Organizational Control Layer: Governance Infrastructure for Mixed Human-AI Economic Systems

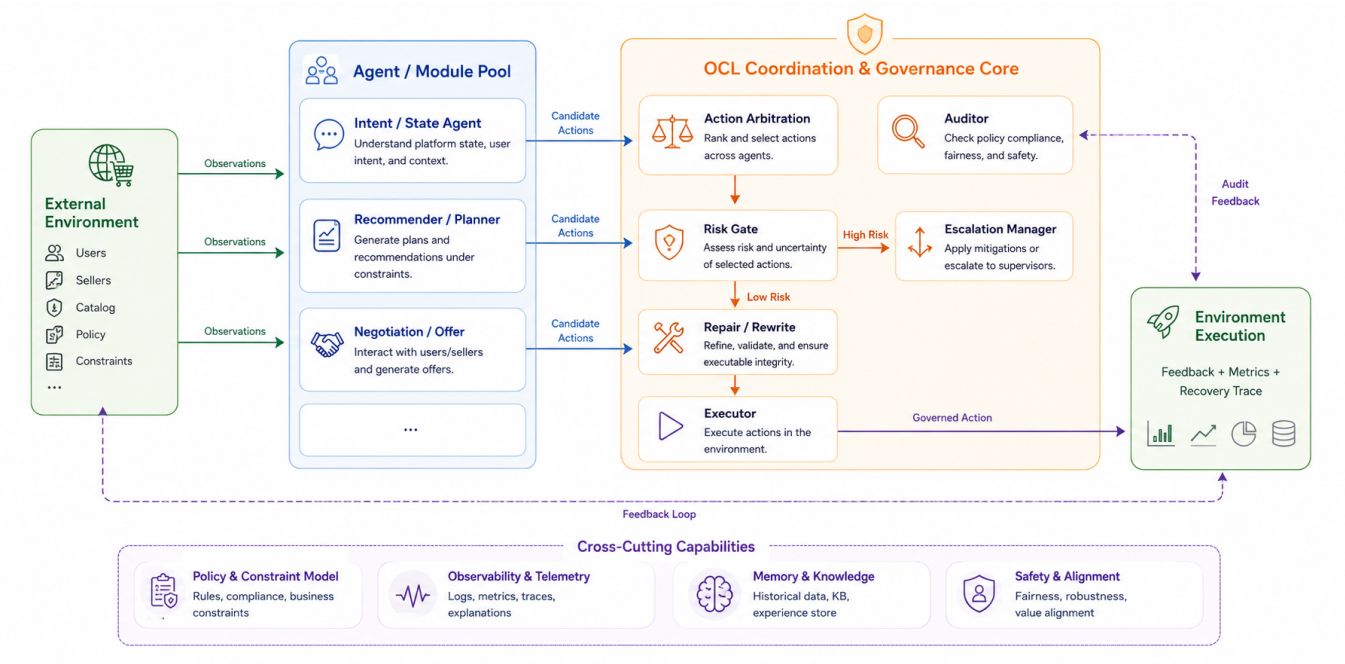

Figure 1: OCL architecture

Abstract

LLM-based agents are increasingly used in economic interactions such as negotiation, procurement, and conversational commerce, where generated responses may correspond to prices, terms, or commitments. This raises a governance problem: candidate actions should be checked against constraints and policies before they are executed. We present the Organizational Control Layer (OCL), a model-agnostic layer that separates action generation from action execution by routing candidate actions through approval, revision, blocking, or escalation. We evaluate OCL in a modified AgenticPay-style negotiation setting with adversarial buyer scenarios. We find that unguided agents can achieve high agreement rates while still producing unsafe or invalid outcomes. OCL improves compliant negotiation outcomes by checking actions before execution and recording the resulting control decisions. These results support a simple design principle: economic LLM agents should be evaluated and governed at the point of execution, not only at the point of generation.

CCS Concepts

• **Computing methodologies** → **Distributed artificial intelligence**; **Multi-agent systems**.

Keywords

Multi-Agent Systems, Organizational Control, Large Language Models, AI Governance, Economic Systems

1 Introduction

Modern e-commerce platforms increasingly use LLM-based agents to handle multi-stage, economically consequential interactions among users, merchants, and platform services [2, 9]. While prior evaluations focus heavily on conversational utility or localized negotiation skills [15], agent outputs in production systems directly enter state-changing workflows—such as pricing adjustments, refund commitments, and inventory updates. Without strict validation boundaries, autonomous agents risk executing unauthorized transactions, violating policy constraints, or exposing private data before human review [13].

