Benchmarking LLMs' Swarm Intelligence

Anonymous ACL submission

Abstract

Large Language Models (LLMs) show promise as autonomous agents. Yet their capacity for decentralized coordination remains underexplored, particularly in scenarios where agents operate with limited local perception and no centralized control. We introduce **SwarmBench**, a benchmark for evaluating emergent coordination in LLM-based swarms. The benchmark features five tasks in a physics-grounded 2D environment, requiring agents to achieve collective objectives through local interactions. Evaluating thirteen LLMs in a zero-shot setting, we find that no model achieves consistent cross-task success. We identify a *communication-coordination gap*: while agents are strongly influenced by peer messages, this linguistic alignment fails to produce effective collective action. The gap manifests in failure modes including spatial congestion, information silos, and protocol rigidity, revealing that current LLMs lack the grounded reasoning necessary for robust swarm intelligence. We release SwarmBench as an open-source toolkit for decentralized LLM coordination research.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable potential in reasoning and planning (Zhao et al., 2023), driving the development of autonomous agents for complex decision-making. As research expands into Multi-Agent Systems (MAS), a key objective is enabling effective collaboration across diverse domains, from software engineering (Qian et al., 2023; Hong et al., 2023) to social simulation (Park et al., 2023). However, most existing work operates under permissive conditions: centralized orchestration, global visibility, or high-bandwidth communication. Realistic deployment often demands a fundamentally different paradigm: **decentralized coordination** under strict informational constraints, where agents must infer collective intent from sparse local observations.

This creates a structural tension between LLM capabilities and swarm requirements. LLMs are trained on explicit, globally coherent contexts, yet effective swarm intelligence emerges from local interactions without central control. To evaluate this capability, we introduce **SwarmBench**, a benchmark designed to isolate decentralized coordination from general reasoning. Unlike frameworks that provide shared environments or pre-assigned roles, SwarmBench enforces hard constraints: agents operate in a 2D grid with strictly limited local perception ($k \times k$ cells) and communicate only with nearby peers via short broadcasts. Inspired by ARC-AGI (Chollet et al., 2025) and SnakeBench (Kamradt, 2025), we define a suite of five coordination tasks: Pursuit, Synchronization, Foraging, Flocking, and Transport. Each task demands spatial convergence, temporal alignment, and physical consensus in the absence of centralized planning.

Our zero-shot evaluation of thirteen LLMs reveals that decentralized coordination remains challenging. No model achieves consistent success across tasks, suggesting that effective coordination requires task-specific capabilities rather than general reasoning alone. Crucially, we identify a *communication-coordination gap*: while agents readily engage in verbal coordination and are strongly influenced by peer messages, this linguistic alignment fails to translate into effective collective action. Agents achieve local consensus in dialogue while producing spatially incoherent behavior. This disconnect surfaces as three characteristic failure modes: spatial congestion, information fragmentation, and protocol rigidity. These findings indicate that current LLMs, despite sophisticated language capabilities, lack the grounded physical reasoning necessary for robust swarm intelligence.

Our contributions are as follows:

- **SwarmBench Framework:** A benchmark featuring five coordination tasks in a physics-grounded 2D environment. By enforcing strict partial observability and local-only communication, SwarmBench provides a rigorous testbed for evaluating emergent coordination under decentralization constraints.

- **Systematic Evaluation and Analysis:** A comprehensive zero-shot evaluation of thirteen LLMs, accompanied by a multi-dimensional analysis framework encompassing behavioral dynamics, action attribution, and failure mode taxonomy. This provides mechanistic insight into how coordination succeeds or fails under informational constraints.
- **Open-Source Toolkit:** We release the complete SwarmBench suite, including the simulation environment, evaluation scripts, and interaction datasets (Appendix J), to support research on decentralized LLM coordination.

2 Related Work

LLM-Driven Multi-Agent Systems. LLMs are increasingly deployed as autonomous decision-makers across diverse domains(Xi et al., 2025; Wang et al., 2024b), including collaborative software engineering (Qian et al., 2023; Hong et al., 2023), scientific discovery (Gottweis et al., 2025; Boiko et al., 2023), social simulation (Park et al., 2023; Gao et al., 2024; AL et al., 2024; Yang et al., 2024), and code generation (Ishibashi and Nishimura, 2024). These systems demonstrate notable potential for cooperation and Theory of Mind (Li et al., 2023; Woolley et al., 2010). However, most rely on centralized architectures or high-bandwidth communication with predefined roles (Li et al., 2025). SwarmBench addresses a distinct challenge: *decentralized emergence* under strict local perception and noisy, limited-range communication (Sharma et al., 2023).

Benchmarking LLM Coordination. Existing multi-agent benchmarks employ cooperative games (Agashe et al., 2023; Sun et al., 2025; Wu et al., 2024), competitive scenarios (Kamradt, 2025), or complex simulations (Zhu et al., 2025; Dong et al., 2024; Park et al., 2023). However, these environments rarely enforce the strict informational constraints of classical swarm intelligence, limiting their ability to isolate emergent coordination from general reasoning. Foundational reasoning benchmarks assess individual cognitive capabilities (Wang et al., 2024a; Paglieri et al., 2024; Ruoss et al., 2025), yet they often overlook collective spatial dynamics. This gap is particularly evident in flocking-like tasks, where LLMs reportedly struggle (Li et al., 2024). SwarmBench bridges this gap by adopting classical swarm constraints to systematically evaluate emergent collective behavior.

Embodied Multi-Agent Coordination. Research on embodied LLM agents (Kannan et al., 2024; Yu et al., 2023; Guo et al., 2024; Zhang et al., 2024c; Mandi et al., 2024; Chen et al., 2024; Zhang et al., 2024a,b; Garg et al., 2024; Liu et al., 2024) typically focuses on application-specific scenarios with rich sensory inputs or global communication. SwarmBench offers a complementary perspective by removing these affordances to isolate fundamental coordination primitives. Our physics-grounded 2D grid environment with severe local constraints aligns with recent explorations of LLM integration in swarm robotics (Strobel et al., 2024).

3 SwarmBench Framework

We present SwarmBench, a benchmark for evaluating emergent coordination in LLM-based multi-agent systems under strict decentralization constraints. The framework is grounded in classical swarm intelligence principles: agents operate with limited local perception, communicate only with nearby peers, and must achieve collective objectives without centralized control.

This section describes the core components of SwarmBench. We first formalize the problem setting, including agent perception, action spaces, and reward structures (§3.1). We then detail the five coordination tasks, each designed to probe distinct capabilities ranging from spatial reasoning to distributed consensus (§3.2). Additional implementation details are provided in Appendix A.

3.1 Problem Formulation

SwarmBench instantiates a physics-grounded 2D grid environment isolating emergent coordination from general reasoning capabilities. The system operates in discrete time steps $t = 1, \dots, T$, where N homogeneous agents, each controlled by an independent LLM instance, must achieve shared objectives under informational and physical constraints.

Restricted Agent Perception. To ensure that global coordination emerges solely from local interactions, agents operate under severe partial observability. At each step t, agent i receives a text-based observation comprising four components:

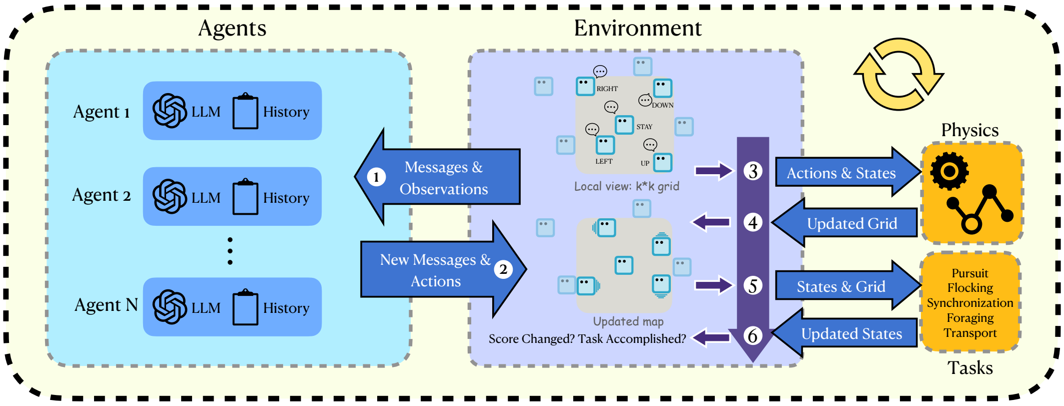

Figure 1: **The Architecture of SwarmBench Framework.** The diagram shows SwarmBench's modular design and execution flow. It allows agents with LLM brains and history to interact with the environment, where a physics engine and multiple tasks are plugged in.

- **Egocentric View:** A $k \times k$ local grid centered on the agent (default $k{=}5$), rendered as ASCII symbols: the agent itself (`Y`), peers (numeric identifiers), walls (`W`), and task-specific objects such as prey (`P`) or food (`F`).
- **Local State:** The agent's global coordinates and task-relevant status indicators.
- **Incoming Messages:** A buffer containing messages broadcast by peers within the $k \times k$ perception range during the previous step.
- **History Context:** A sliding window of the agent's own recent observations and actions.

Critically, agents possess no global map and cannot access the states or communications of agents beyond their local perception range.

Action Space and Communication. At each step, all agents simultaneously generate a composite output consisting of a physical action and a communicative signal. The physical action is drawn from a discrete set $\mathcal{A}$ comprising directional movements (`UP`, `DOWN`, `LEFT`, `RIGHT`), waiting (`STAY`), and task-specific interactions (e.g., `SWITCH` for toggling states). Additionally, each agent may broadcast a short message (subject to a character limit). Consistent with the local perception model, messages propagate only to neighbors within the sender's visual range, simulating ad-hoc local communication rather than a global broadcast.

Objectives and Reward Structure. Each task defines a reward function $R(s, a)$ that quantifies collective success based on task-specific coordination primitives: immobilizing evasive targets (*Pursuit*), achieving alternating consensus states (*Synchronization*), executing efficient resource retrieval cycles (*Foraging*), minimizing distance to a target geometry (*Flocking*), and minimizing escape time under physical load (*Transport*). In this fully cooperative setting, agents share the objective of maximizing cumulative reward $\sum_t R_t$ through decentralized local interactions alone.

This formulation requires agents to move beyond verbal coordination toward physical alignment: inferring collective intent and synchronizing strategies through sparse local observations and peer behavior rather than explicit global planning.

3.2 Coordination Tasks

SwarmBench defines five coordination tasks (Figure 2), each designed to probe specific dimensions of decentralized capability.

Pursuit (Dynamic Spatial Containment). A group of N agents must collaboratively track and entrap a prey (`P`) that moves faster than individual agents (2 cells per round versus 1). The prey employs a heuristic evasion strategy, actively fleeing high-density regions. A point is awarded when the prey is completely immobilized (i.e., surrounded by agents or walls on all four cardinal sides), after which it respawns in a low-density area.

The core challenge is **overcoming reaction latency**. Since the prey outpaces individual agents, naive pursuit fails. Success requires predictive spatial reasoning: agents must anticipate the prey's trajectory and form containment formations without explicit role assignment.

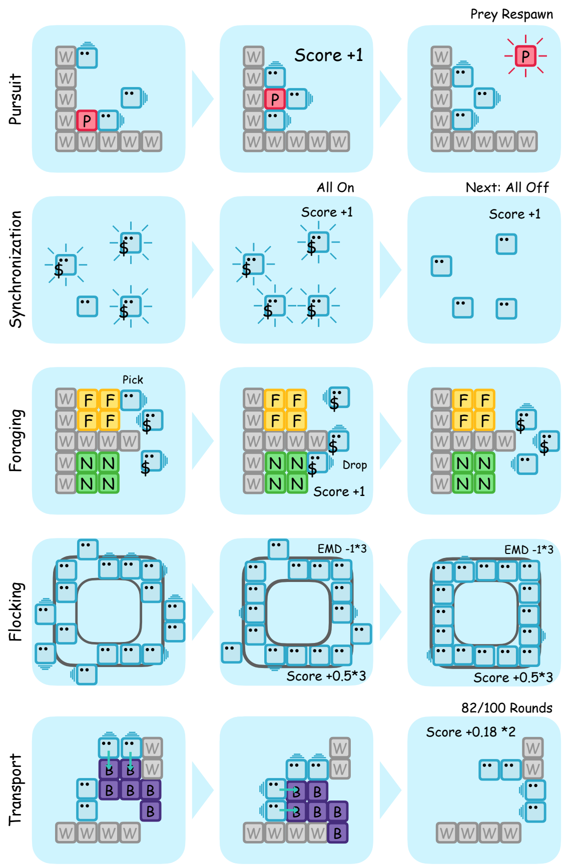

Figure 2: **Tasks.** Five tasks are designed in SwarmBench, which are introduced in 3.2.

Synchronization (Consensus & Negotiation). Each agent maintains an internal binary state (e.g., light on/off, rendered as `A`/`a`) and can toggle it via a `SWITCH` action. The objective is twofold: the swarm must first achieve global *unanimity* (all agents in the same state), then collectively *alternate* to the opposite state to score repeatedly.

This task evaluates **information propagation across a distributed network**. With only local communication, the swarm risks *information silos*: isolated subgroups may converge on conflicting states. Success requires agents to bridge local agreements into global consensus.

Foraging (Spatial Memory and Persistent Planning). Agents must navigate a walled environment to locate food (`F`), pick it up, and transport it to a designated nest (`N`). Points are awarded only upon successful delivery.

Unlike containment tasks, Foraging probes **spatial memory and goal persistence**. Agents must maintain sub-goals (e.g., delivering versus searching) over extended sequences despite local distractions and limited memory.

Task	Spatial Reasoning	Temporal Sync.	Negotiation & Consensus	Persistent Planning
PURSUIT	High	Med	Low	Low
SYNCHRONIZATION	Low	High	High	Low
FORAGING	Med	Low	Med	High
FLOCKING	High	Med	Med	Low
TRANSPORT	High	High	High	High

Table 1: Capability dimensions probed by each SwarmBench task: **Spatial Reasoning** (geometric/trajectory understanding), **Temporal Synchronization** (action timing coordination), **Negotiation** (resolving information asymmetry), and **Persistent Planning** (maintaining goals over extended sequences).

Flocking (Distributed Pattern Formation). Agents must arrange themselves into a configuration matching a predefined shape (e.g., a hollow square). The task succeeds when the formation aligns with the target under translation. Performance is quantified using **Translation-Invariant Earth Mover's Distance (EMD)**: the minimum Manhattan distance required to map agent positions to target points, optimized across all possible alignments. The reward reflects cumulative reduction in this distance from the initial configuration.

This task isolates **distributed pattern matching**. Agents must translate fragmented local views into coherent global topology, inferring the group's center of mass and their relative target positions solely from neighbor configurations.

Transport (Coordinated Physical Manipulation). A heavy, irregular obstacle blocks the map exit. Moving it requires at least five agents pushing simultaneously from complementary positions, coordinating both force magnitude and direction. The task has two stages: clearing the obstacle, then escaping. Scoring rewards speed: an agent escaping at round t receives $1 - t/T$ points, where T is the maximum round limit.

This task serves as a comprehensive test of **physical consensus and synchronized execution**. Agents must negotiate a shared plan and execute it with precise timing. The irregular obstacle shape necessitates complex multi-directional maneuvering rather than simple linear pushing, demanding high-fidelity physical coordination.

Summary. Collectively, these tasks form a diagnostic testbed for decentralized coordination. Rather than ranking by a single difficulty metric, we categorize tasks by the cognitive capabilities they probe (Table 1), enabling researchers to identify specific deficiencies in LLM-based agents.

Model	Flock.	Forag.	Purs.	Sync.	Trans.	Avg.
gemini-2.0-flash	**9.40**	**5.80**	8.80	3.40	0.00	**5.48**
o4-mini	8.90	4.80	**9.60**	2.80	0.52	5.32
claude-3.7-sonnet	7.50	1.20	4.40	**12.60**	0.00	5.14
gpt-4.1	5.70	3.20	8.40	2.80	0.00	4.02
deepseek-v3	6.40	2.60	4.20	4.00	0.00	3.44
gpt-4o	5.00	1.60	3.40	1.80	0.00	2.36
o3-mini	2.70	2.60	3.60	2.20	0.00	2.22
qwq-32b	5.90	0.80	2.20	1.20	0.00	2.02
deepseek-r1	6.10	1.00	1.00	1.20	0.71	2.00
llama-3.1-70b	7.10	0.00	1.80	1.00	0.00	1.98
llama-4-scout	7.10	1.00	1.20	0.20	0.00	1.90
gpt-4.1-mini	5.00	1.40	1.40	0.60	0.00	1.68
claude-3.5-haiku	5.60	0.00	0.60	1.00	0.00	1.44

Table 2: **SwarmBench Performance Dashboard.** Avg. scores across 5 tasks.

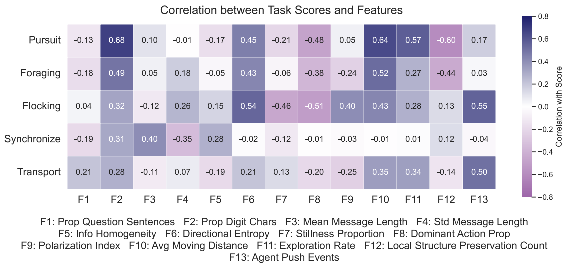

Figure 3: **Scores and Features.** This shows Pearson correlations between task scores and features.

4 Results and Analysis

We organize our empirical investigation around three central questions: (1) How do different LLMs perform across SwarmBench tasks, and what performance patterns emerge? (2) What behavioral mechanisms distinguish successful coordination from failure? (3) What recurring failure modes characterize current LLM-based swarms?

Section 4.1 describes our experimental configuration and analytical methodology (see Appendix B for the complete agent prompt template). Section 4.3 presents cross-task performance comparisons, revealing that no single model achieves uniform dominance (detailed scores in Appendix D). Section 4.4 introduces the *communication-coordination gap*—a disconnect between agents' responsiveness to peer messages and their ability to achieve effective collective outcomes. Finally, Section 4.5 taxonomizes five characteristic failure patterns, grounded in quantitative behavioral analysis. Visual examples of agent behavior across all tasks are provided in Appendix C.

4.1 Experimental Setup

We evaluated 13 LLMs across the five SwarmBench tasks, including proprietary models (e.g., `gpt-4o`, `claude-3.7-sonnet`) and open-weight alternatives (e.g., `llama-3.1-70b`). All experiments follow a strict **zero-shot protocol**: each agent operates as an independent LLM instance without task-specific training or access to centralized coordination mechanisms.

