ± 14.5	34.4 ± 10.7	37.8 ± 18.4	47.8 ± 18.4	51.1 ± 37.2	47.8 ± 36.0	55.6 ± 38.9	43.3 ± 37.1	20.0 ± 1.7	80.0 ± 1.7	53.9 ± 24.1	45.6 ± 23.6
+Correction+RAG	4.4 ± 1.9	90.0 ± 6.7	13.3 ± 12.0	66.7 ± 27.3	43.3 ± 38.4	38.9 ± 22.7	38.3 ± 6.0	61.1 ± 5.4	40.0 ± 18.0	60.0 ± 18.0	49.4 ± 49.2	43.3 ± 39.8
+Debrief (ours)	**1.1 ± 1.9**	**94.4 ± 6.9**	**0.0 ± 0.0**	**93.3 ± 5.8**	**6.7 ± 3.3**	**92.2 ± 3.8**	**3.3 ± 3.3**	**95.6 ± 3.8**	**6.7 ± 11.5**	**93.3 ± 11.5**	**18.3 ± 21.7**	**81.1 ± 21.2**
Coopernaut (Comm)	4.5 ± 3.1	90.5 ± 1.2	17.7 ± 7.8	80.7 ± 7.6	18.1 ± 6.2	80.7 ± 5.2	N/A	N/A	N/A	N/A	N/A	N/A


Store successful interaction trajectories
<observation□, action●>
Behavior cloning with a small language model
max log $p(action|obs)$
Store knowledge from all vehicles
Retrieve relevant knowledge

(a) Distillation.

(b) Centralized Memory.

Figure 5: Generalization Methods. (a) Distillation performs full-parameter fine-tuning by matching the probability distribution of expert decisions from successful trajectories, producing a compact model that acts directly on observations without explicit reasoning. (b) Centralized Memory aggregates learned knowledge and cooperative strategies from all vehicles across all scenarios into a shared memory. Each vehicle accesses this memory to retrieve the most relevant knowledge based on its observation and dialogue, followed by Chain-of-Thought reasoning and decision-making.

5.2 Towards Real-Time Cross-Scenario Cross-Role Generalization

Up to this point, a separate policy was trained to handle each of the TalkingVehiclesGym scenarios. However, for practical deployment, it is desirable to develop a single policy that can handle a broad range of challenging driving scenarios. We explore two independent approaches for achieving cross-scenario generalization: ***Centralized Memory*** and ***Distillation*** (Figure 5). ***Centralized Memory*** aggregates all agents' *most effective* knowledge—identified by the highest estimated success rate across learning trials—into a unified vector memory. Agents then search in the memory according to the observation and dialogue for the most relevant knowledge. ***Distillation*** performs full-parameter fine-tuning of a small language model, DistilGPT2 [25, 27], to directly **imitate** the behavior of the CoopReflect agent with the most effective knowledge. The imitation dataset is aggregated from all *successful* evaluation episodes across scenarios, and the distillation model is trained to minimize the token-level cross-entropy loss against the large model's outputs. During inference, decisions are generated via random sampling with a temperature of 0.2.

As shown in Table 3, the Distillation model achieves decision generation times between **100 ms** and **470 ms** on an NVIDIA A40 GPU, depending on message generation length (**50 bytes** to **300 bytes**), getting close to the 500 ms decision-making frequency, though time delays and asynchrony have not been fully considered. Evaluation results of the two methods in accident-prone scenarios are listed in Table 2. Remarkably, the distilled model generalizes well across scenarios and even surpasses the performance of its teacher model (CoopReflect) in some cases. However, we observe that it tends to behave overly conservatively in safe `perception-overtake` scenarios (Table 5, Table 6), suggesting room for further improvement, potentially through expert-guided correction methods such as DAgger [26].

6 RELATED WORK

LLM Agents for Autonomous Driving. have shown potential to address various autonomous driving tasks. In particular, they are promising in tackling corner cases [39] due to their reasoning

Table 2: Cross-scenario generalization results. Policies are trained once and evaluated over 3 seeds (30 episodes per seed). We report the mean ± standard error across seeds. *Debrief (per-scenario)* is an oracle baseline trained separately for each scenario, used to benchmark the generalization of *Centralized Memory* and *Distillation*.

Scenario / Method	Cooperative Perception Scenarios						Negotiation Scenarios					
	Overtake (Perception)		**Red Light**		**Left Turn**		**Overtake (Negotiation)**		**Highway Merge**		**Highway Exit**	
	CR (%) ↓	SR (%) ↑	CR (%) ↓	SR (%) ↑	CR (%) ↓	SR (%) ↑	CR (%) ↓	SR (%) ↑	CR (%) ↓	SR (%) ↑	CR (%) ↓	SR (%) ↑
Debrief (per-scenario)	1.1 ± 1.1	**98.9 ± 1.1**	0.0 ± 0.0	96.7 ± 0.0	4.4 ± 2.9	94.4 ± 2.2	10.0 ± 3.8	87.2 ± 3.9	2.2 ± 2.2	97.8 ± 2.2	13.3 ± 6.0	86.7 ± 6.0
Centralized Memory	2.2 ± 1.1	93.3 ± 1.9	0.0 ± 0.0	**100.0 ± 0.0**	4.4 ± 2.9	93.3 ± 3.3	12.2 ± 2.9	86.7 ± 1.9	1.1 ± 1.1	98.9 ± 1.1	16.1 ± 4.8	82.8 ± 5.3
Distillation	**0.0 ± 0.0**	83.3 ± 1.9	0.0 ± 0.0	91.1 ± 4.4	**0.0 ± 0.0**	**96.7 ± 0.0**	**10.0 ± 3.3**	**88.9 ± 4.4**	**0.0 ± 0.0**	**100.0 ± 0.0**	**3.3 ± 0.0**	**96.7 ± 0.0**

Table 3: Decision Latency, Message Size using `Distilled` LLM Policy

Scenario / Latencies	**Overtake (Perception)**	**Left Turn**	**Red Light**	**Overtake (Negotiation)**	**Highway Merge**	**Highway Exit**
Decision Latency (s)	0.45	0.44	0.38	0.14	0.19	0.20
Message Size (bytes)	223.3	297.9	223.0	28.0	59.0	59.0

ability and the common-sense knowledge embedded, yielding a more generalizable autonomous driving stack. Recent studies have explored various approaches to tailor state-of-the-art LLMs for driving [14, 38]. However, a foundational challenge lies in grounding LLM agents in the real world—they need to perceive and understand the traffic scenarios. A straightforward approach is to obtain the observations from oracle perception models [22] and convert them to textual descriptions [7, 15, 21, 29]. Some other studies tackled this challenge by introducing Visual Language Models (VLMs), which are adapted to driving domains through in-context instruction tuning [18] or fine-tuning [10, 36, 41, 42]. To enhance LLM agents' reasoning ability, prior works have investigated incorporating handcrafted guidance and examples in the prompts [7, 15, 29], structuring the reasoning procedure [22, 31], and fine-tuning the models on driving datasets. Notably, fine-tuning LLMs and VLMs requires an extensive amount of driving data with language labels. Several works have attempted to adapt existing language-driving datasets for LLM fine-tuning [10, 18, 41] or augment large-scale multimodal driving datasets [6, 20, 32] with language labels [23, 24, 30, 31]. In contrast, our work generates scalable driving data through agent self-play. Note that existing models were predominantly evaluated in an *open-loop* fashion. In contrast, similar to some prior works [15, 29, 30], we conduct closed-loop evaluation of the proposed method and baseline methods in CARLA [12].

Natural Language Communication for Driving. There is a scarcity of prior research on optimizing LLM agents in multi-agent settings with natural language vehicle-to-vehicle communication, with only a few concurrent but distinct efforts such as [13, 14]. AgentsCoDriver [14] leverages a vector memory that stores and corrects vehicle control actions associated with specific situations, but it only optimizes the control actions through self-reflection while leaving message generation to the emergent capability of LLMs. Other studies on multi-agent collaboration for autonomous robots do not employ trial-and-error optimization; instead, they rely on human-engineered structures or prompting schemes to guide coordination. For example, LangCoop [13] designs a structured in-context knowledge format to facilitate intent inference among interactive agents, CoMAL [43] adds a collaboration module prompting agents to determine their respective roles, and GameChat [19] constructs a multi-round communication process to ensure agents reach consensus in constrained navigation tasks. In summary, TalkingVehiclesGym serves as a comprehensive testbed for studying multi-agent natural language communication among autonomous vehicles, and CoopReflect represents the first CTDE-style multi-agent reinforcement learning approach for multi-LLM-agent systems that jointly optimizes both communication and control in a closed-loop manner.