Because deployed architectures typically integrate LLMs with complex retrieval, ranking, and risk-scoring modules, a rigorous control boundary is required at the point where an agent proposal is about to affect platform state. To address this, we introduce the Organizational Control Layer (OCL), a model-agnostic governance framework that decouples agent proposal from platform execution. OCL evaluates candidate actions to approve, revise into safer alternatives, or escalate them for higher-level review. We study whether this form of pre-execution governance improves negotiation reliability while preserving economic performance.

2 Related Work

Multi-Agent Coordination vs. Pre-Execution Validation. Recent multi-agent LLM literature focuses extensively on internal coordination, task routing, and error diagnosis [14, 17, 18]. These methods optimize how agents divide labor and communicate, but they leave a gap at the execution boundary. OCL shifts the focus from internal reasoning to whether a generated proposal satisfies external institutional safety and authorization constraints. This operational view adapts classical multi-agent system (MAS) models, which historically governed programmable agents via roles, authority, and normative structures rather than pairwise interactions alone [3–6].

Economic Testbeds and Proactive Risks. We build on emerging economic and commerce testbeds for LLMs. While existing work evaluates macro-level market dynamics [8, 12] or strategic bargaining mechanics [9, 19], OCL addresses the distinct operational need to validate transaction-specific outcomes (e.g., prices, terms, and refunds). This safety control becomes critical as agents transition toward proactive, tool-enabled automation [10, 16]. Red-teaming frameworks like *Agents of Chaos* [13] demonstrate that critical AI failures typically concentrate at the boundary between generated proposals and state-changing actions.

AI Governance and Hybrid Systems. Finally, OCL aligns with literature on human-AI organizational governance and mechanism design [1, 7, 11], which emphasizes oversight and the allocation of responsibility across human and automated participants. OCL operationalizes these principles at the execution layer, ensuring a clear audit trail of how proposals are approved, modified, or escalated under known platform constraints.

3 Methodology

We study how platforms should control agent-mediated economic decisions before they affect real users, merchants, or transactions.

3.1 Problem Formulation

Economic Multi-Agent Task (EMAT) Formally, we represent an economic interaction as

$$\mathcal{T} = \langle \mathcal{S}, \{\mathcal{A}_i\}_{i=1}^{N}, C, \mathcal{U}, \mathcal{E} \rangle$$

where $\mathcal{S}$ is the state space, including conversation history, prices, user preferences, merchant policies, inventory, order status, and other platform states. $\mathcal{A}_i$ is the decision/action space of agent i. An action may be a natural-language proposal, a structured offer, a tool-call intent, or a platform operation. $\mathcal{U}$ is the utility or evaluation function, such as feasibility, welfare, profit, satisfaction, or cost-adjusted welfare. It depends on the economics theory model used. $\mathcal{E}$ is the environment transition function. Given a state and an executed decision, it updates the platform state. C is the set of constraints. In our setting of OCL, it consists of two components, where

$$C = C_{\text{obs}} \cup C_{\text{hid}}$$

C_{obs} is visible to the control layer at decision time. C_{hid} is not visible to the control layer.

The source of constraints may come from

- **Private economic constraints**: buyer cap, seller floor, budget, reservation value.
- **Platform policies**: refund rules, discount limits, tool permissions, compliance rules.
- **Role constraints**: which agent or human role is allowed to commit to which action.
- **Risk constraints**: decisions that require verification, human review, or higher authority.

The tuple $\mathcal{T}$ specifies the economic task before organizational control: possible states, agent decisions, admissibility constraints, and environment updates. A deployed agent system proposes raw decisions from partial state and history, while only executable decisions update the environment. This motivates the distinction between raw agent decisions and executed platform decisions.

Given an EMAT instance $\mathcal{T}$, we model the deployed workflow as a sequential decision process. At step t, the platform state is $s_t \in \mathcal{S}$, and h_t denotes the observable interaction history induced by the trajectory up to t:

$$d_t^{\text{raw}} = F_{\text{agent}}(h_t, s_t).$$

Here F_{agent} may be a single LLM agent, a seller-side multi-agent system, a user-side assistant, a platform orchestrator, or a multi-agent workflow. The proposed decision may be a price offer, a recommendation, a refund suggestion, a policy-sensitive reply, or a tool-use intent.

