Bridging Formal Strategic Reasoning and Video Games*

Salvatore Romano[1]

[1]University of Naples Federico II

Abstract

The video game industry continuously demands highly sophisticated, realistic, and autonomous Non-Player Characters (NPCs). While commercial game engines have historically approached AI from a strictly technical perspective, the core of video game decision-making architectures remains largely distanced from formal Game Theory. Recently, formal methods have emerged to bridge this gap, recognizing games as Multi-Agent Systems. In our foundational work, we introduced a novel framework utilizing Alternating-time Temporal Logic (ATL) directly within a modern game engine. This paper reports the salient points of that offline-to-runtime architecture, provides a comparative analysis with recent formal developments, such as logics for imperfect information and natural strategic ability, and discusses future extensions, including Probabilistic ATL (PATL) and Neuro-Symbolic integration, to enhance the strategic reasoning of video game AI.

Keywords

Alternating-Time Temporal Logic, Multi-Agent Systems, Video Games

1. Introduction

The video game industry demands increasingly rich, immersive virtual worlds where Non-Player Characters (NPCs) display strategic, coherent, and human-like behaviors [2, 3]. Despite this demand, game AI is predominantly approached from a software-engineering perspective rather than as a formal application of Game Theory. Commercial decision-making relies heavily on reactive architectures like Behavior Trees or opaque machine learning models [4, 5]. While effective, these methods lack transparency and offer no formal guarantees regarding an agent's strategic abilities in complex environments.

Conversely, formalizing video games as *Multi-Agent Systems* (MAS) enables the use of strategic reasoning logics to rigorously verify agent coalitions [6, 7]. Alternating-time Temporal Logic (ATL) provides a foundational framework for this purpose [8]. Subsequent research has significantly broadened this expressive power, introducing Strategy Logic and imperfect information extensions [9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]. These critical developments move the field beyond idealized perfect information, better reflecting the uncertainty and partial observability typical of real games. Building on this tradition, our foundational work introduced one of the first attempts to embed formal strategic reasoning directly into a commercial engine, integrating an ATL model checker natively into Unreal Engine 5 [1]. That work demonstrated that offline abstraction and precomputation can successfully support verifiable runtime decision-making without violating real-time constraints. In this paper, we report the salient points of that framework and analyze how recent advances, such as imperfect information and natural strategies, can bridge the gap between symbolic verification and modern game AI, moving toward virtual agents that are explainable, verifiable, and strategically robust.

2. Related Works

The evolution of Game AI and Multi-Agent Systems The development of game AI has historically relied on rigid structures like Finite State Machines, Behavior Trees [4], or Dynamic Scripting [21]. As

8th International Workshop on Artificial Intelligence and fOrmal VERification, Logic, Automata, and sYnthesis. OVERLAY 2026 @ FLoC, July 18–19, Lisbon, Portugal

* Revisiting [1], this short paper streamlines the main results, updates the related work, and broadens the discussion.

salvatore.romano2@unina.it (S. Romano)
salvo-romano.github.io (S. Romano)
0009-0007-4640-8705 (S. Romano)

© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073
published 2026-11-09

CEUR-WS.org/Vol-4324/paper7.pdf

these methods struggle in unscripted scenarios, Deep Reinforcement Learning (DRL) and Behavioral Cloning have revolutionized the field, achieving superhuman performance across various competitive genres [22, 5, 23, 24]. Furthermore, recognizing games as Multi-Agent Systems (MAS), reinforcement learning has been utilized to optimize multi-agent coordination [25, 26, 27]. However, neural and RL-based controllers lack the reasoning required to offer formal guarantees that an NPC will strictly adhere to specific narrative boundaries or critical operational constraints, such as ensuring an allied agent never irreversibly blocks a player's escape route.

