VEsNA Goes Very Fast: Event-Based BDI Control in Formula One Racing

Jacopo Nazzaro[1[0009−0006−0913−1199]], Andrea Gatti[2[0009−0003−0992−4058]], and Angelo Ferrando[1[0000−0002−8711−4670]]

[1] University of Modena and Reggio Emilia, Modena 41125, Italy
333698@studenti.unimore.it
angelo.ferrando@unimore.it
[2] University of Genoa, Genoa 16146, Italy
andrea.gatti@edu.unige.it

Abstract. Symbolic BDI agents are often perceived as ill-suited for high-speed, continuous, and adversarial domains due to the computational demands of deliberation and the challenges of interfacing with low-level control. This paper presents an engineering study showing that unmodified Jason/AgentSpeak(L) agents, embodied through VEsNA in a Godot-based Formula One simulation, can achieve competitive multi-car racing behaviour at speed using an event-driven symbolic perception-action loop. Our focus is not end-to-end low-level vehicle control: the environment handles continuous vehicle dynamics, low-level actuation, and last-resort collision avoidance, while agents deliberate over compact add/remove percepts (e.g., curves and proximity-based interaction) to select tactical behaviours such as braking/acceleration, racing-line switching, overtaking, and defensive manoeuvres. We describe the architecture, the symbolic plan library, and an empirical evaluation based on telemetry logs, providing evidence that symbolic BDI tactical control can remain responsive in high-frequency continuous domains without modifications to the Jason/AgentSpeak(L) reasoning cycle. We further show that propensities can be integrated by annotating plans with temperaments (e.g., `aggressive`/`calm`) to bias plan selection and induce heterogeneous driving styles without modifying the BDI reasoning cycle.

Keywords: BDI agents · symbolic tactical control · multi-agent systems · racing simulation · VEsNA.

1 Introduction

Symbolic BDI (Belief–Desire–Intention) agents are valued for their explicit, modular, and interpretable model of rational decision making based on beliefs, goals, and plans [5,15]. However, they are often considered ill-suited to high-speed, continuous, and highly dynamic domains, where rapid sensing–action loops, frequent environmental change, and strict timing constraints dominate. This is commonly attributed to the deliberative nature of BDI reasoning, which can increase latency and reduce responsiveness in time-critical settings. Rather than accelerating symbolic deliberation, our approach preserves responsiveness by confining

symbolic reasoning to sparse, qualitative events and delegating continuous dynamics, low-level actuation, and safety-critical control to the environment.

This limitation has been widely acknowledged. Standard BDI models have been described as able to reason *about* time, but not *in* time, limiting their applicability in real-time and safety-critical systems [1]. Empirical studies also show that reaction times in symbolic agent architectures can increase significantly under load, motivating scheduling mechanisms, architectural extensions, or hybrid symbolic–subsymbolic solutions [2]. As a result, empirical evidence that symbolic BDI agents can support responsive *tactical* control in realistic, continuous, adversarial domains remains limited.

To investigate this issue, we consider *competitive Formula One racing*, a demanding domain combining continuous dynamics, high vehicle speeds, adversarial multi-agent interaction, and frequent tactical decisions such as overtaking, defensive driving, and collision avoidance. We present an engineering study in which multiple autonomous racing cars are controlled in real time by independent symbolic BDI agents within a physics-based simulation developed in the Godot game engine [14]. Each agent performs high-level tactical decision making, while the environment handles vehicle dynamics, low-level actuation, and interaction safety. The system is built on VEsNA [12,10], a general-purpose open-source ecosystem for embodying Jason [3] agents in virtual environments. Unlike approaches relying on learning-based components, subsymbolic controllers, or architectural extensions, the agents use unmodified Jason/AgentSpeak(L) for the *decision layer*, together with domain-specific plan libraries and an event-based symbolic interface.

The paper makes three contributions. First, through an engineering case study, we demonstrate the feasibility of using symbolic BDI agents for high-speed *tactical* decision making in a continuous, adversarial multi-agent domain without modifying the Jason/AgentSpeak(L) reasoning cycle or introducing real-time scheduling mechanisms. Second, we show that symbolic propensities, realised through temperament-annotated plans at the *plan-selection level*, induce heterogeneous driving styles and measurable differences in racing outcomes while preserving the standard BDI cycle. Third, we describe an agent–environment integration architecture in which the environment maps continuous state to event-based symbolic percepts and enforces safe, timely execution, while agents deliberate over tactical choices; we report an empirical evaluation illustrating observed behaviours, responsiveness, and scalability up to seven concurrent cars. An accompanying video is available at https://youtu.be/2J2tvfKhaGo, illustrating overtaking, close interactions, and the event-driven symbolic decision process discussed in the paper.

We emphasise that our contribution is not to replace low-level continuous control or safety-critical reflexes with symbolic reasoning. Rather, it is to show that unmodified Jason/AgentSpeak(L) [3,15] can remain effective as a *tactical decision layer* in a continuous, adversarial, time-constrained domain when paired with an event-based symbolic interface and an environment-side execu-

tion substrate. This distinction is essential for interpreting both the strengths and limitations of the proposed approach.

The remainder of the paper is structured as follows. Section 2 introduces the execution environment and agent framework. Section 3 describes the simulation environment and domain abstractions. Section 4 details the agent architecture and perception–action pipeline. Section 5 discusses multi-agent interactions and emergent behaviours. Section 6 presents the experimental evaluation, Section 7 reviews related work, and Section 8 concludes and outlines directions for future research.

2 Background

This section introduces the technical context required to understand the remainder of the paper. We briefly characterise the class of agents considered, the role of the VEsNA framework in supporting agent embodiment, and the nature of the execution environment used in the experimental scenario.

2.1 Symbolic Agents in Situated, Continuous Environments

The agents considered in this work are symbolic BDI agents implemented in Jason [3] and executed according to the standard AgentSpeak(L) [15] reasoning cycle. We assume familiarity with the BDI model and with symbolic agent programming languages, and therefore do not reiterate their theoretical foundations. From an operational perspective, the agents reason over discrete beliefs, select plans symbolically, and commit to intentions that generate high-level actions. These actions are not low-level control commands, but abstract directives whose execution is delegated to an external environment. As a result, symbolic deliberation remains decoupled from continuous dynamics, while still allowing agents to operate in situated and time-sensitive domains.