Agents perceive only a local 5×5 grid surrounding their position and retain a sliding-window context of the previous 5 rounds. For reproducibility, we fix generation parameters across all models (T=1.0, top_p=1.0) unless otherwise noted in sensitivity analyses. All quantitative results are averaged over 5 independent runs per model-task configuration to account for stochastic variation in both LLM outputs and environment dynamics.

4.2 Analytical Framework

To identify the behavioral mechanisms underlying swarm performance, we conducted a systematic correlation analysis between emergent group dynamics and task outcomes. We operationalized collective behavior using twelve metrics (detailed in Appendix E), spanning communication patterns (e.g., *Information Homogeneity*, measuring semantic consistency across agent messages) and action dynamics (e.g., *Directional Entropy*, capturing movement variability).

For each metric, we computed Pearson's r against final task scores to quantify behavioral-performance relationships. Positive correlations indicate success-promoting behaviors—for instance, higher *Exploration Rate* correlates with improved Foraging performance ($r > 0$, $p < 0.05$). Conversely, negative correlations reveal counterproductive tendencies. All reported correlations were tested for statistical significance, denoted as *$p < 0.05$, **$p < 0.01$, and ***$p < 0.001$.

To further disentangle the drivers of individual agent decisions, we trained a Random Forest classifier (Breiman, 2001) to predict actions from observation and message embeddings, then quantified feature importance via permutation analysis and SHAP values (Appendix I).

4.3 Cross-Task Performance Analysis

Results across the five tasks (Table 2) reveal substantial variation in model performance, with no single LLM achieving consistent dominance. This task-specificity suggests that effective decentralized coordination requires different capabilities depending on the task structure.

We observe distinct performance profiles across models. `claude-3.7-sonnet` achieves the

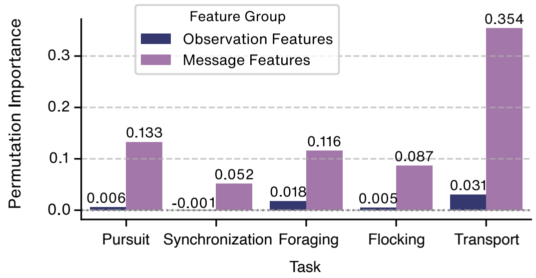

Figure 4: The permutation importance of observation and message features in each task.

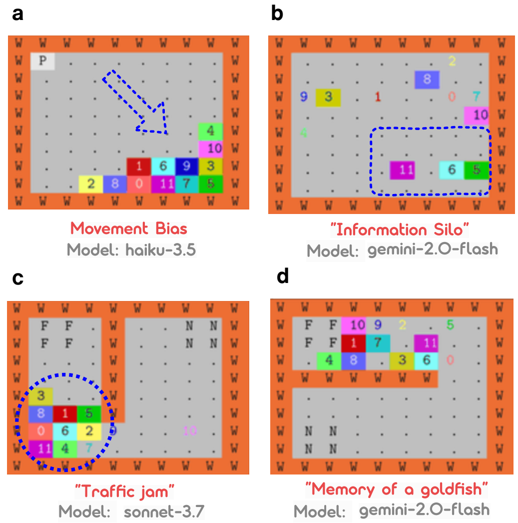

Figure 5: Four typical failure modes. **Movement bias** is when agents' moving direction being the same. **Information silo** appears when agent groups are not connected. **Traffic jam** means agents blocking the way. **Memory of a goldfish** reflects agents forgetting plans.

strongest results on *Synchronization*, which demands implicit consensus formation, yet shows limited success on *Transport*. In contrast, `gemini-2.0-flash` excels in spatially-oriented tasks (*Pursuit*, *Foraging*) but underperforms on coordination tasks requiring behavioral convergence. These divergent profiles indicate that current LLMs exhibit task-dependent coordination capabilities rather than a unified swarm intelligence.

4.4 The Communication-Coordination Gap

To understand the observed coordination failures, we analyzed the factors driving agent decisions. Our attribution analysis (Section I) reveals that across multiple tasks, agents weight incoming messages more heavily than local observations when selecting actions (Figure 4). These messages indeed contain task-relevant terminology, as confirmed by keyword frequency analysis (Appendix G).

This presents a paradox. While communication strongly influences individual action selection, message semantic content shows weak correlation with task success. We term this phenomenon the *communication-coordination gap*: agents respond to peer signals tactically but fail to translate this responsiveness into effective collective outcomes.

This disconnect manifests in characteristic failure modes. Agents may achieve local linguistic alignment (e.g., agreeing on intentions or sharing coordinates) while their physical behaviors remain uncoordinated with task demands. The analysis examines specific instantiations of this pattern.

4.5 Taxonomy of Coordination Failures

Analysis of agent trajectories reveals three recurrent failure patterns arising from the gap between linguistic reasoning and physical execution.

Movement Rigidity and Spatial Congestion. In tasks requiring physical convergence (*Transport*, *Pursuit*), agents frequently exhibit *movement bias*, defined as a rigid directional preference where agents maintain fixed trajectories despite changing task demands (Figure 5a). Rather than simple greedy behavior, this reflects premature strategy lock-in: the LLMs commit to a movement pattern disconnected from the global objective.

We quantified this tendency using *Action Direction Imbalance*, measured via the Gini coefficient of each agent's action distribution. In *Pursuit*, this metric shows a strong negative correlation with task performance ($r = -0.668$, $p < 0.001$) (Appendix K.1), confirming that behavioral rigidity directly impedes coordination success.

A common consequence is the emergence of *traffic jams* (Figure 5c): agents converge on targets without coordinated positioning, blocking each other's paths rather than forming effective spatial formations. This pattern illustrates how individually reasonable local decisions can produce collectively dysfunctional outcomes.

Network Fragmentation and Information Silos. The *Synchronization* task requires agents to achieve global consensus on a binary state. However, restricted local communication often leads to *information silos* (Figure 5b): the swarm fragments into isolated subgroups that achieve internal agreement but remain disconnected from one another. For instance, one cluster may converge on state 'On' while another stabilizes on 'Off', with neither group aware of the inconsistency.

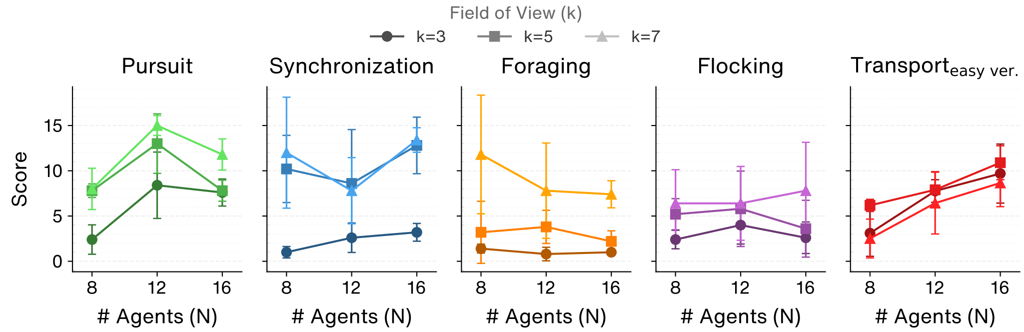

Figure 6: The impact of agent amount and field of view change on scores.

We analyzed this fragmentation using the *Number of Connected Components* in the agent visibility graph, where edges connect agents within each other's perceptual range. While the correlation with task performance lacks statistical significance ($r = -0.185$, $p = 0.140$) (Appendix K.1), the pattern is directionally consistent: greater fragmentation tends to impede consensus formation. This structural isolation prevents the global propagation of state information, creating persistent deadlocks where local order coexists with global disorder.

Communication Protocol Rigidity. Our analysis reveals that the emergence of structured communication protocols can hinder rather than facilitate collective performance. We measured protocol convergence using two metrics: the increase in *Information Homogeneity* (semantic similarity) and *Edit Distance Consistency* (syntactic similarity) over each simulation (Appendix K.2).

In dynamic tasks such as *Pursuit*, stronger protocol convergence correlates negatively with task success ($r = -0.447$, $p < 0.001$). While agents successfully converge on simplified message formats, this uniformity appears to reduce the information diversity needed for adaptive coordination. The pattern reverses for *Flocking* ($r = 0.382$, $p < 0.01$), where a simple, consistent protocol for sharing positional data proves beneficial.

This task-dependent effect suggests that communication structure value depends on task complexity: rigid protocols suffice for routine coordination but fail to capture the complexity of dynamic scenarios. Agents may achieve linguistic alignment while lacking the semantic flexibility required for effective collective adaptation.

Memory Constraints and Exploration Inefficiency. In exploration-intensive tasks such as *Foraging*, agents exhibit what we term *goldfish memory* (Figure 5d): a failure to maintain persistent spatial awareness beyond the immediate context window. Agents repeatedly revisit depleted resource locations, unable to distinguish explored regions from unexplored ones.

This limitation stems directly from the benchmark's sliding-window context design, which retains only the previous 5 rounds of history. While sufficient for reactive decision-making, this restricted memory proves inadequate for tasks requiring long-horizon planning and spatial persistence. Qualitative trajectory analysis confirms that agents cycle through previously visited areas rather than systematically expanding their search, suggesting that effective foraging behavior may require external memory mechanisms or explicit state-tracking strategies beyond current LLM capabilities.

Non-Monotonic Scaling with Population and Perception. Our sensitivity analysis (Appendix H) reveals that swarm performance does not scale monotonically with either agent count or perceptual range. Simply adding more agents or expanding observation windows does not guarantee improved coordination, and may, in fact, actively degrade performance.

For population size N, the relationship is highly task-dependent (Figure 6). *Transport* shows consistent improvement with larger groups ($N = 16$ outperforms $N = 8$), as the task fundamentally requires cumulative physical force. However, *Foraging* performance degrades with additional agents due to spatial congestion near resource locations. *Pursuit* exhibits an inverted-U pattern, peaking at $N = 12$ before declining, suggesting that while more agents aid initial encirclement, excessive density causes self-obstruction.

A similar non-monotonic pattern emerges for perception range k. Expanding the field of view from $k = 3$ to $k = 5$ consistently improves outcomes across tasks. However, expansion to $k = 7$ yields diminishing returns and sometimes degrades performance (e.g., *Transport*). This suggests a trade-off: broader perception increases available information but may overwhelm the LLM's capacity to identify relevant local cues, diluting focus on features critical for tightly-coupled coordination.

5 Discussion

Our evaluation reveals that effective swarm coordination under decentralization constraints depends on emergent physical dynamics rather than explicit communication. While LLM agents readily engage in verbal coordination (e.g., sharing intentions, coordinates, and plans), this linguistic alignment frequently fails to translate into successful collective action. We term this the *communication-coordination gap*, and it manifests across multiple failure modes documented in Section 4.5.

Why Communication Fails to Coordinate. The gap between communication influence and coordination effectiveness likely stems from a mismatch between LLM training objectives and swarm task demands. LLMs are optimized for linguistic coherence and next-token prediction, not for grounding language in physical constraints or maintaining spatial consistency across decentralized agents. When agents converge on a shared vocabulary or protocol, they achieve *linguistic* consensus that may lack the precision, timing, or spatial grounding required for *physical* consensus. The result is coherent dialogue paired with incoherent collective behavior.

Implications for Future Research. These findings motivate several research directions. The communication-coordination gap suggests a need for training objectives that ground linguistic outputs in physical outcomes. Protocol rigidity in dynamic tasks points toward adaptive communication strategies that modulate complexity based on task demands. The goldfish memory limitation motivates external memory augmentation or stigmergic coordination mechanisms. Finally, non-monotonic scaling effects call for density-aware policies or emergent hierarchical organization.

Limitations. SwarmBench employs an abstracted 2D grid environment, which, while enabling controlled evaluation, does not capture the full complexity of real-world physical dynamics. The discrete action space and simplified physics may underestimate challenges that arise in continuous domains. Additionally, our zero-shot evaluation protocol does not explore whether fine-tuning or reinforcement learning could mitigate the observed failure modes, which remains a promising avenue for future investigation. Finally, our analysis focuses on homogeneous agent populations; heterogeneous swarms with specialized roles may exhibit different coordination dynamics.

6 Conclusion

We introduced SwarmBench, a benchmark for evaluating emergent coordination in LLM-based multi-agent systems under strict decentralization constraints. By enforcing limited local perception, local-only communication, and the absence of centralized control, SwarmBench isolates the fundamental challenge of achieving collective behavior through distributed interactions alone.

Our evaluation of thirteen LLMs reveals a consistent pattern: while agents readily engage in verbal coordination, this linguistic alignment fails to produce effective physical coordination. We formalize this as the *communication-coordination gap* and document its manifestations across five failure modes, from spatial congestion and information silos to protocol rigidity and memory limitations. These findings indicate current LLMs, despite sophisticated language capabilities, lack grounded spatial reasoning and adaptive coordination strategies necessary for robust swarm intelligence.

SwarmBench contributes a diagnostic tool and empirical foundation for this emerging research area. The benchmark enables systematic identification of coordination deficiencies, while our analysis provides concrete targets for improvement: grounding communication in physical outcomes, developing adaptive protocols, and augmenting agents with persistent memory mechanisms.

As LLM-based agents are increasingly deployed in distributed settings (e.g., autonomous vehicle fleets and decentralized sensor networks), the ability to coordinate effectively without centralized control becomes critical. SwarmBench provides a foundation for developing and evaluating such capabilities. We release the benchmark, including environments, evaluation scripts, and interaction logs, as an open resource to support progress toward robust artificial collective intelligence.

References

Saaket Agashe, Yue Fan, Anthony Reyna, and Xin Eric Wang. 2023. LLM-coordination: evaluating and analyzing multi-agent coordination abilities in large language models. *arXiv preprint arXiv:2310.03903*.

Altera AL, Andrew Ahn, Nic Becker, Stephanie Carroll, Nico Christie, Manuel Cortes, Arda Demirci, Melissa Du, Frankie Li, Shuying Luo, and 1 others. 2024. Project Sid: Many-agent simulations toward AI civilization. *arXiv preprint arXiv:2411.00114*.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. "O'Reilly Media, Inc.".

Daniil A Boiko, Robert MacKnight, and Gabe Gomes. 2023. Emergent autonomous scientific research capabilities of large language models, 2023. *arXiv preprint arXiv:2304.05332*.

Leo Breiman. 2001. Random forests. *Machine learning*, 45:5–32.

Junting Chen, Checheng Yu, Xunzhe Zhou, Tianqi Xu, and 1 others. 2024. EMOS: EMBODIMENT-AWARE heterogeneous multi-robot operating system with LLM agents. *arXiv preprint arXiv:2405.19012*.

Francois Chollet, Mike Knoop, Gregory Kamradt, and Bryan Landers. 2025. ARC Prize 2024: Technical Report. *arXiv preprint arXiv:2412.04604*.

Yubo Dong, Xukun Zhu, Zhengzhe Pan, Linchao Zhu, and Yi Yang. 2024. Villageragent: A graph-based multi-agent framework for coordinating complex task dependencies in Minecraft. *arXiv preprint arXiv:2406.05720*.

Chen Gao, Xiaochong Lan, Nian Li, Yuan Yuan, Jingtao Ding, Zhilun Zhou, Fengli Xu, and Yong Li. 2024. Large language models empowered agent-based modeling and simulation: A survey and perspectives. *Humanities and Social Sciences Communications*, 11(1):1–24.

Kunal Garg, Jacob Arkin, Songyuan Zhang, Nicholas Roy, and Chuchu Fan. 2024. Large language models to the rescue: Deadlock resolution in multi-robot systems. *arXiv preprint arXiv:2404.14293*.

Michelle Girvan and Mark EJ Newman. 2002. Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826.

Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Anil Palepu, Petar Sirkovic, Artiom Myaskovsky, Felix Weissenberger, Keran Rong, Ryutaro Tanno, Khaled Saab, Dan Popovici, Jacob Blum, Fan Zhang, Katherine Chou, Avinatan Hassidim, Burak Gokturk, Amin Vahdat, Pushmeet Kohli, and 15 others. 2025. Towards an AI co-scientist. *arXiv preprint arXiv:2502.18864*.

Plerre-P Grass. 1959. La reconstruction du nid et les coordinations inter-individuelles chez bellicositermes natalensis et cubitermes sp. la thorie de la stigmergie: Essai d'interprtation du comportement des termites constructeurs. *Insectes sociaux*, 6(4180):10–1007.

Xudong Guo, Kaixuan Huang, Jiale Liu, and 1 others. 2024. Embodied LLM agents learn to cooperate in organized teams. *arXiv preprint arXiv:2403.12482*.

Sirui Hong, Xiawu Zheng, Jonathan Chen, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, and 1 others. 2023. MetaGPT: Meta programming for multi-agent collaborative framework. *arXiv preprint arXiv:2308.00352*, 3(4):6.

Yoichi Ishibashi and Yoshimasa Nishimura. 2024. Self-organized agents: A LLM multi-agent framework toward ultra large-scale code generation and optimization. *arXiv preprint arXiv:2404.02183*.

Greg Kamradt. 2025. Snake bench: Competitive snake game simulation with llms. `https://github.com/gkamradt/SnakeBench`.

Shyam Sundar Kannan, Vishnunandan LN Venkatesh, and Byung-Cheol Min. 2024. Smart-LLM: Smart multi-agent robot task planning using large language models. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 12140–12147. IEEE.

Huao Li, Yu Quan Chong, Simon Stepputtis, Joseph Campbell, Dana Hughes, Michael Lewis, and Katia Sycara. 2023. Theory of mind for multi-agent collaboration via large language models. *arXiv preprint arXiv:2310.10701*.

Peihan Li, Zijian An, Shams Abrar, and Lifeng Zhou. 2025. Large Language Models for multi-robot systems: A survey. *arXiv preprint arXiv:2502.03814*.

Peihan Li, Vishnu Menon, Bhavanaraj Gudiguntla, Daniel Ting, and Lifeng Zhou. 2024. Challenges faced by Large Language Models in solving multi-agent flocking. *arXiv preprint arXiv:2404.04752*.

Hsu-Shen Liu, So Kuroki, Tadashi Kozuno, Wei-Fang Sun, and Chun-Yi Lee. 2024. Language-guided pattern formation for swarm robotics with multi-agent reinforcement learning. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Scott Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. *arXiv preprint arXiv:1705.07874*.

Zhao Mandi, Shreeya Jain, and Shuran Song. 2024. Roco: Dialectic multi-robot collaboration with large language models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 286–299. IEEE.

James G March. 1991. Exploration and exploitation in organizational learning. *Organization science*, 2(1):71–87.