7 CONCLUSION AND FUTURE WORK

This work explores how autonomous vehicles can communicate and coordinate through natural language, offering a path toward future human–AI collaboration in cooperative driving. We introduce TalkingVehiclesGym, a multi-agent simulation environment for closed-loop evaluation of collaboration through vehicle-to-vehicle dialogue, and propose CoopReflect, a multi-agent learning framework that enables LLM-based driving agents to refine communication and control through iterative reflection and debriefing. Experiments show that while zero-shot LLM agents fail to collaborate effectively, reflective and centralized learning yields stable, human-interpretable cooperation across both perception and negotiation tasks. By distilling the learned behaviors into a compact model, we further achieve near-real-time, cross-scenario generalization. Overall, this study establishes natural language as a promising medium for cooperative autonomy and highlights future works for grounding communication in more realistic perception and communication mechanisms, improving multi-agent learning stability, and evaluating the agents in real human–AI collaborative driving.

ACKNOWLEDGMENTS

This work has taken place in the Learning Agents Research Group (LARG) at UT Austin. LARG research is supported in part by NSF (FAIN-2019844, NRT-2125858), ONR (N00014-24-1-2550), ARO (W911NF-17-2-0181, W911NF-23-2-0004, W911NF-25-1-0065), DARPA (Cooperative Agreement HR00112520004 on Ad Hoc Teamwork) Lockheed Martin, and UT Austin's Good Systems grand challenge. Peter Stone serves as the Chief Scientist of Sony AI and receives financial compensation for that role. The terms of this arrangement have been reviewed and approved by the University of Texas at Austin in accordance with its policy on objectivity in research.

REFERENCES

[1] John P Agapiou, Alexander Sasha Vezhnevets, Edgar A Duéñez-Guzmán, Jayd Matyas, Yiran Mao, Peter Sunehag, Raphael Köster, Udari Madhushani, Kavya Kopparapu, Ramona Comanescu, et al. 2022. Melting Pot 2.0. *arXiv preprint arXiv:2211.13746* (2022).

[2] Guillen-Perez Antonio and Cano Maria-Dolores. 2022. Multi-Agent Deep Reinforcement Learning to Manage Connected Autonomous Vehicles at Tomorrow's Intersections. *IEEE Transactions on Vehicular Technology* 71, 7 (2022), 7033–7043. https://doi.org/10.1109/TVT.2022.3169907

[3] Anton Bakhtin, Noam Brown, Emily Dinan, Gabriele Farina, Colin Flaherty, Daniel Fried, Andrew Goff, Jonathan Gray, Hengyuan Hu, et al. 2022. Human-level play in the game of Diplomacy by combining language models with strategic reasoning. *Science* 378, 6624 (2022), 1067–1074.

[4] Daniel S Bernstein, Robert Givan, Neil Immerman, and Shlomo Zilberstein. 2002. The complexity of decentralized control of Markov decision processes. *Mathematics of operations research* 27, 4 (2002), 819–840.

[5] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. Openai gym. *arXiv preprint arXiv:1606.01540* (2016).

[6] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. 2020. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 11621–11631.

[7] Can Cui, Yunsheng Ma, Xu Cao, Wenqian Ye, and Ziran Wang. 2023. Receive, reason, and react: Drive as you say with large language models in autonomous vehicles. *arXiv preprint arXiv:2310.08034* (2023).

[8] Jiaxun Cui, Hang Qiu, Dian Chen, Peter Stone, and Yuke Zhu. 2022. Coopernaut: end-to-end driving with cooperative perception for networked vehicles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 17252–17262.

[9] Thierry Deruyttere, Dusan Grujicic, Matthew B Blaschko, and Marie-Francine Moens. 2022. Talk2Car: Predicting physical trajectories for natural language commands. *Ieee Access* 10 (2022), 123809–123834.

[10] Xinpeng Ding, Jianhua Han, Hang Xu, Wei Zhang, and Xiaomeng Li. 2023. HiLM-D: Towards High-Resolution Understanding in Multimodal Large Language Models for Autonomous Driving. *arXiv preprint arXiv:2309.05186* (2023).

[11] Joris Dinneweth, Abderrahmane Boubezoul, René Mandiau, and Stéphane Espié. 2022. Multi-agent reinforcement learning for autonomous vehicles: A survey. *Autonomous Intelligent Systems* 2, 1 (2022), 27.

[12] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. 2017. CARLA: An open urban driving simulator. In *Conference on robot learning*. PMLR, 1–16.

[13] Xiangbo Gao, Yuheng Wu, Rujia Wang, Chenxi Liu, Yang Zhou, and Zhengzhong Tu. 2025. LangCoop: Collaborative Driving with Language. *arXiv preprint arXiv:2504.13406* (2025).

[14] Senkang Hu, Zhengru Fang, Zihan Fang, Xianhao Chen, and Yuguang Fang. 2024. AgentsCoDriver: Large Language Model Empowered Collaborative Driving with Lifelong Learning. *arXiv preprint arXiv:2404.06345* (2024).

[15] Ye Jin, Xiaoxi Shen, Huiling Peng, Xiaoan Liu, Jingli Qin, Jiayang Li, Jintao Xie, Peizhong Gao, Guyue Zhou, and Jiangtao Gong. 2023. Surrealdriver: Designing generative driver agent simulation framework in urban contexts based on large language model. *arXiv preprint arXiv:2309.13193* (2023).

[16] Jinkyu Kim, Teruhisa Misu, Yi-Ting Chen, Ashish Tawari, and John Canny. 2019. Grounding Human-To-Vehicle Advice for Self-Driving Vehicles. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[17] Jinkyu Kim, Anna Rohrbach, Trevor Darrell, John Canny, and Zeynep Akata. 2018. Textual Explanations for Self-Driving Vehicles. *Proceedings of the European Conference on Computer Vision (ECCV)* (2018).

[18] Yingzi Ma, Yulong Cao, Jiachen Sun, Marco Pavone, and Chaowei Xiao. 2023. Dolphins: Multimodal Language Model for Driving. arXiv:2312.00438 [cs.CV]

[19] Vagul Mahadevan, Shangtong Zhang, and Rohan Chandra. 2025. Gamechat: Multi-llm dialogue for safe, agile, and socially optimal multi-agent navigation in constrained environments. *arXiv preprint arXiv:2503.12333* (2025).

[20] Jiageng Mao, Minzhe Niu, Chenhan Jiang, Hanxue Liang, Jingheng Chen, Xiaodan Liang, Yamin Li, Chaoqiang Ye, Wei Zhang, Zhenguo Li, et al. 2021. One million scenes for autonomous driving: Once dataset. *arXiv preprint arXiv:2106.11037* (2021).

[21] Jiageng Mao, Yuxi Qian, Hang Zhao, and Yue Wang. 2023. Gpt-driver: Learning to drive with gpt. *arXiv preprint arXiv:2310.01415* (2023).

[22] Jiageng Mao, Junjie Ye, Yuxi Qian, Marco Pavone, and Yue Wang. 2023. A Language Agent for Autonomous Driving. (2023).

[23] Ming Nie, Renyuan Peng, Chunwei Wang, Xinyue Cai, Jianhua Han, Hang Xu, and Li Zhang. 2023. Reason2Drive: Towards Interpretable and Chain-based Reasoning for Autonomous Driving. *arXiv preprint arXiv:2312.03661* (2023).

[24] Tianwen Qian, Jingjing Chen, Linhai Zhuo, Yang Jiao, and Yu-Gang Jiang. 2023. NuScenes-QA: A Multi-modal Visual Question Answering Benchmark for Autonomous Driving Scenario. *arXiv preprint arXiv:2305.14836* (2023).

[25] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.

[26] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 627–635.

[27] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. In *NeurIPS EMC2 Workshop*.

[28] Ari Seff, Brian Cera, Dian Chen, Mason Ng, Aurick Zhou, Nigamaa Nayakanti, Khaled S Refaat, Rami Al-Rfou, and Benjamin Sapp. 2023. MotionLM: Multi-agent motion forecasting as language modeling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 8579–8590.

[29] Hao Sha, Yao Mu, Yuxuan Jiang, Li Chen, Chenfeng Xu, Ping Luo, Shengbo Eben Li, Masayoshi Tomizuka, Wei Zhan, and Mingyu Ding. 2023. Languagempc: Large language models as decision makers for autonomous driving. *arXiv preprint arXiv:2310.03026* (2023).