2.2 The VEsNA Framework

VEsNA [12,10] is an open-source agent-based ecosystem designed to support the embodiment of symbolic agents within virtual environments. It provides the infrastructure required to synchronise symbolic reasoning with an external simulation loop, translating environmental observations into agent beliefs and symbolic actions into environment-specific commands.

The framework supports the embodiment of Jason agents in different execution contexts, including virtual environments developed using Godot [14] (via the `vesna-light` module) and Unity [17] (via `vesna-unity` [6]). In all cases, agents operate as independent symbolic entities, each associated with a single embodied instance in the environment. Perception and actuation are mediated through a well-defined interface, allowing the symbolic reasoning layer to remain agnostic with respect to the concrete implementation of sensing, physics, or rendering. This separation enables the same symbolic agent to be deployed across different environments without modifications to its internal reasoning structure.

In addition to embodiment, the VEsNA ecosystem includes optional extensions for natural language interaction between humans and agents. In particular, the ChatBDI framework [8,13] enables dialogue-based interaction by connecting symbolic BDI agents with large language models, while preserving the agents' internal symbolic reasoning cycle. These components are orthogonal to the control mechanisms discussed in this paper and are not used in the experimental scenario.

The VEsNA ecosystem has also been extended with VEsNA-Pro [11], a module that introduces agent propensities[3], such as personality traits and mutable temper states, which can be used to annotate plans and bias intention selection through compatibility measures and post-execution updates. In the experimental system presented in this paper, we explicitly adopt the *conceptual model* of VEsNA-Pro by annotating plans with temperament labels (e.g., `aggressive` and `calm`) and using these annotations to bias plan selection. As a result, behavioural variability is realised through domain-level symbolic modelling rather than through a dedicated framework extension, while remaining fully compatible with propensity-aware execution models. While VEsNA offers general-purpose locomotion actions (e.g., movement, rotation, and discrete interactions), these primitives are intended as conceptual abstractions rather than fixed control mechanisms. Domain-specific environments may therefore redefine or extend the action layer to expose symbolic operations that better reflect the characteristics of the simulated domain. This design choice allows symbolic agents to be reused across heterogeneous environments while adapting the action interface to the dynamics and constraints of a given scenario.

2.3 Game Engines as Agent Execution Environments

Godot provides the computational substrate for the virtual environment in which the agents are embedded [14]. Its support for continuous physics simulation, high-frequency update loops, spatial querying, and the management of multiple interacting entities makes it well suited to dynamic, adversarial, physically grounded domains such as competitive vehicle racing.

The engine maintains world state, simulates vehicle dynamics, detects collisions, and generates raw perceptual data through constructs such as collision shapes and spatial areas. Agents do not interact with the engine directly; instead, perceptual inputs are abstracted into discrete beliefs, and symbolic actions are mapped via VEsNA to parameterised control routines executed by the environment. This yields a clear separation of concerns: Godot provides the physical and temporal substrate, while symbolic agents retain responsibility for decision-making and behavioural control. Symbolic tactical reasoning is thus separated from continuous control and hard safety enforcement. The contribution of this study is therefore the feasibility of symbolic BDI *tactical* control in a high-speed environment, not the replacement of low-level real-time vehicle control by symbolic agents alone.

[3] Propensities function as symbolic plan-selection biases rather than beliefs.

3 Simulation Environment

The racing scenario is implemented in the Godot game engine using a publicly available 3D model[4] of the *Red Bull Ring, Spielberg, Austria*. Although the circuit's visual representation is externally sourced, all functional components of the simulation were developed for this work, including vehicle controllers, perception mechanisms, symbolic abstractions, and integration with the agent reasoning layer. Each vehicle is simulated as a fully dynamic body under continuous physics-based motion. The environment handles low-level dynamics such as engine force, braking, and steering, while symbolic agents make high-level control decisions. Vehicles follow predefined racing lines represented as parametric trajectories sampled along the track, including an optimal line and alternative overtaking trajectories. Switching is achieved by smoothly blending between trajectories, allowing tactical deviation without constraining motion.

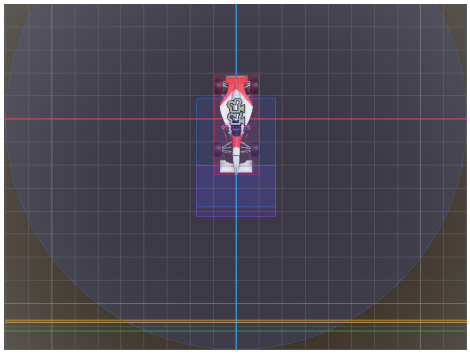

Fig. 1. Vehicle perceptual areas used to generate symbolic percepts. Blue: obstacles; green: slipstream; yellow: close-range competition and overtaking.

To support symbolic perception, the environment implements a custom sensing layer based on spatial volumes attached to each vehicle (Figure 1). These invisible sensing areas detect relevant spatial and situational conditions, such as proximity to opponents, side-by-side racing, and entry into braking zones. Detected conditions are translated into qualitative symbolic percepts and transmitted to the controlling agent. In addition to racing lines, the track includes a set of static curve zones implemented as spatial areas within the environment. These zones act as qualitative track annotations rather than navigation targets, encoding metadata such as ideal cornering speed, braking distance, and throttle pressure. When a vehicle enters or exits a zone, the environment generates corresponding symbolic percepts that allow the agent to anticipate upcoming curves and reason about braking and acceleration in advance, analogously to braking reference points used by human drivers.

[4] https://tinyurl.com/mry222dt

4 Agent Architecture

Each vehicle in the simulation is controlled by an independent symbolic BDI agent implemented in Jason and embodied through the VEsNA framework. Agents interact with the environment via an event-driven perception–deliberation–action loop mediated by a WebSocket-based bridge. Rather than operating at a fixed control frequency, agents react to the addition and removal of symbolic percepts. Deliberation is driven only by relevant environmental changes, avoiding continuous sensing and bounding symbolic reasoning costs.

4.1 Abstraction Level and Granularity of the BDI Model

Designing symbolic BDI agents for a high-speed racing domain requires a careful choice of abstraction level. Decisions must be sufficiently expressive to capture tactical behaviour, while remaining decoupled from low-level numerical control and physics-specific details. Figure 2 illustrates the adopted separation of concerns. The symbolic agent represents the *driver* perspective and reasons at a strategic and tactical level. Its beliefs encode qualitative information such as relative position, racing line, and interaction state with opponents, while its intentions correspond to abstract manoeuvres including braking, acceleration, racing-line changes, and overtaking.

