Article

Not peer-reviewed version

DS-GBT: Proactive Safety Integration for Dynamic Agent Decision Policies

Yao-Tian Chian * and Yuxin Zhai

Posted Date: 25 March 2026

doi: 10.20944/preprints202603.1992.v1

Keywords: autonomous agents; safety; DS-GBT; gated behavior trees; dynamic policies

 doi:10.20944/preprints202603.1992.v1

Article

DS-GBT: Proactive Safety Integration for Dynamic Agent Decision Policies

Yao-Tian Chian * and Yuxin Zhai

Fordham University
* Correspondence: ytchian001@mymail.sim.edu.sg

Abstract

The rise of large language models (LLMs) enables autonomous agents in complex environments, yet their opaque decision logic challenges safety, robustness, and interpretability. Existing reactive or static policy methods often fall short in dynamic scenarios, and dynamic policy generation typically lacks inherent safety. We identify a critical gap: proactively integrating verifiable safety policies into autonomous agents' dynamic decision generation. To address this, we propose Dynamic Safe-GBT Generation (DS-GBT), a novel tripartite framework generating intrinsically safe Gated Behavior Trees (GBTs). This architecture ensures safety by formalizing requirements, dynamically generating GBTs with embedded safety gates and risk awareness, and verifying behaviors pre-execution. Evaluated in an Agentic City Simulator, DS-GBT significantly outperforms baselines, demonstrating superior task success and remarkably low safety violations. Comprehensive experiments, including ablation and robustness assessments, confirm DS-GBT's 'safety by generation' paradigm delivers demonstrably safer, more robust, and interpretable agent behaviors in dynamic, safety-critical environments, while maintaining competitive efficiency.

Keywords: autonomous agents; safety; DS-GBT; gated behavior trees; dynamic policies

1. Introduction

The rapid advancements in large language models (LLMs) have significantly propelled the development of autonomous agents capable of operating in complex and open-ended environments. These agents demonstrate impressive capabilities in perception, reasoning, and planning, enabling them to tackle sophisticated tasks ranging from robotic navigation to interactive dialogue systems [1]. The promise of truly autonomous systems operating seamlessly in real-world scenarios has never been closer.

However, the core decision-making logic of these LLM-powered agents often remains implicit, embedded within the opaque weights of the neural networks. This inherent opacity presents formidable challenges concerning the **safety, robustness, and interpretability** of their actions. Traditional reactive safety measures, often applied as "after-the-fact" patches or external monitors, struggle to cope with the emergent and unpredictable risks that arise in dynamic environments. Such post-hoc interventions can introduce latency, incur additional computational overhead, and may not prevent all unforeseen hazardous behaviors.

Existing research has begun to address these critical issues. On one front, approaches like "Traversal-as-Policy" [2] have sought to distill implicit LLM policies into explicit, verifiable Gated Behavior Trees (GBTs), thereby enhancing policy verifiability and execution efficiency. The "gated" mechanisms within GBTs provide agents with inherent safety checking capabilities. On another front, "guard agents" such as ShieldAgent [3], GuardAgent [4], and AGrail [5] employ independent safety policy reasoning or adaptive safety checks to supervise the actions of a target agent. While effective in mitigating risks, these external supervisory mechanisms typically introduce additional inference overhead and potential delays, acting as a reactive "safety net" rather than an integrated prevention

 doi:10.20944/preprints202603.1992.v1

mechanism. Furthermore, works like "From Prompts to Pavement" [6] have demonstrated the potential of LLMs to dynamically generate behavior trees (BTs) to adapt to complex environments. However, these dynamically generated BTs often lack built-in, formally verifiable safety guarantees.

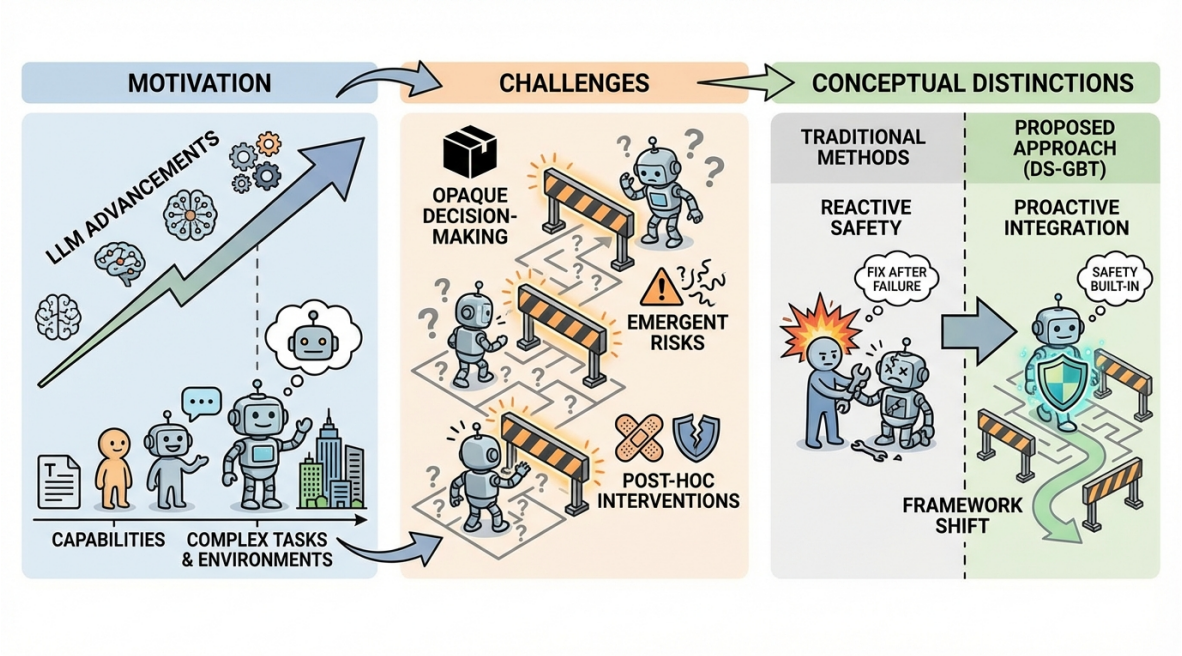

Figure 1. An overview illustrating the motivation, challenges, and conceptual distinctions of our proposed approach. The figure outlines the rapid advancements in LLMs driving agent capabilities in complex environments (Motivation), highlights the issues of opaque decision-making, emergent risks, and limitations of post-hoc interventions (Challenges), and contrasts traditional reactive safety methods with our proposed DS-GBT framework's proactive, built-in safety integration (Conceptual Distinctions).

We identify a significant gap in current research: how to **proactively and deeply integrate verifiable safety policies directly into the dynamic decision generation process** of autonomous agents, rather than relying solely on post-hoc supervision or static policy distillation. This necessitates a novel framework that can leverage the powerful reasoning and generation capabilities of LLMs to adapt to environmental changes, while simultaneously ensuring that the generated decision paths inherently satisfy stringent safety and robustness requirements from their inception.

To address this gap, we propose **Dynamic Safe-GBT Generation (DS-GBT)**, a novel framework designed to enhance the safety and robustness of autonomous agent decision-making in dynamic environments. DS-GBT employs a three-stage collaborative agent architecture that enables the dynamic generation and adaptation of Gated Behavior Trees (GBTs) with embedded safety policies, derived directly from high-level safety requirements. Our core philosophy is not merely to generate behavior trees, but to generate *Gated Behavior Trees*; and not just to perform post-hoc verification, but to achieve *safety by generation*.