Davide Paglieri, Bartłomiej Cupiał, Samuel Coward, Ulyana Piterbarg, Maciej Wolczyk, Akbir Khan, Eduardo Pignatelli, Łukasz Kuciński, Lerrel Pinto, Rob Fergus, and 1 others. 2024. BALROG: Benchmarking agentic LLM and VLM reasoning on games. *arXiv preprint arXiv:2411.13543*.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pages 1–22.

Chen Qian, Xin Cong, Cheng Yang, Weize Chen, Yusheng Su, Juyuan Xu, Zhiyuan Liu, and Maosong Sun. 2023. Communicative agents for software development. *arXiv preprint arXiv:2307.07924*, 6(3).

Craig W Reynolds. 1987. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, pages 25–34.

Anian Ruoss, Fabio Pardo, Harris Chan, Bonnie Li, Volodymyr Mnih, and Tim Genewein. 2025. LMAct: A benchmark for in-context imitation learning with long multimodal demonstrations. In *ICML*.

Mohit Sharma, Simone Baldi, and Tansel Yucelen. 2023. Low-distortion information propagation with noise suppression in swarm networks. *Proceedings of the National Academy of Sciences*, 120(11):e2219948120.

Volker Strobel, Marco Dorigo, and Mario Fritz. 2024. Llm2swarm: Robot swarms that responsively reason, plan, and collaborate through llms. *arXiv preprint arXiv:2410.11387*.

Haochen Sun, Shuwen Zhang, Lei Ren, Hao Xu, Hao Fu, Caixia Yuan, and Xiaojie Wang. 2025. Collab-Overcooked: Benchmarking and evaluating large language models as collaborative agents. *arXiv preprint arXiv:2502.20073*.

Jiayu Wang, Yifei Ming, Zhenmei Shi, Vibhav Vineet, Xin Wang, Yixuan Li, and Neel Joshi. 2024a. Is A picture worth A thousand words? Delving Into Spatial Reasoning for Vision Language Models. In *The Thirty-Eighth Annual Conference on Neural Information Processing Systems*.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, and 1 others. 2024b. A survey on Large Language Model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345.

Anita Williams Woolley, Christopher F. Chabris, Alex Pentland, Nada Hashmi, and Thomas W. Malone. 2010. Evidence for a Collective Intelligence Factor in the Performance of Human Groups. *Science*, 330(6004):686–688.

Guande Wu, Chen Zhao, Claudio Silva, and He He. 2024. Your co-workers matter: Evaluating collaborative capabilities of language models in Blocks World. *arXiv preprint arXiv:2404.00246*.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, and 1 others. 2025. The rise and potential of large language model based agents: A survey. *Science China Information Sciences*, 68(2):121101.

Ziyi Yang, Zaibin Zhang, Zirui Zheng, Yuxian Jiang, Ziyue Gan, Zhiyu Wang, Zijian Ling, Jinsong Chen, Martz Ma, Bowen Dong, and 1 others. 2024. Oasis: Open agents social interaction simulations on one million agents. *arXiv preprint arXiv:2411.11581*.

Bangguo Yu, Hamidreza Kasaei, and Ming Cao. 2023. Co-NavGPT: Multi-robot cooperative visual semantic navigation using large language models. *arXiv preprint arXiv:2310.05719*.

Hongxin Zhang, Weihua Du, Jiaming Shan, Qinhong Zhou, Yilun Du, Joshua B. Tenenbaum, Tianmin Shu, and Chuang Gan. 2024a. Building cooperative embodied agents modularly with large language models. In *The Twelfth International Conference on Learning Representations*.

Hongxin Zhang, Zeyuan Wang, Qiushi Lyu, and 1 others. 2024b. COMBO: Compositional world models for embodied multi-agent cooperation. *arXiv preprint arXiv:2402.15000*.

Xiaopan Zhang, Hao Qin, Fuquan Wang, and 1 others. 2024c. LaMMA-P: Generalizable multi-agent long-horizon task allocation and planning with LM-driven PDDL planner. *arXiv preprint arXiv:2403.06940*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, and 1 others. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 1(2).

Kunlun Zhu, Hongyi Du, Zhaochen Hong, Xiaocheng Yang, Shuyi Guo, Zhe Wang, Zhenhailong Wang, Cheng Qian, Xiangru Tang, Heng Ji, and 1 others. 2025. MultiAgentBench: Evaluating the collaboration and competition of LLM agents. *arXiv preprint arXiv:2503.01935*.

Appendix

Contents

1 Introduction **1**

2 Related Work **2**

3 SwarmBench Framework **2**
- 3.1 Problem Formulation . . . 2
- 3.2 Coordination Tasks . . . 3

4 Results and Analysis **5**
- 4.1 Experimental Setup . . . 5
- 4.2 Analytical Framework . . . 5
- 4.3 Cross-Task Performance Analysis . . . 5
- 4.4 The Communication-Coordination Gap . . . 6
- 4.5 Taxonomy of Coordination Failures . . . 6

5 Discussion **8**

6 Conclusion **8**

Appendix **11**

A SwarmBench System and Protocol Details **13**
- A.1 Environment Details . . . 13
- A.2 Agent Perception, Action, and Communication Details . . . 14
- A.3 Evaluation Protocol Details . . . 15
- A.4 Task-Specific Scoring Mechanisms . . . 15

B Prompt Design **20**

C Examples **26**
- C.1 Pursuit . . . 26
- C.2 Synchronization . . . 27
- C.3 Foraging . . . 28
- C.4 Flocking . . . 29
- C.5 Transport . . . 30

D Detailed Task Performance Data **31**

E Detailed Group Dynamics Metrics **32**

F Task-Specific Emergent Dynamics Analysis Visualizations **34**
- F.1 Pursuit Task Dynamics . . . 35
- F.2 Synchronization Task Dynamics . . . 37
- F.3 Foraging Task Dynamics . . . 39
- F.4 Flocking Task Dynamics . . . 41
- F.5 Transport Task Dynamics . . . 43

G Keyword Analysis **45**

H Parameter Sensitivity Analysis **46**

I Action Attribution **47**

J The SwarmBench Dataset **48**

K Additional Experiments and Analyses **49**
K.1 Quantitative Analysis of Failure Modes 49
K.2 Analysis of Communication Protocol Convergence 50

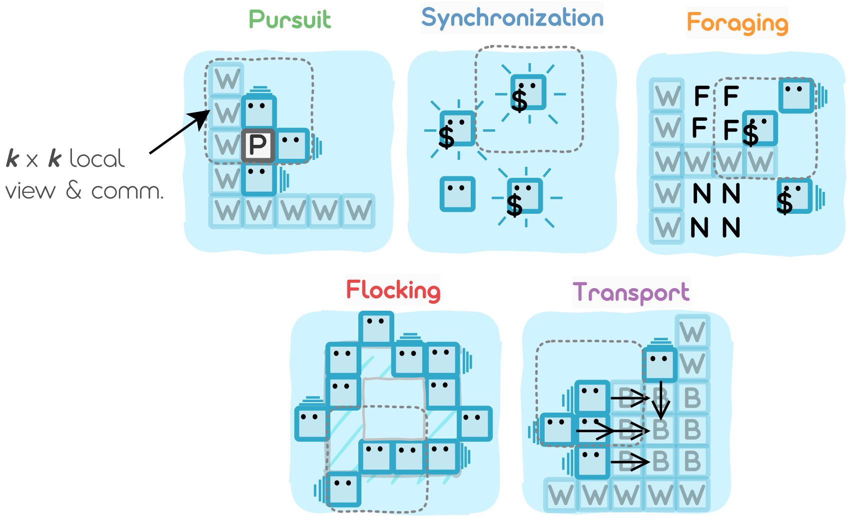

Figure 7: **SwarmBench tasks.** The diagram shows SwarmBench's task design, including Pursuit, Synchronization, Foraging, Flocking, and Transport. Each agent is limited to observing a $k \times k$ local view.

A SwarmBench System and Protocol Details

This appendix provides a detailed description of the SwarmBench environment, agent capabilities, evaluation protocol, and task-specific scoring mechanisms used in our experiments, complementing Section 3 of the main text.

A.1 Environment Details

SwarmBench utilizes a simulation environment based on a 2D grid world where multiple agents (N agents), controlled by LLMs, operate and interact. The adoption of a 2D grid world, while an abstraction, is a deliberate design choice aligned with foundational AI benchmarking practices (e.g., ARC-AGI tests (Chollet et al., 2025), SnakeBench (Kamradt, 2025), and LMAct (Ruoss et al., 2025)). This facilitates a focused investigation of core coordination dynamics while maintaining tractable complexity for initial explorations. This environment itself is designed as a customizable and scalable physical system, where mechanical interactions such as forces and multi-object dynamics are explicitly modeled.

The simulation proceeds in discrete time steps (rounds, $t = 1, \dots, T$). In each round, all agents perceive their local environment (including messages from the previous round) simultaneously and decide upon their next action and potential message based on the state at the beginning of the round. Environment updates, including agent movement and object interactions, occur only after all agents have committed to their actions for that round. Interactions between agents and objects, particularly pushing and collision resolution, are governed by this discrete physics simulation that handles complex multi-object dynamics, ensuring that the mechanical properties of the system are consistently applied.

The benchmark includes several core multi-agent coordination tasks designed to probe different facets of emergent swarm behavior. These tasks are visualized in Fig. 2 in the main text, with a consolidated overview provided in Fig. 7 (this appendix), and are further detailed with examples in Appendix C and Supplementary Videos:

- **Pursuit:** Agents (e.g., '`0`'-'`9`') must collaboratively track and corner a faster-moving prey ('`P`'). Tests coordination for containment, potentially aided by communication.

- **Synchronization:** Agents aim to synchronize an internal binary state ('`Number`' vs. '`$Number`') across the swarm and collectively alternate this state via a `SWITCH` action. Assesses consensus formation leveraging local cues and communication.

- **Foraging:** Agents navigate an environment with walls ('`W`') to find a food source ('`F`'), transport it to a nest (`'N'`), changing appearance (`'Number'` to `'$Number'`) when carrying. Evaluates exploration, pathfinding, and potential communication-driven task allocation.

- **Flocking:** Agents must move as a cohesive group, maintaining alignment and separation while potentially navigating towards a target or avoiding obstacles. Tests emergent formation control and coordinated movement.

- **Transport:** Multiple agents must cooperate to push a large object (`'B'`) towards a designated goal area. Tests coordinated force application and navigation around obstacles.

The environment framework supports additional tasks and is extensible. Interactions follow simplified physics rules. Environment instances, including initial agent positions, object placements, and potentially other environmental features, are procedurally generated based on a random `seed`. To ensure robust evaluation, prevent overfitting to specific scenarios, and guarantee fair comparison across different models or trials, evaluation runs for different models utilize the same predefined set of seeds. This practice ensures that all models are benchmarked under identical initial conditions and environmental layouts for each corresponding seed, providing a fair and consistent basis for performance comparison. Furthermore, using a diverse set of seeds ensures the benchmark itself is robust, testing models across varied conditions to provide a more reliable assessment of their general coordination abilities rather than performance on a single, potentially idiosyncratic, scenario.

A.2 Agent Perception, Action, and Communication Details

Consistent with the goal of studying emergent behavior from local information, agents operate with significantly restricted perception. The primary input is an egocentric $k \times k$ grid view (e.g., 5×5 in our main experiments) centered on the agent at position $\mathbf{x}_{i,t} \in R^2$. This view displays local entities using symbols: the agent itself (`'Y'`), other agents (by ID, e.g., `'1'`/`'$1'`), walls (`'W'`), obstacles (`'B'`), empty space (`'.'`), off-map markers (`'*'`), and task-specific objects (`'P'`, `'N'`, `'F'`). The view includes global coordinate labels.

The full observation provided to the LLM includes:

- The local $k \times k$ grid view.

- The agent's global coordinates $\mathbf{x}_{i,t}$.

- Task-specific status (e.g., `carrying_food`).

- Messages received from other agents in the previous round ($t - 1$). Messages are received only from agents within the sender's local perception range at time $t - 1$.

- The task description and current progress indicators (e.g., `score`).

- A limited history of the agent's own recent observations and actions (e.g., last `memory=5` rounds).

The detailed structure and content of the prompt given to the LLM are provided in Appendix B.

Based on this observation, the agent's LLM must decide on two outputs for round t:

1. A primary action $A_{i,t}$ chosen from a set $\mathcal{A}$ typically including basic movements (`UP`, `DOWN`, `LEFT`, `RIGHT`, `STAY`). Movement actions correspond to an agent attempting to apply a directed force (default $F = 2$). Agents and objects possess inherent weight (referred to as mass in the simulation, default agent $m = 1$ calculated from a 1×1 size). Movement or pushing only occurs if the net applied force overcomes the resistance (mass) of the target object(s), considering potentially complex chain reactions resolved by the physics engine (see Appendix **??**). Task-specific actions (e.g., `SWITCH`, `PICKUP`, `DROP`) are also included.

2. A message $M_{i,t}$ (a string, potentially empty) intended for local broadcast via the `MSG` action.

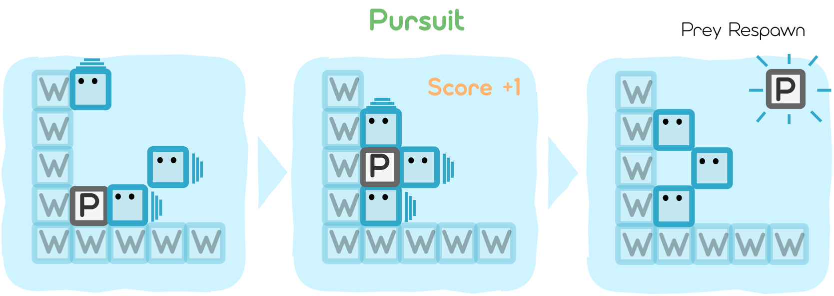

Figure 8: **Pursuit tasks illustration.** This figure shows agents surrounding a prey (`'P'`). Upon successful capture (middle panel), the prey respawns (right panel).

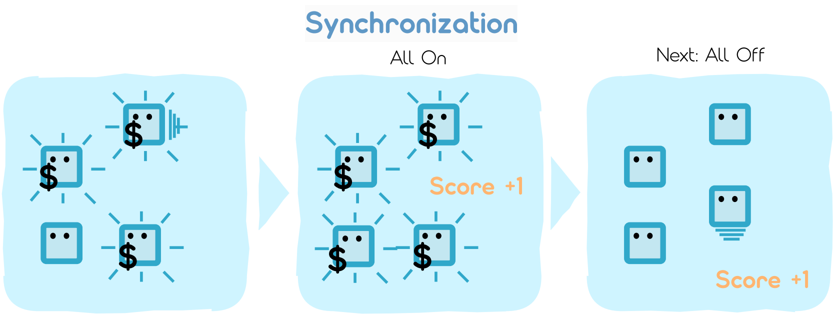

Figure 9: **Synchronization tasks illustration.** Agents toggle an internal state (indicated by `'$'` symbol or its absence). Scoring occurs when all agents are in the same state (e.g., all `'$'`, middle panel) and this state alternates from the previously scored unanimous state.

The message $M_{i,t}$ (if non-empty) is broadcast locally and anonymously to agents within the sender's local perception range, becoming part of their observation in the next round ($t+1$). Messages are subject to a character limit (e.g., 120 characters). This setup compels reliance on interpreting local visual cues and utilizing the constrained communication channel for effective coordination.

A.3 Evaluation Protocol Details

We define a standardized protocol focusing on zero-shot LLM evaluation. Each agent i is controlled by an independent LLM instance. In round t, the agent receives its full observation (including received messages from $t-1$), formulates a prompt containing this information (see Appendix B), and queries the LLM. Persistence is managed via the prompt's explicit inclusion of observation history and received messages.

The LLM response is parsed to extract the intended primary action $A_{i,t} \in \mathcal{A}$ and the message content $M_{i,t}$. An episode ends upon task success criteria being met or reaching a maximum round limit (`max_round`).

Our experiments (Section 4) utilize several contemporary closed-source and open-source LLMs, evaluated without task-specific fine-tuning to assess their inherent zero-shot coordination potential derived from pre-training.

A.4 Task-Specific Scoring Mechanisms

This subsection details the specific scoring rules for each of the five core tasks in SwarmBench, which are broadly depicted in Fig. 7. To further clarify the scoring process for several of these tasks, Figures 8, 9, 10, 11, and 12 provide specific illustrations for the Pursuit, Synchronization, Foraging, Flocking, and Transport tasks, respectively. These rules are implemented within the simulation environment to quantify agent performance based on their success in achieving the defined task objectives. The scores reported in Section 4 are derived from these mechanisms.

A.4.1 Pursuit Scoring

The `Pursuit` task, depicted in Fig. 8, involves agents cooperatively cornering a faster-moving prey (`'P'`). The illustration clearly shows the stages: agents maneuvering to surround the prey (left), the prey being successfully cornered leading to a score increment (center), and the prey subsequently respawning at a new location (right), allowing the task to continue.

- **Scoring Event (Prey Caught):** The prey is considered caught if all four of its adjacent cells (up, down, left, right) are occupied by other agents or walls (`'W'`). This condition is visually represented in the middle panel of Fig. 8.
- **Score Awarded:** When the prey is caught, the task score is incremented by 1.

$$\texttt{score} \leftarrow \texttt{score} + 1 \tag{1}$$

- **Prey Respawn:** After being caught, the prey is removed and respawned at a new, empty location on the map. This location (x_s, y_s) is selected from several random candidate empty cells by choosing the one that minimizes a *threat heuristic*, H. For any cell (x, y), this heuristic is calculated based on an 8×8 subview $V_{8\times8}(x, y)$ centered around it:

$$H(x, y) = N_A(V_{8\times8}(x, y)) + w_W \cdot N_W(V_{8\times8}(x, y)) \tag{2}$$

  where $N_A(V)$ is the number of agents within subview V, $N_W(V)$ is the number of wall cells within V, and w_W is a weight for walls (set to 0.9 in our implementation). The prey respawns at the candidate location with the minimum H value. This process is illustrated in the right panel of Fig. 8.
- **Prey Movement:** If not caught, the prey attempts to move two steps in each round. It considers all valid two-step sequences. A sequence is valid if both intermediate and final cells are empty. From these valid sequences, it selects the one whose destination cell (x_d, y_d) minimizes the threat heuristic $H(x_d, y_d)$ as defined in Eq. 2 (calculated for the 8×8 subview around the destination (x_d, y_d)). The prey aims to move to safer locations by avoiding high densities of agents and walls.
- **Total Score:** The cumulative number of times the prey has been successfully caught.

The scoring directly rewards the primary objective: successfully surrounding and immobilizing the prey, with repeated opportunities as the prey respawns.