The simulated vehicle acts as an execution substrate and perceptual interface, translating abstract symbolic actions into continuous mechanical effects such as steering, throttle, braking, and collision dynamics. It does not perform autonomous decision making or reason about goals or strategies; its role is limited to executing the agent's intentions and generating perceptual events reflecting the current physical situation. This abstraction enables complex racing behaviour to be expressed symbolically without exposing agents to low-level numerical control. For example, proximity to another vehicle is perceived as a symbolic competitive interaction with an assigned role, supporting tactical reasoning while the environment ensures safe and continuous physical execution.

4.2 Event-Driven Perception Pipeline

Perceptual information is generated when relevant spatial conditions occur, such as entering or exiting a curve, approaching another vehicle, or engaging in side-by-side racing. These conditions are detected by the environment and translated into symbolic beliefs, including `curve` beliefs for anticipatory braking and acceleration, and `battle` and `wheel_to_wheel` beliefs for competitive interaction. All percepts follow a uniform add/remove lifecycle: the environment emits an event when a condition becomes true and a corresponding event when it ceases to hold. This ensures that the belief base reflects the current situated context, preventing the accumulation of stale information and enabling controlled reasoning over transient situations.

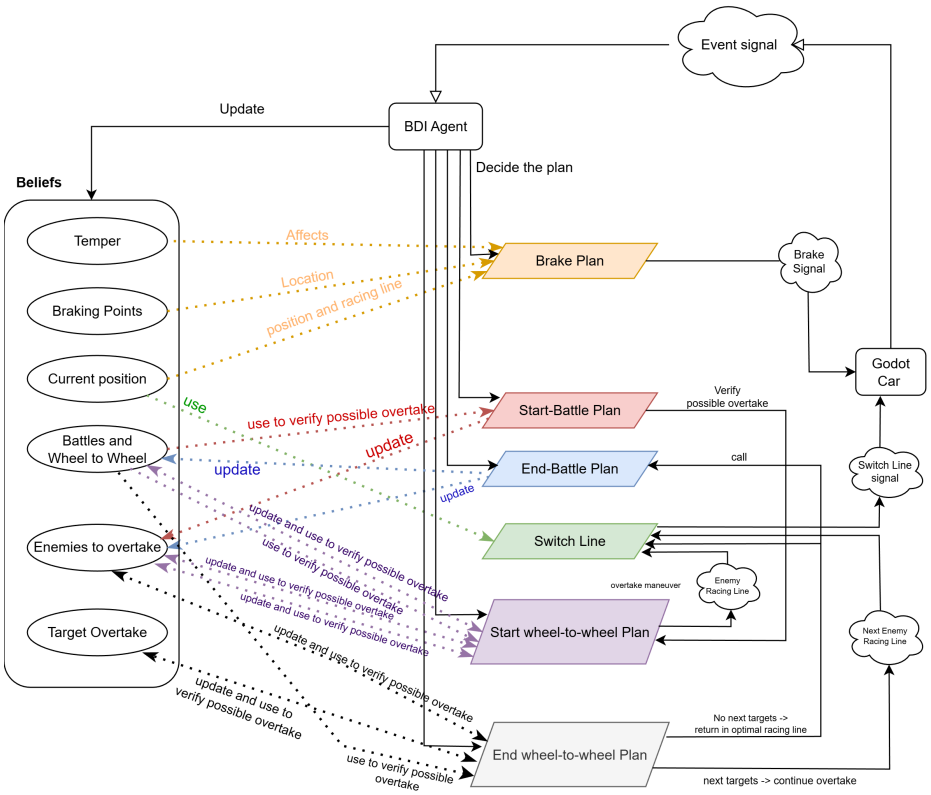

Fig. 2. Separation between symbolic decision making and physical execution in the proposed architecture.

```
void handleCurvePerception(perception) {
    if (perception.operation == ADD) {
        add belief curve(
            name, id, distance, currentSpeed,
            idealSpeed, acceleration, throttlePressure);
    } else if (perception.operation == REMOVE) {
        remove all matching curve beliefs;
    }
}
```

Listing 1.1. Handling of curve perceptions at the agent–environment interface.

The arguments of the `curve` belief provide a qualitative abstraction of the upcoming track segment rather than raw sensor data. Parameters such as remaining distance, current speed, and ideal cornering speed support anticipatory symbolic reasoning without continuous control or numeric optimisation.

4.3 Symbolic Control Actions

Agents do not manipulate continuous control variables directly. Instead, they select among abstract symbolic actions, such as braking, accelerating, racing-line selection, and overtaking, which are interpreted by environment-side controllers responsible for continuous dynamics. This separation allows symbolic agents to focus on tactical decision making while the environment ensures physically consistent execution.

4.4 Plan Structure and Atomicity

Domain-specific behaviour is encoded in plan libraries combining reactive and goal-driven plans. Reactive plans are triggered by perceptual beliefs and handle safety-critical situations such as curve entry, braking, and overtaking. Several plans are marked as `@atomic` to prevent interleaving during critical transitions.

Listing 1.2 shows representative plans for anticipatory curve handling. Rather than mapping a `curve` percept to a single action, the agent selects among multiple behaviours depending on contextual conditions and agent propensities.

```
@curve_entering_bp_aggr[atomic, temper([aggressive(1.0)])]
+curve(Name, Id, Distance, CurSpeed, IdealSpeed, Acc, _)
    : brake_point(Name, Space, aggressive)
      & CurSpeed > IdealSpeed
    <- !brake(Distance, CurSpeed, IdealSpeed, Acc, Space).

@curve_entering_bp_calm[atomic, temper([calm(1.0)])]
+curve(Name, Id, Distance, CurSpeed, IdealSpeed, Acc, _)
    : brake_point(Name, Space, calm)
      & CurSpeed > IdealSpeed
    <- !brake(Distance, CurSpeed, IdealSpeed, Acc, Space).

@curve_entering_no_brake[atomic]
+curve(Name, Id, _, CurSpeed, IdealSpeed, _, _)
    : CurSpeed =< IdealSpeed
    <- true.

@curve_exit[atomic]
-curve(Name, Id, _, _, _, _, Throttle)
    : accelerating_point(Name)
    <- !accelerate(Throttle).
```

Listing 1.2. AgentSpeak(L) plans for anticipatory curve handling.