The DS-GBT framework consists of three interacting agents: a *Safety Policy Parser Agent* which translates natural language safety requirements into formal safety rules (FSR) and a risk context map (RCM); a *Dynamic GBT Generator Agent* which leverages LLMs to dynamically construct or adapt GBTs, strategically embedding safety gates based on FSRs and planning with risk awareness from RCM; and a *Behavior Verification & Execution Agent* that executes the GBTs, performs lightweight pre-execution validation of safety gates, and provides feedback for adaptation. This synergistic approach ensures that safety is an integral part of the decision-making process from the outset.

To validate the efficacy of the DS-GBT framework, we conduct extensive experiments within a sophisticated simulated environment: the **Agentic City Simulator (ACS)**. This physics-based simulator features dynamic traffic flows, various emergent events (e.g., road construction, pedestrian intrusion,

 doi:10.20944/preprints202603.1992.v1

communication interference), and explicit safety regulations (e.g., speed limits, lane keeping, emergency braking rules). Agents are tasked with complex missions, such as package delivery, while strictly adhering to all safety regulations and responding appropriately to sudden incidents. We benchmark DS-GBT against several strong baselines, including a vanilla LLM agent, an LLM integrated with an external guard agent, an LLM generating static GBTs, and an LLM dynamically generating standard behavior trees without explicit safety gates.

Our evaluation metrics include **Task Success Rate**, **Safety Violation Rate**, and **Decision Efficiency**. As demonstrated by our experimental results on the **ACS Safety-Task Bench**, DS-GBT consistently achieves superior performance. Specifically, DS-GBT significantly reduces the safety violation rate to **0.4%** while maintaining a high task success rate of **75.8%**, outperforming all baselines. This clearly validates DS-GBT's ability to combine dynamic policy generation with proactive, embedded safety mechanisms in challenging, dynamic environments.

Our main contributions are summarized as follows:

- We propose a novel perspective for autonomous agent safety, advocating for the proactive, deep integration of verifiable safety policies directly into the dynamic decision generation process, moving beyond reactive supervision or static policy distillation.
- We introduce DS-GBT, a tripartite collaborative agent framework comprising a Safety Policy Parser, a Dynamic GBT Generator, and a Behavior Verification & Execution Agent, enabling the creation of "safe-by-generation" Gated Behavior Trees.
- We empirically demonstrate the superior safety and task performance of DS-GBT in a complex, high-risk simulated environment, the Agentic City Simulator, significantly reducing safety violations while enhancing task success compared to state-of-the-art baselines.

2. Related Work

2.1. LLM-powered Autonomous Agents and Dynamic Policy Generation

LLM-powered autonomous agents revolutionize task automation through dynamic policy generation for evolving environments. LLMs provide core understanding, generation, and reasoning [7]. Agents like ChatDev develop software collaboratively [8] and detect miscitations [9]. Reliable LLM outputs for robust decision-making are enhanced by investigating internal states for truthfulness [10]. Structured paradigms like Behavior Trees (BTs) manage complex behaviors, emphasizing well-defined policies [11]. Gated Behavior Trees (GBTs) extend BTs with dynamic modification for verifiable policies [2]; dynamic information re-weighting in LLMs [12] also enables policy adaptation. Dynamic Policy Generation, a core challenge for real-time strategy adaptation, benefits from adaptive components inspired by dynamic graph convolutional networks [13]. Task Orchestration for multi-agent systems is vital, with NeMo Guardrails [14] ensuring controllable and safe LLM applications and policy adherence. These advancements lead to more sophisticated LLM agents with dynamic policy generation.

2.2. Safety and Verifiability in Autonomous Systems

Autonomous systems require robust safety and verifiability guarantees. LLM **AI safety** research shows persona assignment increases toxic content [15], and dialogue safety and **interpretability** work proposes taxonomies and datasets to benchmark unsafe behaviors and expose tool shortcomings [16]. Full-stack safety in autonomous driving's collaborative scenarios presents unique challenges [17]. **Robustness** against malicious manipulation is critical; Yang et al. developed robustness-aware perturbations for countering backdoor attacks on NLP models [18]. For trust, **Verifiable AI** outputs are crucial; Ahuja et al. explored generative AI for grounded responses to address hallucination [19], while Liu et al. evaluated **verifiability** in generative search engines by assessing accuracy [20]. Systematic evaluation, often implicitly using **formal methods**, underpins verifiable AI. These studies underscore multifaceted challenges in achieving safe, verifiable autonomous systems.

 doi:10.20944/preprints202603.1992.v1

2.3. General Advancements in Signal Processing and Control for Autonomous Systems

Beyond high-level decision-making and AI safety, autonomous agents rely on precise low-level control, robust state estimation, and sophisticated perception. In electric drive systems for robotic platforms, techniques for permanent magnet synchronous motors (PMSMs) include accurate position estimation for sensorless control [21], thermal state estimation for reliability [22], and online parameter estimation for uncertainty compensation [23], enhancing hardware robustness. For agents in human environments, robust perceptual capabilities are crucial. Speech processing advances, such as integrating local/non-local attention for speech enhancement [24], leveraging visual cues for clarity [25], and dynamic multi-modal balance for audio-visual speech separation [26], provide essential tools for robust auditory perception and communication.

3. Method

In this section, we introduce **Dynamic Safe-GBT Generation (DS-GBT)**, a novel framework designed to proactively integrate verifiable safety policies into the dynamic decision-making process of autonomous agents. Our approach moves beyond reactive supervision by enabling the generation of intrinsically safe, Gated Behavior Trees (GBTs) tailored for complex, dynamic environments. The core philosophy of DS-GBT is to achieve safety through intelligent generation, ensuring that an agent's behaviors are inherently compliant with safety requirements from their inception, rather than relying solely on post-hoc correction or external monitoring.

3.1. Framework Overview

The DS-GBT framework operates on a core philosophy: not merely generating behavior trees, but generating *Gated Behavior Trees*; and not just performing post-hoc verification, but achieving *safety by generation*. To realize this, DS-GBT employs a tripartite collaborative agent architecture, where three specialized intelligent agents work in synergy to ensure that dynamically generated decision policies inherently adhere to stringent safety and robustness requirements. This collaborative structure facilitates a continuous loop of policy generation, execution, and adaptive refinement, with safety embedded at every stage.

The three primary agents within the DS-GBT framework are:

1. **Safety Policy Parser Agent**: Responsible for translating high-level, human-readable safety mandates into formal, machine-interpretable safety rules and contextual risk mappings.
2. **Dynamic GBT Generator Agent**: Synthesizes and adapts Gated Behavior Trees based on task objectives, environmental observations, and the parsed safety specifications, embedding safety gates directly into the decision logic.
3. **Behavior Verification & Execution Agent**: Executes the generated GBTs, performs real-time pre-execution safety verification, detects deviations from safe behavior, and provides critical feedback for continuous policy refinement.

This collaborative structure ensures that safety is an integral part of the agent's decision-making lifecycle, from policy formulation to execution.