Formal Verification in Gaming Environments To bridge the gap between pure adaptability and strategic guarantees, formal logic has been applied to MAS. However, traditional model checkers, such as NuSMV [28] and MCMAS [29], are predominantly designed for offline execution. Recent frameworks like VITAMIN [30] offer compositional verification to mitigate the state explosion problem, but real-time in-game applications remain scarce. Previous studies have successfully verified MAS properties, but they were largely constrained to offline external tools [31], simplified engines or arcade games like Tetris [32, 33]. Some research has verified Unreal Engine Blueprints using formal model checking, but strictly through offline pre-computation [34, 35]. Since continuous 3D environments suffer from the state explosion problem, performing formal verification directly within the game loop is a complex frontier.

Imperfect Information and Natural Strategies Standard ATL assumes perfect global knowledge [8], clashing with the partial observability (e.g., "fog of war") essential to realistic video games. Fortunately, broader formal methods have long addressed this, yielding Strategy Logic and ATL with imperfect information [12], pushdown module checking [36, 37], and specialized model checkers like STV [38]. To further prevent agents from acting on unobservable global variations, *Natural Strategic Ability* limits actions strictly to those deducible from local observations [39]. Additionally, intuitionistic variants of strategic logics [40, 41] prioritize the actual synthesizability of a strategy over its mere logical existence. Integrating these advanced formalisms represents the necessary next step for applying rigorous Game Theory to highly uncertain, partially observable virtual environments.

3. Formal Background

This section reviews the formal syntax and semantics of ATL, whose formulas are evaluated over Concurrent Game Structures (CGS), labeled transition graphs that formally capture the agents' interactive behaviors.

Definition 1 (Concurrent Game Structure). *A CGS consists of a tuple $S = \langle Agt, Q, Act, \Pi, \pi, d, \delta \rangle$ where: (i) $Agt \geq 1$ is a natural number indicating the number of players. (ii) Q is a finite set of states. (iii) Act is a finite set of actions. (iv) Π is finite set of propositions. (v) $\pi : Q \to 2^{\Pi}$ is a labelling function such that for each state q, $\pi(q) \subseteq \Pi$ is a set of true propositions at q. (vi) $d_a(q) \geq 1$ indicates for each player $a \in Agt$ and each state $q \in Q$, the number of actions available in q. A move vector at q consists of a tuple $\langle j_1, ..., j_{Agt} \rangle$ such that $1 \leq j_a \leq d_a(q)$ for each player a. (vii) δ is a transition function that, for each state q and move vector, returns a state resulting from q if each player a chooses the move in the vector.*

Definition 2 (ATL Syntax). *An ATL formula can be expressed using the syntax:*

$$\varphi := p \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \langle\langle A \rangle\rangle X\varphi \mid \langle\langle A \rangle\rangle G\varphi \mid \langle\langle A \rangle\rangle \varphi_1 U \varphi_2$$

where p is an atomic proposition, and X, G and U are temporal operators standing for "next", "globally" and "until". The $\langle\langle A \rangle\rangle$ operator represents the capability of a coalition $A \subseteq Agt$ of enforcing states where φ holds true.

Definition 3 (ATL Semantics). *A **path** (or computation) is an infinite sequence of states $\lambda = q_0 q_1 q_2 \dots$ such that for every $i \geq 0$, there exists a valid move vector transitioning the system from q_i to q_{i+1}. We denote the i-th state of the path λ as $\lambda[i]$.*

*A **strategy** for a player $a \in Agt$ is a function $f_a : Q^+ \to Act$ that maps every finite non-empty sequence of states (representing the history of the game) to a valid action available in the last state. For a coalition $A \subseteq Agt$, a collective strategy $F_A = \{f_a \mid a \in A\}$ is a set comprising one strategy for each agent belonging to A.*

*Given a state $q \in Q$ and a collective strategy F_A for a coalition A, the **outcome** denoted as $out(q, F_A)$ is the set of all possible paths λ such that: (i) $\lambda[0] = q$ (ii) For all $i \geq 0$, the transition from $\lambda[i]$ to $\lambda[i+1]$ is the result of the agents in A executing the actions dictated by $F_A(\lambda[0] \dots \lambda[i])$, while the opposing agents in $Agt \setminus A$ choose any arbitrary available actions.*

Relying on these concepts, the full semantics of the strategic operator $\langle\langle A \rangle\rangle$ can be strictly defined. For any state $q \in Q$ and a temporal path formula φ (which includes operators like X, G, or U):

$$q \models \langle\langle A \rangle\rangle \varphi \iff \exists F_A \text{ such that } \forall \lambda \in out(q, F_A), \lambda \models \varphi$$

This mathematical foundation guarantees that if the model checker verifies the formula as true, the coalition A has the absolute power to enforce the condition φ regardless of any possible adversarial counter-strategy.