4.5 From Reactive Events to Temporally Extended Tactical Reasoning

Although decisions in the racing domain are triggered by local perceptual events, several core behaviours, most notably overtaking, cannot be reduced to single-step reactive responses. Instead, they require temporally extended reasoning in which agents commit to intermediate intentions and condition future decisions on the outcome of previous manoeuvres.

Overtaking is therefore modelled as a structured symbolic process. When a competitive situation is detected, the agent establishes a symbolic representation of the interaction, selects a target opponent, and commits to an overtaking intention, understood here operationally as a committed course of action

persisting across multiple reasoning cycles. While one could instead post an overtaking goal and treat subsequent manoeuvres as goal-directed substeps, an intention-centred formulation more directly captures commitment, continuation, and cancellation across changing interaction phases, while remaining close to Jason's operational semantics. In practice, this intention may span phases such as `battle` and `wheel_to_wheel`, whose outcomes shape subsequent decisions, including whether to pursue further overtakes or return to the optimal racing line.

This modelling choice allows symbolic BDI reasoning to capture the strategic and sequential nature of racing tactics while remaining event-driven and responsive. The concrete realisation of this reasoning pattern is detailed in the following section, with additional implementation details provided in Appendix A.

5 Multi-Agent Interactions

We now shift from the individual to the system level, examining how interaction emerges when multiple symbolic agents operate concurrently in the same racing environment. The scenario is a competitive multi-agent system in which all vehicles are controlled by independent agents sharing the same reasoning architecture and plan library. There is no explicit agent-to-agent communication or centralised coordination. Instead, interaction is mediated by the environment, which detects spatial relations among vehicles and provides a shared perceptual substrate. Agents therefore adapt their behaviour to the environment-mediated effects of other agents' actions.

Competitive interaction arises mainly in close racing situations, where agents reason symbolically about relative position and role while maintaining safety at high speed. Although overtaking and defensive manoeuvres are triggered by local perceptual events, their effects are inherently multi-agent. A racing-line change by one vehicle alters the perceptual context of nearby agents and may trigger further symbolic reactions. Interaction thus unfolds as sequences of interdependent decisions rather than isolated stimulus–response pairs. Listing 1.3 illustrates this symbolic representation of competitive interaction. The plan is triggered by a belief describing a battle situation and initiates an overtaking manoeuvre only when the agent is in a following role. Although local to one agent, its execution has system-level consequences through changes in the shared environment.

```
@react_to_competition[atomic]
+battle(Opponent, Id, RacingLine, following)
    : not overtaking(_)
    <- +overtaking(Opponent);
       !switch_line(RacingLine).
```

Listing 1.3. Symbolic reasoning over competitive interaction with another agent.

Safety becomes a fundamentally multi-agent concern when vehicles operate in close proximity. Unlike in single-agent scenarios, where safety mainly reduces to staying on track, here it is relational and depends on the behaviour of nearby agents. The environment therefore continuously monitors inter-vehicle distances and spatial configurations, detecting hazardous situations such as side-by-side racing or rapid frontal approach.

Emergency collision avoidance is enforced by a lightweight environment-side safety layer that applies corrective actions when necessary. Operating independently of symbolic deliberation, it ensures physical feasibility in time-critical situations while complementing rather than replacing agent decision making.

At the system level, anticipatory behaviour, competitive interaction, and reactive safety responses give rise to emergent phenomena such as congestion, opportunistic overtakes, defensive driving, and dynamic racing groups. These behaviours emerge from the interaction between homogeneous symbolic controllers and a continuous adversarial environment, showing that symbolic BDI tactical reasoning can support complex, high-speed multi-agent interaction without learning-based components, continuous optimisation, or architectural extensions.

The safety layer does not encode tactical or strategic behaviour. Symbolic agents remain responsible for high-level decisions, including braking, acceleration, racing-line selection, and overtaking intent, while the environment enforces the physical constraints needed to prevent collisions. Agents also coordinate only indirectly through the environment and exchange no symbolic messages. This design isolates the effects of local symbolic reasoning and shared environmental dynamics, while leaving explicit communication to future work on team tactics, cooperative manoeuvres, and richer opponent modelling.

6 Evaluation

This section assesses whether symbolic BDI tactical control remains sufficiently responsive at racing speeds and whether multi-agent interaction can be handled safely. Throughout this paper, *real-time* is understood as *soft real-time tactical responsiveness*: symbolic decisions must be generated quickly enough to sustain competitive behaviour under racing conditions, while hard instantaneous safety constraints are enforced by the environment-side safety layer.

6.1 Experimental Setup

Experiments were conducted in the Godot simulation (Section 3) with one Jason agent per vehicle, embodied through VEsNA and connected via an event-driven WebSocket bridge (Section 4). At startup, each agent issues an engine start command followed by maximum acceleration (100% = 1.0). We consider a solo time-trial and multi-vehicle races with three and seven cars. Results are intended as indicative of system behaviour rather than statistically conclusive.

All agents are equipped with *propensities* (temperaments) that bias plan selection and action parameterisation. Since propensities introduce run-to-run variability, results are reported over multiple independent simulations, using both lap-time summaries and aggregated speed statistics.

To support reproducibility and inspection of the agent design, the complete implementation of the system, including the Jason agents, environment integration, and simulation setup, is publicly available at: https://github.com/NazzaroJacopo/VesnaGoesFast.

Telemetry and metrics. We log runtime telemetry, interaction events, emergency-braking triggers, and overtaking attempts (success/failure). Braking is driven by event-based curve zones: entering a zone produces a `curve` belief (including speed, distance, and ideal speed), while zone exit provides a recommended acceleration. Interactions are detected through proximity phases (*battle* and *wheel-to-wheel*), which generate corresponding symbolic beliefs; emergency braking and collision avoidance are enforced environment-side. Evaluation metrics cover **responsiveness** (delay, update rate), **stability** (on-track rate, oscillations), **interaction safety** (minimum distance, emergency braking), and **racing performance** (lap completion, lap times, overtaking outcomes).