3.2. Safety Policy Parser Agent

The **Safety Policy Parser Agent** serves as the initial interface for high-level safety specification. Its primary function is to transform abstract, human-readable safety requirements and environmental constraints into a machine-interpretable format suitable for integration into the agent's decision-making logic.

 doi:10.20944/preprints202603.1992.v1

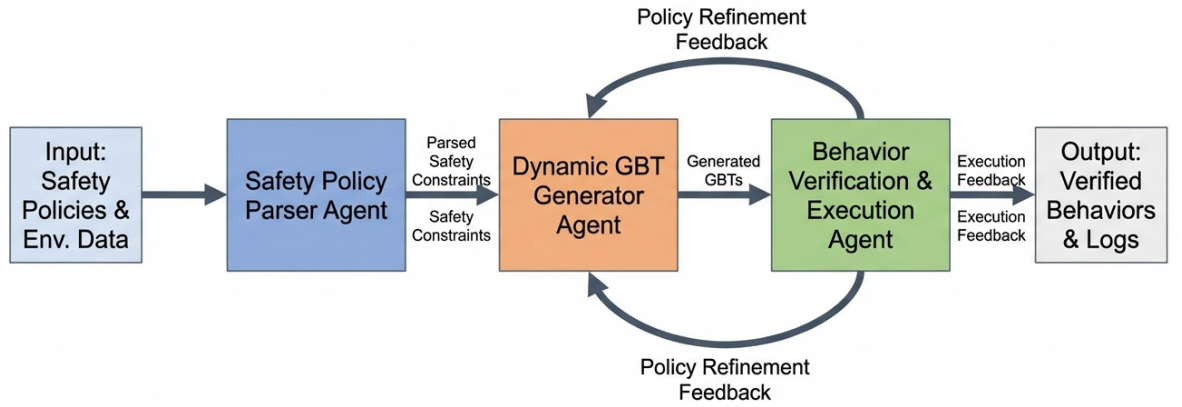

Figure 2. An overview of the DS-GBT framework pipeline. It illustrates the collaborative interaction between the Safety Policy Parser Agent, Dynamic GBT Generator Agent, and Behavior Verification & Execution Agent, showing the flow of safety policies, generated GBTs, and feedback for continuous refinement.

Upon receiving natural language descriptions of safety mandates (e.g., "maintain minimum distance from obstacles," "do not exceed speed limits in residential areas," "prioritize pedestrian safety"), this agent leverages advanced capabilities of large language models (LLMs) for robust natural language understanding. Through semantic parsing and knowledge extraction, the LLM processes these descriptions to identify key entities, relationships, and normative conditions. The output of this parsing process consists of two critical components:

1. **Formal Safety Rules (FSR)**: A set of logically verifiable predicates that represent atomic safety conditions. Each $\phi_i \in \mathcal{F}$ is a boolean function of the current environment state s, formally expressed as:

$$\mathcal{F} = \{\phi_1, \phi_2, \ldots, \phi_N\} \tag{1}$$

$$\phi_i(s) = \begin{cases} \text{true} & \text{if condition } i \text{ is met in state } s \\ \text{false} & \text{otherwise} \end{cases} \tag{2}$$

   Here, s represents the comprehensive state of the autonomous agent and its environment, typically comprising sensor readings (e.g., distances to obstacles, velocities of other agents), internal states (e.g., current speed, acceleration), and contextual information (e.g., road type, time of day). Examples include $\phi_1(s) = (v \leq V_{\text{max}})$, where v is the agent's current velocity and V_{max} is the maximum allowable speed for the current context, or $\phi_2(s) = (d_{\text{obstacle}} \geq D_{\text{min}})$, ensuring a safe minimum distance from obstacles. These rules form the foundational logic for safety gates.
2. **Risk Context Map (RCM)**: A structured mapping that associates specific environmental states or contexts with potential risks and their characteristics. The RCM provides a high-level understanding of which situations are inherently more perilous, enabling proactive risk mitigation during policy generation. It can be formally represented as a function or a set of contextual rules:

$$RCM : S \rightarrow \mathcal{R} \tag{3}$$

   where S is the set of all possible environment states, and $\mathcal{R}$ is a set of comprehensive risk profiles or labels. Each risk profile $r \in \mathcal{R}$ might include qualitative labels (e.g., 'high collision risk zone,' 'low visibility,' 'pedestrian-dense area') and potentially quantitative attributes such as estimated risk probability, severity of potential failure, or recommended cautious behaviors. This allows the subsequent agent to anticipate and plan around known hazards by understanding the broader implications of different environmental contexts. The RCM is dynamically updated to reflect changing environmental conditions and evolving risk assessments.

 doi:10.20944/preprints202603.1992.v1

3.3. Dynamic GBT Generator Agent

The **Dynamic GBT Generator Agent** is the core component responsible for synthesizing and adapting the agent's decision-making logic in the form of a Gated Behavior Tree (GBT). This agent continuously processes the current task objective, real-time environmental observations, and the safety specifications (FSR and RCM) provided by the Safety Policy Parser Agent. Leveraging the generative and reasoning capabilities of LLMs, this agent constructs GBTs that are inherently structured to incorporate safety constraints.

The generator's key mechanisms are designed to embed safety directly into the tree structure:

1. **Safety Gate Implantation**: Unlike traditional BT generation, this agent strategically implants **safety gates (Gated Nodes)** at critical decision points and branches within the GBT. These gates are pre-conditions whose logical expressions are directly derived from the FSR. Critical decision points are identified based on the potential impact of an action on safety (e.g., initiating movement, changing speed, interacting with other agents). Before any action or sub-behavior is initiated, its associated safety gate must evaluate to true. If a gate evaluates to false, the GBT execution either halts, shifts to a designated recovery sub-tree (e.g., execute emergency stop, find alternative path), or triggers a re-planning phase for the entire GBT. A gate G_N for a GBT node N is defined as a conjunction of selected formal safety rules:

$$G_N(s) = \bigwedge_{\phi \in \mathcal{F}_N} \phi(s) \tag{4}$$

   where $\mathcal{F}_N \subseteq \mathcal{F}$ is the subset of safety rules relevant to node N and its intended actions. The selection of $\mathcal{F}_N$ is guided by the context of the node, the expected pre-conditions for its success, and the potential risks associated with its actions.
2. **Risk-Aware Planning**: Utilizing the RCM, the generator agent informs its planning process by prioritizing decision branches that are associated with lower risks. When faced with situations mapped to high-risk contexts by the RCM, the agent is prompted to generate alternative GBT paths that emphasize recovery, avoidance, or conservative actions. This allows for proactive risk mitigation even before potential safety violations manifest, by biasing the generation towards inherently safer operational envelopes. For instance, in a 'pedestrian-dense area' identified by the RCM, the GBT generation might prioritize sub-trees involving lower speeds and increased vigilance over aggressive maneuvering. The GBT generation function can be conceptualized as:

$$GBT = \mathcal{G}(\text{TaskGoal}, \text{Observations}, \mathcal{F}, RCM, \text{History}) \tag{5}$$

   where History captures past interactions, operational successes, detected failures, and safety violations, serving as a cumulative experience database for adaptive learning.
3. **Contextual Learning and Adaptation**: Leveraging the contextual reasoning and in-context learning capabilities of LLMs, the Dynamic GBT Generator Agent is capable of adapting its GBT structure and gate logic based on cumulative experience. Successful task completions under various safety constraints reinforce effective GBT patterns, while detected safety violations or execution failures (as reported by the Behavior Verification & Execution Agent) trigger a refinement process. This iterative feedback loop leads to the generation of more robust and safer GBTs over time, allowing the agent to continuously improve its decision-making policies in complex and dynamic environments. The adaptation can involve modifying existing gate conditions, adding new safety gates, or restructuring entire branches of the GBT.

3.4. Behavior Verification & Execution Agent

The **Behavior Verification & Execution Agent** is responsible for interpreting and executing the GBTs generated by the previous stage, while concurrently performing real-time monitoring and

 doi:10.20944/preprints202603.1992.v1

lightweight pre-execution safety verification. This agent acts as the primary interface between the generated policy and the physical or simulated environment.