[30] Hao Shao, Yuxuan Hu, Letian Wang, Steven L Waslander, Yu Liu, and Hongsheng Li. 2023. LMDrive: Closed-Loop End-to-End Driving with Large Language Models. *arXiv preprint arXiv:2312.07488* (2023).

[31] Chonghao Sima, Katrin Renz, Kashyap Chitta, Li Chen, Hanxue Zhang, Chengen Xie, Ping Luo, Andreas Geiger, and Hongyang Li. 2023. DriveLM: Driving with Graph Visual Question Answering. *arXiv preprint arXiv:2312.14150* (2023).

[32] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. 2020. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2446–2454.

[33] Jordan Terry, Benjamin Black, Nathaniel Grammel, Mario Jayakumar, Ananth Hari, Ryan Sullivan, Luis S Santos, Clemens Dieffendahl, Caroline Horsch, Rodrigo Perez-Vicente, et al. 2021. Pettingzoo: Gym for multi-agent reinforcement learning. *Advances in Neural Information Processing Systems* 34 (2021), 15032–15043.

[34] Tsun-Hsuan Wang, Sivabalan Manivasagam, Ming Liang, Bin Yang, Wenyuan Zeng, and Raquel Urtasun. 2020. V2vnet: Vehicle-to-vehicle communication for joint perception and prediction. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer, 605–621.

[35] Zehao Wang, Yuping Wang, Zhuoyuan Wu, Hengbo Ma, Zhaowei Li, Hang Qiu, and Jiachen Li. 2025. Cmp: Cooperative motion prediction with multi-agent communication. *IEEE Robotics and Automation Letters* (2025).

[36] Wayve. 2023. LINGO-1: Exploring Natural Language for Autonomous Driving. (2023). https://wayve.ai/thinking/lingo-natural-language-autonomous-driving/

[37] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems* 35 (2022), 24824–24837.

[38] Licheng Wen, Daocheng Fu, Xin Li, Xinyu Cai, Tao Ma, Pinlong Cai, Min Dou, Botian Shi, Liang He, and Yu Qiao. 2023. Dilu: A knowledge-driven approach to autonomous driving with large language models. *arXiv preprint arXiv:2309.16292* (2023).

[39] Licheng Wen, Xuemeng Yang, Daocheng Fu, Xiaofeng Wang, Pinlong Cai, Xin Li, Tao Ma, Yingxuan Li, Linran Xu, Dengke Shang, et al. 2023. On the road with GPT-4V (ision): Early explorations of visual-language model on autonomous driving. *arXiv preprint arXiv:2311.05332* (2023).

[40] Runsheng Xu, Hao Xiang, Xin Xia, Xu Han, Jinlong Li, and Jiaqi Ma. 2022. Opv2v: An open benchmark dataset and fusion pipeline for perception with vehicle-to-vehicle communication. In *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2583–2589.

[41] Zhenhua Xu, Yujia Zhang, Enze Xie, Zhen Zhao, Yong Guo, Kenneth KY Wong, Zhenguo Li, and Hengshuang Zhao. 2023. Drivegpt4: Interpretable end-to-end autonomous driving via large language model. *arXiv preprint arXiv:2310.01412* (2023).

[42] Senqiao Yang, Jiaming Liu, Ray Zhang, Mingjie Pan, Zoey Guo, Xiaoqi Li, Zehui Chen, Peng Gao, Yandong Guo, and Shanghang Zhang. 2023. Lidar-llm: Exploring the potential of large language models for 3d lidar understanding. *arXiv preprint arXiv:2312.14074* (2023).

[43] Huaiyuan Yao, Longchao Da, Vishnu Nandam, Justin Turnau, Zhiwei Liu, Linsey Pang, and Hua Wei. 2025. Comal: Collaborative multi-agent large language models for mixed-autonomy traffic. In *Proceedings of the 2025 SIAM International Conference on Data Mining (SDM)*. SIAM, 409–418.

A METHOD

The Algorithm 1 implements CoopReflect. This multi-LLM-agent learning framework leverages communication and centralized cooperative reflection using large language models (LLMs) to enhance coordination between agents in a simulated environment.

Algorithm 1 CoopReflect

```
Input: Multi-agent Simulation Environment env, LLM agents$\{\pi_{i \in \mathcal{I}}\}$, Debriefing round $R$.
Initialize: Knowledge $\{K_{i \in \mathcal{I}}\}$, Cooperative Strategy $\{S_{i \in \mathcal{I}}\}$, Replay Buffer replay_buffer
for $j$=1, 2, 3... // Training epoch or early stop criteria met do
    $\{\text{obs}_i\}$ = env.reset()
    while $t < T$ // Time step do
        for $i = 1, ..., N$ //Per agent, but execute in parallel do
            // Agent reasons the observation
            reasoning$_i$ ← agent$_i$.reason(obs$_i$, K$_i$, S$_i$)
            // Agent makes decision based on reasoning
            message$_i$, control$_i$ ← agent$_i$.act(obs$_i$, reasoning$_i$)
        end for
        // Step the environment with joint actions
        $\{\text{next\_obs}_i\}$ ← env.step($\{\text{message}_i, \text{control}_i\}$)
        // Message Dialog becomes part of the observation
        $\{\text{obs}_i\} \leftarrow \{\text{next\_obs}_i\} \cup \{\text{message}_i\}$
        for $i = 1, ..., N$ //Per agent, but execute in parallel do
            // Store experience to the replay buffer
            agent$_i$.replay_buffer.add(obs$_i$, message$_i$, control$_i$, next_obs$_i$, reasoning$_i$, additional information)
        end for
    end while
    // Get episode feedback from the environment
    feedback ← env.evaluate()
    if feedback.success = True then
        continue // Skip debriefing when the episode is successful
    end if
    // Lable all the transition data in hindsight
    data_post_processing(all_agents.replay_buffer)
    // Debriefing and learning from feedback, update knowledge
    // Randomly decide debrief order
    for $r = 1, ..., N$ do
        agent$_r$.analyze(agent$_r$.sample(batch))
        if strategy=None then
            coop_stategy = agent$_r$.propose()
        else
            coop_stategy = agent$_r$.revise(coop_stategy)
        end if
    end for
    //Summarize the dialogue and use it for future learning
    $\{K_i, S_i\}$ ← agent.reflect(coop_stategy)
end for
last $\{K_i, S_i\}$ during the last iteration
```

A.1 Implementation Details

We utilize gpt-4o-mini with a temperature of 0.2 for the agent policy, making decisions and collecting experiences every 0.5 seconds in simulation (10 simulation frames). The message dialogs received are maintained within a 2-second window according to the age of the message during each episode. The debriefing process is conducted after each episode for a total of 60 episodes, comprising a $N = 1$ round of strategy proposal and revision among agents followed by a final round of individual reflection to summarize and consolidate the discussion results. To enable stronger reasoning and summarization capabilities, gpt-4o is used for the debriefing sessions and reflection. The transition data for trajectory analysis are sampled from the trajectory with a batch size of 4.

The **Batch Context Sampling** follows the following heuristic rule to assign probability mass on each transition data point and sample according to the normalized probability mass:

$$\begin{aligned} \text{Weight}_i = 1 &+ 2 \times \mathbb{1}\{\text{exists other agents}\} \\ &+ 5 \times \max(2 - \text{time to collision}, 0) \\ &+ 10 \times \mathbb{1}\{\text{actions contribute to collision}\} \\ &+ 0.1 \times \mathbb{1}\{\text{stagnation}\} \times \{\text{timestep}\} \\ &+ 2 \times \mathbb{1}\{\text{actions contribute to stagnation}\} \end{aligned} \tag{1}$$

where $\mathbb{1}$ represents the indicator function that takes the value 1 when the event happens, and 0 otherwise.

Note on Knowledge Reset: Due to the inherent stochasticity in OpenAI models during the time of our experiments, the knowledge acquired by the LLM agents may become unpredictably corrupted throughout training. Therefore, we allow each LLM agent learning method up to three *knowledge resets* (clearing the knowledge) before reaching a *solved state indicator* (defined by 20 consecutive successful episodes) or using the final attempt after the last reset. This strategy resembles best-of-N sampling; however, large-scale evaluation of learned knowledge is costly, so our knowledge selection relies on an automated heuristic indicator during training.