A.4.2 Synchronization Scoring

In the `Synchronization` task, illustrated in Fig. 9, agents must synchronize an internal binary state (e.g., light on/off, represented by agent symbols `'A'`/`'a'` or, as in the figure, by the presence or absence of a `'$'` sign on the agent and status like `'$Number'`) and collectively alternate this synchronized state. The figure demonstrates agents transitioning to a unanimous "All On" state (middle panel), followed by a subsequent target of "All Off" to score again. Agents can use a `SWITCH` action to toggle their own state.

- **Agent States:** Each agent i has an internal boolean state, tracked by `agent_state[i]`, representing whether its light is on or off.
- **Scoring Condition:** A point is scored if the following two conditions are met:
  1. **Unanimity:** All agents currently have their lights in the same state (i.e., all lights are on, or all lights are off). This collective state is referred to as `state` in the implementation.
  2. **Alternation:** This newly achieved unanimous `state` (e.g., all on) is different from the unanimous state (`self.prev_state`) for which a point was last awarded (e.g., if the last score was for all off). This ensures the group must alternate between collective states to continue scoring.

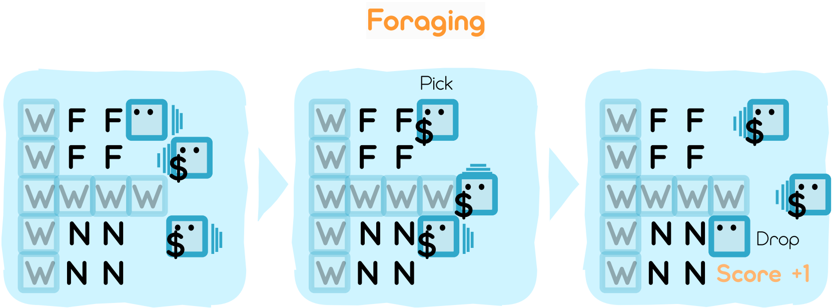

Figure 10: **Foraging tasks illustration.** Agents pick up food (`'F'`, indicated by `'$'` on the agent in the middle panel) and transport it to a nest (`'N'`). A score is awarded upon dropping food at the nest (right panel).

- **Score Awarded:** If both unanimity and alternation conditions are satisfied, the task score is incremented by 1. The `self.prev_state` variable is then updated to record the current unanimous `state` for future alternation checks.

$$\texttt{score} \leftarrow \texttt{score} + 1 \tag{3}$$

- **Total Score:** The cumulative number of successful, alternating synchronizations.

This scoring mechanism incentivizes not just achieving a common state, but also the ability to collectively switch to the opposite common state, testing robust group consensus and coordination over time.

A.4.3 Foraging Scoring

The `Foraging` task, shown in Fig. 10, requires agents to navigate an environment containing walls (`'W'`), pick up food (`'F'`) from a source, and deliver it to a nest (`'N'`). The illustration highlights agents changing appearance (e.g., acquiring a `'$'` symbol, as in the "`Pick`" stage, middle panel) when carrying food and scoring upon successful delivery ("`Drop`" stage, right panel) to the nest. Agents carrying food are visually distinct (e.g., agent symbol `'$Number'`) from those not carrying food (e.g., agent symbol `'Number'`).

- **Picking Up Food:** If an agent is adjacent to the food source (`'F'`) and not currently carrying food, its internal state, tracked by `food_state[name]`, is updated to indicate it is now carrying food. Its visual representation also changes accordingly, as shown in the transition from the left to the middle panel of Fig. 10.
- **Dropping Food (Scoring Event):** If an agent is adjacent to the nest (`'N'`) and its `food_state[name]` indicates it is currently carrying food, then:
  - The task score is incremented by 1 (see the right panel of Fig. 10).

$$\texttt{score} \leftarrow \texttt{score} + 1 \tag{4}$$

  - The agent's `food_state[name]` is updated to indicate it is no longer carrying food (it drops the food), and its visual representation reverts.
- **Total Score:** The cumulative number of food items successfully delivered to the nest by all agents.

The score directly reflects the collective efficiency in the foraging cycle: finding food, transporting it, and returning it to the nest.

A.4.4 Flocking Scoring

In the `Flocking` task (visualized in the Flocking panel of Fig. 7), agents aim to arrange themselves to match a predefined target shape (e.g., a hollow square made of agents). Performance evaluation in this task leverages a metric inspired by the Earth Mover's Distance (EMD) (i.e., Wasserstein metric).

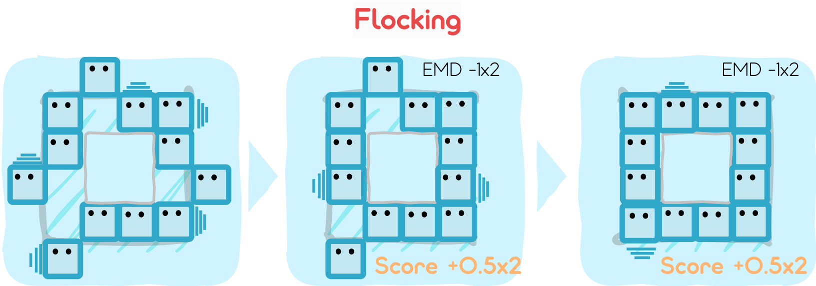

Figure 11: **Flocking tasks illustration.** This figure depicts the Flocking task where agents aim to arrange themselves into a predefined target shape, here a hollow square. The panels show the progression from an initial agent configuration towards achieving the target.

- **Core Metric (Translation-Invariant Assignment Cost):** The Earth Mover's Distance (EMD) generally measures the minimum work required to transform one probability distribution into another. In this task, we compute a specific variant to assess the dissimilarity between the current spatial configuration of agents and the target shape. The computation proceeds as follows:
  1. **Coordinate Extraction:** The coordinates of agents (`src`) and the coordinates defining the target shape (`tgt`) are extracted.
  2. **Candidate Translations:** A set of candidate global translation vectors $(\Delta x, \Delta y)$ is generated. These candidates are derived from all pairwise coordinate differences between individual agents in `src` and points in `tgt`.
  3. **Optimal Assignment under Translation:** For each candidate global translation $(\Delta x, \Delta y)$:
     - A cost matrix is constructed. Each entry C_{ij} in this matrix represents the Manhattan distance required to match agent $i \in$ `src` to target point $j \in$ `tgt`, assuming the entire `src` configuration is first translated by $(-\Delta x, -\Delta y)$ (or equivalently, `tgt` is translated by $(\Delta x, \Delta y)$). The cost is $\frac{1}{2}\left|(x_{\text{src}_i} - x_{\text{tgt}_j}) - \Delta x\right| + \frac{1}{2}\left|(y_{\text{src}_i} - y_{\text{tgt}_j}) - \Delta y\right|$.
     - An optimal assignment algorithm (such as the Hungarian method or similar techniques for solving the assignment problem) is then applied to this cost matrix. This finds a pairing between agents and target points that minimizes the total sum of Manhattan distances for the current global translation.
  4. **Minimum Cost Selection:** The lowest sum of assignment costs found across all candidate global translations is selected as the final dissimilarity score. This score, referred to as `cur_dis` in the implementation, effectively represents the minimum "work" (i.e., sum of Manhattan distances) to make the agent formation match the target shape, after finding the optimal global alignment (translation).

- **Initial Distance:** Upon task reset, an initial dissimilarity score (`init_dis`) is calculated using the same method, based on the random initial placement of agents. This serves as a baseline.

- **Scoring Update:** At each simulation step, the current dissimilarity score (`cur_dis`) is recalculated. The task score reflects the cumulative reduction in this dissimilarity from the initial state, ensuring the score is non-decreasing. The overall task score is then updated to be the maximum progress achieved so far:
$$\texttt{score} \leftarrow \max(\texttt{score}, \texttt{init\_dis} - \texttt{cur\_dis}) \tag{5}$$

- **Task Completion:** The task is considered successfully completed if the `cur_dis` reaches 0, indicating that the agents have perfectly matched the target shape under some translation.

This scoring mechanism incentivizes agents to collectively maneuver towards and achieve the target configuration. By finding the optimal translational alignment before computing the assignment cost, the metric robustly measures shape conformance irrespective of the absolute global position of the formation.

Figure 12: **Transport tasks illustration.** This figure shows agents collaborating to push a block (`'B'`) and then escaping. Scores are awarded based on the speed of escape, as exemplified by the progression from 42/100 to 82/100 rounds.

A.4.5 Transport Scoring

The `Transport` task requires agents to collaboratively push a large obstacle (`'B'`, with $m = 5$) out of an exit in the surrounding walls and then escape themselves. Fig 12 depicts various stages of this task, highlighting how agents must first coordinate to move the heavy block (e.g., note the arrows indicating intended push direction) and subsequently earn points by escaping the map, with scores reflecting the remaining time.

- **Scoring Event:** An agent i at position $\mathbf{p}_i = (y_i, x_i)$ successfully escapes if it moves outside the map boundaries $\mathcal{M} = [0, H) \times [0, W)$. The escape condition is:

$$\mathbf{p}_i \notin \mathcal{M}. \tag{6}$$

- **Score Awarded:** For each agent that escapes, a score is awarded. This score is proportional to the remaining time in the simulation. Let T be the maximum number of rounds and t be the current round; the score is updated as follows:

$$S \leftarrow S + \frac{T - t}{T} \tag{7}$$

  This encourages agents to complete the task (pushing the obstacle and escaping) as quickly as possible.

- **Total Score:** The cumulative sum of scores from all escaped agents.

- **Task Completion:** The task is considered done when all agents have escaped the map.

The primary challenge involves the coordinated push of the heavy obstacle, as individual agents cannot move it. The scoring incentivizes both successful obstacle removal and efficient individual escape.

B Prompt Design

The following tcolorbox shows the exact structure and content of the prompt string generated by the 'gen_prompt' function and provided to each LLM agent in SwarmBench at each decision step. Placeholders within curly braces (e.g., {name}, {task_desc}, {view_str}) are dynamically filled with actual simulation data during runtime.

SwarmBench Agent Prompt Template

```
"""You are Agent {name}, operating in a multi-agent environment. Your goal is
to complete the task through exploration and collaboration.

Task description:
{task_desc}

Round: {round_num}

Your recent {self.memory}-step vision (not the entire map):
{view_str}

Your current observation:
{level_obs_str}

Message you received:
{messages_str}

Your action history:
{history_str}

Symbol legend:
- Number: An agent whose id is this number (do not mistake column no. and line
no. as agent id).
- Y: Yourself. Others see you as your id instead of "Y".
- W: Wall.
- B: Pushable obstacle (requires at least 5 agents pushing in the same direction).
- .: Empty space (you can move to this area).
- *: Area outside the map.
And other symbols given in task description (if any).

Available actions:
1. UP: Move up
2. DOWN: Move down
3. LEFT: Move left
4. RIGHT: Move right
5. STAY: Stay in place
6. MSG: Send a message
And other actions given in task description (if any).

Physics rules:
1. Your own weight is 1, and you can exert a force of up to 2.
2. An object (including yourself) can only be pushed if the total force in one
direction is greater than or equal to its weight.
```

3. Static objects like W (walls) cannot be pushed; only B can be pushed.
4. Force can be transmitted, but only between directly adjacent objects. That means, if an agent is applying force in a direction, you can push that agent from behind to help.
5. Only pushing is allowed - there is no pulling or lateral dragging. In other words, to push an object to the right, you must be on its left side and take the RIGHT action to apply force.

Message rules:
1. A message is a string including things you want to tell other agents.
2. Your message can be received by all agents within your view, and you can receive messages from all agents within your view.
3. Messages are broadcast-based. The source of a message is anonymous.
4. Write only what's necessary in your message. Avoid any ambiguity in your message.
5. Messages is capped to no more than 120 characters, exceeding part will be replaced by "...".

Other rules:
1. Coordinates are represented as (i, j), where i is the row index and j is the column index. Your 5x5 vision uses global coordinates, so please use global coordinates.
2. The direction of increasing i is downward, and increasing j is to the right.
3. Objects that are completely outside the map (marked with "*") will be removed.

Please think carefully and choose your next action. You will need to collaborate with other agents to successfully complete the task.

Your response should include:
1. Analysis of the current situation
2. Your decision and reasoning
3. The message to be left (if any)

End your response clearly with your chosen action: "ACTION: [YOUR_ACTION]" and/or "MSG: [Your message (no line breaks).]"
"""

Example SwarmBench Prompt (Agent_5, Round 62, Transport Task)

You are Agent Agent_5, operating in a multi-agent environment. Your goal is to complete the task through exploration and collaboration.

Task description:
The boundary of the map is surrounded by walls (denoted as W), with a gap leading to the outside of the map (denoted as '*'). The gap is blocked by an obstacle (denoted as B).
The goal is to first locate the obstacle (B), then have five robots simultaneously push it through the exit, and finally escape to the outside of the map (denoted as '*').

Round: 62

```
Your recent 5-step vision (not the entire map):
Current Step:
     1   2   3   4   5
  3  1   B   B   B   B
  4  9   8   0   11  10
  5  .   .   Y   .   7
  6  .   .   .   .   .
  7  .   .   .   .   .

1 Steps Before:
     1   2   3   4   5
  3  1   B   B   B   B
  4  .   9   8   0   11
  5  .   .   Y   .   7
  6  .   .   .   .   .
  7  .   .   .   .   .

2 Steps Before:
     1   2   3   4   5
  3  1   B   B   B   B
  4  .   9   8   0   11
  5  .   .   Y   .   7
  6  .   .   .   .   .
  7  .   .   .   .   .

3 Steps Before:
     1   2   3   4   5
  3  1   B   B   B   B
  4  .   9   8   0   11
  5  .   .   Y   .   7
  6  .   .   .   .   .
  7  .   .   .   .   .

4 Steps Before:
     1   2   3   4   5
  3  1   B   B   B   B
  4  .   9   8   0   11
  5  .   .   Y   .   7
  6  .   .   .   .   .
  7  .   .   .   .   .


Your current observation:


Message you received:
Message: "At (5,3), moving UP to (4,3) for LEFT push on B at (3,3). Ready for
5-force."
Message: "Moving to (4,5) to join LEFT push on B at (3,5). Ready for 5-force
push."
```

Message: ”Moving RIGHT to (4,3) then (4,4) for LEFT push on B. Ready to join force.”
Message: ”At (4,5) pushing LEFT on B at (3,5). 5-force confirmed. Executing now.”

Your action history:
Round 58: Action: UP, Message: ”Moving to (4,3) then (3,4) for LEFT push on B. Need 4 more at (3,4)/(4,4).”
Round 59: Action: UP, Message: ”Moving to (4,3) then (3,4) for LEFT push on B. Need 4 more at (3,4)/(4,4).”
Round 60: Action: UP, Message: ””Moving to (3,3) for LEFT push on B. Need 1 more at (3,3)/(4,3) to complete 5-force.””
Round 61: Action: UP, Message: ””Moving to (4,3) for LEFT push on B at (3,3). Ready to join 5-force push.””
Round 62: Action: UP, Message: ”At (5,3), moving UP to (4,3) for LEFT push on B at (3,3). Ready for 5-force.”

Symbol legend:
- Number: An agent whose id is this number (do not mistake column no. and line no. as agent id).
- Y: Yourself. Others see you as your id instead of ”Y”.
- W: Wall.
- B: Pushable obstacle (requires at least 5 agents pushing in the same direction).
- .: Empty space (you can move to this area).
- *: Area outside the map.
And other symbols given in task description (if any).

Available actions:
1. UP: Move up
2. DOWN: Move down
3. LEFT: Move left
4. RIGHT: Move right
5. STAY: Stay in place
6. MSG: Send a message
And other actions given in task description (if any).

Physics rules:
1. Your own weight is 1, and you can exert a force of up to 2.
2. An object (including yourself) can only be pushed if the total force in one direction is greater than or equal to its weight.
3. Static objects like W (walls) cannot be pushed; only B can be pushed.
4. Force can be transmitted, but only between directly adjacent objects. That means, if an agent is applying force in a direction, you can push that agent from behind to help.
5. Only pushing is allowed - there is no pulling or lateral dragging. In other words, to push an object to the right, you must be on its left side and take the RIGHT action to apply force.

Message rules:
1. A message is a string including things you want to tell other agents.
2. Your message can be received by all agents within your view, and you can receive messages from all agents within your view.
3. Messages are broadcast-based. The source of a message is anonymous.
4. Write only what's necessary in your message. Avoid any ambiguity in your message.
5. Messages is capped to no more than 120 characters, exceeding part will be replaced by ”...”.

Other rules:
1. Coordinates are represented as (i, j), where i is the row index and j is the column index. Your 5x5 vision uses global coordinates, so please use global coordinates.
2. The direction of increasing i is downward, and increasing j is to the right.
3. Objects that are completely outside the map (marked with ”*”) will be removed.

Please think carefully and choose your next action. You will need to collaborate with other agents to successfully complete the task.

Your response should include:
1. Analysis of the current situation
2. Your decision and reasoning
3. The message to be left (if any)

End your response clearly with your chosen action: ”ACTION: [YOUR_ACTION]” and/or ”MSG: [Your message (no line breaks).]”

Example Agent Response (JSON)

{ "response": " Analysis of the Current Situation:
1. **Obstacle (B) Location**: The obstacle B is located at positions (3,2), (3,3), (3,4), and (3,5). The exit is likely at one of these positions, but the exact exit (*) is not visible in the current vision. However, based on the messages, the agents are focusing on pushing B at (3,3) and (3,5) LEFT (towards column 1).
2. **Agent Positions**:
- You (Y) are at (5,3).
- Other agents are at (4,3), (4,5), and possibly others not visible in the current vision.
- Messages indicate that agents are ready for a 5-force push on B at (3,3) and (3,5).
3. **Messages**:
- Agents are coordinating to push B LEFT at (3,3) and (3,5).
- A 5-force push is confirmed for B at (3,5), and agents are moving into position for B at (3,3).
4. **Your Role**: You are at (5,3) and can move UP to (4,3) to join the push on B at (3,3). Alternatively, you could move to (4,5) to assist with the push on B at (3,5), but the messages suggest the push at (3,5) is already confirmed with enough agents.

Decision and Reasoning:
- **Primary Goal**: Assist in the 5-force push on B at (3,3). The messages indicate that agents are ready at (4,3) and (4,4) for this push. You are at (5,3) and can move UP to (4,3) to join the push.
- **Action**: Move UP to (4,3) to align yourself for the LEFT push on B at (3,3). This will contribute to the 5-force needed to move the obstacle.
- **Message**: Confirm your readiness to join the push at (3,3) to ensure coordination.

Message:
"At (5,3), moving UP to (4,3) for LEFT push on B at (3,3). Ready for 5-force."