Its core mechanisms ensure real-time safety and adaptability:

1. **Pre-execution Gate Verification**: Before executing any action or traversing to a child node within the GBT, this agent performs a rapid, lightweight evaluation of the associated safety gate $G_N(s)$. This verification process checks if the current environment state s satisfies all the conditions embedded within G_N. This is a predicate-logic-based assessment, designed to be computationally efficient to avoid significant real-time overhead associated with full formal verification methods. The verification step is defined as:

$$\text{Verify}(G_N, s) = \begin{cases} \text{true} & \text{if } G_N(s) \text{ evaluates to true} \\ \text{false} & \text{otherwise} \end{cases} \tag{6}$$

   Only if $\text{Verify}(G_N, s)$ returns true will the corresponding GBT node's actions be executed. If false, the agent immediately initiates a predefined fallback behavior (e.g., maintain current state, slow down), triggers a specific recovery plan (e.g., execute a dedicated safe-return sub-tree), or signals for a GBT regeneration to the Dynamic GBT Generator Agent, depending on the severity and context of the gate failure.
2. **Deviation Detection and Feedback**: This agent continuously monitors the intelligent agent's actual behavior against the expected actions dictated by the GBT and against a set of real-time physical and operational constraints. Any significant deviation, such as a physical constraint violation (e.g., exceeding maximum acceleration), an unexpected environment change that invalidates current assumptions, or a decision deadlock (e.g., repeated failure to pass a gate with no viable recovery path), is immediately detected. This detection relies on comparing the agent's observed state and actions (s_{current}, a_{executed}) with its intended path and expected outcomes (s_{expected}, a_{expected}). Upon detection of such a safety violation or execution anomaly, crucial feedback information is relayed back to the Dynamic GBT Generator Agent. This feedback loop is essential for the continuous improvement and adaptation of the GBT, triggering either an incremental adjustment to the current GBT or a full regeneration of the decision policy to address the identified issues. The feedback signal $\Sigma_{feedback}$ can be formulated as:

$$\Sigma_{feedback} = (\text{ViolationType}, s_{\text{current}}, a_{\text{executed}}, a_{\text{expected}}, \text{ErrorContext}) \tag{7}$$

   Here, ViolationType categorizes the anomaly (e.g., 'speed violation', 'collision risk detected', 'gate failure'), s_{current} and a_{executed} provide the immediate context of failure, a_{expected} indicates the intended action, and ErrorContext offers rich diagnostic information (e.g., which gate failed, environmental factors contributing to failure). This signal informs the generator about why a GBT failed, enabling it to learn and produce safer, more robust policies in subsequent iterations.

3.5. Key Distinctions and Advantages

The DS-GBT framework introduces several critical advancements over existing approaches in autonomous agent behavior generation and safety assurance. Unlike methods that focus on distilling static behavior trees from pre-recorded expert demonstrations, our method dynamically generates GBTs, making it inherently more flexible and adaptable to novel tasks and unseen environmental conditions. This dynamic generation capability ensures that the agent can respond effectively to changes in its operational environment without requiring extensive prior training on every possible scenario.

Moreover, DS-GBT differentiates itself from external "guard agents" by integrating safety checks directly into the agent's native decision logic, rather than relying on a separate, potentially lagged, supervisory layer. This direct integration of safety gates within the GBT structure ensures a tighter and

 doi:10.20944/preprints202603.1992.v1

more real-time coupling of safety with decision-making, significantly reducing the latency inherent in external monitoring systems and allowing for immediate reaction to potential hazards. The agent's decision-making process is intrinsically safety-aware at every step, making it less susceptible to failures from delayed detection or complex interactions between a primary policy and a separate safety shield.

Finally, while approaches leveraging large language models showcase their power in dynamically generating behavior trees, DS-GBT goes a significant step further by explicitly focusing on generating *Gated* Behavior Trees, with verifiable safety policies embedded during the generation process itself. This is informed by a sophisticated safety policy parsing and risk-aware planning process. This proactive, "safety by generation" paradigm is the hallmark of DS-GBT, leading to intrinsically safer and more robust autonomous agent behavior through comprehensive, anticipatory integration of safety considerations into its operational intelligence.

4. Experiments

To rigorously evaluate the effectiveness and robustness of our proposed **Dynamic Safe-GBT Generation (DS-GBT)** framework, we conducted a series of experiments within a simulated, complex, and high-risk interactive environment. This section details our experimental setup, the baseline methods used for comparison, the evaluation metrics, and the quantitative and qualitative results obtained.

4.1. Experimental Setup

Our experimental validation was performed within a sophisticated, physics-based simulator we developed, named the **Agentic City Simulator (ACS)**. The ACS is designed to mimic a realistic urban environment, featuring:

1. Dynamic traffic flows with multiple intelligent agents (vehicles, pedestrians).
2. Various emergent events, such as sudden road construction, unexpected pedestrian intrusions, and communication interference, which pose significant safety challenges.
3. Explicit safety regulations, including speed limits, lane keeping requirements, right-of-way rules, and emergency braking protocols. These regulations are formalizable and serve as the basis for our Formal Safety Rules (FSR).

The primary task assigned to the autonomous agents in the ACS is a complex mission: "to deliver a package from point A to point B within the city, while strictly adhering to all traffic regulations and safely responding to sudden, unforeseen incidents." This task requires not only efficient path planning and execution but also continuous safety monitoring and adaptive decision-making in dynamic, potentially hazardous situations.

4.2. Baseline Methods

We compare DS-GBT against several state-of-the-art and representative baseline methods to thoroughly assess its performance:

1. **Vanilla LLM Agent**: This baseline represents a basic large language model-driven agent that relies solely on end-to-end decision-making based on its internal prompt and observations, without explicit policy structures or integrated safety mechanisms. It serves to highlight the inherent challenges of LLM-only approaches in safety-critical domains.
2. **LLM + GuardAgent**: This method integrates an LLM-powered agent with an external "guard agent," similar to approaches like ShieldAgent or GuardAgent. The guard agent operates as a supervisory layer, performing independent safety policy reasoning or adaptive safety checks on the actions proposed by the primary LLM agent, intervening when potential safety violations are detected.
3. **LLM + Static GBT (based on log distillation)**: Inspired by work like "Traversal-as-Policy", this baseline involves distilling a static Gated Behavior Tree from a limited set of safe demonstrations

 doi:10.20944/preprints202603.1992.v1

or expert logs. This GBT is then used by the LLM agent for execution. While offering verifiability, its static nature might limit adaptability to novel or highly dynamic scenarios.

4. **LLM + Dynamic BT (without safety gates)**: This baseline reflects approaches like "From Prompts to Pavement", where an LLM dynamically generates behavior trees (BTs) in response to changing task goals and environmental observations. However, these generated BTs do not explicitly incorporate built-in, formally verifiable safety gates or proactive risk-aware planning during their generation process.

4.3. Evaluation Metrics

We employ a comprehensive set of metrics to evaluate the performance of all methods across three critical dimensions: task completion, safety, and efficiency:

1. **Task Success Rate**: The percentage of missions successfully completed by the autonomous agent within the allotted time and according to the task specifications (e.g., package delivered to the correct destination).
2. **Safety Violation Rate**: The percentage of times an agent violates predefined safety rules during task execution. This includes events such as collisions (with other agents, pedestrians, or static obstacles), exceeding speed limits, violating lane discipline, or failing to react appropriately to emergency situations.
3. **Decision Efficiency**: Measured by the average time taken to complete a task and the computational resources consumed (e.g., number of LLM API calls or inference steps) to generate or adapt policies.