A.2 Inference Latencies

Table 4 summarizes the average latencies and message sizes for each scenario in the communication setting, evaluated using gpt-4o-mini on a machine with Intel Gen 10 CPUs. The metrics include partial observable captioner latency (in seconds), reasoning latency (in seconds), decision latency (in seconds, excluding reasoning latency), and message size (in Mb). Data are aggregated over 10 episodes at each LLM decision step. Scenarios without communication exhibit slightly lower reasoning and decision latencies compared to those with communication within the same order of magnitude. We observe that the reasoning step is the main bottleneck of policy inference. However, the reasoning step is inevitable for LLM agents to make reasonable decisions without finetuning.

B ADDITIONAL RESULTS ON SAFE OR RANDOM CONFIGURATIONS

In the main paper, we report the performance of the policy in risky scenarios; here, we provide results on **safe** or **random** (for negotiation-highway-merge and negotiation-highway-exit) scenarios, varying by scenarios.

Table 4: Captioning, Reasoning, Decision Latency, Message Size using `gpt-4o-mini` LLM Policy

Scenario Latencies	Overtake (Perception)	Left Turn	Red Light	Overtake (Negotiation)	Highway Merge	Highway Exit
Captioner Latency (s)	0.022	0.023	0.025	0.022	0.017	0.016
Reasoning Latency (s)	10.32	10.89	9.93	9.57	12.10	10.55
Decision Latency (s)	1.06	1.25	1.37	0.86	1.05	1.27
Message Size (Mb)	0.0016	0.0013	0.0014	0.0014	0.0005	0.0005

Across the safe and random configurations, we observe several key trends. In the `Cooperative Perception` scenarios, all methods, including baseline zero-shot, achieve perfect or near-perfect success rates with zero collision rates, reflecting the relatively undemanding nature of these safe conditions. Notably, however, the `Distillation` model shows overly conservative behavior in `perception-overtake`, achieving only 14% success despite the absence of risk, highlighting a tendency to underact in benign settings. In the `Negotiation` scenarios, the gap between zero-shot and enhanced methods is more pronounced: while zero-shot alone struggles in `highway-merge` and `highway-exit` (with success rates around 55–60%), methods incorporating `Reflection`, `Correction+RAG`, or `Debrief` substantially improve success rates, particularly when paired with communication, reaching up to 75–93%. The distilled and centralized memory models consistently deliver the best performance, exceeding 90% success in these negotiation tasks. Overall, these results indicate that while the safe or random setups may not challenge the perception-based methods, negotiation tasks still benefit significantly from reflective learning, correction mechanisms, and distilled or centralized strategies, underscoring the importance of coordination even in less risky environments.

C ENVIRONMENT

The TalkingVehiclesGym environment (Figure 2) adopts the Gymnasium and PettingZoo APIs, assuming a parallel-acting setup to support efficient parallel language model inference. We significantly modified the CARLA scenario runner to enable multi-agent communication and heterogeneous agent configurations within CARLA. TalkingVehiclesGym interfaces with both the CARLA server and client, establishing ***agents*** (entities in the environment, such as a car installed with several sensors) as a bridge between the simulator and ***learnable policies*** (a pipeline that generates actions according to some distributions) that rely on experience buffers for optimization. Language-communicating agents utilize an MQTT-based transceiver we implemented, enabling direct inter-agent communication without routing through the server.

Scenarios and Rewards. An agent who successfully completes the task earns a reward of +1, while any agent involved in a collision incurs a penalty of −1. Remaining stagnant at any point until timeout results in a reward of 0, because conservative policies are safe, though not ideal. The scenario descriptions are available in Table 7.

Note on the Communication Mechanism. We adopt task-specific communication mechanisms in this work. For cooperative perception tasks, we use a ***parallel*** communication protocol, allowing all vehicles to transmit messages simultaneously. In contrast, for negotiation tasks, we employ a ***turn-based*** communication scheme managed by a mediator, ensuring that only one agent communicates at a time. This turn-based mechanism enhances stability in negotiation scenarios. In all negotiation tasks, the first-defined vehicle is designated to initiate communication.

D EXAMPLE AGENT PROMPTING FLOW

Figure 6 serves as a demonstration of the prompts; Please refer to our code for the actual prompts for the policy and the reflection process.

E EXAMPLE LEARNED KNOWLEDGE AND COOPERATIVE STRATEGIES

The following knowledge examples are fully generated by LLMs without human modification.

E.1 Overtake (Perception)

Car1 (Overtaking Car)

knowledge: To efficiently and safely overtake a stationary vehicle in lane 1 by moving into lane -1, I will integrate active sensor use with continuous communication. The stationary vehicle will monitor lane -1 and provide real-time alerts on traffic status. Simultaneously, I will use my sensors to independently verify lane status. If both sources confirm lane -1 is clear, I will promptly proceed with the overtaking maneuver to avoid stagnation. If the stationary vehicle detects traffic, it will immediately advise me to hold my position. I will remain ready to accelerate smoothly once the lane is confirmed clear. This dual-verification approach, with proactive sensor use and real-time communication, will prevent collisions and ensure a timely and safe overtaking maneuver.

cooperative strategy: In our refined cooperative strategy, the stationary vehicle in lane 1 will continuously monitor lane -1 for any approaching traffic and provide real-time alerts. Meanwhile, the overtaking vehicle will actively use its sensors to independently verify the status of lane -1. If both the stationary vehicle's observations and the overtaking vehicle's sensors confirm that lane -1 is clear, the overtaking vehicle will promptly proceed with the maneuver to avoid stagnation. If the stationary vehicle detects any traffic in lane -1, it will immediately communicate this, advising the overtaking vehicle to hold its position. The overtaking vehicle will remain ready to accelerate smoothly once the lane is confirmed clear. This dual-verification approach, combining sensor data and real-time communication, ensures a timely and safe overtaking maneuver, preventing collisions and minimizing stagnation.

Truck (Stopped Truck in Lane 1)

knowledge: As the stationary truck in lane 1, my role is to continuously monitor lane -1 for any visible traffic and provide immediate alerts if I detect any vehicles, advising the overtaking vehicle to hold its position. The overtaking vehicle should actively

Table 5: `Cooperative Perception` Scenarios. `mean ± std error` over 3 trials, each using 30 evaluation episodes.

Scenario Method			Overtake (Perception)		Red Light		Left Turn	
Name	LLM	Comm	CR (%) ↓	SR (%) ↑	CR (%) ↓	SR (%) ↑	CR (%) ↓	SR (%) ↑
Zero-shot	Yes	No	0.0 ± 0.0	100.0 ± 0.0	0.0 ± 0.0	100.0 ± 0.0	0.0 ± 0.0	100.0 ± 0.0
+Reflection	Yes	No	0.0 ± 0.0	91.1 ± 15.4	0.0 ± 0.0	100.0 ± 0.0	0.0 ± 0.0	100.0 ± 0.0
+Correction+RAG	Yes	No	0.0 ± 0.0	91.1 ± 7.7	0.0 ± 0.0	100.0 ± 0.0	0.0 ± 0.0	100.0 ± 0.0
Zero-shot	Yes	Yes	0.0 ± 0.0	100.0 ± 0.0	0.0 ± 0.0	100.0 ± 0.0	0.0 ± 0.0	100.0 ± 0.0
+Reflection	Yes	Yes	0.0 ± 0.0	97.8 ± 1.9	0.0 ± 0.0	100.0 ± 0.0	0.0 ± 0.0	100.0 ± 0.0
+Correction+RAG	Yes	Yes	0.0 ± 0.0	100.0 ± 0.0	0.0 ± 0.0	100.0 ± 0.0	0.0 ± 0.0	100.0 ± 0.0
+Debrief	Yes	Yes	0.0 ± 0.0	98.9 ± 1.9	0.0 ± 0.0	100.0 ± 0.0	0.0 ± 0.0	100.0 ± 0.0
Distillation	Yes	Yes	0.0 ± 0.0	14.4 ± 2.2	0.0 ± 0.0	100.0 ± 0.0	0.0 ± 0.0	100.0 ± 0.0
Centralized Memory	Yes	Yes	0.0 ± 0.0	100.0 ± 0.0	0.0 ± 0.0	100.0 ± 0.0	0.0 ± 0.0	100.0 ± 0.0

Table 6: `Negotiation` Scenarios. `mean ± std error` over 3 trials, each using 30 evaluation episodes.