4. The Core Framework: Offline Verification and Real-Time Execution

Our foundational framework [1] successfully established a pipeline to integrate formal strategic verification within the Unreal Engine 5 game loop. The primary challenge in merging formal logic with commercial physics engines is the disparity in execution time: while game engines must resolve logic within milliseconds to maintain a 60 frames-per-second (FPS) target, exhaustive temporal logic synthesis can take seconds or minutes. To bypass this immense computational bottleneck, our architecture strictly decouples the heavy reasoning phase from the real-time execution phase.

Environment Abstraction To evaluate ATL formulas, the continuous 3D game environment must first be manually abstracted into a discrete CGS. In our experimental setup, a labyrinth escape scenario, two opposing forces interact: the Chaser Team (composed of a human player and an allied NPC) and the Escapist Team (an enemy NPC). The environment features dynamic elements, such as two operable escape doors. Rather than processing absolute 3D coordinates, actions in this CGS represent high-level strategic goals (e.g., navigating to a specific quadrant, opening a door, or intercepting the enemy). The environment was abstracted into 8 discrete Boolean variables (such as $Enemy_{prox}$, indicating the enemy is in close proximity to an exit, and $D1_{open}$, indicating the state of a door). This specific abstraction resulted in exactly 160 distinct global states. By mapping all possible combinations of agent actions, we generated a comprehensive transition matrix comprising 14,400 state transitions.

Offline Model Checking Crucially, the evaluation of the CGS is executed entirely offline. A custom ATL model checker built into Unreal Engine 5 explores the discrete state space and pre-computes the valid collective strategies F_A for the agent coalitions against all possible adversarial counter-moves. Instead of running complex logical solvers in C++, the pre-computed strategies are stored as highly optimized Data Tables within the engine's Blueprint visual scripting system. Each distinct combination of agent actions is encoded as an integer value to minimize memory overhead. This offline approach completely shields the active game engine from the exponential time complexity associated with formal model checking, reducing the in-game computational cost to a simple $O(1)$ memory lookup.

Runtime Decision Making via Prioritized Queues While the verification is offline, the gameplay is highly dynamic. Human players are inherently unpredictable, and their choices can instantaneously alter the game state, potentially invalidating an ongoing strategy. To manage this epistemic uncertainty without triggering a computationally expensive runtime re-evaluation, the framework employs a Prioritized Formula Queue to extract best-effort strategies [42]. The queue holds a list of propositional formulas representing desirable states for the team, strictly ordered by their strategic priority (e.g., φ_1: "Ensure the enemy never escapes", φ_2: "Close all doors"). During each game loop iteration, the engine reads the live 3D variables, translates them into the discrete Boolean format, and identifies the current state q_{curr}. The system then utilizes the current state q_{curr} and the highest priority formula φ_1 as joint search keys within the pre-computed Data Table to instantaneously extract the optimal associated action. While the ATL strategic operator $\langle\langle A \rangle\rangle$ dictates the absolute power to enforce a condition, unpredictable human interference can force the game into a critical state where the primary goal becomes mathematically unsatisfiable. The best-effort approach resolves this by intentionally failing the lookup for φ_1, instantaneously aborting the invalidated strategy, and degrading gracefully to the verifiable fallback formula φ_2. Because the engine merely queries pre-calculated results, it seamlessly adapts to chaotic player behavior on the fly, generating best-effort responses with near-zero latency. Recent theoretical optimizations by our research group have further expanded this concept, demonstrating that shifting the focus to Computation Tree Logic (CTL) can optimize the synthesis algorithms, paving the way for even faster and more scalable in-game performance [43, 44].