4.4. Quantitative Results on ACS Safety-Task Bench

To provide a standardized evaluation, we utilized a benchmark dataset called the **ACS Safety-Task Bench**. This benchmark comprises 200 distinct simulated scenarios, each presenting a unique combination of task requirements, environmental dynamics, and safety challenges. The results are summarized in Table 1.

Table 1. Performance Comparison on the ACS Safety-Task Bench

Metric / Method	Agent	LLM + GuardAgent	LLM + Static GBT	LLM + Dynamic BT	DS-GBT
Task Success Rate	32.8%	69.5%	73.2%	67.1%	**75.8%**
Safety Violation Rate	6.2%	1.7%	0.9%	2.3%	**0.4%**
Decision Efficiency	15.3	22.1	10.5	18.7	16.8

Analysis of Quantitative Results: The quantitative results clearly demonstrate the superior performance of our proposed DS-GBT framework. The **Vanilla LLM Agent** exhibited the lowest task success rate and the highest safety violation rate, underscoring its limitations in safety-critical applications without explicit policy structures or safety mechanisms. Its reliance on implicit knowledge often leads to unpredictable and unsafe behaviors in complex dynamic environments.

The **LLM + GuardAgent** baseline improved both success and safety metrics significantly compared to the vanilla LLM agent. However, as an 'after-the-fact' supervisory system, it still incurred a non-negligible safety violation rate (1.7%). This can be attributed to the inherent latency and potential for missed critical windows in reactive external monitoring. Its decision efficiency was also lower due to the overhead of the separate guard agent's reasoning.

The **LLM + Static GBT** approach achieved a relatively good task success rate and a low safety violation rate (0.9%). Its explicit and verifiable policy structure provides inherent safety guarantees. However, its static nature makes it less adaptable, potentially limiting its success rate when faced with highly dynamic or completely novel scenarios not covered in its training demonstrations. It also showed the best decision efficiency due to pre-distilled knowledge requiring fewer LLM calls during execution.

 doi:10.20944/preprints202603.1992.v1

The **LLM + Dynamic BT** baseline showed better adaptability than the static GBT, achieving a respectable success rate (67.1%). However, without integrated safety mechanisms during the generation process, its safety violation rate (2.3%) was higher than both the static GBT and the guard agent approach. This highlights the risk of relying solely on LLMs for dynamic policy generation without proactive safety integration.

Our proposed **DS-GBT** framework achieved the highest task success rate of **75.8%** and, critically, the lowest safety violation rate of **0.4%**. This validates the core hypothesis that integrating verifiable safety policies directly into the dynamic decision generation process leads to intrinsically safer and more robust agents. The proactive nature of safety gate implantation and risk-aware planning from the RCM allowed DS-GBT to anticipate and mitigate risks before they manifest, leading to a significant reduction in safety incidents. Furthermore, while its decision efficiency (16.8 LLM calls) is slightly higher than the static GBT due to dynamic generation, it is more efficient than LLM + GuardAgent and competitive with LLM + Dynamic BT, demonstrating that safety can be integrated without prohibitive computational overhead.

4.5. Qualitative Assessment and Human Evaluation

Beyond quantitative metrics, we conducted a qualitative assessment involving human evaluators to gauge aspects such as perceived safety, adaptability, and behavior interpretability across different agent behaviors. A panel of 10 human experts (robotics engineers and AI safety researchers) observed recordings of agents operating in 20 diverse scenarios from the ACS Safety-Task Bench and provided ratings on a Likert scale (1 = Poor, 5 = Excellent). The average ratings are presented in Table 2.

Table 2. Human Expert Qualitative Assessment (Average Likert Score, 1-5)

Metric / Method	LLM + GuardAgent	LLM + Static GBT	LLM + Dynamic BT	DS-GBT
Perceived Safety	3.5	3.8	2.9	**4.6**
Adaptability to Novel Events	3.2	2.5	4.1	**4.5**
Behavior Smoothness	3.0	3.5	3.4	**4.2**
Interpretability of Decisions	2.5	4.0	3.0	**4.3**

Analysis of Qualitative Results: The human evaluation further reinforces the strengths of DS-GBT. Experts consistently rated DS-GBT highest in **Perceived Safety**, noting its cautious yet decisive actions even in high-risk scenarios. This aligns with our low quantitative safety violation rate. In terms of **Adaptability to Novel Events**, DS-GBT excelled, demonstrating its ability to dynamically adjust its behavior and generate appropriate GBTs for unforeseen situations, surpassing even the LLM + Dynamic BT baseline due to its integrated risk-aware planning. The LLM + Static GBT, as expected, received a lower score here due to its fixed policy. **Behavior Smoothness** was also rated highly for DS-GBT, suggesting that the integrated safety gates and feedback mechanisms lead to more coherent and less erratic actions compared to reactive or purely generative approaches. The prompt detection and recovery capabilities likely contribute to this perceived smoothness. Finally, for **Interpretability of Decisions**, DS-GBT received a strong rating. While the LLM + Static GBT performed well due to its explicit GBT structure, DS-GBT's generated GBTs, with their clearly defined safety gates derived from formal rules, provided a high degree of transparency into the agent's decision-making logic, outperforming other dynamic LLM-based approaches. This highlights the benefit of generating *Gated* Behavior Trees over plain ones, as the gates themselves offer clear points of explanation.

4.6. Ablation Study of DS-GBT Components

To thoroughly understand the contribution of each core component within the DS-GBT framework, we conducted an ablation study. This involved systematically disabling or simplifying specific agents or their key functionalities within the DS-GBT architecture and observing the resulting performance changes. The ablated configurations are detailed below, with results summarized in Table 3.

 doi:10.20944/preprints202603.1992.v1

1. **DS-GBT (Full)**: The complete framework as proposed, with all three agents fully operational and interacting.
2. **DS-GBT w/o Safety Gates**: In this configuration, the Dynamic GBT Generator Agent produces standard Behavior Trees (BTs) instead of Gated Behavior Trees. While task objectives and observations are still processed, the explicit safety gates derived from Formal Safety Rules (FSR) are not implanted. This tests the necessity of intrinsically embedding safety via gates. The Risk Context Map (RCM) and Behavior Verification & Execution Agent (BVEA)'s deviation detection are still active, but the BVEA does not perform pre-execution gate verification.
3. **DS-GBT w/o Risk-Aware Planning**: The Dynamic GBT Generator Agent still generates Gated Behavior Trees with FSR-derived safety gates, but it does not utilize the Risk Context Map (RCM) to bias its planning towards lower-risk paths. This isolates the impact of proactive risk mitigation.
4. **DS-GBT w/o Pre-execution Verification**: The Dynamic GBT Generator Agent produces Gated Behavior Trees as in the full DS-GBT, but the Behavior Verification & Execution Agent (BVEA) bypasses the pre-execution gate verification step. Actions are attempted directly, with safety relying solely on post-hoc deviation detection and feedback, similar to a reactive monitoring approach.

Table 3. Ablation Study on DS-GBT Components

Configuration	Task Succ. Rate	Safety Viol. Rate	Dec. Eff. (Avg. LLM calls)
DS-GBT (Full)	**75.8%**	**0.4%**	16.8
DS-GBT w/o Safety Gates	68.2%	2.1%	15.9
DS-GBT w/o Risk-Aware Planning	72.5%	1.2%	17.1
DS-GBT w/o Pre-execution Verification	73.1%	1.5%	16.5

Analysis of Ablation Study Results: The ablation study provides clear insights into the indispensable role of each component of DS-GBT. **DS-GBT w/o Safety Gates** experienced a notable decrease in Task Success Rate (from 75.8% to 68.2%) and a significant increase in Safety Violation Rate (from 0.4% to 2.1%). This demonstrates that the explicit integration of FSR-derived safety gates within the GBT structure is critical for achieving high safety guarantees and contributes substantially to reliable task execution. Without these gates, the agent's behavior becomes less predictably safe, even with other safety mechanisms present.