6.2 Quantitative Results

Since propensities introduce run-to-run variability, we report results over multiple independent simulations and include a no-propensity baseline to isolate their effect from the rest of the symbolic control loop. Table 1 reports lap-time statistics for a representative three-car run with propensities disabled, while Table 2 summarises three independent 15-lap runs with propensities enabled. Three further 15-lap simulations[5] were used to compute per-vehicle speed statistics (minimum, maximum, and mean speed), reported in Table 3.

Claim 1: Temporal stability of symbolic control. The no-propensity baseline exhibits highly stable behaviour across laps. Excluding the slower standing-start lap, lap times quickly converge and remain within a narrow band for laps 2–15. As shown in Table 1, average lap times differ by less than 0.1 s across agents, and the standard deviation over steady-state laps is minimal. This indicates that symbolic deliberation does not accumulate delay over time and remains responsive at racing speed throughout extended execution.

Table 1. Lap-time statistics over 15 laps with propensities disabled. Standard deviation is computed over laps 2–15. Times are reported in seconds (s).

Agent	Best lap	Avg. lap	Std. dev.
M1	47.08	47.35	0.09
M2	47.10	47.34	0.08
M3	47.27	47.41	0.10

Claim 2: Structured behavioural divergence induced by propensities. When propensities are enabled, lap-time distributions exhibit substantially greater variability across executions (Table 2). In particular, agent M2 achieves markedly faster best laps in simulations S1 and S3 (45.65 s and 45.52 s), while simulation S2 is globally slower and more homogeneous. The gap between average and best lap time (Δ) highlights differences in consistency and risk-taking: larger gaps correspond to more opportunistic driving styles, while smaller gaps indicate more

[5] Speed-statistics simulations are independent from the lap-time simulations.

conservative behaviour. These differences persist across multiple laps, indicating that propensity-annotated plan selection induces sustained driving styles rather than isolated stochastic effects.

Table 2. Lap-time statistics with propensities enabled: three independent 15-lap simulations (S1–S3). Δ denotes the difference between average and best lap time. Times are reported in seconds (s).

Sim.	Agent	Avg. lap	Best lap	Δ
S1	M1	47.56	47.27	0.29
	M2	46.17	45.65	0.52
	M3	47.36	46.55	0.81
S2	M1	47.81	46.82	0.99
	M2	47.71	46.75	0.96
	M3	47.84	47.55	0.29
S3	M1	47.48	47.25	0.23
	M2	46.22	45.52	0.70
	M3	46.38	45.53	0.85

Claim 3: Performance is driven by decision timing rather than raw speed. Speed statistics aggregated over three independent propensity-enabled simulations are reported in Table 3. While average mean speeds are very similar across agents, differences in maximum and minimum speeds reveal distinct driving dynamics. Notably, higher maximum speed does not systematically correspond to faster lap times, indicating that symbolic decisions about braking, acceleration, and racing-line selection (*i.e.*, *when* speed is gained or lost) play a more decisive role than sustained speed advantages.

Table 3. Speed statistics with propensities enabled, aggregated over three independent simulations. Values are reported in km/h.

Metric	M1	M2	M3
Avg. max speed	194.78	196.19	194.62
Avg. min speed	69.22	70.87	70.37
Avg. mean speed	143.76	144.89	144.72

Claim 4: Scalable interaction under multi-agent competition. Aggregated interaction statistics from a representative 15-lap, three-car run include 14 emergency-braking events and 42 overtaking attempts, 27 of which succeeded (64.29%). This corresponds to approximately 0.31 emergency interventions per car-lap. Although sensitive to race-start congestion and close-proximity traffic, these values suggest that the environment-side safety mechanism acts as an occasional safeguard rather than as the primary driver of behaviour. Close racing and overtaking remain largely governed by symbolic tactical decisions, while emergency intervention is reserved for situations in which symbolic reaction alone would

be insufficiently timely. This observation is reinforced by a 16-lap race with seven concurrent vehicles (M1–M7). After initial position changes, the system converges to a stable ordering without oscillatory behaviour or performance collapse. The fastest lap was 46.53 s (M6), and average lap times remain within a narrow range (47.38 s to 47.93 s), comparable to the three-vehicle scenario. Taken together, these results suggest that the same perception-driven plan library and environment-side safety layer scale beyond small multi-agent settings without requiring additional architectural mechanisms.

6.3 Reaction Latency and Mitigation Strategies

Experimental runs revealed that, while symbolic deliberation remains stable over time, end-to-end perception–decision–action latency can be non-negligible in time-critical situations, particularly during high-speed braking. Environment-side measurements indicate average reaction times on the order of several hundred milliseconds, with occasional higher peaks. At racing speeds, such delays can correspond to significant spatial distances, potentially affecting braking precision and competitiveness.

To mitigate this limitation, braking decisions are shifted from reactive execution to anticipatory reasoning. When entering a curve zone, agents compute a prediction of their future speed at a predefined braking point, which is encoded symbolically as part of their prior knowledge of the track. Based on this prediction, the agent commits to a braking intention in advance, while the actual braking action is executed by the vehicle controller only when the braking point is reached. This approach decouples symbolic deliberation from immediate execution, allowing braking to remain responsive despite deliberation latency.

Not all situations, however, are amenable to prediction. In particular, emergency collision avoidance requires reaction times below those achievable through symbolic reasoning alone. For this reason, a lightweight safety layer is enforced directly by the environment, monitoring inter-vehicle distances and applying immediate corrective actions when necessary. This mechanism operates independently of agent goals and plans, ensuring physical safety while preserving the symbolic nature of tactical decision making.

A detailed analysis of reaction-time measurements, predictive braking implementation, and environment-side safety mechanisms is provided in Appendix B.

6.4 Qualitative Racing Behaviour

Qualitatively, agents exhibit three characteristic classes of behaviour: (i) anticipatory braking and re-acceleration driven by curve-zone percepts, (ii) tactical racing-line switching during overtaking, and (iii) sustained close racing under environment-level safety enforcement. These behaviours are illustrated by the curve-based braking and acceleration areas in Figure 3 and by a representative wheel-to-wheel interaction in Figure 4.

Figure 3 highlights the explicit lifecycle of symbolic percepts used for anticipatory control: entering a curve zone adds a `curve` belief that triggers braking

plans, while exiting the zone removes the belief and activates re-acceleration plans. This add/remove mechanism tightly couples symbolic deliberation to spatial context, enabling predictive yet reactive behaviour without continuous sensing or numeric optimisation.