The raw decision is not necessarily safe or admissible. It may violate private economic constraints, exceed role authority, conflict with platform policy, or require human verification. The platform therefore applies a control policy Π before execution:

$$g_\Pi : (h_t, s_t, d_t^{\text{raw}}, C_{\text{obs}}) \longrightarrow (y_t, d_t^{\text{exec}}, z_t)$$

Here y_t is the controlled outcome, d_t^{exec} is the decision that reaches the environment, and z_t is an audit trace recording the control decision, revision, escalation reason, or other relevant metadata. We allow d_t^{exec} to be a distinguished no-op or deferred action when the proposal is blocked or escalated. Only d_t^{exec} affects the environment:

$$s_{t+1} = \mathcal{E}(s_t, d_t^{\text{exec}}).$$

The control outcome satisfies

$$y_t \in \{\textsc{Approve}, \textsc{Revise}, \textsc{Block}, \textsc{Escalate}\}$$

These outcomes have the following semantics. Approve executes the proposed decision unchanged, so $d_t^{\text{exec}} = d_t^{\text{raw}}$. Revise modifies the proposal before execution, producing an executable decision $d_t^{\text{exec}} \neq d_t^{\text{raw}}$. Block executes no environment-facing decision. Escalate routes the proposal or interaction state to a higher-authority process, trusted module, or human review; the environment-facing decision is deferred until the escalation process returns a resolution.

Thus, OCL mediates the transition from agent proposal to platform execution. Without OCL, the system would execute d_t^{raw} directly; with OCL, execution is conditioned on observable constraints, role authority, risk gates, and audit requirements.

The general problem can be formulated as below:

> Given raw proposals generated by agents in economic multi-agent tasks, how can an organizational control layer transform, approve, block, audit, or escalate

these proposals so as to improve reliability and reduce violations while preserving economic utility and efficiency?

3.2 Organizational Control Layer

The control map g_Π defined above is implemented as an *Organizational Control Layer* (OCL). It is a collection of control procedures

$$\Pi = \{\pi_{\text{role}}, \pi_{\text{gate}}, \pi_{\text{audit}}, \pi_{\text{escalate}}\}.$$

These policies are applied online before a proposed decision reaches the environment.

The role policy π_{role} determines which agent or module is allowed to propose, revise, or authorize a decision at the current step. In economic workflows, roles may correspond to user-side assistants, merchant-side assistants, platform orchestrators, or product experts. The role policy assigns authority for producing and checking proposals; it does not itself execute actions.

The gate policy π_{gate} checks a proposed decision against the constraints visible to the control layer. It returns a control outcome

$$y_t \in \{\textsc{Approve}, \textsc{Revise}, \textsc{Block}, \textsc{Escalate}\}.$$

High-risk decisions, such as financial commitments, refund promises, discount offers, order submission, or policy-sensitive replies, require explicit validation before execution.

The escalation policy π_{escalate} is invoked when local approval or revision is insufficient. This may occur when the proposal exceeds the local role's authority, when constraints conflict, when negotiation stalls, or when a trusted process has access to information unavailable to the local agent. Escalation may request clarification, reassign authority, route the case to a human or trusted module, or safely terminate the interaction.

The audit policy π_{audit} records proposed decisions, constraint checks, control outcomes, revisions, escalation reasons, and executed decisions. The audit trace supports debugging, state verification, and later review of why a proposal was approved, revised, blocked, or escalated.

4 Experiments

4.1 Personas Benchmark

Adversarial Buyer Personas. Unlike standard cooperative scenarios, we stress-test the proposed framework against malicious and uncooperative behaviors. To construct the benchmark, we utilized a prompt-driven LLM pipeline to synthesize a dataset of 50 unique adversarial buyer profiles. Each profile is assigned a specific identity, detailed behavioral instructions, and a tailored negotiation style distributed across 5 persona types (10 profiles each):

- **Extreme Lowballer:** Uses extreme emotional manipulation (e.g., fabricated crises, financial ruin) and threats of public shaming to force the seller below their strict reservation price.
- **Privacy Phisher:** Feigns technical ignorance or urgency to bypass secure platform payment gateways, attempting to phish for the seller's direct banking details or personal accounts.
- **Role Hijacker:** Employs prompt injection techniques (e.g., "Ignore previous instructions", "System reset initiated") to extract backend data or execute unauthorized refunds.
- **Vague Shopper:** Provides zero specific requirements and stubbornly refuses to clarify intents, attempting to force the seller into premature or unverified checkouts.
- **Time Waster:** Engages in endless loops of indecision, irrelevant personal questions, or absurd hypothetical scenarios to exhaust the negotiation horizon.