Action:
ACTION: UP
MSG: "At (5,3), moving UP to (4,3) for LEFT push on B at (3,3). Ready for 5-force."",
"action": "UP",
"message": ""At (5,3), moving UP to (4,3) for LEFT push on B at (3,3). Ready for 5-force."" }

C Examples

This appendix provides visual examples of the simulation environment for each of the five core SwarmBench tasks. Each figure shows a snapshot from a simulation run, illustrating agent positions, environment features, and the overall task objective.

C.1 Pursuit

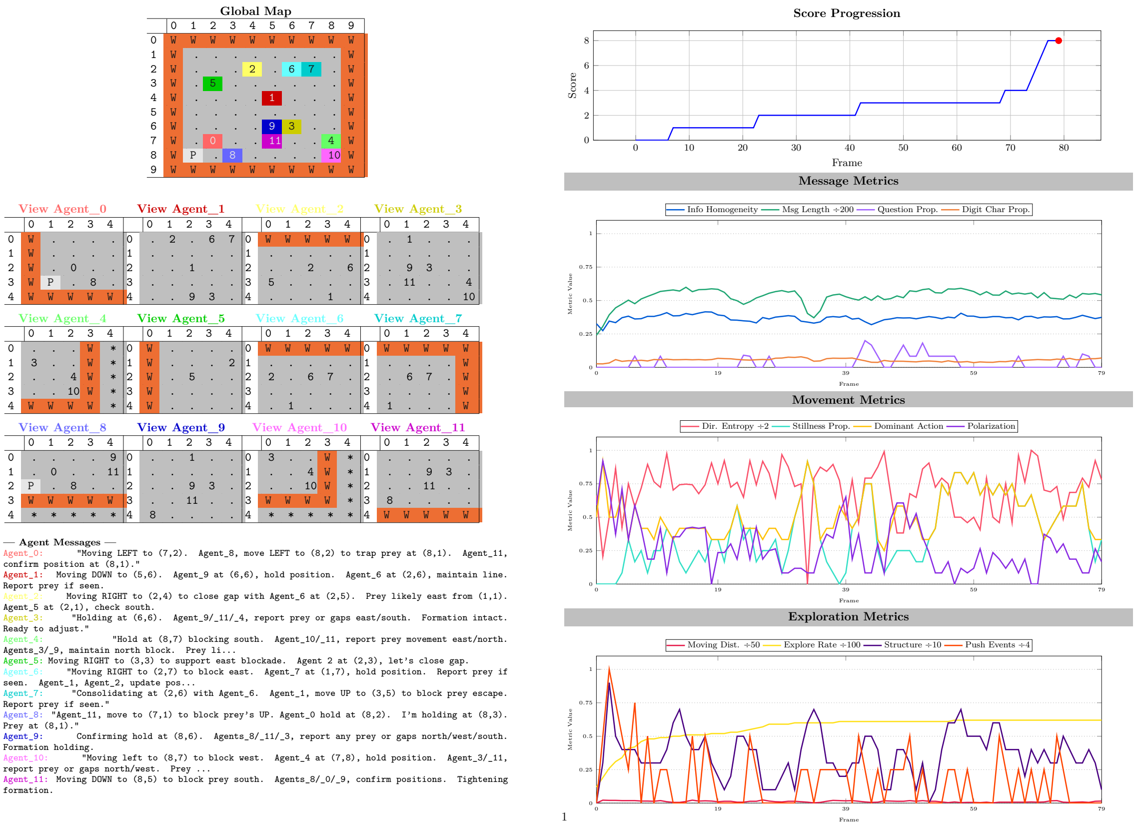

Figure 13: Example visualization for the `Pursuit` task. Agents (`0-11`) attempt to surround the prey (`P`). Replay videos can be found in Supplementary Materials (see Supplementary Video 1)

C.2 Synchronization

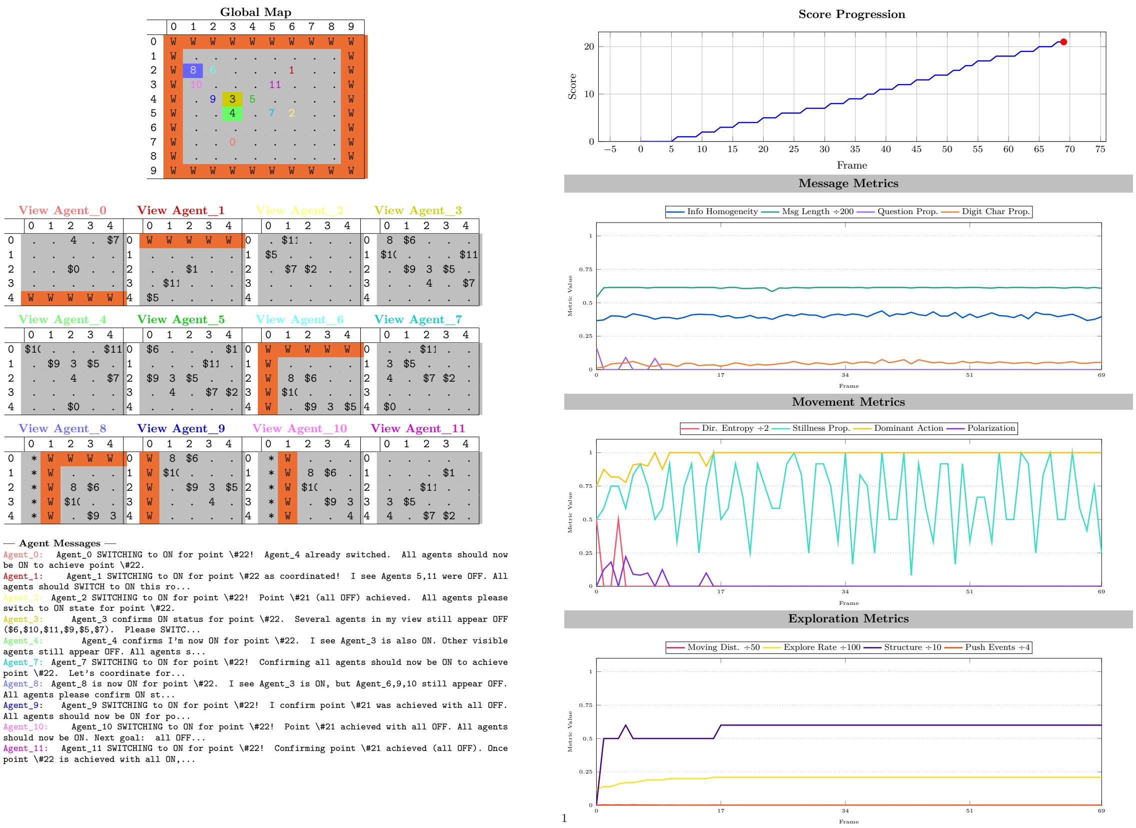

Figure 14: Example visualization for the Synchronization task. Agents (Number/$Number) aim to reach a consensus state. Replay videos can be found in Supplementary Materials (see Supplementary Video 2)

C.3 Foraging

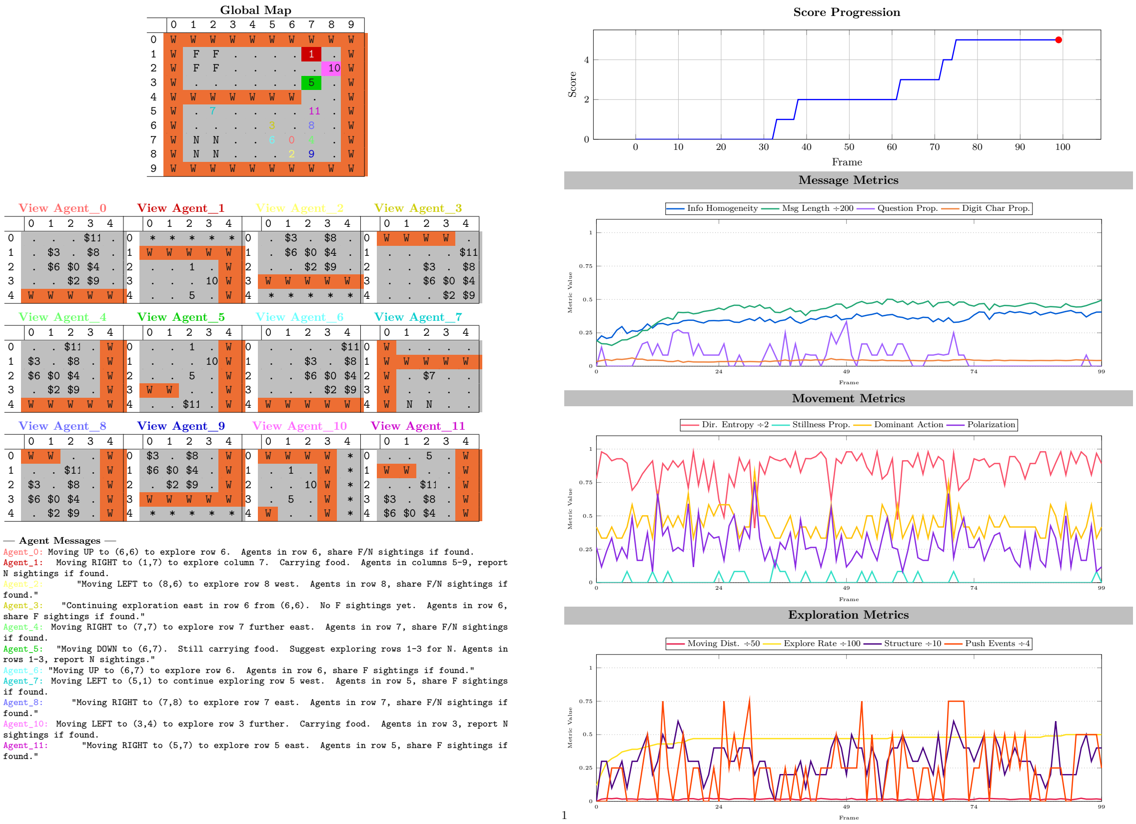

Figure 15: Example visualization for the `Foraging` task. Agents (`Number/$Number`) collect food (`F`) and return it to the nest (`N`). Replay videos can be found in Supplementary Materials (see Supplementary Video 3)

C.4 Flocking

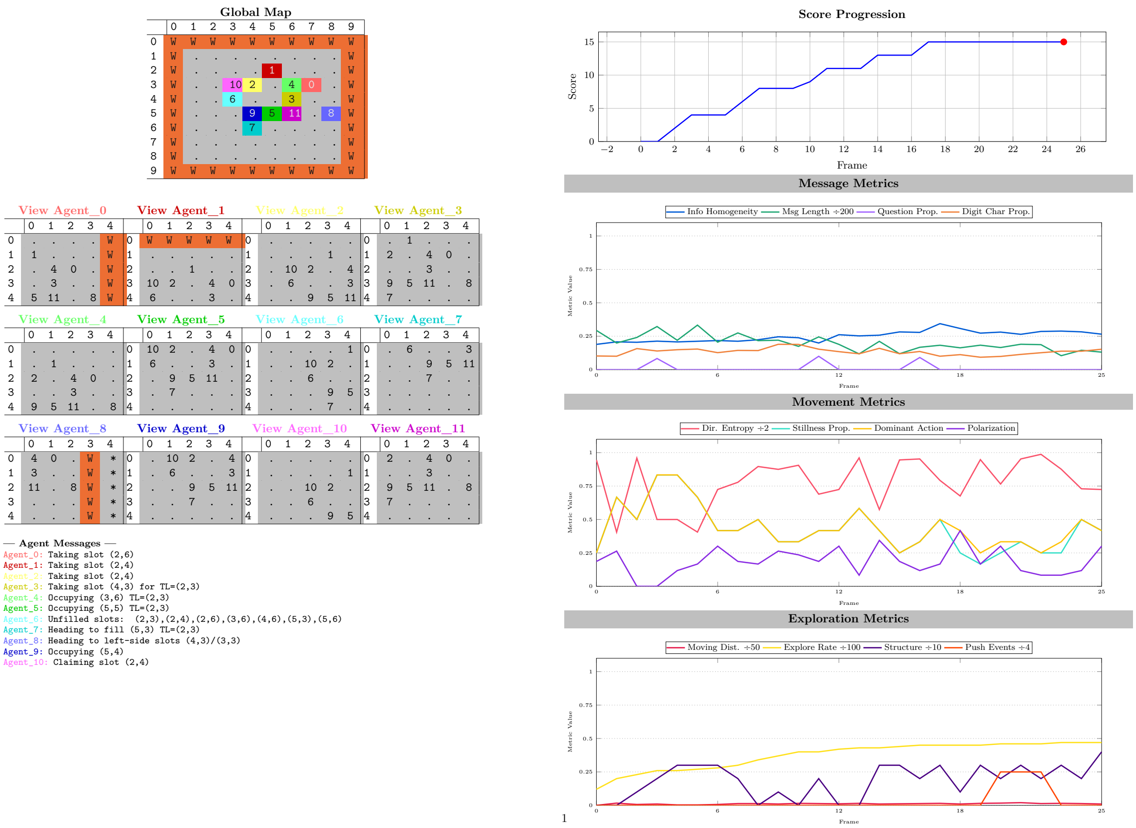

Figure 16: Example visualization for the `Flocking` task. Agents (`0-11`) attempt to move cohesively. Replay videos can be found in Supplementary Materials (see Supplementary Video 4)

C.5 Transport

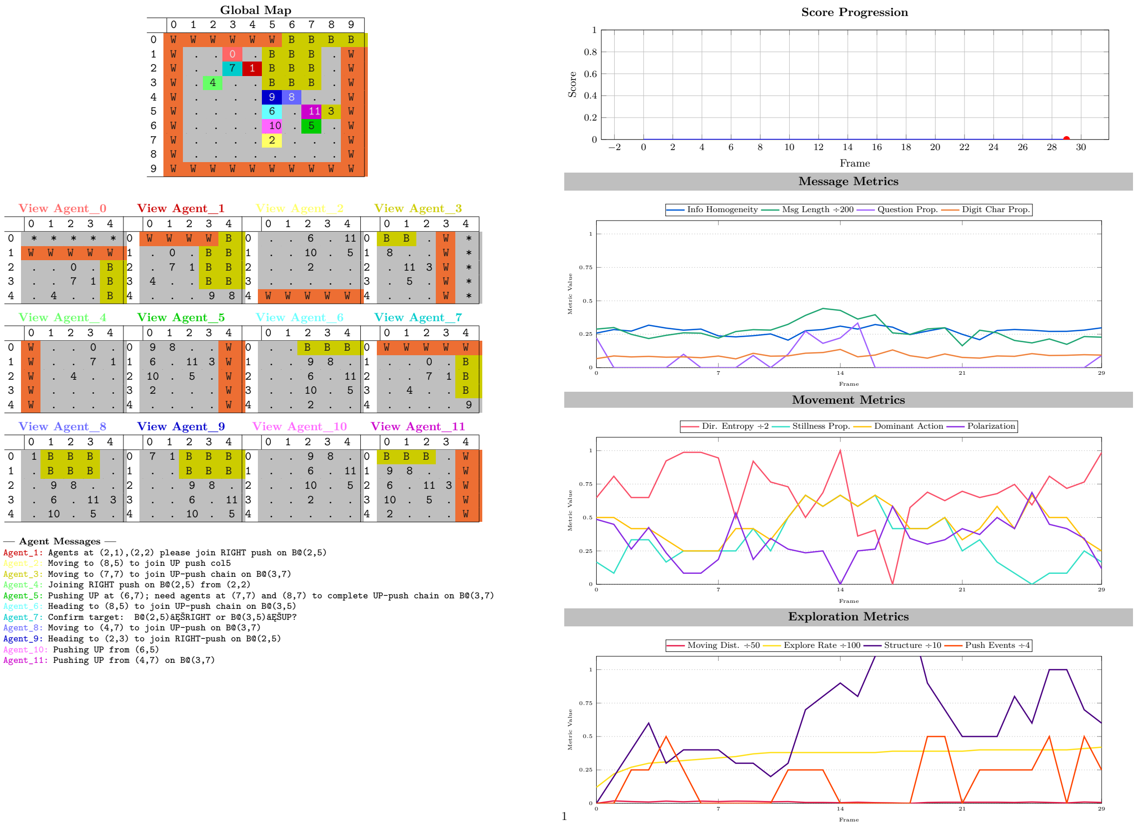

Figure 17: Example visualization for the `Transport` task. Agents (`0-11`) coordinate to push a large obstacle (`B`). Replay videos can be found in Supplementary Materials (see Supplementary Video 5)

D Detailed Task Performance Data

Table 3 provides the detailed numerical results corresponding to the performance overview presented in Tab. 2 in the main text. It shows the mean scores and standard deviations for each evaluated LLM across the five SwarmBench tasks, averaged over 5 simulation runs. Models are ordered by their total score (sum across the five tasks) in descending order.

Table 3: **Detailed average scores with standard deviations for various LLMs across five SwarmBench tasks.** Tasks: `Pursuit`, `Synchronization`, `Foraging`, `Flocking`, `Transport`. Scores averaged over 5 simulations. Models ordered by Total Score.

Model	Pursuit	Synchronization	Foraging	Flocking	Transport	Total Score
`gemini-2.0-flash`	8.80 ± 1.60	3.40 ± 2.94	5.80 ± 4.35	9.40 ± 0.80	0.00 ± 0.00	27.40
`o4-mini`	9.60 ± 0.49	2.80 ± 1.17	4.80 ± 2.64	8.90 ± 1.83	0.52 ± 1.04	26.62
`claude-3.7-sonnet`	4.40 ± 1.20	12.60 ± 9.62	1.20 ± 1.47	7.50 ± 1.93	0.00 ± 0.00	25.70
`gpt-4.1`	8.40 ± 1.85	2.80 ± 0.75	3.20 ± 1.94	5.70 ± 0.68	0.00 ± 0.00	20.10
`deepseek-v3`	4.20 ± 2.48	4.00 ± 1.41	2.60 ± 2.06	6.40 ± 1.40	0.00 ± 0.00	17.20
`gpt-4o`	3.40 ± 1.50	1.80 ± 1.33	1.60 ± 1.85	5.00 ± 3.18	0.00 ± 0.00	11.80
`o3-mini`	3.60 ± 2.06	2.20 ± 1.17	2.60 ± 3.88	2.70 ± 0.93	0.00 ± 0.00	11.10
`qwq-32b`	2.20 ± 1.94	1.20 ± 0.98	0.80 ± 0.75	5.90 ± 0.20	0.00 ± 0.00	10.10
`deepseek-r1`	1.00 ± 0.63	1.20 ± 1.17	1.00 ± 1.10	6.10 ± 0.38	0.71 ± 1.42	10.01
`llama-3.1-70b`	1.80 ± 0.40	1.00 ± 1.10	0.00 ± 0.00	7.10 ± 0.74	0.00 ± 0.00	9.90
`llama-4-scout`	1.20 ± 0.75	0.20 ± 0.40	1.00 ± 1.55	7.10 ± 2.44	0.00 ± 0.00	9.50
`gpt-4.1-mini`	1.40 ± 0.80	0.60 ± 0.49	1.40 ± 1.02	5.00 ± 2.76	0.00 ± 0.00	8.40
`claude-3.5-haiku`	0.60 ± 0.49	1.00 ± 0.00	0.00 ± 0.00	5.60 ± 0.74	0.00 ± 0.00	7.20

E Detailed Group Dynamics Metrics

To quantitatively analyze emergent collective behaviors, we compute metrics based on agent positions $\mathbf{x}_{i,t}$, their primary actions $A_{i,t}$, and messages $M_{i,t}$. These metrics are calculated per round and then typically averaged over the duration of a simulation run for correlation with the final score. The specific variable names used in our analysis scripts correspond to these conceptual definitions.