5. Discussion and Future Developments

While our framework successfully integrates ATL into a commercial engine, its reliance on perfect information and rigid offline matrices limits its scalability for AAA development. Enhancing this symbolic approach requires the following critical evolutions:

Imperfect Information and Natural Strategies In realistic scenarios, NPCs lack omniscient map knowledge. Upgrading the framework to support ATL with imperfect information is essential. By implementing *Natural Strategic Ability* [39], NPCs will synthesize strategies based solely on local observations (e.g., ray-casts). This ensures formal guarantees hold even with hidden enemies, yielding fairer and more realistic AI behavior.

Probabilistic Logics and Fuzzy Systems Games rely heavily on stochastic events that standard ATL cannot model. Extending the engine to support Probabilistic ATL (PATL) [45] and Probabilistic Timed ATL [46] will allow synthesizing strategies with guaranteed probability thresholds, natively embracing in-game randomness. Additionally, incorporating Fuzzy Logic could enable the Prioritized Formula Queue to handle degrees of truth, smoothing transitions between aggressive and defensive strategies.

Towards Neuro-Symbolic Integration A fundamental limitation of purely symbolic methods is the unscalable manual abstraction of continuous 3D spaces into discrete states. A promising future direction involves exploring Neuro-Symbolic architectures [47]. Recent theoretical advances have demonstrated the efficacy of neuro-symbolic game models that integrate autoencoders, capable of extracting latent structural representations from complex data, with dynamic formal logic modules [48]. Adapting this architecture to video games would allow us to delegate spatial perception to a neural network. The network would dynamically extract discrete semantic states from the continuous game environment, serving as direct inputs for an optimized runtime logic solver. This pipeline could provide a pathway to bridge the fluid adaptability of data-driven learning with the rigorous, verifiable guarantees of Game Theory [49].

Acknowledgments

This work has been partially supported by the following projects: COSMI ALGORITMICI (E63C26000840001), GAMEL (F53C25001420001), SHINE-QC (B69J25000260005). I would like to sincerely thank my co-authors of [1], Aniello Murano and Marco Aruta, for their invaluable collaboration and the foundational research that made this framework possible.

Declaration on Generative AI

During the preparation of this work, the author(s) used Gemini in order to: Grammar and spelling check. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