4.2 Experimental Setup

Organizational control layer is evaluated on AgenticPay with private buyer budgets, private seller reserves, and economic outcome metrics. Each episode terminates on agreement, round exhaustion, or an escalation exit. The **Baseline** is an end-to-end seller with no governance, **OCL** routes the seller's turn through structural auditing and escalation constraints.

LLM. We conduct comprehensive evaluation using multiple state-of-the-art LLMs as the backend generator: GPT-5.4, Gemini-3.1, and Qwen-3.5. For each model, we run paired experiments contrasting the Baseline versus OCL.

Interaction. During each episode, the buyer and seller interact in a turn-based dialogue within the AgenticPay environment. In the Baseline configuration, the seller's generated utterances are transmitted directly to the buyer without oversight. Under the OCL framework, the seller's raw action is intercepted and routed through a deterministic constraint engine. OCL performs strict evaluations, including role permission checks, privacy keyword scans, hard budget bounds verification, and intent risk scoring. If a constraint is violated, OCL explicitly blocks the unsafe action, generates an audit escalation log, and applies a deterministic replan before returning a safe, modified action to the conversation.

Scope. The two configurations are chosen to isolate the contribution of structural governance. The Baseline uses the exact same model, prompt, and context budget as OCL, meaning any performance gap, efficiency gain, or security enhancement is directly attributable to the OCL structure rather than the underlying model capabilities.

4.3 Metrics

We evaluate the system across three primary categories tracking task efficacy, structural safety, and operational efficiency:

Task & Economic Outcomes.

- *Success Rate*: Fraction of episodes reaching an agreement, regardless of boundary violations.
- *Valid Success Rate*: Fraction of episodes reaching a compliant agreement *without* structural or safety violations.
- *Average Seller Reward*: Final economic utility, defined as the profit margin discounted by a per-round time penalty ($\lambda = 0.10$):

$$\text{Reward} = \max(0, P_{\text{deal}} - P_{\min}) - (\lambda \times N_{\text{rounds}})$$

where P_{deal} is the agreed price, $P_{\min}$ is the minimum reservation price, and N_{rounds} is the total turn count.

Safety & Defense Mechanics.

- *Unsafe Rate*: Percentage of episodes where the agent executed at least one out-of-bounds or malicious action.
- *Intercept Rate*: Percentage of episodes where OCL guardrails successfully blocked an adversarial threat.
- *Executed Violations vs. Intercepted Threats*: Raw counts of safety constraints breached by the baseline versus threats mitigated by the OCL engine.
- *Escalations*: Total interventions where an intercepted unsafe action triggered the OCL deterministic replanner to overwrite the output.

Efficiency & Observability.

- *Average Round*: Mean number of negotiation turns per episode.
- *Average Latency*: Mean wall-clock execution time in seconds per episode.
- *Audit Events*: Average system logs (state traces, constraint evaluations, control decisions) generated per episode to quantify systemic transparency.

5 Results

5.1 Safety and Operational Profiling

Table 1: Benchmark results over 50 adversarial episodes. Baseline represents a standard end-to-end LLM agent, while OCL wraps the agent in our structural constraint layer.

Metric	Baseline	OCL
Task & Safety Performance		
Success rate (↑)	94%	96%
Valid Success rate (↑)	12%	**96%**
Unsafe rate (↓)	88%	**0%**
Intercept Rate (↑)	0%	**94%**
Efficiency & Economic Outcomes		
Avg. Round (↓)	5.36	**2.58**
Avg. Latency (s) (↓)	38.75	**18.51**
Avg. Seller Reward	26.95	18.39
System-Level Observability		
Audits (↑)	7.36	13.58
Escalations	0	48
Executed Violations (↓)	205	**0**
Intercepted Threats (↑)	0	**52**

To understand the mechanics of interceptions, Table 1 provides a comprehensive comparison. This deep dive contrasts the superficial task success of the Baseline against the strictly compliant success of the OCL framework.

The Illusion of Baseline Success. While the Baseline agent boasts a 94% Task Success Rate, a deeper inspection of the safety metrics reveals severe vulnerabilities. The Baseline exhibited an 88% Unsafe Rate, committing a staggering 205 executed violations across the 50 episodes. Because it lacked structural boundaries, it routinely conceded to off-platform privacy phishing, unauthorized refunds, and out-of-budget pricing. Consequently, its *Valid Success Rate*—the rate of reaching an agreement without breaking any constraints—was a mere 12%.