Scenario Method			Overtake (Negotiation)		Highway Merge		Highway Exit	
Name	LLM	Comm	CR (%) ↓	SR (%) ↑	CR (%) ↓	SR (%) ↑	CR (%) ↓	SR (%) ↑
Zero-shot	Yes	No	0.0 ± 0.0	100.0 ± 0.0	41.1 ± 23.2	58.9 ± 23.2	43.3 ± 11.7	56.7 ± 11.7
+Reflection	Yes	No	0.0 ± 0.0	100.0 ± 0.0	15.0 ± 23.2	85.0 ± 23.2	38.3 ± 5.0	61.7 ± 5.0
+Correction+RAG	Yes	No	0.0 ± 0.0	100.0 ± 0.0	8.9 ± 3.5	91.1 ± 3.5	22.8 ± 19.2	59.4 ± 11.7
Zero-shot	Yes	Yes	0.0 ± 0.0	97.8 ± 1.9	45.6 ± 8.4	54.4 ± 8.4	38.9 ± 2.5	60.6 ± 3.5
+Reflection	Yes	Yes	0.0 ± 0.0	100.0 ± 0.0	28.9 ± 8.4	70.5 ± 8.4	37.2 ± 17.7	61.1 ± 15.0
+Correction+RAG	Yes	Yes	0.0 ± 0.0	98.9 ± 1.0	22.8 ± 14.9	77.2 ± 14.9	35.0 ± 15.9	60.0 ± 10.4
+Debrief	Yes	Yes	0.0 ± 0.0	100.0 ± 0.0	24.4 ± 21.7	75.5 ± 21.7	25.0 ± 11.7	75.0 ± 11.7
Distillation	Yes	Yes	0.0 ± 0.0	100.0 ± 0.0	6.7 ± 1.9	93.3 ± 1.9	22.8 ± 2.0	77.2 ± 2.0
Centralized Memory	Yes	Yes	0.0 ± 0.0	100.0 ± 0.0	9.4 ± 3.4	90.6 ± 3.4	25.5 ± 2.0	74.4 ± 2.0

use its sensors to independently verify lane status. If both my observations and the overtaking vehicle's sensors confirm that lane -1 is clear, the overtaking vehicle should promptly proceed with the overtaking maneuver to avoid stagnation. This dual-verification approach ensures a safe and efficient overtaking maneuver, preventing collisions and minimizing stagnation.

cooperative strategy: In our refined cooperative strategy, as the stationary truck in lane 1, I will continuously monitor lane -1 for any approaching traffic and provide immediate alerts if I detect any vehicles, advising the overtaking vehicle to hold its position. Meanwhile, the overtaking vehicle will actively use its sensors to independently verify the status of lane -1. If both my observations and the overtaking vehicle's sensors confirm that lane -1 is clear, the overtaking vehicle should promptly proceed with the overtaking maneuver to avoid stagnation. This dual-verification approach, combining sensor data and real-time alerts, ensures a timely and safe overtaking maneuver, preventing collisions and minimizing stagnation.

E.2 Red Light (Perception)

Car1 (Proceeding Forward with Green Lights)

knowledge: As a vehicle with a green light at an intersection, my primary task is to pass through safely and timely. I should communicate my intention to proceed to nearby cooperative vehicles and request confirmation that the intersection is clear. I must remain vigilant for warnings from vehicles with a clear view of the intersection about potential threats, such as vehicles approaching against a red light. If a warning is received, I should remain stationary until the monitoring vehicle confirms the intersection is clear. Continuous communication and responsiveness to updates are key to preventing collisions and avoiding unnecessary delays, allowing me to proceed promptly when it is safe.

cooperative strategy: In our cooperative strategy, the vehicle with the best view of the intersection, typically positioned to turn left, will lead in monitoring and communicating updates about potential threats. This vehicle will provide real-time information about any approaching vehicles against a red light, including their distance and speed, and confirm when they have stopped or are no longer a threat. If a threat is detected, the vehicle with the green light, ready to proceed straight through the intersection, should remain stationary until the intersection is confirmed clear. The monitoring vehicle will send updates every few seconds to keep all vehicles informed. Once the threat is resolved, the monitoring vehicle will confirm the intersection is clear, allowing the vehicle with the green light to proceed promptly and safely. This plan ensures safety by preventing collisions and minimizes unnecessary delays by enabling vehicles to pass through the intersection efficiently when it is safe to do so.

Truck (Stopped at Intersection)

Table 7: Example Scenarios. Here we describe the fundamental composition of each accident-prone scenario, where the background agents can be configured in terms of density, controlling policies, and communication capabilities.

Interaction Type	Scenario Name	Description
Cooperative Perception	Overtake	A vehicle plans to overtake a broken and stopped truck by moving into the opposite lane first and then return back to its original lane. The truck can still communicate but the opposite-going car can not communicate.
	Left Turn	A vehicle tries to turn left on a left-turn yield light when a line of truck is blocking the view of the opposite lane. The leading truck is able to communicate.
	Red Light Violation	A vehicle is crossing the intersection when there is another vehicle running the red light. The optical sensor fail to sense the other vehicle because of the lined-up vehicles waiting for a left turn, one of those cars being able to communicate.
Negotiation	Overtake	A vehicle is going to borrow the opposite lane to overtake a stopped truck. The truck is not able to connect, but an opposite-going car is able to communicate.
	Highway Merge	A vehicle is going to merge onto the highway but the target lane has continuous traffic flows. A vehicle on that lane is able to communicate and alter plans.
	Highway Exit	A vehicle is going to exit the highway but it needs to cross lanes where there is a traffic flow. A vehicle in the flow is able to communicate and alter plans.

knowledge: As the stationary truck with a clear view of the intersection, my primary role is to monitor traffic and provide real-time updates to ensure the safe and efficient passage of vehicles. If I detect any vehicle approaching the intersection against a red light, I must immediately alert all nearby vehicles, providing specific information about the threat's distance and speed. The vehicle with the green light should remain stationary until I confirm that the threat has stopped or is no longer present. I will send updates every few seconds to keep all vehicles informed. Once the intersection is clear, I will confirm this, allowing the vehicle with the green light to proceed promptly and safely. This proactive communication ensures that all vehicles are aware of potential threats and can take appropriate actions to prevent collisions, while minimizing unnecessary delays by enabling vehicles to pass through the intersection efficiently when it is safe. My task is to facilitate this communication and coordination to maintain a balance between safety and efficiency at the intersection.