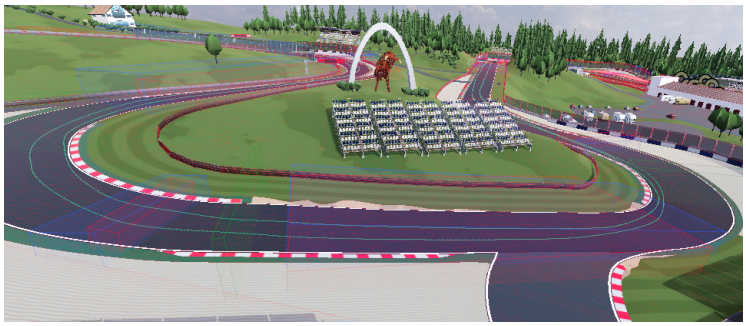

Fig. 3. Combined braking/acceleration/reset areas. Exiting a curve area removes the `curve` belief and triggers symbolic re-acceleration plans.

Figure 4 illustrates close-range interaction between vehicles. In such situations, agents reason symbolically over competitive percepts (e.g., `wheel_to_wheel`) to initiate overtaking or defensive manoeuvres, while an environment-side safety layer ensures collision avoidance in real time. As a result, coordination and competition emerge from shared perceptual abstractions rather than explicit agent-to-agent communication.

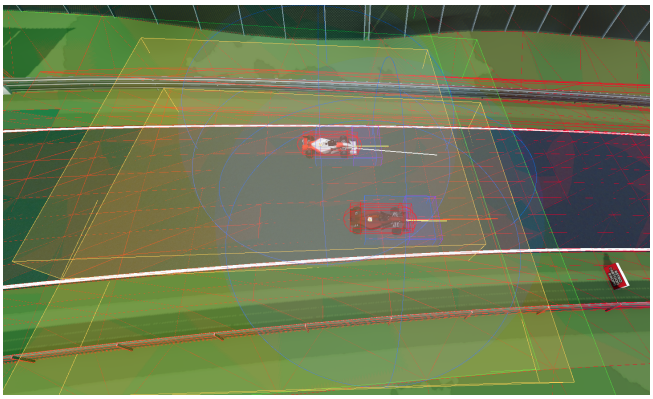

Fig. 4. Wheel-to-wheel situation captured during multi-agent interaction.

6.5 Discussion of Findings

Taken together, the quantitative and qualitative results support the paper's main engineering claim: symbolic BDI tactical control can remain responsive and stable in a high-speed, continuous, adversarial domain when agents reason over compact, event-based percepts and delegate low-level continuous execution to the environment. Responsiveness is preserved not through architectural extensions or subsymbolic components, but through careful design of the agent–environment interface: perception is limited to add/remove events, and safety constraints are enforced environment-side, keeping symbolic deliberation bounded even under sustained multi-agent interaction.

At the same time, the proposed architecture is best understood as a layered design pattern for continuous domains in which a reactive execution substrate already exists. It is most applicable when symbolic agents must reason over sparse, qualitative, event-based state to select tactics, while lower layers handle continuous dynamics, actuator-level control, and hard safety constraints. Conversely, it is less suitable for domains in which symbolic reasoning is expected to close the low-level control loop directly or provide hard real-time safety guarantees without such a substrate. This boundary should be kept in mind when generalising the present results beyond the racing scenario considered here.

Although CPU load is not measured directly, and the latency analysis is limited to the reaction-time measurements reported in Appendix B, the temporal stability of lap times across extended runs provides further evidence that symbolic deliberation does not introduce accumulating delay.

Finally, propensity annotations provide a lightweight symbolic mechanism for inducing heterogeneous driving styles: plan-selection biases affect lap-time variability and interaction outcomes without altering the underlying BDI reasoning cycle, indicating that behavioural diversity and real-time responsiveness can coexist within a purely symbolic control architecture.

7 Related Work

Symbolic BDI agents have long been valued for their modularity and interpretable decision making. However, their deployment in real-time, high-speed domains remains limited by concerns over deliberation latency and the integration of symbolic reasoning with continuous control loops.

Early work addressed these issues through soft real-time extensions. Vincent et al. [19] introduced task prioritisation and deadline-aware scheduling into agent execution cycles, while Vikhorev et al. [18] extended AgentSpeak(L) with plan priorities and deadlines to support reasoning under temporal constraints. Although these approaches demonstrated bounded responsiveness, they were evaluated in relatively simple, often single-agent, environments.

More recent work has incorporated formal scheduling theory into symbolic agent frameworks. Alzetta et al. [1] proposed RT-BDI, which integrates feasibility analysis and CPU-aware scheduling, including Earliest Deadline First, into

the BDI reasoning cycle. Traldi et al. [16] later deployed RT-BDI on robotic platforms, showing that symbolic agents can satisfy hard real-time constraints through dynamic intention management and task preemption. Calvaresi et al. [7] provided a broader formalisation of real-time rationality in multi-agent systems, highlighting challenges such as internal scheduling overhead and communication delays under stringent timing requirements.

While these studies address symbolic control under temporal constraints, the use of symbolic agents in high-speed, continuous, and adversarial domains remains underexplored. One relevant contribution is the Jason-based framework of Bosello and Ricci [4], which integrates learning into BDI agent development, but focuses on agent design and plan refinement rather than real-time execution in dynamic environments. Our work differs in both focus and contribution. We show that symbolic BDI agents for tactical decision making, implemented on the unmodified VEsNA platform and without subsymbolic components, architectural extensions, or learning mechanisms, can operate effectively in a high-speed multi-agent racing scenario. Using a physics-based Formula One simulation, we provide empirical evidence that symbolic plans alone can support responsive control and complex behaviours such as overtaking and defensive driving. This provides evidence for the feasibility of symbolic BDI reasoning in continuous, adversarial, and time-constrained multi-agent environments.