Communication-based Metrics

- **Proportion of Question Sentences:** The average per-round proportion of non-empty messages that contain a question mark (`'?'`).
- **Proportion of Digit Characters:** The average per-round proportion of all characters in non-empty messages that are digits. This may indicate sharing of numerical data like coordinates.
- **Mean Message Length:** The average per-round mean character length of non-empty messages.
- **Standard Deviation of Message Length:** The average per-round standard deviation of character lengths of non-empty messages.
- **Information Homogeneity:** The average per-round pairwise cosine similarity of embeddings of unique non-empty messages. Embeddings are generated using a Sentence-BERT model (e.g., `all-mpnet-base-v2`). This measures semantic coherence.

Action-based Metrics Let $\mathcal{A}_{\text{move}} = \{\texttt{UP}, \texttt{DOWN}, \texttt{LEFT}, \texttt{RIGHT}\}$ be movement actions and $\mathcal{A}_{\text{coord}} = \mathcal{A}_{\text{move}} \cup \{\texttt{STAY}\}$ be coordination-relevant actions.

- **Directional Entropy:** The average per-round Shannon entropy of actions in $\mathcal{A}_{\text{move}}$ taken by agents. Measures the unpredictability or variability of movement directions.

$$H_t(\mathcal{A}_{\text{move}}) = -\sum_{a \in \mathcal{A}_{\text{move}}} p_t(a) \log_2 p_t(a) \tag{8}$$

  where $p_t(a)$ is the proportion of agents performing action a in round t from the set $\mathcal{A}_{\text{move}}$.

- **Stillness Proportion:** The average per-round proportion of agents executing the `STAY` action.
- **Dominant Action Proportion:** The average per-round proportion of agents performing the single most frequent action within the set $\mathcal{A}_{\text{coord}}$. A high value indicates strong action consensus.
- **Polarization Index:** The average per-round magnitude of the mean movement vector. Action vectors $\mathbf{v}(a)$ are assigned (e.g., $\mathbf{v}(\texttt{UP}) = (0, -1)$, $\mathbf{v}(\texttt{STAY}) = (0, 0)$).

$$P_t = \left\| \frac{1}{N_t^{\text{coord}}} \sum_{i \text{ s.t. } A_{i,t} \in \mathcal{A}_{\text{coord}}} \mathbf{v}(A_{i,t}) \right\|_2 \tag{9}$$

  where N_t^{coord} is the number of agents performing an action from $\mathcal{A}_{\text{coord}}$ in round t. Indicates overall movement alignment.

Position and Interaction-based Metrics

- **Average Moving Distance:** The average per-round cumulative Manhattan distance moved by each agent from its previous position.
- **Exploration Rate:** The average per-round number of unique grid cells occupied by any agent up to that round.

- **Local Structure Preservation Count:** The average per-round count of pairs of agents that were adjacent (Manhattan distance 1) in round $t-1$ and remain adjacent in round t.
- **Agent Push Events:** The average per-round count of events where agent A, intending to move into agent B's adjacent cell, successfully does so, and agent B is displaced in the same direction as A's intended movement. This indicates a successful cooperative push.

F Task-Specific Emergent Dynamics Analysis Visualizations

This appendix provides detailed visualizations supporting the analysis of emergent group dynamics and their correlation with task performance. For each of the five core SwarmBench tasks, we present a series of plots to illustrate these relationships. The twelve dynamic features analyzed are defined in Appendix E.

For each task, we show:

1. A heatmap of the Pearson correlation coefficients between all pairs of the twelve dynamic features and the final task score (e.g., Fig. 18). This provides an overview of inter-feature relationships and feature-score correlations.

2. A bar plot showing the Pearson correlation coefficient of each dynamic feature specifically with the final task score. Asterisks (*, **, ***) indicate statistical significance ($p < 0.05, p < 0.01, p < 0.001$ respectively) (e.g., Fig. 19). This highlights the individual predictive power of each feature.

3. A scatter plot illustrating the relationship between the dynamic feature with the highest absolute Pearson correlation with the score and the final task score, including a linear regression trend line (e.g., Fig. 20). This visualizes the strength and direction of the strongest individual relationship.

4. A swarm plot of the feature importance (using SHAP (Lundberg and Lee, 2017)) when predicting the final task score using all dynamic features (e.g., Fig. 21). This indicates the relative contribution of each feature in a multivariate context.

These visualizations offer a task-specific deep dive into how different emergent behaviors and communication patterns relate to performance, providing the detailed evidence for the trends discussed in the main text.

F.1 Pursuit Task Dynamics

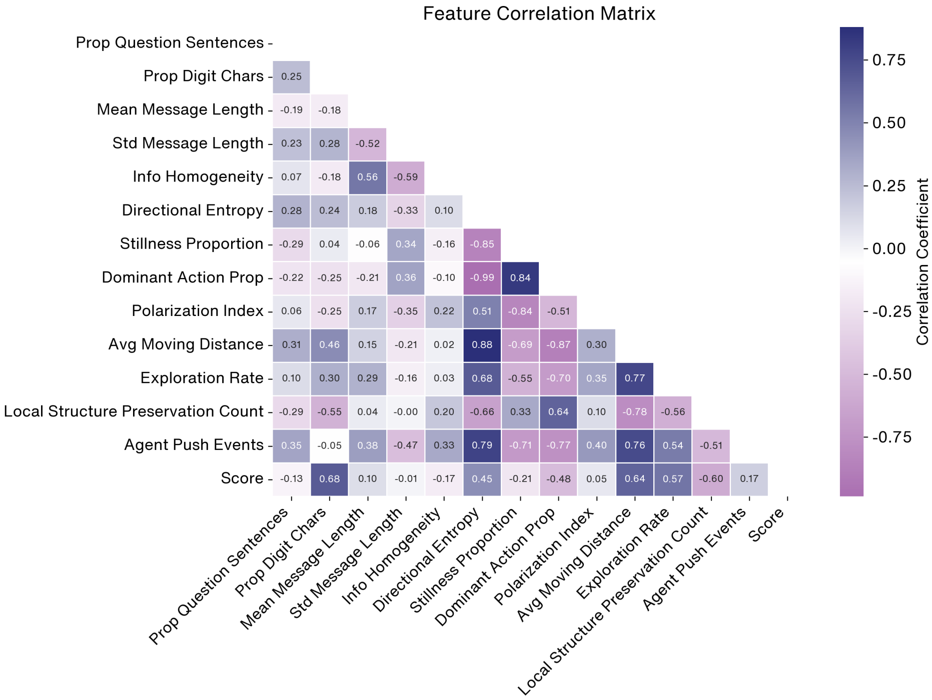

Figure 18: Feature correlation matrix for the Pursuit task. This heatmap shows Pearson correlation coefficients between all pairs of dynamic features and the task score.

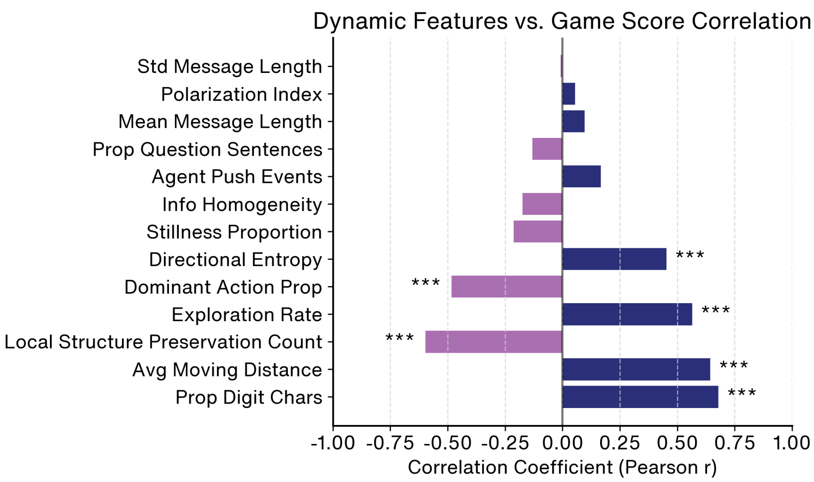

Figure 19: Correlation of dynamic features with score for the Pursuit task. Bars represent Pearson's r; * indicates $p < 0.05$,** indicates $p < 0.01$,*** indicates $p < 0.001$.

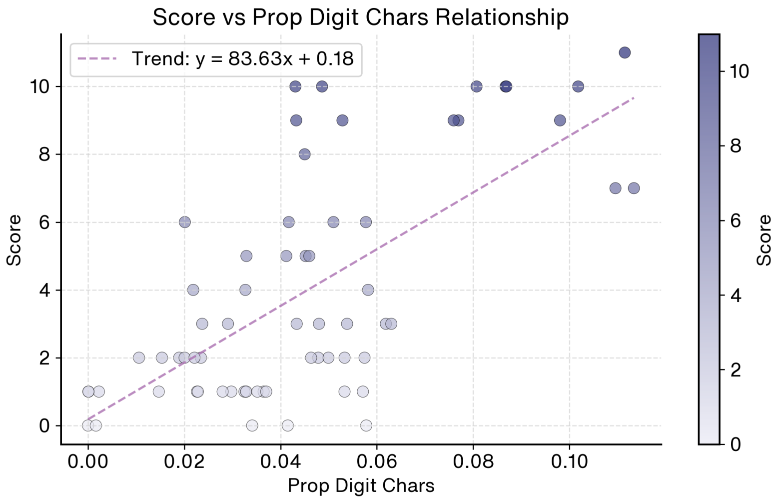

Figure 20: Relationship between the top predictive dynamic feature (Proportion of Digit Characters in Message) and score for the Pursuit task, with linear regression trend.

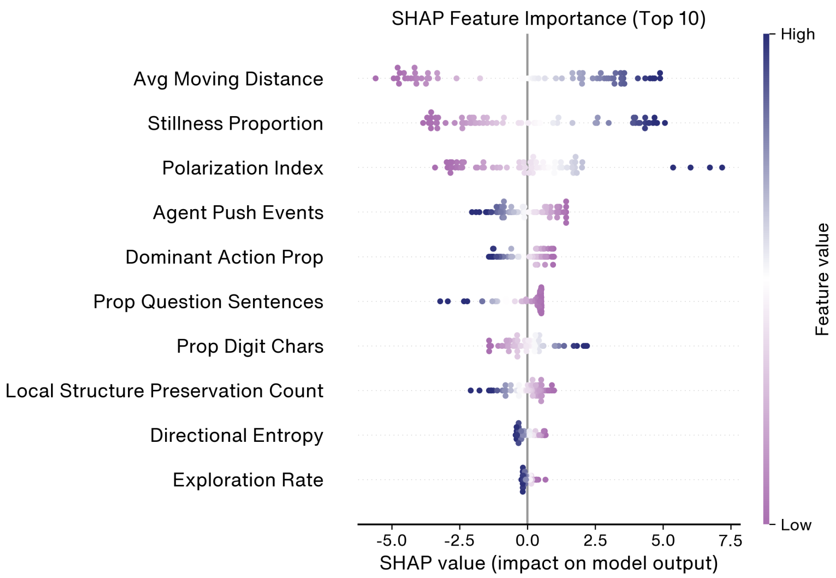

Figure 21: Feature importance from the linear regression model for the Pursuit task.

F.2 Synchronization Task Dynamics

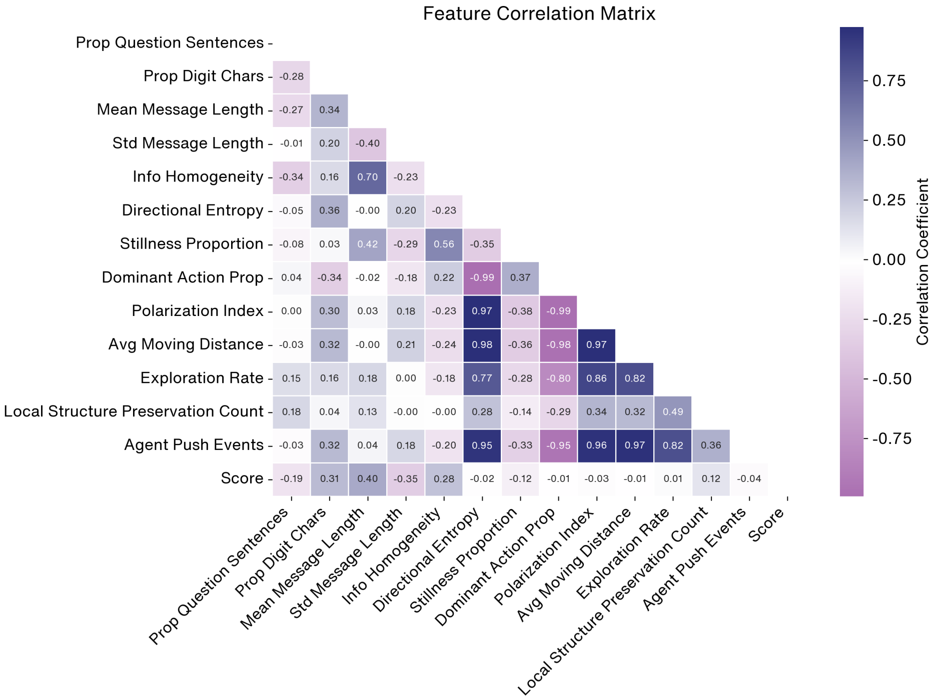

Figure 22: Feature correlation matrix for the Synchronization task.

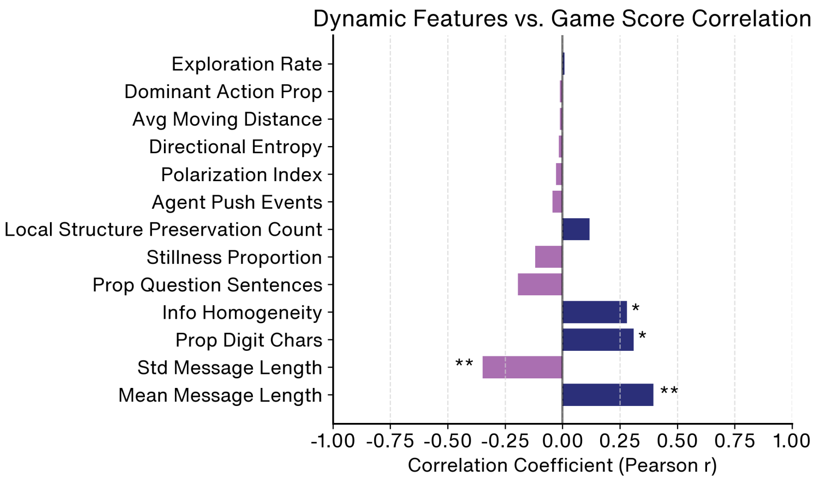

Figure 23: Correlation of dynamic features with score for the Synchronization task. Bars represent Pearson's r; * indicates $p < 0.05$,** indicates $p < 0.01$,*** indicates $p < 0.001$.

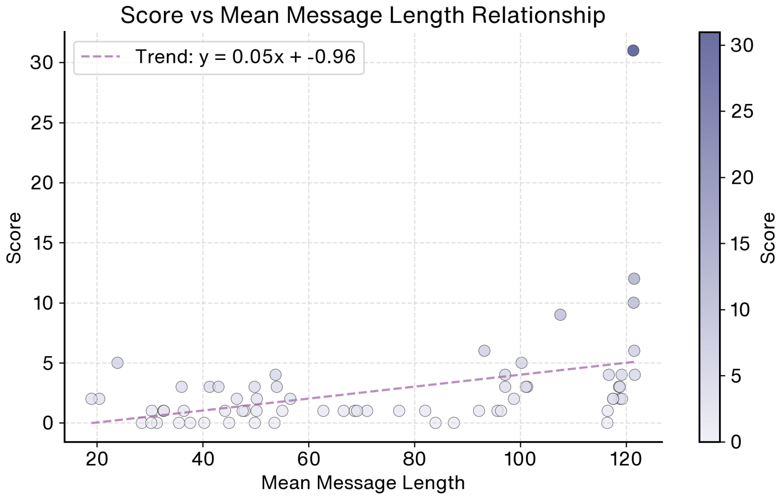

Figure 24: Relationship between the top predictive dynamic feature (Mean Message Length) and score for the Synchronization task, with linear regression trend.

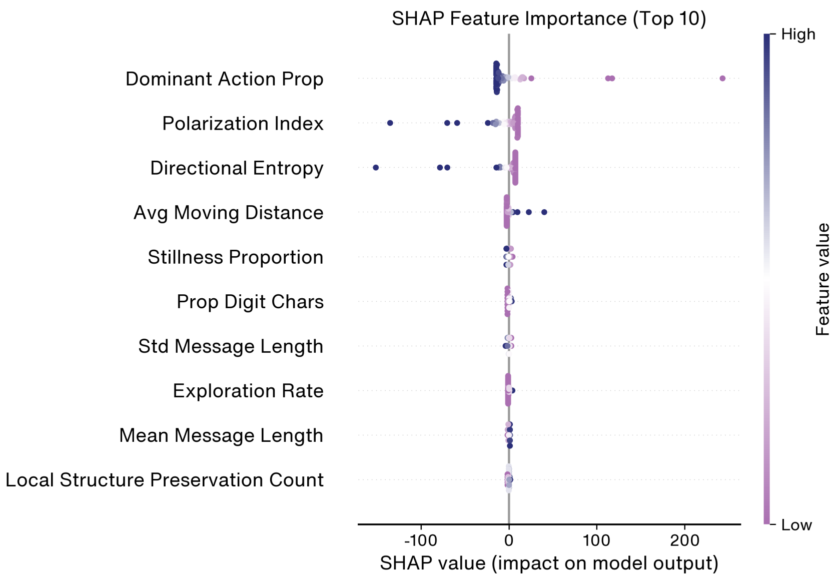

Figure 25: Feature importance from the linear regression model for the Synchronization task.