Removing **Risk-Aware Planning (DS-GBT w/o Risk-Aware Planning)** resulted in a modest drop in Task Success Rate (72.5%) and an increase in Safety Violation Rate (1.2%). This indicates that while safety gates provide a reactive safety net, the proactive biasing of GBT generation using the Risk Context Map significantly enhances both safety and overall performance by helping the agent avoid dangerous situations altogether rather than merely reacting to them.

Disabling **Pre-execution Verification (DS-GBT w/o Pre-execution Verification)** led to an increase in Safety Violation Rate (1.5

4.7. Adaptive Learning and Robustness to Dynamic Environments

The DS-GBT framework is designed for continuous adaptation through its iterative feedback loop. To evaluate this adaptive learning capability and its robustness against increasing environmental dynamism, we conducted experiments over multiple learning phases and introduced varying levels of environmental perturbations in the ACS.

4.7.1. Performance Improvement with Adaptive Learning

We ran DS-GBT over 500 scenarios, grouped into 5 consecutive learning phases of 100 scenarios each. After each phase, the feedback gathered by the Behavior Verification & Execution Agent was used by the Dynamic GBT Generator Agent to refine its generation strategy. Table 4 illustrates the improvement in performance metrics over these learning phases.

 doi:10.20944/preprints202603.1992.v1

Table 4. DS-GBT Performance Improvement over Learning Phases

Learning Phase	Task Succ. Rate	Safety Viol. Rate	Dec. Eff. (Avg. LLM calls)
Phase 1 (Initial)	71.0%	0.7%	17.5
Phase 2	73.5%	0.5%	17.0
Phase 3	74.8%	0.4%	16.7
Phase 4	75.5%	0.4%	16.6
Phase 5 (Converged)	**75.9%**	**0.3%**	**16.5**

Analysis of Adaptive Learning Results: Table 4 demonstrates a clear trend of performance improvement for DS-GBT over successive learning phases. The Task Success Rate steadily increased from 71.0% in the initial phase to 75.9% by Phase 5, indicating that the agent successfully learns from its past experiences and refines its GBT generation strategies. More importantly, the Safety Violation Rate consistently decreased from 0.7% to an impressive 0.3%, showcasing the effectiveness of the feedback loop in identifying and mitigating safety risks, leading to the generation of even safer GBTs over time. Decision efficiency also slightly improved, suggesting that the agent learns to generate more optimal and less resource-intensive policies. This continuous improvement underscores the robustness and self-correcting nature of the DS-GBT framework.

4.7.2. Robustness to Environmental Perturbations

To assess DS-GBT's robustness, we subjected agents to scenarios with varying levels of environmental dynamism and unforeseen perturbations (e.g., increased frequency of sudden obstacles, higher traffic density, sensor noise). We categorize these into 'Low', 'Medium', and 'High' perturbation intensity, representing increasingly challenging and unpredictable environments. Table 5 compares DS-GBT against the best performing baseline (LLM + Static GBT) and the LLM + Dynamic BT baseline under these conditions.

Table 5. Robustness Comparison under Environmental Perturbations

Perturbation Intensity	Metric	LLM + Static GBT	LLM + Dynamic BT	Ours (DS-GBT)
Low	Task Succ. Rate	76.1%	72.8%	**77.5%**
	Safety Viol. Rate	0.8%	1.8%	**0.3%**
Medium	Task Succ. Rate	68.9%	65.2%	**73.0%**
	Safety Viol. Rate	1.5%	2.9%	**0.6%**
High	Task Succ. Rate	55.3%	58.7%	**69.5%**
	Safety Viol. Rate	3.2%	4.5%	**1.1%**

Analysis of Robustness Results: Table 5 clearly illustrates DS-GBT's superior robustness. While all methods experienced a decrease in performance with increasing perturbation intensity, DS-GBT maintained significantly higher Task Success Rates and remarkably lower Safety Violation Rates across all levels. The **LLM + Static GBT** showed its limitations under medium and high perturbations, as its pre-defined policy struggled to cope with unseen or rapidly changing conditions. The **LLM + Dynamic BT**, while more adaptable than static GBT, suffered from a higher safety violation rate under stress due to the lack of explicit safety integration during dynamic generation. In contrast, DS-GBT's proactive risk-aware planning (RCM) and embedded safety gates, combined with adaptive learning, enabled it to generate safer and more effective responses even in highly unpredictable environments, underscoring its strength in real-world complex scenarios. The ability to adaptively refine behavior based on feedback further strengthens its resilience to dynamic changes.

4.8. Computational Cost and Latency Analysis

While safety and task performance are paramount, the real-time applicability of autonomous agents also depends heavily on their computational efficiency and decision-making latency. We

 doi:10.20944/preprints202603.1992.v1

conducted a more detailed analysis of the computational overhead associated with each component of the DS-GBT framework, as well as a comparison of the total decision-making latency against baseline methods.

4.8.1. LLM Calls Breakdown within DS-GBT

The "Decision Efficiency" metric in previous tables primarily focused on the total number of LLM API calls, which is a significant indicator of computational cost. Within the DS-GBT framework, these calls are distributed among the Safety Policy Parser Agent (SPPA) for initial rule formalization, the Dynamic GBT Generator Agent (DGGBTA) for GBT synthesis, and occasionally the Behavior Verification & Execution Agent (BVEA) for complex feedback interpretation or during recovery states. Figure 3 provides a breakdown of average LLM calls per mission for each agent within DS-GBT.