8 Conclusion and Future Work

This paper showed that symbolic BDI agents, implemented in Jason and embodied through VEsNA, can provide the tactical decision layer for multiple vehicles in a physics-based Formula One simulation using event-based percepts, while environment-managed dynamics and a safety layer ensure continuous execution and physical feasibility. We also showed that temperament-annotated plans can bias plan selection without changing the BDI reasoning cycle, producing heterogeneous driving styles and measurable performance differences. Experiments with up to seven concurrent vehicles suggest that the same perception-driven plan library and safety mechanisms scale beyond small multi-agent settings. Although curve zones and racing lines are domain-specific, the underlying design pattern is general: agents reason over compact symbolic state derived from event-based percepts, while the environment handles low-level control and safety. This pattern may extend to other time-critical domains requiring symbolic reasoning alongside continuous dynamics.

Beyond simulation, the results suggest applications in motorsport strategy support and driver training, where symbolic BDI agents could provide interpretable opponents or decision-support tools over real-time telemetry.

Future work includes direct measurements of perception–action latency and CPU load, stronger opponent modelling, team-level strategies, locality-aware communication [9], and hybrid approaches combining learned tactical parameters with symbolic goal and plan selection.

References

1. Alzetta, F., Giorgini, P., Marinoni, M., Calvaresi, D.: RT-BDI: A real-time BDI model. In: Demazeau, Y., Holvoet, T., Corchado, J.M., Costantini, S. (eds.) Advances in Practical Applications of Agents, Multi-Agent Systems, and Trustworthiness. The PAAMS Collection - 18th International Conference, PAAMS 2020, L'Aquila, Italy, October 7-9, 2020, Proceedings. Lecture Notes in Computer Science, vol. 12092, pp. 16–29. Springer (2020). https://doi.org/10.1007/978-3-030-49778-1_2
2. Becker, L.B., de Oliveira Silvestre, I., Hübner, J.F., Fisher, M.: An expedited BDI agent architecture: Improving the responsiveness of agent-based autonomous systems for handling critical situations. Robotics Auton. Syst. **186**, 104917 (2025). https://doi.org/10.1016/J.ROBOT.2025.104917
3. Bordini, R.H., Hübner, J.F., Wooldridge, M.: Programming multi-agent systems in AgentSpeak using Jason. J. Wiley (2007)
4. Bosello, M., Ricci, A.: From programming agents to educating agents - A jason-based framework for integrating learning in the development of cognitive agents. In: Dennis, L.A., Bordini, R.H., Lespérance, Y. (eds.) Engineering Multi-Agent Systems - 7th International Workshop, EMAS 2019, Montreal, QC, Canada, May 13-14, 2019, Revised Selected Papers. Lecture Notes in Computer Science, vol. 12058, pp. 175–194. Springer (2019). https://doi.org/10.1007/978-3-030-51417-4_9
5. Bratman, M.: Intention, Plans, and Practical Reason. Cambridge, MA: Harvard University Press (1987)
6. Briola, D., Vizzari, G., Mirra, G., Gatti, A., Martini, M., Mascardi, V.: Integrating JaCa-Unity in VEsNA for behavioural realistic VR simulations. In: 22nd European Conference on Multi-Agent Systems (EUMAS 2025). Springer (2025)
7. Calvaresi, D., Cid, Y.D., Marinoni, M., Dragoni, A.F., Najjar, A., Schumacher, M.: Real-time multi-agent systems: rationality, formal model, and empirical results. Auton. Agents Multi Agent Syst. **35**(1), 12 (2021). https://doi.org/10.1007/S10458-020-09492-5
8. Das, S., Nowé, A., Vorobeychik, Y. (eds.): ChatBDI: Think BDI, Talk LLM. International Foundation for Autonomous Agents and Multiagent Systems / ACM (2025). https://doi.org/10.5555/3709347.3743930
9. Ferrando, A., Gatti, A., Mascardi, V.: Oops, i heard that! situated communication with locality-aware KQML. In: Proceedings of the 13th International Workshop on Engineering Multi-Agent Systems (EMAS 2025), co-located with AAMAS 2025, Detroit, Michigan, USA. p. 157. Springer Nature (2025)
10. Ferrando, A., Gatti, A., Mascardi, V.: Integrating virtual reality, chatbots, and bdi agents: Vesna goes fast! In: Agents and Multi-Agent Systems Development: Platforms, Toolkits, Technologies, pp. 289–322. Springer (2026)
11. Gatti, A., Casale, F., Mascardi, V., Stucchi, A., Ferrando, A.: VEsNA-Pro: Exploiting BDI agents with propensities for emergent narrative. In: Amato, C., Dennis, L., Mascardi, V., Thangarajah, J. (eds.) Proceedings of the 25th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2026, Paphos, Cyprus, May 25-29, 2026. International Foundation for Autonomous Agents and Multiagent Systems / ACM (2026)
12. Gatti, A., Mascardi, V.: Vesna, a framework for virtual environments via natural language agents and its application to factory automation. Robotics **12**(2), 46 (2023). https://doi.org/10.3390/ROBOTICS12020046

13. Gatti, A., Mascardi, V., Ferrando, A.: Let me talk to you! natural language interaction between humans and BDI agents via ChatBDI. In: ECAI 2025 (2025)
14. Godot foundation: Godot web site (2014), https://godotengine.org/, accessed on June 11, 2026
15. Rao, A.S.: AgentSpeak(L): BDI agents speak out in a logical computable language. In: 7th European Workshop on Modelling Autonomous Agents in a Multi-Agent World, Eindhoven, The Netherlands, January 22-25, 1996. Lecture Notes in Computer Science, vol. 1038, pp. 42–55. Springer (1996). https://doi.org/10.1007/BFb0031845
16. Traldi, A., Bruschetti, F., Robol, M., Roveri, M., Giorgini, P.: Real-time BDI agents: A model and its implementation. In: Raedt, L.D. (ed.) Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022. pp. 511–517. ijcai.org (2022). https://doi.org/10.24963/IJCAI.2022/73
17. Unity Technologies: Unity web site (2005), https://unity.com/, accessed on June 11, 2026
18. Vikhorev, K., Alechina, N., Logan, B.: Agent programming with priorities and deadlines. In: Sonenberg, L., Stone, P., Tumer, K., Yolum, P. (eds.) 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011), Taipei, Taiwan, May 2-6, 2011, Volume 1-3. pp. 397–404. IFAAMAS (2011), http://portal.acm.org/citation.cfm?id=2030529&CFID=69153967&CFTOKEN=38069692
19. Vincent, R., Horling, B., Lesser, V.R., Wagner, T.: Implementing soft real-time agent control. In: André, E., Sen, S., Frasson, C., Müller, J.P. (eds.) Proceedings of the Fifth International Conference on Autonomous Agents, AGENTS 2001, Montreal, Canada, May 28 - June 1, 2001. pp. 355–362. ACM (2001). https://doi.org/10.1145/375735.376329

A Extended Overtaking Example

This appendix provides additional material to support the understanding of the agent decision process and the symbolic reasoning involved in overtaking manoeuvres. The content is intended as supplementary documentation and does not introduce new mechanisms beyond those described in Section 4.