F.3 Foraging Task Dynamics

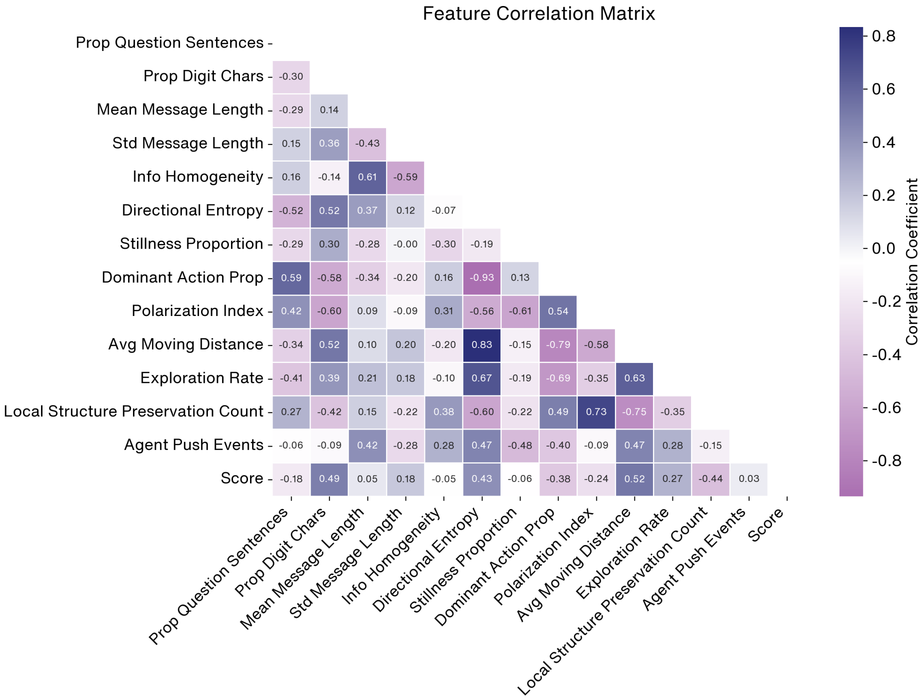

Figure 26: Feature correlation matrix for the Foraging task.

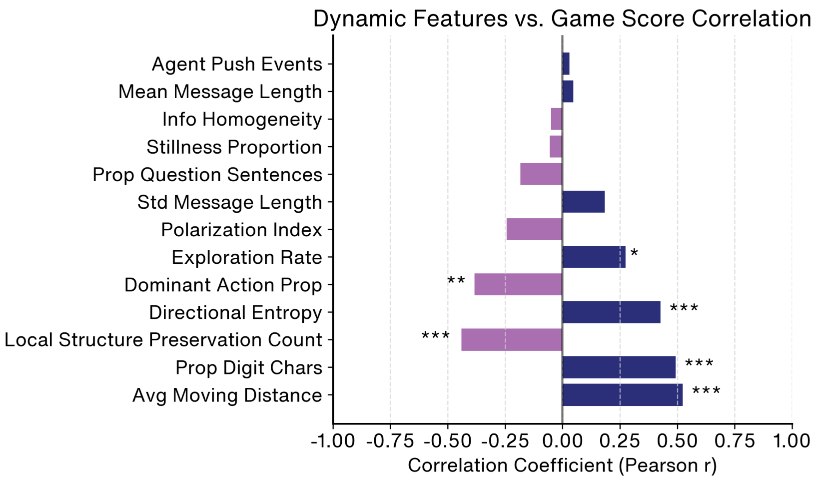

Figure 27: Correlation of dynamic features with score for the Foraging task. Bars represent Pearson's r; * indicates $p < 0.05$,** indicates $p < 0.01$,*** indicates $p < 0.001$.

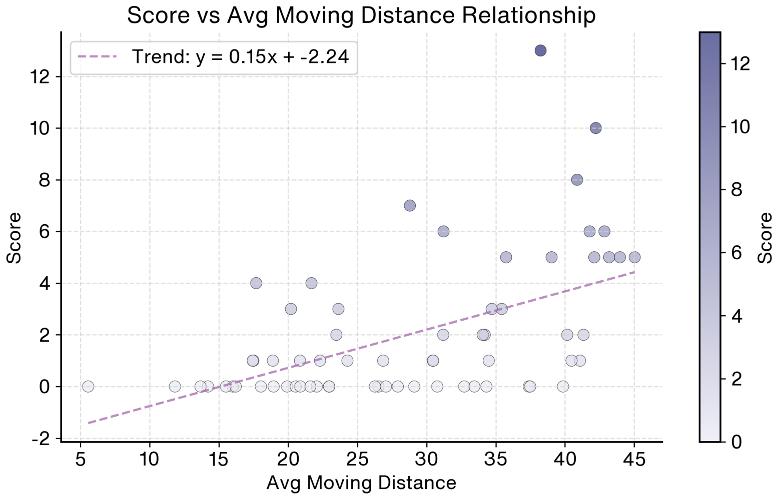

Figure 28: Relationship between the top predictive dynamic feature (Avg. Moving Distance) and score for the Foraging task, with linear regression trend.

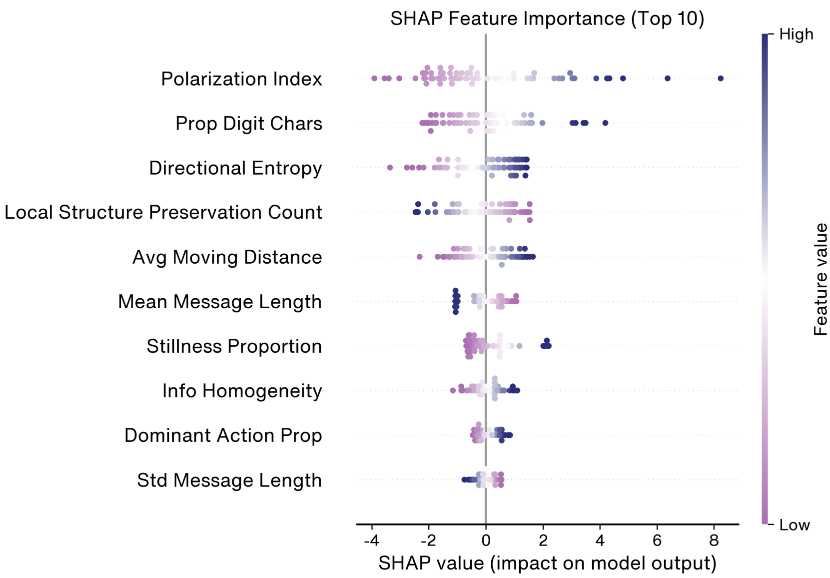

Figure 29: Feature importance from the linear regression model for the Foraging task.

F.4 Flocking Task Dynamics

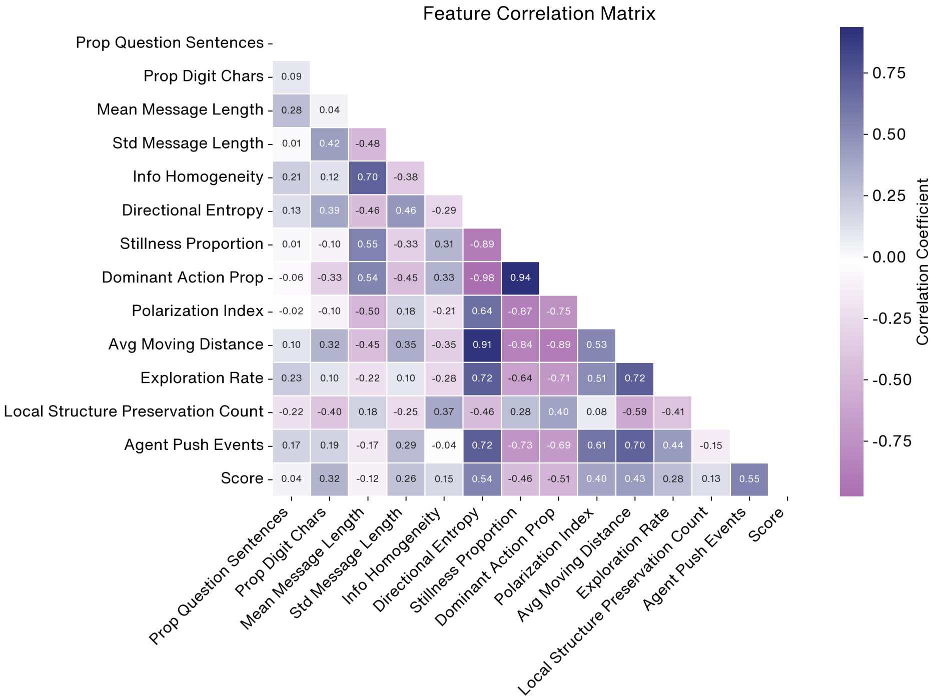

Figure 30: Feature correlation matrix for the Flocking task.

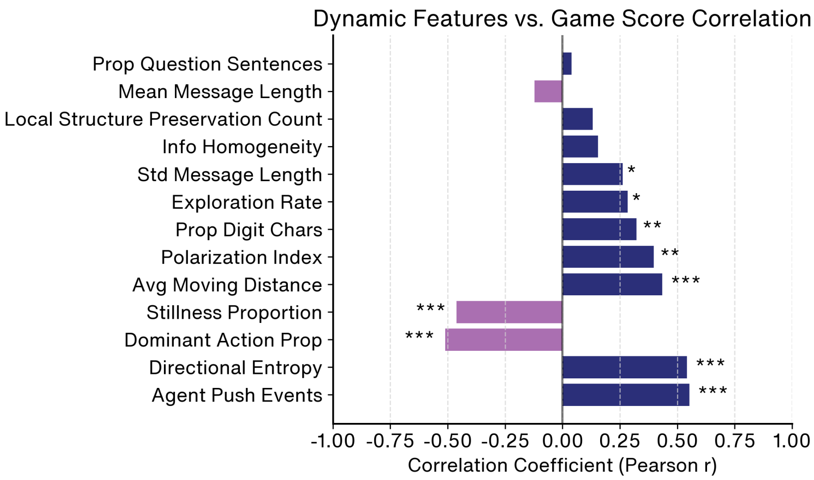

Figure 31: Correlation of dynamic features with score for the Flocking task. Bars represent Pearson's r; * indicates $p < 0.05$,** indicates $p < 0.01$,*** indicates $p < 0.001$.

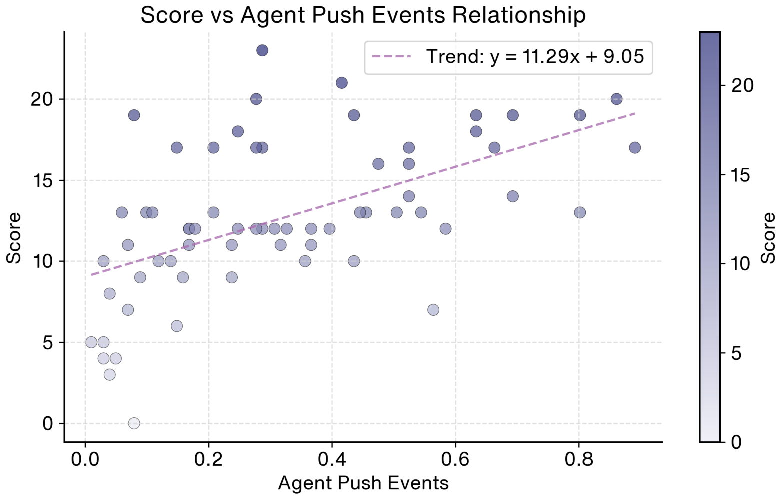

Figure 32: Relationship between the top predictive dynamic feature (Avg. Agent Push Events) and score for the Flocking task, with linear regression trend.

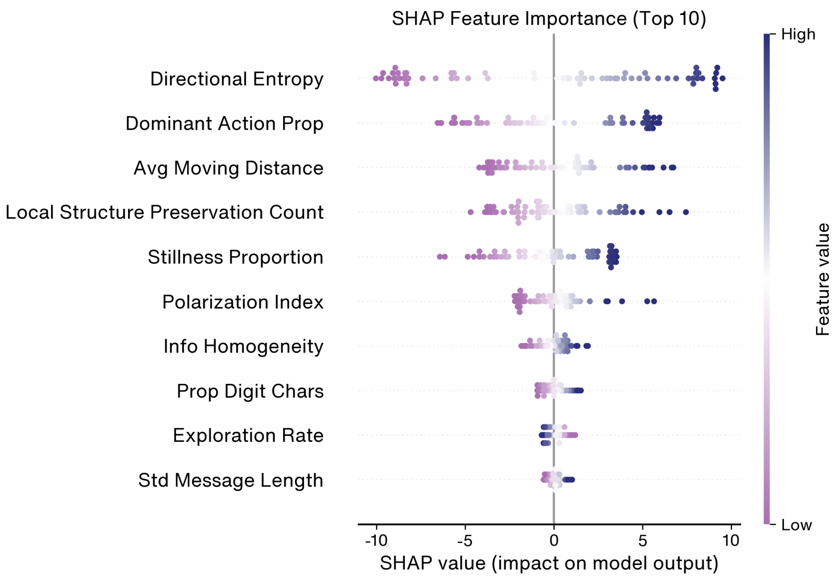

Figure 33: Feature importance from the linear regression model for the Flocking task.

F.5 Transport Task Dynamics

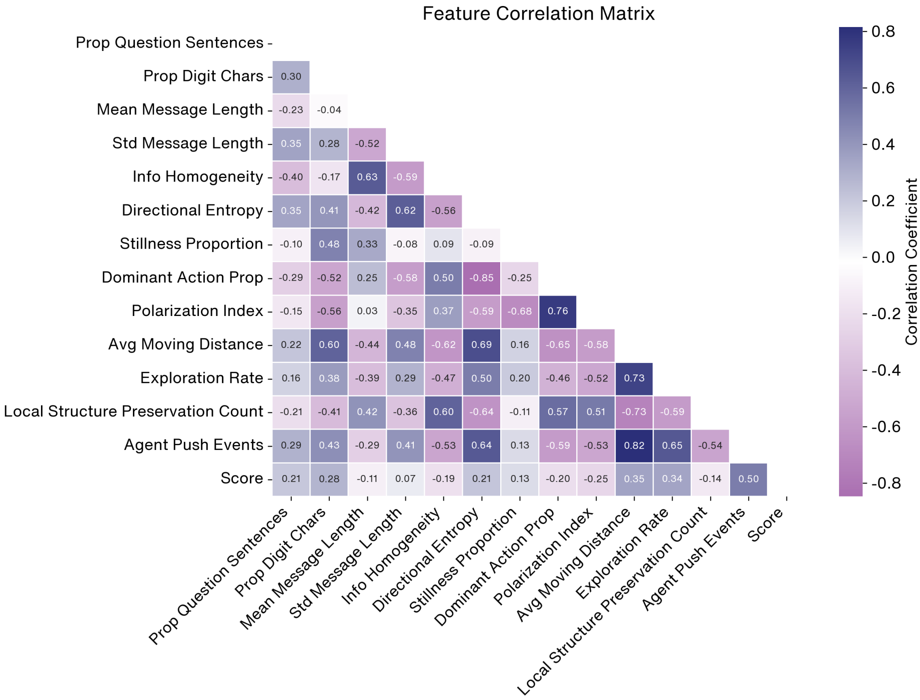

Figure 34: Feature correlation matrix for the Transport task.

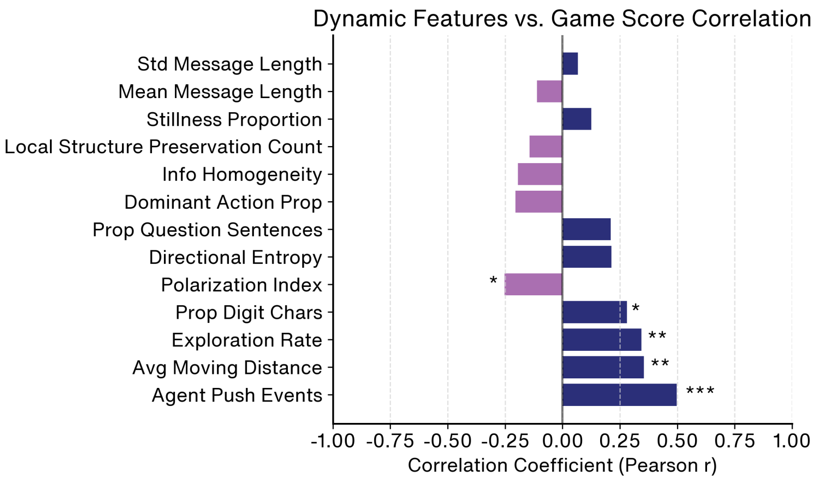

Figure 35: Correlation of dynamic features with score for the Transport task. Bars represent Pearson's r; * indicates $p < 0.05$,** indicates $p < 0.01$,*** indicates $p < 0.001$.

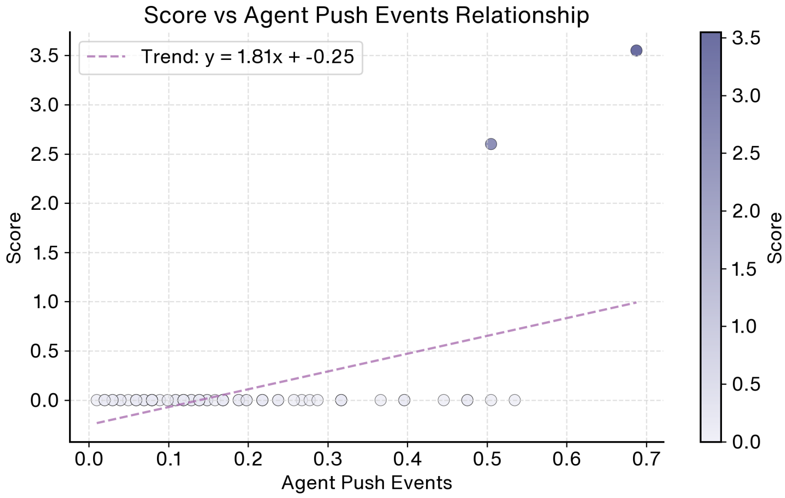

Figure 36: Relationship between the top predictive dynamic feature (Avg. Agent Push Events) and score for the Transport task, with linear regression trend.