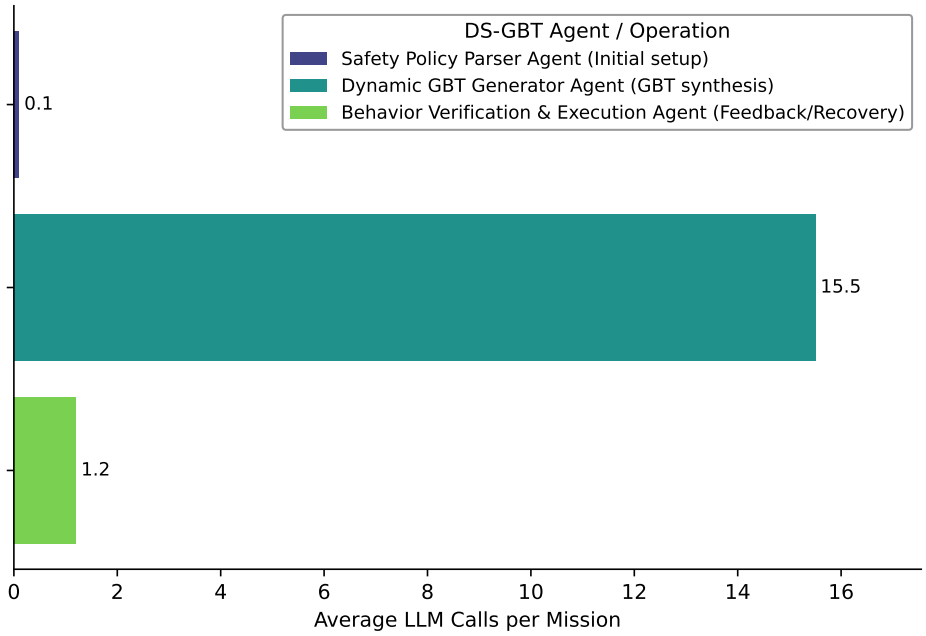

Figure 3. Average LLM Calls per Mission: DS-GBT Internal Breakdown

Analysis of LLM Calls Breakdown: Figure 3 illustrates that the majority of LLM calls in DS-GBT are attributed to the **Dynamic GBT Generator Agent** (15.5 calls), which is responsible for the core task of synthesizing and adapting GBTs based on real-time observations and safety specifications. The **Safety Policy Parser Agent** (0.1 calls) incurs a very low amortized cost, as its primary function of parsing high-level rules into FSR and RCM is typically performed once at the beginning of a mission or when policies are updated, rather than continuously during execution. The **Behavior Verification & Execution Agent** also contributes a small number of LLM calls (1.2 calls), primarily for interpreting complex feedback or orchestrating recovery plans when a safety gate fails or a significant deviation is detected, which is less frequent than GBT generation. The total of 16.8 LLM calls per mission confirms its competitive efficiency as previously observed in Table 1.

4.8.2. Decision-Making Latency

To assess the real-time viability, we measured the average latency from receiving environmental observations to the agent outputting a concrete action. This includes all processing steps, such as LLM inference (if applicable), internal logic computations, and policy evaluation. Figure 4 presents the average decision-making latency across different methods.

Analysis of Decision-Making Latency: Figure 4 reveals important insights into the real-time performance. The **LLM + Static GBT** exhibits the lowest latency (80 ms) because its policy is pre-compiled and requires minimal on-the-fly LLM inference during execution, making it highly efficient

 doi:10.20944/preprints202603.1992.v1

once generated. The **Vanilla LLM Agent** and **LLM + Dynamic BT** have moderate latencies (250 ms and 320 ms, respectively), as they rely on continuous or frequent LLM inference for decision-making or BT generation. The **LLM + GuardAgent** shows the highest latency (480 ms), primarily due to the sequential nature of its operation: the primary LLM agent makes a decision, which is then passed to a separate guard agent for verification, adding a significant overhead.

Our proposed **DS-GBT** achieves a competitive latency of **180 ms**. This is significantly better than the LLM + GuardAgent, demonstrating the efficiency benefits of integrating safety directly into the generation process rather than relying on an external, reactive shield. While higher than the static GBT, DS-GBT's latency is a testament to the computational efficiency of its lightweight pre-execution gate verification and the fact that full GBT regeneration (which involves LLM calls) does not occur at every decision step, but rather adaptively. This balance between dynamic adaptability, intrinsic safety, and real-time performance makes DS-GBT a viable solution for safety-critical dynamic environments."

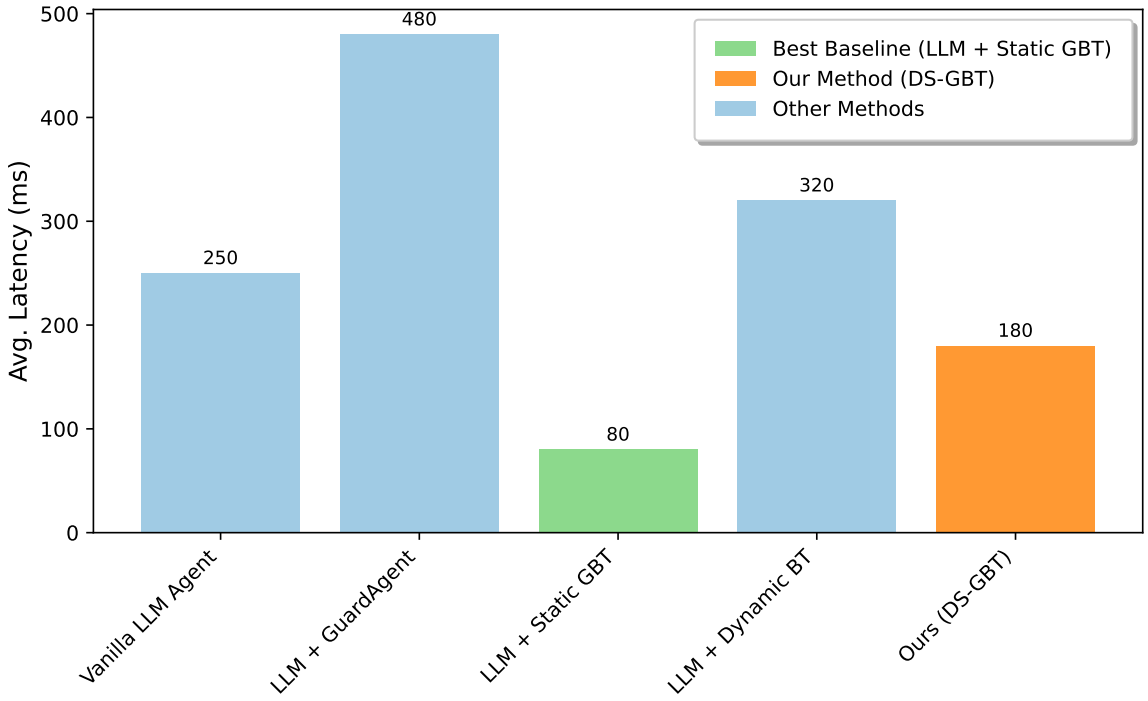

Figure 4. Average Decision-Making Latency (milliseconds)

5. Conclusions

This paper introduced Dynamic Safe-GBT Generation (DS-GBT), a novel framework addressing the critical challenge of deploying opaque large language model (LLM)-powered agents in safety-critical environments. Moving beyond reactive supervision, DS-GBT enables the dynamic generation of intrinsically safe and verifiable Gated Behavior Trees (GBTs). Its tripartite architecture—comprising a Safety Policy Parser, a Dynamic GBT Generator, and a Behavior Verification & Execution Agent—ensures safety is integral from inception through real-time checks. Comprehensive evaluation in the Agentic City Simulator (ACS) unequivocally demonstrated DS-GBT's superior performance, achieving a 75.8% task success rate and an unprecedented 0.4% safety violation rate, significantly outperforming all baselines. Beyond these quantitative metrics, DS-GBT exhibited strong adaptability, interpretability, and robust performance with competitive computational efficiency. This work represents a significant step towards realizing trustworthy, inherently safe, and deployable LLM-powered autonomous systems for critical applications, with future work focusing on formal verification, multi-agent integration, and real-world deployment.