Bibliography

[1] M. Aruta, A. Murano, S. Romano, Atl for dynamic gaming environments, in: European Conference on Multi-Agent Systems, Springer, 2024, pp. 129–137.
[2] E. Goh, O. Al-Tabbaa, Z. Khan, Unravelling the complexity of the video game industry: An integrative framework and future research directions, Telematics and informatics reports 12 (2023) 100100.
[3] Y.-H. Chang, R. Maheswaran, T. Levinboim, V. Rajan, Learning and evaluating human-like npc behaviors in dynamic games, in: AAAI 2011, volume 7, 2011, pp. 8–13.
[4] K. Zhang, C. Xu, A evolutionary behavior tree ai for neural mmo challenge, in: AIAM 2022, 2022, pp. 327–332.
[5] G. Lample, D. S. Chaplot, Playing FPS games with deep reinforcement learning, CoRR abs/1609.05521 (2016).
[6] M. Wooldridge, An introduction to multiagent systems, John Wiley & Sons, 2009.
[7] B. Parasumanna Gokulan, D. Srinivasan, An Introduction to Multi-Agent Systems, volume 310, 2010, pp. 1–27. doi:10.1007/978-3-642-14435-6_1.
[8] R. Alur, T. A. Henzinger, O. Kupferman, Alternating-time temporal logic, Journal of the ACM (JACM) 49 (2002) 672–713. doi:10.1145/585265.585270.
[9] F. Mogavero, A. Murano, G. Perelli, M. Y. Vardi, Reasoning about strategies: On the model-checking problem, ACM Trans. Comput. Log. 15 (2014) 34:1–34:47.
[10] F. Mogavero, A. Murano, G. Perelli, M. Y. Vardi, Reasoning about strategies: on the satisfiability problem, Log. Methods Comput. Sci. 13 (2017).
[11] F. Mogavero, A. Murano, L. Sauro, On the boundary of behavioral strategies, in: 2013 28th Annual ACM/IEEE Symposium on Logic in Computer Science, IEEE, 2013, pp. 263–272.
[12] R. Berthon, B. Maubert, A. Murano, S. Rubin, M. Y. Vardi, Strategy logic with imperfect information, ACM Transactions on Computational Logic (TOCL) 22 (2021) 1–51.
[13] W. Jamroga, V. Malvone, A. Murano, Natural strategic ability, Artif. Intell. 277 (2019).
[14] F. Belardinelli, W. Jamroga, M. Mittelmann, A. Murano, Verification of stochastic multi-agent systems with forgetful strategies, in: AAMAS 2024, 2024, pp. 160–169.
[15] A. Murano, D. Neider, M. Zimmermann, Robust alternating-time temporal logic, in: JELIA 2023, LNCS 14281, Springer, 2023, pp. 796–813.
[16] W. Jamroga, W. Van Der Hoek, Agents that know how to play, Fundamenta Informaticae 63 (2004) 185–219.
[17] M. Aruta, F. Improta, V. Malvone, A. Murano, When natural strategies meet fuzziness and resource-bounded actions, in: AAAI 2026, 2026, pp. 29305–29313.
[18] F. Laroussinie, N. Markey, Augmenting ATL with strategy contexts, Inf. Comput. 245 (2015) 98–123.
[19] K. Chatterjee, T. A. Henzinger, N. Piterman, Strategy logic, Inf. Comput. 208 (2010) 677–693.
[20] W. Jamroga, T. Ågotnes, Constructive knowledge: what agents can achieve under imperfect information, Journal of Applied Non-Classical Logics 17 (2007) 423–475.
[21] P. Spronck, M. Ponsen, I. Sprinkhuizen-Kuyper, E. Postma, Adaptive game ai with dynamic scripting, Machine Learning 63 (2006) 217–248.
[22] T. Pearce, J. Zhu, Counter-strike deathmatch with large-scale behavioural cloning, 2021. arXiv:2104.04258.
[23] N. A. Barriga, M. Stanescu, F. Besoain, M. Buro, Improving rts game ai by supervised policy learning, tactical search, and deep reinforcement learning, IEEE Computational Intelligence Magazine 14 (2019) 8–18. doi:10.1109/MCI.2019.2919363.