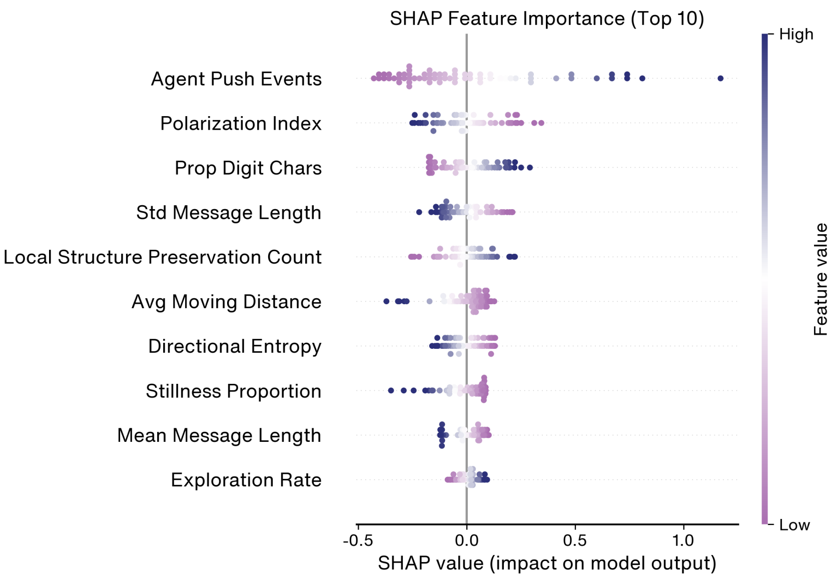

Figure 37: Feature importance from the linear regression model for the Transport task.

G Keyword Analysis

We performed keyword extraction on sampled message data to understand terminology used by different models across tasks. Messages were preprocessed (lowercasing, punctuation removal, English stopword removal using NLTK (Bird et al., 2009)) and frequent terms identified for each model-task combination. This analysis, visualized in Fig. 38, confirms agents' messages contained task-relevant vocabulary. The figure reveals keyword usage variations between LLM models performing the same task, suggesting model-specific communication styles. While relevant keywords indicate task understanding in communication, their frequency does not translate to coordination effectiveness, which appeared more linked to emergent physical dynamics and semantic consistency than keyword usage.

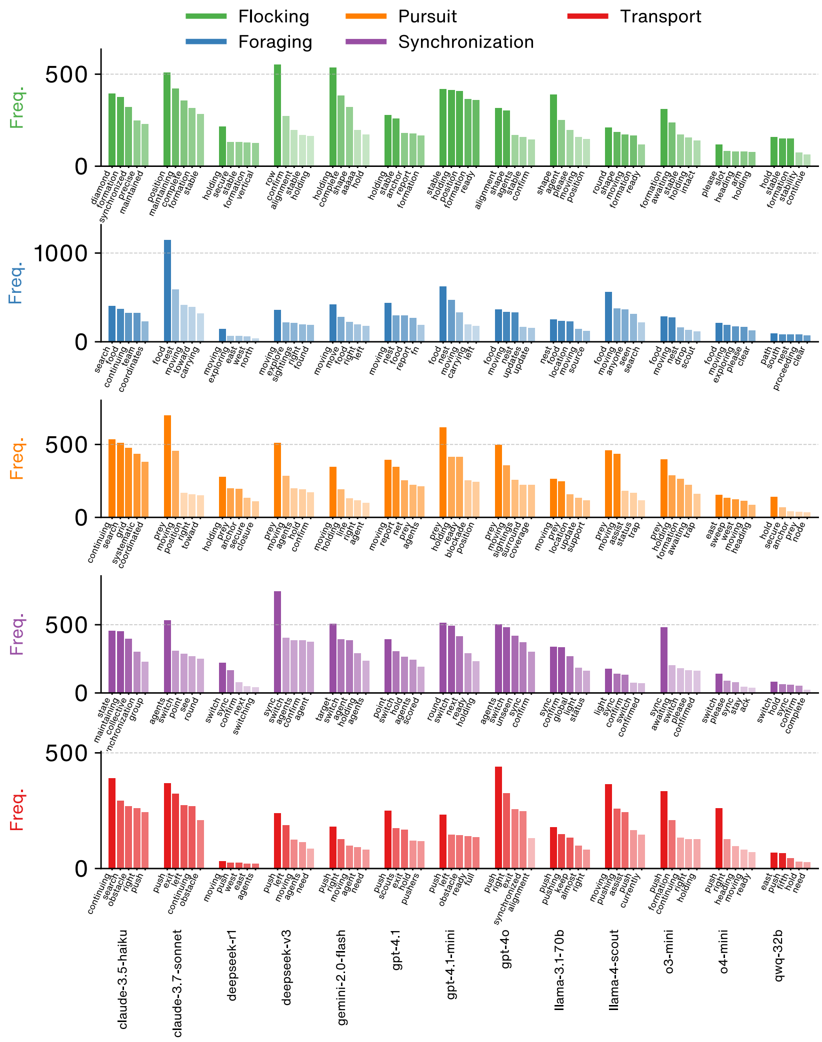

Figure 38: **Keyword Frequency Analysis from Agent Messages.** Frequency of the top keywords extracted from agent messages, grouped by LLM model and task. Message data was preprocessed before frequency counting. The visualization highlights task-specific terminology (e.g., 'push' in `Transport`, 'food' in `Foraging`) and reveals variations in keyword usage across different models for the same task.

H Parameter Sensitivity Analysis

This appendix provides a more detailed textual elaboration on how agent performance responds to changes in local perception range (k, the size of the square view) and group size (N, the number of agents). A summary of these findings and their implications, along with a visual representation of the key trends, is presented in Section 4.5 of the main text, specifically in Figure 6. Here, we expand on the specific observations and the nuances of the analysis that underpin those summarized conclusions.

Our systematic investigation involved varying $N \in \{8, 12, 16\}$ and $k \in \{3, 5, 7\}$ for key tasks, using the `gemini-2.0-flash` model. Performance was measured by task-specific scores averaged over multiple simulation runs, the results of which are visually summarized in Figure 6 in the main text.

The data, as shown in Figure 6, reveals several important trends. Expanding the field of view from $k = 3$ to $k = 5$ consistently improved outcomes across diverse tasks like `Pursuit`, `Synchronization`, `Foraging`, and `Flocking`. This suggests that a minimal level of environmental awareness is crucial for agents to effectively coordinate, likely enabling better anticipation and response to neighbors' actions. However, as also indicated by Figure 6 and discussed in the main text, further increasing the view to $k = 7$ yielded only marginal gains and was sometimes less effective than $k = 5$, particularly in the `Transport` task which demands precise collective alignment. This plateau, and in some cases like the `Transport` task a performance dip with $k = 7$ compared to $k = 5$, implies a potential trade-off. While more information can be beneficial, an overly broad view might lead to information overload, making it harder for the LLM agents to discern critical local cues from a larger, potentially noisier, perceptual field. This could dilute focus on immediately relevant neighbors or environmental features crucial for tightly coupled maneuvers, such as the precise alignment needed in `Transport`. The increased cognitive load of processing a larger input space without a corresponding improvement in strategic depth might thus be counterproductive in certain scenarios. The effectiveness of $k = 5$ in our main experiments (Section 4) likely reflects a more optimal balance between sufficient environmental awareness and manageable perceptual complexity for the current LLM architectures in these zero-shot settings.

The influence of group size (N) presented a more complex picture, strongly modulated by task demands, as also detailed visually in Figure 6. Predictably, performance in `Transport` improved with more agents ($N = 16$ vs $N = 8$), as the task fundamentally relies on accumulating sufficient physical force. Conversely, `Foraging` performance deteriorated as N increased, suggesting that larger groups introduced detrimental effects like congestion or interference near critical locations (nest '`N`', food '`F`'), outweighing any potential benefits. Intriguingly, `Pursuit` exhibited peak performance at an intermediate size ($N = 12$ compared to $N = 8$ and $N = 16$), hinting that while more agents can help initially encircle a target, too many may hinder coordinated containment through increased complexity and potential self-obstruction. `Flocking` remained relatively robust to changes in N within the tested range.

These detailed textual elaborations are intended to complement the summarized findings and the visual data presented in Section 4.5. The varied scaling behaviors highlight a core challenge for LLM-based swarms: managing the increased interaction density and potential for conflicting local decisions in larger groups without centralized control. The sensitivity to both k and N underscores that robust swarm intelligence requires strategies adaptable to varying information availability and group dynamics, motivating evaluation across diverse parametric settings.

I Action Attribution

To quantify the influence of different information sources, we trained a Random Forest classifier (Breiman, 2001) to predict agent actions based on embeddings of their local observation and received messages. We then used permutation importance to assess the relative influence of each feature type.

Across several tasks, messages received from other agents showed higher permutation importance than visual observations in predicting an agent's next action. This indicates that communicated information strongly influences immediate, local decisions.

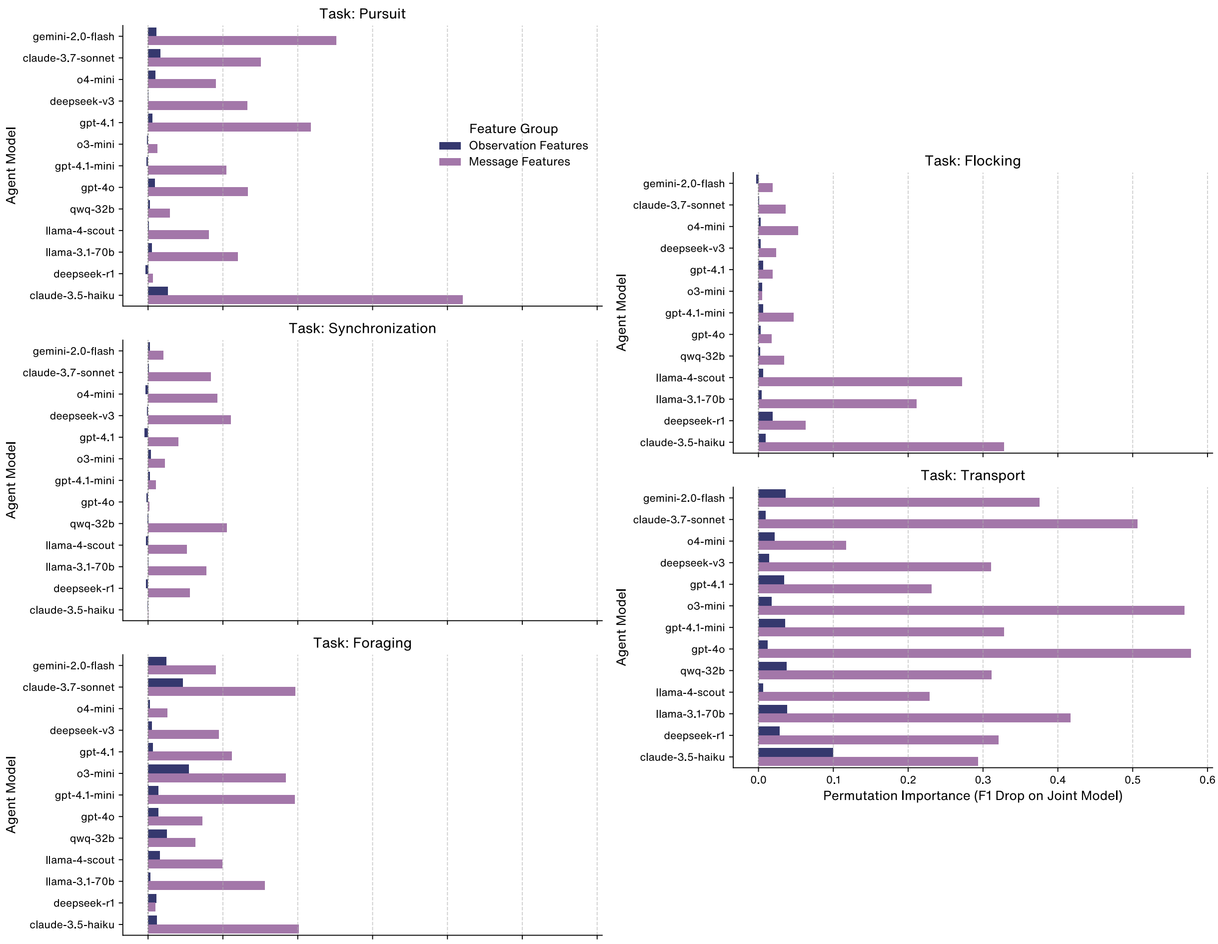

Figure 39: Permutation importance (F1 Drop) of Observation Features and Message Features for predicting agent actions, broken down by task and individual LLM agent model. The subplots detail these importance scores across the five SwarmBench tasks for each evaluated model.

J The SwarmBench Dataset

To support reproducibility and further research, we will release a comprehensive dataset encompassing all experimental runs detailed in this paper. The dataset is structured into experimental batches, with each batch containing a collection of JSON files that log the simulation parameters and detailed execution traces.

The primary components for each experimental batch are:

- **meta_log.json**: This central JSON file serves as an index for all individual simulation runs within a batch. It is a dictionary where each key is a unique run identifier (run_id). The corresponding value for each run_id is an object detailing the high-level configuration of that specific run, including parameters such as the Large Language Model employed (model), the number of participating agents (num_agents), and the maximum configured simulation rounds (max_round).
- **agent_log_<run_id>.json** files: For every run identified in meta_log.json, a corresponding agent log file is generated. This file stores a JSON array, with each element representing a detailed record for a single agent at a specific simulation round. These records capture the agent's local perception (view), the full prompt provided to its controlling LLM, the raw response from the LLM, and the subsequently parsed action (e.g., movement, task-specific command) and any message the agent chose to broadcast.
- **game_log_<run_id>.json** files: Complementing the agent logs, a game log file is also generated for each run. This file contains a JSON array, where each element chronicles the global state of the simulation environment at each round. This includes the complete 2D environment grid, the current score for the task, an array detailing the id, and global x, y coordinates for all agents, and a list of all messages that were broadcast by agents in the immediately preceding round and are thus available for perception in the current round.

This structured data will allow for in-depth analysis of agent behavior, communication patterns, and emergent group dynamics.

K Additional Experiments and Analyses

K.1 Quantitative Analysis of Failure Modes

Here, we try to categorize LLM's failures more formally in terms of swarm theory.

For example, we frame the "Movement Bias" as premature convergence (March, 1991), where an LLM's pattern-matching capabilities cause it to lock into a suboptimal strategy. This can be directly measured by calculating the action imbalance of the agent. We use the Gini coefficient as the measure.

"Information Silos" result from spontaneous strong community structure formation in the agent network, where agents create tightly-knit groups with sparse inter-group connections (Girvan and Newman, 2002), causing network fragmentation and preventing global consensus, which can be measured by the number of connected components (constructing a graph using the agent's visual range).

In terms of the "Traffic Jams", for example, we can directly modify the Separation Rule in the Boids model (Reynolds, 1987) to calculate the repulsive forces that each agent should experience in order to evaluate whether congestion phenomena exist.

Finally, we attribute the "Memory of a Goldfish" to the LLM's volatile context window, which prevents the formation of a persistent, environment-mediated memory, a function served by stigmergy in natural swarms (Grass, 1959). This failure mode may be relatively difficult to measure directly, but we believe it can be indirectly measured by analyzing the impact of increasing the LLM's context window (number of memory frames) on the overall score.

We analyzed three failure modes quantitatively. As shown in Table 4, we examined the tasks consistent with Figure 5 (i.e., Pursuit, Synchronization, and Foraging) to explore how these metrics correlate with scores. Interestingly, all three metrics showed negative correlations with the final scores, which aligns with our expectations.

Table 4: Quantitative Analysis of Failure Modes

Task	**Metric**	r	p-value	**Significance**
`Pursuit`	Action Direction Imbalance	-0.668	0.000	$***$
`Synchronization`	Number of Connected Components	-0.185	0.140	–
`Foraging`	Separation Force	-0.309	0.012	$*$

K.2 Analysis of Communication Protocol Convergence

In our supplementary videos (e.g., `flocking_o4-mini_best.gif`), agents' messages often start as varied and verbose, but over time, they converge to a shorter, more structured format (e.g., "`Taking slot (2,4) TL=(2,3)`").

To quantify this phenomenon and its connection to task success, we introduced two new metrics: (a) the Increase in Information Homogeneity (semantic similarity) and (b) the Increase in Edit Distance Consistency (syntactic similarity) over the course of each game. We then correlated these metrics with the final task score (see Table 5 and Table 6, $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$)

Table 5: Correlation of Protocol Convergence with Final Score (by Task)

Task	**Homogeneity**		**Edit Consistency**	
	r	p-**value**	r	p-**value**
`Flocking`	0.382	0.002**	0.458	0.000***
`Foraging`	-0.102	0.419	-0.027	0.832
`Pursuit`	-0.447	0.000***	-0.439	0.000***
`Sync.`	-0.134	0.286	-0.159	0.205
`Transport`	0.056	0.660	-0.115	0.361

Table 6: Correlation of Protocol Convergence with Final Score (by Model)

Model	**Homogeneity**		**Edit Consistency**	
	r	p-**value**	r	p-**value**
`gemini-2.0-flash`	0.241	0.246	-0.061	0.774
`o4-mini`	0.305	0.138	0.372	0.067
`claude-3.7-sonnet`	-0.147	0.483	-0.141	0.500
`gpt-4.1`	-0.044	0.835	-0.031	0.884
`deepseek-v3`	-0.307	0.136	-0.255	0.218
`llama-3.1-70B`	-0.499	0.011*	-0.267	0.197
`gpt-4o`	0.031	0.884	-0.231	0.267
`llama-4-scout`	-0.016	0.941	0.094	0.656
`deepseek-r1`	-0.144	0.493	-0.118	0.574
`qwq-32B`	-0.135	0.520	-0.139	0.508
`o3-mini`	-0.528	0.007**	-0.450	0.024*
`gpt-4.1-mini`	-0.338	0.098	-0.163	0.436
`claude-3.5-haiku`	-0.005	0.980	-0.058	0.783

First, contrary to intuition, a stronger convergence of the communication protocol often correlates with a lower final score. This suggests that for complex, dynamic scenarios, maintaining communicative diversity and richness may be more beneficial than prematurely locking into a rigid, simplistic protocol.

The `Flocking` task, however, is a notable exception with a significant positive correlation. This distinction is illuminating: `Flocking` is a task that benefits from converging on a simple, efficient protocol for sharing positional data without extraneous information. In contrast, more dynamic tasks may require richer communication to adapt. This provides a much deeper, data-driven reason for failure modes: a swarm's failure to converge on a protocol is not always a deficiency; in some cases, a rigid convergence is itself the strategy that fails.