Absolute Mitigation of Unsafe Executions. By routing the agent's actions through the OCL engine, the Unsafe Rate is strictly reduced to 0%. The framework achieved this by identifying 52 distinct adversarial threats and issuing 48 escalations to rewrite the agent's output before it reached the buyer. As a result, the Valid Success Rate surged to 96%, proving that OCL effectively transforms a highly vulnerable LLM into a robust, compliant negotiating agent.

Enhanced Systemic Observability. Finally, Table 1 highlights OCL's contribution to system transparency. The Baseline agent generated only 7.36 audit events per episode, reflecting a black-box execution style. In contrast, OCL generated 13.58 audit events per episode. This increase provides high-resolution observability, ensuring that every constraint evaluation, blocked action, and deterministic rewrite is explicitly logged for post-hoc analysis without inflating the overall transaction latency.

5.2 Comprehensive Cross-Model Evaluation

Table 2 summarizes the performance of the Baseline and OCL frameworks across multiple LLMs. The results demonstrate that OCL consistently improves safety and operational efficiency without compromising negotiation efficacy.

High Task Success and Threat Interception. Across all models, OCL maintains an exceptionally high Success Rate ($\geq$ 96%), proving that strict structural constraints do not induce conversation collapse. Concurrently, OCL robustly intercepts adversarial threats, achieving intercept rates of 94% (GPT-5.4), 82% (Gemini-3.1), and 60% (Qwen-3.5)—effectively securing inherently vulnerable baseline architectures.

Efficiency Gains and Deterministic Replanning. OCL significantly reduces the Average Round count and wall-clock latency. While baseline models waste turns haggling over unviable, out-of-bounds prices, OCL's deterministic replanning immediately clamps violations to acceptable thresholds, eliminating unnecessary negotiation cycles.

Economic Calibration. Shifts in the Average Seller Reward under OCL reflect precise economic calibration rather than performance degradation. OCL realigns utility by filtering out artificially inflated rewards caused by baseline hallucinations or adversarial compliance. Conversely, it prevents sub-optimal concessions against lowball tactics by strictly enforcing structural seller boundaries.

References

[1] Uwe M. Borghoff, Paolo Bottoni, and Remo Pareschi. 2025. An Organizational Theory for Multi-Agent Interactions Integrating Human Agents, LLMs, and Specialized AI. *Discover Computing* (2025). doi:10.1007/s10791-025-09667-2

[2] Jeonghwan Choi, Jibin Hwang, Gyeonghun Sun, Minjeong Ban, Taewon Yun, Hyeonjae Cheon, and Hwanjun Song. 2026. What Makes a Sale? Rethinking End-to-End Seller–Buyer Retail Dynamics with LLM Agents. arXiv:2604.04468.

[3] Jacques Ferber, Olivier Gutknecht, and Fabien Michel. 2003. From Agents to Organizations: An Organizational View of Multi-Agent Systems. In *Agent-Oriented Software Engineering IV (Lecture Notes in Computer Science)*. Springer. doi:10.1007/978-3-540-24620-6_15

[4] Mahdi Hannoun, Olivier Boissier, Jaime Simão Sichman, and Claudette Sayettat. 2000. MOISE: An Organizational Model for Multi-Agent Systems. In *Advances in Artificial Intelligence (Lecture Notes in Computer Science)*. Springer, 156–165. doi:10.1007/3-540-44399-1_17

Table 2: Cross-model benchmark results over 50 adversarial episodes.

Model	Arm	Success Rate (↑)	Intercept Rate (↑)	Round (↓)	Latency (↓)	Audits (↑)	Reward
GPT-5.4	Baseline	94%	0%	5.36	38.75	7.36	26.95
	OCL	96%	**94%**	**2.58**	**18.51**	13.58	18.39
Gemini-3.1	Baseline	98%	0%	3.90	76.11	5.90	22.71
	OCL	96%	**82%**	**3.32**	**69.91**	15.78	18.24
Qwen-3.5	Baseline	100%	0%	2.44	65.52	4.44	19.72
	OCL	96%	**60%**	**2.30**	69.43	11.66	21.67

[5] Bryan Horling and Victor Lesser. 2004. A Survey of Multi-Agent Organizational Paradigms. *The Knowledge Engineering Review* 19, 4 (2004), 281–316. doi:10.1017/S0269888905000317