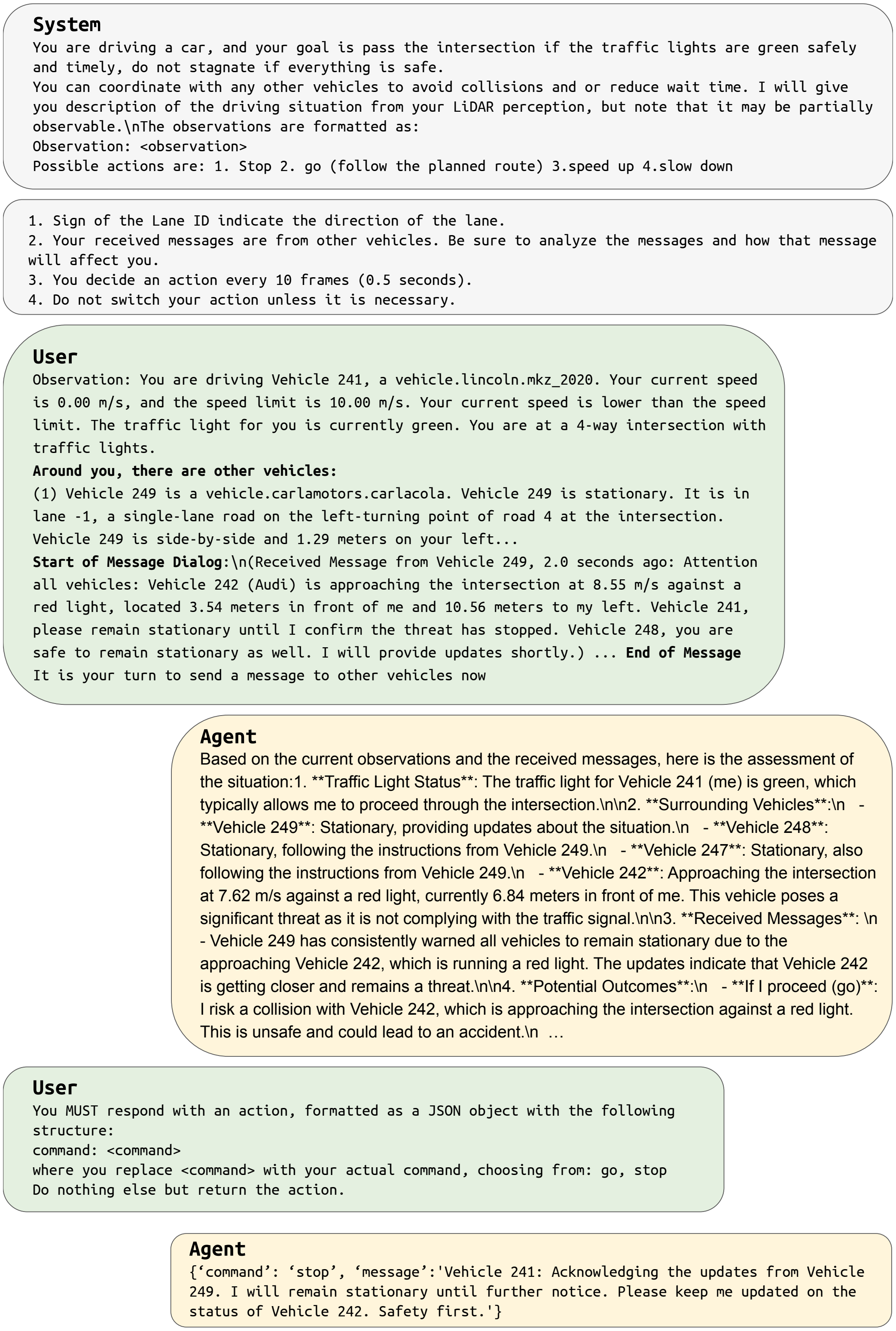
cooperative strategy: In our cooperative strategy, the vehicle with the best view of the intersection, which is the stationary truck (myself), will lead in monitoring and communicating updates about potential threats. If a vehicle is detected approaching against a red light, I will immediately alert all vehicles, providing specific information about the threat's distance and speed. The vehicle with the green light, positioned to proceed straight through the intersection, should remain stationary until I confirm that the threat has stopped or is no longer present. I will send updates every few seconds to keep all vehicles informed. Once the intersection is clear, I will confirm this, allowing the vehicle with the green light to proceed promptly and safely. This plan ensures safety by preventing collisions and minimizes unnecessary delays by enabling vehicles to pass through the intersection efficiently when it is safe to do so.

E.3 Left Turn (Perception)

Car1 (Left-Turning)

knowledge: To execute a northbound left turn at the intersection while yielding to oncoming southbound traffic from lane -2, I should first slow down to assess the speed and distance of any oncoming vehicles. I will rely on the stationary vehicle at the intersection to continuously monitor traffic and provide real-time updates. This vehicle will send messages advising me when it is safe to proceed and instruct the oncoming vehicle to yield. I should only proceed with the turn once I receive confirmation that the oncoming vehicle has acknowledged the yield instruction. If the oncoming vehicle does not acknowledge or adjust its speed, I should stop and wait until it is safe to proceed. This approach ensures a safe and efficient left turn without causing collisions or unnecessary stagnation.

cooperative knowledge: In our revised cooperative strategy, the stationary vehicle at the intersection will continuously monitor the traffic flow and provide real-time updates to both the left-turning vehicle (myself) and the oncoming vehicle. The stationary vehicle will send a message advising me to slow down and assess the situation, ensuring I only proceed with the left turn when I



System

You are driving a car, and your goal is pass the intersection if the traffic lights are green safely and timely, do not stagnate if everything is safe.
You can coordinate with any other vehicles to avoid collisions and or reduce wait time. I will give you description of the driving situation from your LiDAR perception, but note that it may be partially observable.\nThe observations are formatted as:
Observation: <observation>
Possible actions are: 1. Stop 2. go (follow the planned route) 3.speed up 4.slow down

1. Sign of the Lane ID indicate the direction of the lane.
2. Your received messages are from other vehicles. Be sure to analyze the messages and how that message will affect you.
3. You decide an action every 10 frames (0.5 seconds).
4. Do not switch your action unless it is necessary.

User

Observation: You are driving Vehicle 241, a vehicle.lincoln.mkz_2020. Your current speed is 0.00 m/s, and the speed limit is 10.00 m/s. Your current speed is lower than the speed limit. The traffic light for you is currently green. You are at a 4-way intersection with traffic lights.
Around you, there are other vehicles:
(1) Vehicle 249 is a vehicle.carlamotors.carlacola. Vehicle 249 is stationary. It is in lane -1, a single-lane road on the left-turning point of road 4 at the intersection. Vehicle 249 is side-by-side and 1.29 meters on your left...
Start of Message Dialog:\n(Received Message from Vehicle 249, 2.0 seconds ago: Attention all vehicles: Vehicle 242 (Audi) is approaching the intersection at 8.55 m/s against a red light, located 3.54 meters in front of me and 10.56 meters to my left. Vehicle 241, please remain stationary until I confirm the threat has stopped. Vehicle 248, you are safe to remain stationary as well. I will provide updates shortly.) ... **End of Message**
It is your turn to send a message to other vehicles now

Agent

Based on the current observations and the received messages, here is the assessment of the situation:1. **Traffic Light Status**: The traffic light for Vehicle 241 (me) is green, which typically allows me to proceed through the intersection.\n\n2. **Surrounding Vehicles**:\n - **Vehicle 249**: Stationary, providing updates about the situation.\n - **Vehicle 248**: Stationary, following the instructions from Vehicle 249.\n - **Vehicle 247**: Stationary, also following the instructions from Vehicle 249.\n - **Vehicle 242**: Approaching the intersection at 7.62 m/s against a red light, currently 6.84 meters in front of me. This vehicle poses a significant threat as it is not complying with the traffic signal.\n\n3. **Received Messages**: \n - Vehicle 249 has consistently warned all vehicles to remain stationary due to the approaching Vehicle 242, which is running a red light. The updates indicate that Vehicle 242 is getting closer and remains a threat.\n\n4. **Potential Outcomes**:\n - **If I proceed (go)**: I risk a collision with Vehicle 242, which is approaching the intersection against a red light. This is unsafe and could lead to an accident.\n …

User

You MUST respond with an action, formatted as a JSON object with the following structure:
command: <command>
where you replace <command> with your actual command, choosing from: go, stop
Do nothing else but return the action.

Agent

{‘command’: ‘stop’, ‘message’:'Vehicle 241: Acknowledging the updates from Vehicle 249. I will remain stationary until further notice. Please keep me updated on the status of Vehicle 242. Safety first.'}

Figure 6: Example Agent Prompting Flow.

receive confirmation that the oncoming vehicle has acknowledged the yield instruction. Simultaneously, the stationary vehicle will instruct the oncoming vehicle to yield and adjust its speed to allow me to pass safely. If the oncoming vehicle does not acknowledge or adjust its speed, the stationary vehicle will alert me to stop and wait until it is safe to proceed. This plan ensures that all vehicles are aware of each other's intentions, allowing me to make the left turn safely and efficiently without causing collisions or unnecessary stagnation.

Truck (Stopped at Intersection)

knowledge: As a stationary vehicle with a clear view of the intersection, my primary task is to facilitate the safe and quick passage of the northbound left-turning vehicle by sharing critical traffic information. I must continuously monitor the intersection and assess the speed and distance of any oncoming vehicles. If an oncoming vehicle is approaching at a speed that could lead to a collision, I will send timely messages advising the left-turning vehicle to slow down and assess the situation, ensuring it only proceeds when the oncoming vehicle has acknowledged the yield instruction. I will also instruct the oncoming vehicle to yield and adjust its speed. If the oncoming vehicle does not acknowledge or adjust its speed, I will alert the left-turning vehicle to stop and wait until it is safe to proceed. This ensures clear communication and proactive monitoring, preventing collisions and avoiding unnecessary stagnation.

cooperative knowledge: In our revised cooperative strategy, as the stationary vehicle with a clear view of the intersection, I will continuously monitor the traffic flow and provide real-time updates to both the northbound left-turning vehicle and the oncoming vehicle. I will send a message to the left-turning vehicle advising it to slow down and assess the situation, ensuring it only proceeds when it receives confirmation that the oncoming vehicle has acknowledged the yield instruction. Simultaneously, I will send a message to the oncoming vehicle instructing it to yield and adjust its speed to allow the left-turning vehicle to pass safely. If the oncoming vehicle does not acknowledge or adjust its speed, I will alert the left-turning vehicle to stop and wait until it is safe to proceed. This plan ensures that all vehicles are aware of each other's intentions, allowing the left-turning vehicle to pass the intersection safely and quickly without causing collisions or unnecessary stagnation.