A.1 Extended Example of Overtaking Strategy

To complement the compact overtaking plans presented in Section 4, we report here a longer excerpt from the AgentSpeak(L) plan library governing competitive interaction and multi-step overtaking behaviour. This example illustrates how symbolic reasoning supports temporally extended strategies conditioned on interaction outcomes.

The overtaking process is initiated when an agent enters a competitive situation in which it assumes a `following` role. At this stage, opponents ahead are collected into an internal target list, representing potential overtaking candidates. When the interaction progresses to a close-range phase, represented by a `wheel_to_wheel` belief, the agent selects a specific target opponent and commits to an overtaking intention by switching to an alternative racing line.

Subsequent behaviour depends explicitly on the outcome of the manoeuvre. If the overtaking attempt succeeds, as detected through a role inversion in the competitive beliefs, the agent may immediately continue the strategy by selecting a new target from the remaining list. If the attempt fails, the agent abandons the manoeuvre and returns to the optimal racing line, deferring further attempts until new interaction conditions arise. Throughout this process, internal control beliefs prevent re-triggering and ensure coherent execution across multiple perception–action cycles.

The following listing shows a representative subset of the plans involved in this extended reasoning pattern.

```
@battle_following[atomic]
+battle(Name, Id, RacingLine, following)
    <- !append_overtake(Name);
       !verify_switch(Name, Id, RacingLine).

@wheel_to_wheel_not_overtaking[atomic]
+wheel_to_wheel(Name, Id)
    : battle(Name, Id, RacingLine, following)
      & not overtaking(_)
    <- +overtaking(Name);
       !switch_line(RacingLine).

@wheel_to_wheel_exit[atomic]
-wheel_to_wheel(Name, Id)
    : overtaking(Name)
    <- !verify_next_move(Name).

@verify_next_move_followed[atomic]
+!verify_next_move(Name)
    : battle(Name, Id, _, followed)
      & overtaking(Name)
    <- !verify_next_attack.
```

```
@verify_next_move_following_failed[atomic]
+!verify_next_move(Name)
    : battle(Name, Id, _, following)
      & overtaking(Name)
    <- -overtaking(Name);
       !switch_line_to(optimal_RL).
```

This extended example makes explicit how overtaking behaviour is realised as a sequence of symbolic decisions distributed over time, rather than as a single reactive response. The example complements the compact plans presented in the main body of the paper and provides additional insight into the engineering choices underlying the symbolic control strategy.

B Latency, Predictive Braking, and Environment-Side Safety

This appendix provides a detailed engineering analysis of perception–action latency observed during experimentation, together with the mitigation strategies adopted to preserve responsiveness in high-speed racing conditions. The material complements the evaluation presented in Section 6 and does not introduce additional architectural mechanisms beyond those already described.

B.1 Observed Perception–Action Latency

During experimental runs, a practical limitation of symbolic BDI control emerged in situations requiring very fast reaction times, most notably during braking before high-speed curves. Although symbolic reasoning operations are computationally lightweight, the end-to-end perception–decision–action cycle introduces a non-negligible delay when operating under real-time constraints.

Latency was measured using environment-side timers, capturing the time elapsed between the detection of a curve-entry event by the simulation and the emission of the corresponding braking command by the agent. Across multiple runs, the average reaction time was approximately 0.53 seconds, with observed peaks close to 1 second and minimum values around 36 milliseconds. At typical racing speeds of approximately 144 km/h, this delay corresponds to a reaction distance of roughly 20 meters, which can significantly impact braking precision and competitiveness in narrow braking zones.

These observations highlight that, while symbolic control remains stable over time, raw reaction speed alone is insufficient to handle all time-critical situations if decisions are taken only at the moment a percept becomes available.

B.2 Predictive Braking as a Mitigation Strategy

To mitigate the impact of perception–action latency, a predictive braking mechanism was introduced. Rather than deciding whether to brake at the instant a braking action must be executed, agents anticipate the decision earlier, when entering a curve zone.

Table 4. Reaction time to brakes.

Iteration	Reaction time (ms)
1	124
2	754
3	346
4	363
5	955
6	447
7	978
8	272
9	592
10	522
11	992
12	443
13	956
14	289
15	567
16	494
17	39
18	445
19	938
20	275
21	571
22	497
23	36
24	474
25	956

Upon perceiving a curve, the agent computes a prediction of its future speed at a predefined braking point, which is encoded symbolically as part of its prior knowledge of the track. Based on this prediction, the agent commits to a braking intention in advance. The corresponding braking command is then executed by the vehicle controller only when the braking point is physically reached (see Table 4).

This approach shifts symbolic deliberation earlier in time, effectively decoupling decision making from immediate execution. As a result, braking actions are applied reactively at the correct spatial location while benefiting from anticipatory symbolic reasoning. Although this mechanism does not eliminate latency, it compensates for it in classes of decisions that are spatially predictable, such as curve braking.

B.3 Unpredictable Events and Environment-Side Safety

Not all racing situations can be addressed through prediction. In particular, emergency braking to avoid sudden collisions requires reaction times well below those achievable through symbolic deliberation alone. In close-proximity interactions, vehicles may require corrective action within a fraction of a second, depending on relative speed and spatial configuration.

For this reason, a lightweight safety layer was implemented directly within the simulation environment. This mechanism continuously monitors inter-vehicle distances and applies immediate corrective steering or braking forces when collision risk is detected. The safety layer operates independently of agent goals and

plans, enforcing instantaneous physical constraints without engaging symbolic reasoning.

This design preserves the symbolic nature of agent decision making while ensuring physical feasibility in extreme scenarios. Symbolic agents remain responsible for tactical decisions such as overtaking and braking intent, while the environment enforces last-resort safety constraints analogous to low-level stabilisation mechanisms used in real robotic platforms.