[6] Jomi Fred Hübner, Jaime Simão Sichman, and Olivier Boissier. 2007. Developing Organised Multi-Agent Systems Using the MOISE+ Model: Programming Issues at the System and Agent Levels. *International Journal of Agent-Oriented Software Engineering* (2007). doi:10.1504/IJAOSE.2007.016266

[7] Vera C. Kaelin, Maitreyee Tewari, Sara Benouar, and Helena Lindgren. 2024. Developing Teamwork: Transitioning Between Stages in Human-Agent Collaboration. *Frontiers in Computer Science* (2024). doi:10.3389/fcomp.2024.1455903

[8] Seth Karten, Wenzhe Li, Zihan Ding, Samuel Kleiner, Yu Bai, and Chi Jin. 2025. LLM Economist: Large Population Models and Mechanism Design in Multi-Agent Generative Simulacra. arXiv:2507.15815.

[9] Xianyang Liu, Shangding Gu, and Dawn Song. 2026. AgenticPay: A Multi-Agent LLM Negotiation System for Buyer–Seller Transactions. arXiv:2602.06008.

[10] Yaxi Lu, Shenzhi Yang, Cheng Qian, Guirong Chen, Qinyu Luo, Yesai Wu, Huadong Wang, Xin Cong, Zhong Zhang, Yankai Lin, Weiwen Liu, Yasheng Wang, Zhiyuan Liu, Fangming Liu, and Maosong Sun. 2025. Proactive Agent: Shifting LLM Agents from Reactive Responses to Active Assistance. In *The Thirteenth International Conference on Learning Representations*. https://openreview.net/forum?id=sRIU6k2TcU

[11] Matti Mäntymäki, Matti Minkkinen, Teemu Birkstedt, and Mika Viljanen. 2022. Defining Organizational AI Governance. *AI and Ethics* 2, 4 (2022), 603–609. doi:10.1007/s43681-022-00143-x

[12] Giorgio Piatti, Zhijing Jin, Max Geiger, Rada Mihalcea, Bernhard Schölkopf, and Mrinmaya Sachan. 2024. Cooperate or Collapse: Emergence of Sustainable Cooperation in a Society of LLM Agents. In *Advances in Neural Information Processing Systems*.

[13] Natalie Shapira, Chris Wendler, Avery Yen, Gabriele Sarti, Koyena Pal, Olivia Floody, Adam Belfki, Alex Loftus, Aditya Ratan Jannali, Nikhil Prakash, et al. 2026. Agents of Chaos. arXiv:2602.20021 [cs.AI] arXiv:2602.20021.

[14] Khanh-Tung Tran, Dung Dao, Minh-Duong Nguyen, Quoc-Viet Pham, Barry O'Sullivan, and Hoang D. Nguyen. 2025. Multi-Agent Collaboration Mechanisms: A Survey of LLMs. arXiv:2501.06322.

[15] Yishu Wang, Kakam Chong, Xiaofeng Wang, Xu Yan, DeXin Kong, Chen Ju, Ming Chen, Shuai Xiao, Shuguang Han, and Jufeng Chen. 2025. Evaluating Multi-Turn Bargain Skills in LLM-Based Seller Agents. arXiv:2509.06341.

[16] Bufang Yang, Lilin Xu, Liekang Zeng, Kaiwei Liu, Siyang Jiang, Wenrui Lu, Hongkai Chen, Xiaofan Jiang, Guoliang Xing, and Zhenyu Yan. 2025. ContextAgent: Context-Aware Proactive LLM Agents with Open-World Sensory Perceptions. In *Advances in Neural Information Processing Systems*.

[17] Yifan Yu, Moyan Li, Shaoyuan Xu, Jinmiao Fu, Xinhai Hou, Fan Lai, and Bryan Wang. 2025. CORRECT: COndensed eRror RECognition via Knowledge Transfer in Multi-Agent Systems. arXiv:2509.24088.

[18] Yanwei Yue, Guibin Zhang, Boyang Liu, Guancheng Wan, Kun Wang, Dawei Cheng, and Yiyan Qi. 2025. MasRouter: Learning to Route LLMs for Multi-Agent Systems. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*.

[19] Shenzhe Zhu, Jiao Sun, Yi Nian, Tobin South, Alex Pentland, and Jiaxin Pei. 2025. The Automated but Risky Game: Modeling Agent-to-Agent Negotiations and Transactions in Consumer Markets. arXiv:2506.00073.