[24] D.-W. Kim, S. Park, S.-i. Yang, Mastering fighting game using deep reinforcement learning with self-play, in: 2020 IEEE Conference on Games (CoG), 2020, pp. 576–583. doi:10.1109/CoG47356.2020.9231639.
[25] S. Kapetanakis, D. Kudenko, Reinforcement learning of coordination in heterogeneous cooperative multi-agent systems, in: Symposium on Adaptive Agents and Multi-agent Systems, Springer, 2003, pp. 119–131.
[26] C. Yu, A. Velu, E. Vinitsky, J. Gao, Y. Wang, A. Bayen, Y. Wu, The surprising effectiveness of ppo in cooperative multi-agent games, Advances in Neural Information Processing Systems 35 (2022) 24611–24624.
[27] R. Moghadam, H. Modares, Resilient adaptive optimal control of distributed multi-agent systems using reinforcement learning, IET Control Theory & Applications 12 (2018) 2165–2174.
[28] A. Cimatti, E. Clarke, F. Giunchiglia, M. Roveri, Nusmv: a new symbolic model checker, STTT 2 (2000). doi:10.1007/s100090050046.
[29] A. Lomuscio, H. Qu, F. Raimondi, Mcmas: an open-source model checker for the verification of multi-agent systems, International Journal on Software Tools for Technology Transfer 19 (2017) 9–30.
[30] A. Ferrando, V. Malvone, Vitamin: A compositional framework for model checking of multi-agent systems, in: ICAART 2025, Volume 1, INSTICC, SciTePress, 2025, pp. 648–655. doi:10.5220/0013349600003890.
[31] R. Rezin, I. Afanasyev, M. Mazzara, V. Rivera, Model checking in multiplayer games development, in: AINA 2018, 2018, pp. 826–833.
[32] M. Chover, C. Marín, C. Rebollo, I. Remolar, A game engine designed to simplify 2d video game development, Multimedia Tools and Applications 79 (2020) 12307–12328.
[33] C. Marín-Lora, M. Chover, J. M. Sotoca, A multi-agent specification for the tetris game, in: Distributed Computing and Artificial Intelligence, Springer, 2022, pp. 169–178.
[34] N. Igawa, T. Yokogawa, M. Takahashi, K. Arimoto, Model checking of visual scripts created by ue4 blueprints, in: IIAI-AAI 2020, 2020, pp. 512–515. doi:10.1109/IIAI-AAI50415.2020.00107.
[35] K. Wayama, T. Yokogawa, S. Amasaki, H. Aman, K. Arimoto, Verifying game logic in unreal engine 5 blueprint visual scripting system using model checking, in: ASE 2023, ACM, 2023.
[36] B. Aminof, A. Legay, A. Murano, O. Serre, M. Y. Vardi, Pushdown module checking with imperfect information, Information and Computation 223 (2013) 1–17.
[37] A. Murano, G. Perelli, Pushdown multi-agent system verification, in: IJCAI 2015, AAAI Press, 2015, pp. 1090–1097.
[38] D. Kurpiewski, W. Jamroga, M. L. Knapik, Stv: model checking for strategies under imperfect information, in: AAMAS 2019, IFAAMAS, 2019.
[39] W. Jamroga, V. Malvone, A. Murano, Natural strategic ability, Artificial Intelligence 277 (2019) 103170.
[40] A. Capone, D. Catta, A. Murano, et al., An intuitionistic version of computation tree logic, in: Proc. of 22nd EUMAS 2025, LNCS 16259, Springer, 2026.
[41] L. Bozzelli, A. Capone, D. Catta, A. Murano, An intuitionistic version of alternating-time temporal logic, in: Proc. of KR'25, volume 22, 2025, pp. 185–195.
[42] B. Aminof, G. De Giacomo, A. Lomuscio, A. Murano, S. Rubin, Synthesizing Best-effort Strategies under Multiple Environment Specifications, in: KR 2021, 2021, pp. 42–51.
[43] M. Aruta, V. Malvone, A. Murano, V. Pio Palma, S. Romano, Findme: A prototype videogame ai based on ctl with an optimized synthesis algorithm, in: AAMAS 2025, 2025, pp. 2997–2999.
[44] V. Malvone, A. Murano, V. P. Palma, S. Romano, Findme reforged: Temporal logic AI for richer videogame scenarios, in: Proc. of ICAART'26, Vol. 3, SciTePress, 2026, pp. 2413–2420. doi:10.5220/0014241900004052.
[45] T. Chen, J. Lu, Probabilistic alternating-time temporal logic and model checking algorithm, in: FKSD 2007, volume 2, 2007, pp. 35–39.
[46] W. Jamroga, M. Kwiatkowska, W. Penczek, L. Petrucci, T. Sidoruk, Probabilistic timed atl (2025).
[47] G. Marra, S. Dumančić, R. Manhaeve, L. De Raedt, From statistical relational to neurosymbolic artificial intelligence: A survey, Artificial Intelligence 328 (2024) 104062. URL: https://www.sciencedirect.com/science/article/pii/S0004370223002084. doi:https://doi.org/10.1016/j.artint.2023.104062.
[48] C. Listone, V. Malvone, A. Murano, Toward explainable diagnosis: A neurosymbolic approach, in: Proc. of ICAART'26, Vol. 4, SciTePress, 2026, pp. 3190–3197. doi:10.5220/0014403800004052.
[49] A. S. d'Avila Garcez, M. Gori, L. Lamb, L. Serafini, M. Spranger, S. Tran, Neural-symbolic computing: An effective methodology for principled integration of machine learning and reasoning, FLAP 6 (2019) 611–632. URL: https://api.semanticscholar.org/CorpusID:155092677.