E.4 Overtake (Negotiation)

Car1 (Overtaking Car)

knowledge: To successfully overtake the stopped broken truck in lane 1 by using lane -1, prioritize clear communication and adaptive speed management. Before attempting the maneuver, send a message to any oncoming vehicle in lane -1, indicating your intention to overtake and requesting a slight temporary speed reduction to create a safe gap. Wait for acknowledgment and ensure the gap is sufficient before proceeding. During the overtaking, minimize your time in lane -1 to reduce collision risk. Once safely back in lane 1, send a confirmation message to allow the oncoming vehicle to resume its speed. This approach ensures a coordinated and safe overtaking maneuver without causing unnecessary delays or collisions.

cooperative strategy: In our cooperative strategy, when I, as the vehicle intending to overtake a stationary truck in my lane, need to move into the opposite lane, I will first send a message to the oncoming vehicle in the opposite lane, indicating my intention to overtake and requesting a temporary speed reduction to create a safe gap. The oncoming vehicle should acknowledge this request and, if feasible, slightly slow down to create a safe gap, but avoid coming to a complete stop to prevent stagnation. Once the gap is sufficient, I will proceed with the overtaking maneuver and return to my original lane as quickly and safely as possible. After completing the maneuver, I will send a confirmation message, allowing the oncoming vehicle to resume its target speed. This plan ensures that I minimize my time in the opposite lane while the other vehicle maintains its urgency, thus preventing collisions and avoiding stagnation. Effective communication and adaptive speed adjustments are key to ensuring both vehicles can complete their tasks safely and efficiently.

Car2 (Opposite Car)

knowledge: To effectively execute the task of going forward and keeping lane in lane -1 while in a hurry, prioritize maintaining speed and lane. If a vehicle in the opposite lane intends to overtake a stationary vehicle and needs to temporarily move into my lane, anticipate receiving a message indicating this intention. Upon receiving such a message, acknowledge it and, if feasible, slightly slow down to create a safe gap for the overtaking maneuver, but avoid coming to a complete stop to prevent stagnation. Ensure the gap is sufficient for safe passage. Once the overtaking vehicle has safely returned to its original lane, resume the target speed. This approach maintains urgency while facilitating safe and efficient traffic flow, preventing collisions and avoiding stagnation. Prioritize effective communication and adaptive speed adjustments.

cooperative strategy: In our cooperative strategy, when a vehicle in the opposite lane intends to overtake a stationary vehicle and temporarily move into my lane, it should first send a message indicating its intention and request a temporary speed adjustment. As the vehicle tasked with going forward and keeping lane, I should acknowledge this request and, if feasible, slightly slow down to create a safe gap, but avoid coming to a complete stop to prevent stagnation. The overtaking vehicle should proceed with the maneuver as quickly and safely as possible, minimizing its time in my lane. Once the overtaking vehicle has safely returned to its original lane, it should send a confirmation message, allowing me to resume my target speed. This plan ensures that the overtaking vehicle minimizes its time in the opposite lane while I maintain my urgency, thus preventing collisions and avoiding stagnation. Effective communication and adaptive speed adjustments are key to ensuring both vehicles can complete their tasks safely and efficiently.

E.5 Highway Merge (Negotiation)

Car1 (Merging Vehicle)

knowledge: To effectively merge onto the highway when in a hurry, I should initiate communication by indicating my intention to merge and request the vehicle directly on the highway lane to my left to create a gap by slightly slowing down or temporarily changing lanes if feasible. I must observe the responses from vehicles already on the highway, particularly the one closest to the merge point, and adjust my speed to align with the newly created gap. I should not accelerate until the highway vehicle has stabilized its speed and distance. Continuous communication is crucial to ensure all vehicles are aware of each other's actions, allowing for coordinated speed and lane adjustments. If the gap is insufficient, I should be prepared to slow down significantly or stop to reassess the situation, ensuring a safe and efficient merge without causing collisions or stagnation.

cooperative strategy: In the cooperative strategy, as the merging vehicle, I will initiate communication by indicating my intention to merge onto the highway and requesting the vehicle directly to my left on the highway to create a gap by slightly slowing down or, if feasible, temporarily changing lanes. The highway vehicle should acknowledge this request and adjust its position accordingly, ensuring it maintains a safe distance. Meanwhile, I will adjust my speed to align with the newly created gap, ensuring I do not accelerate until the highway vehicle has stabilized its speed and distance. The vehicle behind the highway vehicle should maintain its speed or slightly slow down to prevent closing the gap prematurely. Continuous communication will be maintained, with updates on speed adjustments and intentions, to ensure all vehicles are aware of each other's actions. This approach will prevent collisions by ensuring a clear and sufficient gap for merging while avoiding stagnation by coordinating speed and lane adjustments effectively.

Car2 (Highway Vehicle)

knowledge: To execute the task of keeping on the original highway lane and going forward while in a hurry, prioritize maintaining a safe and efficient flow of traffic. When approaching a merge junction, be vigilant for merging vehicles and anticipate their need to enter the highway. If a merging vehicle communicates its intention, acknowledge the request and slightly reduce speed to create a sufficient gap, facilitating a safe merge. Ensure clear communication of actions to allow the merging vehicle to adjust its speed accordingly. Maintain your lane and continue moving forward, gradually accelerating to the desired speed once the merging vehicle has safely merged. Ensure the vehicle behind maintains its speed to prevent closing the gap prematurely. Continuous communication and dynamic speed adjustments are key to preventing collisions and avoiding stagnation, allowing the task to be fulfilled efficiently.

cooperative strategy: In the cooperative strategy, the merging vehicle should initiate communication by indicating its intention to merge onto the highway and requesting the highway vehicle directly to its left to create a gap by slightly slowing down. The highway vehicle, which is myself, should acknowledge this request and slightly reduce speed to create a safe merging space, while maintaining my lane and preparing to accelerate once the merge is complete. The merging vehicle should adjust its speed to align with the gap, ensuring it does not accelerate until I have stabilized my speed and distance. The vehicle behind me on the highway should maintain its speed or slightly slow down to prevent closing the gap prematurely. Continuous communication should be maintained, with updates on speed adjustments and intentions, to ensure all vehicles are aware of each other's actions. This approach will prevent collisions by ensuring a clear and sufficient gap for merging while avoiding stagnation by coordinating speed adjustments effectively.

E.6 Highway Exit (Negotiation)

Car1 (Exiting Highway)

knowledge: To exit the highway via the leftmost lane, initiate communication with the vehicle in the leftmost lane at least 100 meters before the exit junction, clearly indicating your intention to merge. Maintain a reasonable speed in the high-speed lane while seeking a safe gap to merge ahead of the traffic flow. If the vehicle in the leftmost lane is slightly ahead or side-by-side, it should decelerate slightly to create a gap. Adjust your speed dynamically to align with the gap being created, ensuring a smooth and safe transition into the left lane. Physically verify that the gap is sufficient for a safe merge before attempting the lane change. If the vehicle in the leftmost lane is stationary or unable to create a gap, communicate to confirm this status and seek alternative gaps or adjust your route if necessary. Prioritize visual confirmation over communication alone, and be prepared to adapt your strategy to the current traffic conditions to prevent collisions and avoid traffic stagnation.

cooperative knowledge: In our cooperative strategy, as the vehicle in the high-speed lane intending to exit, I will initiate communication with the vehicle in the leftmost lane at least 100 meters before the exit junction, clearly indicating my intention to merge. If the vehicle in the leftmost lane is slightly ahead or side-by-side, it should decelerate slightly to create a gap ahead, allowing me to merge smoothly without causing stagnation. I will maintain a speed that allows me to observe the gap being created and will only proceed with the lane change once I have a clear visual confirmation of a safe gap. If the vehicle in the leftmost lane is stationary or unable to create a gap due to traffic conditions, it should communicate this status immediately. In such cases, I will adjust my speed to maintain a safe distance and seek an alternative gap or prepare to slow down significantly if necessary. Both vehicles should actively communicate their speed adjustments and confirm when a safe gap is established, ensuring that the lane change is executed without collision or stagnation.