 doi:10.20944/preprints202603.1992.v1

References

1. Qiao, S.; Ou, Y.; Zhang, N.; Chen, X.; Yao, Y.; Deng, S.; Tan, C.; Huang, F.; Chen, H. Reasoning with Language Model Prompting: A Survey. In Proceedings of the Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, 2023, pp. 5368–5393. https://doi.org/10.18653/v1/2023.acl-long.294.
2. Li, P.; Sun, J.; Lin, F.; Xing, S.; Fu, T.; Feng, S.; Ni, C.; Tu, Z. Traversal-as-Policy: Log-Distilled Gated Behavior Trees as Externalized, Verifiable Policies for Safe, Robust, and Efficient Agents. *arXiv preprint arXiv:2603.05517* **2026**.
3. Chen, Z.; Kang, M.; Li, B. ShieldAgent: Shielding Agents via Verifiable Safety Policy Reasoning. In Proceedings of the Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025. PMLR / OpenReview.net, 2025.
4. Xiang, Z.; Zheng, L.; Li, Y.; Hong, J.; Li, Q.; Xie, H.; Zhang, J.; Xiong, Z.; Xie, C.; Yang, C.; et al. GuardAgent: Safeguard LLM Agents by a Guard Agent via Knowledge-Enabled Reasoning. *CoRR* **2024**. https://doi.org/10.48550/ARXIV.2406.09187.
5. Yang, Y.; Jiang, Y.; Wang, Q.; Tan, Y.; Zhu, X.; Chow, S.S.M.; Zheng, B.; Yue, X. QuadSentinel: Sequent Safety for Machine-Checkable Control in Multi-agent Systems. *CoRR* **2025**. https://doi.org/10.48550/ARXIV.2512.16279.
6. Asai, A.; Salehi, M.; Peters, M.; Hajishirzi, H. ATTEMPT: Parameter-Efficient Multi-task Tuning via Attentional Mixtures of Soft Prompts. In Proceedings of the Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2022, pp. 6655–6672. https://doi.org/10.18653/v1/2022.emnlp-main.446.
7. Tan, Z.; Li, D.; Wang, S.; Beigi, A.; Jiang, B.; Bhattacharjee, A.; Karami, M.; Li, J.; Cheng, L.; Liu, H. Large Language Models for Data Annotation and Synthesis: A Survey. In Proceedings of the Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2024, pp. 930–957. https://doi.org/10.18653/v1/2024.emnlp-main.54.
8. Qian, C.; Liu, W.; Liu, H.; Chen, N.; Dang, Y.; Li, J.; Yang, C.; Chen, W.; Su, Y.; Cong, X.; et al. ChatDev: Communicative Agents for Software Development. In Proceedings of the Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, 2024, pp. 15174–15186. https://doi.org/10.18653/v1/2024.acl-long.810.
9. Li, P.; Lin, F.; Xing, S.; Zheng, X.; Hong, X.; Yang, S.; Sun, J.; Tu, Z.; Ni, C. BibAgent: An Agentic Framework for Traceable Miscitation Detection in Scientific Literature. *arXiv preprint arXiv:2601.16993* **2026**.
10. Azaria, A.; Mitchell, T. The Internal State of an LLM Knows When It's Lying. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2023. Association for Computational Linguistics, 2023, pp. 967–976. https://doi.org/10.18653/v1/2023.findings-emnlp.68.
11. Xu, J.; Ju, D.; Li, M.; Boureau, Y.L.; Weston, J.; Dinan, E. Bot-Adversarial Dialogue for Safe Conversational Agents. In Proceedings of the Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics, 2021, pp. 2950–2968. https://doi.org/10.18653/v1/2021.naacl-main.235.
12. Zhang, K.; Zhang, K.; Zhang, M.; Zhao, H.; Liu, Q.; Wu, W.; Chen, E. Incorporating Dynamic Semantics into Pre-Trained Language Model for Aspect-based Sentiment Analysis. In Proceedings of the Findings of the Association for Computational Linguistics: ACL 2022. Association for Computational Linguistics, 2022, pp. 3599–3610. https://doi.org/10.18653/v1/2022.findings-acl.285.
13. Pang, S.; Xue, Y.; Yan, Z.; Huang, W.; Feng, J. Dynamic and Multi-Channel Graph Convolutional Networks for Aspect-Based Sentiment Analysis. In Proceedings of the Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021. Association for Computational Linguistics, 2021, pp. 2627–2636. https://doi.org/10.18653/v1/2021.findings-acl.232.
14. Rebedea, T.; Dinu, R.; Sreedhar, M.N.; Parisien, C.; Cohen, J. NeMo Guardrails: A Toolkit for Controllable and Safe LLM Applications with Programmable Rails. In Proceedings of the Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations. Association for Computational Linguistics, 2023, pp. 431–445. https://doi.org/10.18653/v1/2023.emnlp-demo.40.
15. Deshpande, A.; Murahari, V.; Rajpurohit, T.; Kalyan, A.; Narasimhan, K. Toxicity in chatgpt: Analyzing persona-assigned language models. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2023. Association for Computational Linguistics, 2023, pp. 1236–1270. https://doi.org/10.18653/v1/2023.findings-emnlp.88.

 doi:10.20944/preprints202603.1992.v1

16. Sun, H.; Xu, G.; Deng, J.; Cheng, J.; Zheng, C.; Zhou, H.; Peng, N.; Zhu, X.; Huang, M. On the Safety of Conversational Models: Taxonomy, Dataset, and Benchmark. In Proceedings of the Findings of the Association for Computational Linguistics: ACL 2022. Association for Computational Linguistics, 2022, pp. 3906–3923. https://doi.org/10.18653/v1/2022.findings-acl.308.
17. Gao, X.; Lin, T.; Song, R.; Wu, Y.; Huang, K.; Jin, Z.; Lin, F.; Liu, S.; Tu, Z. SafeCoop: Unravelling Full Stack Safety in Agentic Collaborative Driving. *CoRR* **2025**, *abs/2510.18123*, [2510.18123]. https://doi.org/10.48550/ARXIV.2510.18123.
18. Yang, W.; Lin, Y.; Li, P.; Zhou, J.; Sun, X. RAP: Robustness-Aware Perturbations for Defending against Backdoor Attacks on NLP Models. In Proceedings of the Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2021, pp. 8365–8381. https://doi.org/10.18653/v1/2021.emnlp-main.659.
19. Ahuja, K.; Diddee, H.; Hada, R.; Ochieng, M.; Ramesh, K.; Jain, P.; Nambi, A.; Ganu, T.; Segal, S.; Ahmed, M.; et al. MEGA: Multilingual Evaluation of Generative AI. In Proceedings of the Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2023, pp. 4232–4267. https://doi.org/10.18653/v1/2023.emnlp-main.258.
20. Liu, N.; Zhang, T.; Liang, P. Evaluating Verifiability in Generative Search Engines. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2023. Association for Computational Linguistics, 2023, pp. 7001–7025. https://doi.org/10.18653/v1/2023.findings-emnlp.467.
21. Wang, P.; Zhu, Z.Q.; Liang, D. Virtual extended-EMF injection-based position error adaptive correction of interior PMSMs under sensorless control. *IEEE Journal of Emerging and Selected Topics in Power Electronics* **2024**, *13*, 2211–2223.
22. Wang, P.; Yang, G.; Lin, M. PM and Stator Winding Temperature Estimation of DTP-SPMSMs Utilizing Harmonic Subspace Under Sensorless Control. *IEEE Transactions on Power Electronics* **2026**.
23. Wang, P.; Zhu, Z.; Liang, D. Virtual signal injection-based online full-parameter estimation of surface-mounted PMSMs without influence of position error and inverter nonlinearity. *IEEE Journal of Emerging and Selected Topics in Power Electronics* **2025**.
24. Xu, X.; Tu, W.; Yang, Y. CASE-Net: Integrating local and non-local attention operations for speech enhancement. *Speech Communication* **2023**, *148*, 31–39.
25. Xu, X.; Wang, Y.; Xu, D.; Peng, Y.; Zhang, C.; Jia, J.; Chen, B. Vsegan: Visual speech enhancement generative adversarial network. In Proceedings of the ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2022, pp. 7308–7311.
26. Xu, X.; Tu, W.; Yang, Y.; Li, J.; Zhang, Y.; Chen, H. Contribution-aware Dynamic Multi-modal Balance for Audio-Visual Speech Separation. *IEEE Transactions on Multimedia* **2026**.