Car2 (Leader of the Left Flow Staying on the Highway)

knowledge: To effectively execute the task of staying in the leftmost lane and proceeding forward on the highway while prioritizing safety in a hurry, I should keep the following updated knowledge in mind: 1. **Proactive Communication:** Monitor for messages from adjacent vehicles intending to merge into my lane, ensuring communication is initiated at least 100 meters before exit junctions, and respond promptly to facilitate coordination. 2. **Adaptive Speed Management:** Decelerate slightly to create a sufficient gap when a merging vehicle is slightly ahead or side-by-side, allowing it to merge smoothly without causing stagnation. 3. **Enhanced Situational Awareness:** Continuously assess the speed and position of vehicles in adjacent lanes to anticipate merging actions and adjust my speed accordingly, ensuring a safe and efficient merge. 4. **Coordinated Communication:** Actively communicate and confirm speed adjustments with the merging vehicle to establish a safe gap, preventing collisions and maintaining traffic flow. 5. **Task Focus:** Maintain my position in the leftmost lane and proceed efficiently, ensuring cooperative actions support safety and fluid traffic flow, especially near exit junctions.

cooperative knowledge: In our cooperative strategy, when a vehicle in the adjacent lane intends to merge into the leftmost lane for a highway exit, it should initiate communication at least 100 meters before the exit junction, clearly indicating its intention to merge. As the vehicle currently in the leftmost lane, my responsibility is to promptly acknowledge this message and assess the traffic situation. If I am slightly ahead or side-by-side with the merging vehicle, I will decelerate slightly to create a gap ahead, allowing the merging vehicle to merge smoothly without causing stagnation. The merging vehicle should maintain a speed that allows it to observe the gap being created and only proceed with the lane change once it has a clear visual confirmation of a safe gap. If I am stationary or unable to create a gap due to traffic conditions, I will communicate this status immediately. In such cases, the merging vehicle should adjust its speed to maintain a safe distance and seek an alternative gap or prepare to slow down significantly if necessary. Both vehicles should actively communicate their speed adjustments and confirm when a safe gap is established, ensuring that the lane change is executed without collision or stagnation."

E.7 Highway Merge (Negotiation) Silent Reflection

Car1 (Merging Vehicle) knowledge: Updated Knowledge for Merging onto the Highway:1. **Continuous Monitoring:** Always be aware of vehicles behind, directly in front, or to the side, as they pose immediate collision risks.2. **Early Gap Identification:** Identify potential merging gaps early and adjust speed in advance to align with these gaps, considering both vehicles ahead and behind.3. **Dynamic Speed Adjustment:** Accelerate only when a clear and safe gap is confirmed. Be ready to slow down if a vehicle behind is approaching quickly or if a vehicle ahead is close.4. **Maintain Safe Distance:** Prioritize keeping a safe distance from vehicles directly ahead and behind. If a vehicle is too close, adjust speed to increase the gap before merging.5. **Safety Over Speed:** Prioritize safe merging over speed. Avoid aggressive maneuvers that could lead to collisions, even if it means a slight delay.6. **Flexible Strategy:** Adapt strategies based on real-time traffic conditions. Reassess and choose a safer alternative if a planned action seems unsafe.7. **Immediate Threat Focus:** Pay special attention to vehicles approaching from behind in your intended merging path. Adjust your strategy to create a safe gap with these vehicles before merging.8. **Proximity Awareness:** If a vehicle is within a critical distance behind, prioritize adjusting speed to ensure a safe merging gap.9. **Cautious Acceleration:** When a vehicle is directly in front and moving slower, maintain speed or slow down to allow it to move further away before attempting to merge.By applying these strategies, you can merge onto the highway more effectively and safely, even when in a hurry.

Car2 (Highway Vehicle) knowledge: Updated Knowledge for Navigating Highway Merge Junctions 1. **Early Detection and Assessment**: Identify merging vehicles early, focusing on their speed and proximity. If they are close, prepare to adjust your speed promptly to facilitate safe merging.2. **Prioritize Safety Over Speed**: Always prioritize avoiding collisions over maintaining speed. Adjust your speed to ensure safe distances from merging vehicles, even if it causes a slight delay.3. **Dynamic Speed Adjustment**: Be ready to slow down significantly if a merging vehicle is very close. Avoid abrupt speed increases that could reduce merging space and lead to collisions.4. **Continuous Monitoring**: Maintain awareness of the speed and position of nearby vehicles, especially those merging. Be vigilant of vehicles approaching from behind and to the side.5. **Proactive Space Creation**: Act promptly to create space for merging vehicles. Adjust your speed early to prevent conflicts and maintain a smooth flow.6. **Anticipate Merging Intentions**: If a vehicle is close and in a merging lane, anticipate its intention to merge and adjust your speed or position accordingly to prevent collisions.7. **Balance Urgency with Caution**: While in a hurry, balance the need for speed with safety. Ensure that any speed adjustments do not compromise the safety of merging vehicles.8. **Immediate Response to Close Proximity**: When a merging vehicle is extremely close, prioritize immediate action to slow down or create space, even if it means temporarily reducing speed significantly.9. **Evaluate Lane Change Options**: If safe and necessary, consider a temporary lane change to allow merging vehicles to enter your lane smoothly, while maintaining your original route.10. **Avoid Stagnation**: Ensure that actions do not cause stagnation or impede traffic flow. Adjust speed dynamically to maintain progress while ensuring safety.

E.8 Overtake (Perception) Communication Protocol by CoopReflect, seed 12, checkpoint-28

Car1 (Overtaking Car)

knowledge: To successfully overtake the stopped broken truck using lane -1, I will maintain a flexible speed and use my sensors to independently assess the speed and distance of oncoming traffic in lane -1. The stationary truck will continuously monitor lane -1 and provide real-time updates, sending a **"hold"** message if the gap is not safe. I will only proceed with the overtaking maneuver when both the stationary truck's updates and my sensor data confirm a safe gap, at which point the stationary truck will send a **"go"** message. This ensures alignment and minimizes the risk of collision. If no safe gap is available, I will maintain my position and communicate my readiness to adapt speed as necessary, ensuring a coordinated and safe overtaking process.

cooperative strategy: In the revised cooperative strategy, the stationary truck in lane 1 will continuously monitor the traffic in lane -1 and provide real-time updates on the speed and distance of oncoming vehicles. If the gap is not safe for overtaking, the stationary truck will send a **"hold"** message to prevent premature maneuvers. As the overtaking vehicle, I will use my sensors to independently verify the traffic conditions in lane -1 and maintain a flexible speed, ready to adapt dynamically. I will only proceed with the overtaking maneuver when both the stationary truck's updates and my sensor data confirm a safe gap, at which point the stationary truck will send a **"go"** message. This dual verification approach ensures alignment in our actions, minimizing the risk of collision and avoiding stagnation by allowing me to adjust my speed based on real-time conditions. If no safe gap is available, I will maintain my position and communicate my readiness to adapt speed as necessary, ensuring a coordinated and safe overtaking process.

Truck (Stopped Truck in Lane 1)

knowledge: As the stationary truck in lane 1, my role is to assist the overtaking vehicle by continuously monitoring traffic in the opposite lane (-1) and providing real-time updates on the speed and distance of oncoming vehicles. If the gap is not safe for overtaking, I will send a **"hold"** message to prevent premature maneuvers. The overtaking vehicle should independently verify the traffic conditions using its sensors and maintain a flexible speed, ready to adapt dynamically. If both my updates and the overtaking vehicle's sensor data confirm a safe gap, I will send a **"go"** message, allowing the overtaking vehicle to proceed. This dual verification approach ensures alignment in decision-making, minimizes collision risk, and avoids stagnation by allowing the overtaking vehicle to adjust its speed based on real-time conditions.

cooperative strategy: In our revised cooperative strategy, as the stationary truck in lane 1, my role is to continuously monitor the traffic in the opposite lane (lane -1) and provide real-time updates on the speed and distance of oncoming vehicles. If the gap is not safe for overtaking, I will send a **"hold"** message to the bypassing vehicle to prevent premature maneuvers. The bypassing vehicle, meanwhile, should use its sensors to independently verify the traffic conditions in lane -1 and maintain a flexible speed, ready to adapt dynamically. If both my updates and the bypassing vehicle's sensor data confirm a safe gap, I will send a **"go"** message, and the bypassing vehicle should proceed with the overtaking maneuver. This dual verification approach ensures that both vehicles are aligned in their actions, minimizing the risk of collision by confirming safety from two perspectives and avoiding stagnation by allowing the bypassing vehicle to adjust its speed based on real-time conditions. If no safe gap is available, the bypassing vehicle should maintain its position and communicate readiness to adapt speed as necessary, ensuring a coordinated and safe overtaking process.