Symbolic Runtime Verification and Adaptive Decision-Making for Robot-Assisted Dressing

Yasmin Rafiq[1][0009−0006−1364−9820], Gricel Vázquez[2][0000−0003−4886−5567], Radu Calinescu[2][0000−0002−2678−9260], and Sanja Dogramadzi[3][0000−0002−0009−7522]
Robert M Hierons[4][0000−0002−4771−1446]

[1] Department of Computer Science, University of Manchester, UK
yasmeen.rafiq@manchester.ac.uk
[2] Department of Computer Science, University of York, UK
{gricel.vazquez, radu.calinescu}@york.ac.uk
[3] School of Electrical and Electronic Engineering, The University of Sheffield, UK
s.dogramadzi@sheffield.ac.uk
[4] School of Computer Science, The University of Sheffield, UK
r.hierons@sheffield.ac.uk

Abstract. We present a control framework for robot-assisted dressing that augments low-level hazard response with runtime monitoring and formal verification. A parametric discrete-time Markov chain (pDTMC) models the dressing process, while Bayesian inference dynamically updates this pDTMC's transition probabilities based on sensory and user feedback. Safety constraints from hazard analysis are expressed in probabilistic computation tree logic, and symbolically verified using a probabilistic model checker. We evaluate reachability, cost, and reward trade-offs for garment-snag mitigation and escalation, enabling real-time adaptation. Our approach provides a formal yet lightweight foundation for safety-aware, explainable robotic assistance.

Keywords: Robot-Assisted Dressing · Symbolic Runtime Verification · Probabilistic Model Checking · Human-Robot Interaction · Adaptive Control.

1 Introduction

As assistive robots become more integrated into daily life, ensuring safety and reliability during physical human-robot interaction (pHRI) remains a critical challenge [5, 7, 9, 12]. In tasks such as Robot-Assisted Dressing (RAD) [4, 14], the robot must perform close-contact assistance in coordination with a human, where garment dynamics, force disturbances, and user motion introduce runtime uncertainty. While low-level control strategies (e.g., force thresholds, compliant motions, and speed modulation) can mitigate immediate issues, they operate reactively and lack long-term task-level reasoning. This motivates the need for high-level control strategies that reason over future states and adapt using runtime feedback.

This paper presents a high-level control framework for safety-critical human-robot interaction, combining symbolic verification with runtime adaptation. While demonstrated through Robo-Assisted Dressing (RAD), the approach is methodological and generalisable to other uncertain human-in-the-loop systems.

We propose a high-level control framework that integrates parametric discrete-time Markov chains (pDTMCs) [13], Bayesian inference [2], and symbolic runtime verification [17]. Safety and performance requirements, derived from a hazard analysis [8], are formally specified using probabilistic computation tree logic (PCTL) [6]. Symbolic expressions for these requirements are precomputed using the probabilistic model checkers PRISM [16] and PARAM [13], and are evaluated at runtime using parameters updated via Bayesian inference.

The main contributions of our paper are:

- A high-level formal verification framework using a pDTMC model guided by hazard analysis.
- Integration of runtime Bayesian inference to update model parameters using sensory data and user feedback.
- Symbolic runtime verification using closed-form expressions generated with PRISM and PARAM.
- Symbolic evaluation of safety, cost, and reward properties to support dynamic adaptation and decision-making.

The RAD case study serves to ground our contribution in a real-world safety-critical setting, illustrating how the proposed framework supports runtime verification and decision-making under uncertainty. The rest of the paper is structured as follows. Section 2 reviews related work. Section 3 describes the system architecture. Section 4 defines the pDTMC model and its formal requirements, and Section 5 presents its symbolic analysis. Section 6 discusses practical insights and limitations. Section 7 summarises our results and outlines future work.

2 Related Work

This section reviews research at the intersection of probabilistic verification, runtime adaptation, and uncertainty quantification in robotics, particularly under pHRI settings.

Probabilistic Verification and Model Checking Probabilistic model checking has been widely applied in robotics to provide formal guarantees over safety, reachability, and performance requirements under uncertainty. Tools such as PRISM [16] support the analysis of Discrete-Time Markov Chains (DTMCs) and verification of requirements specified in PCTL [6].

Zhao et al. [22] applied probabilistic verification to autonomous systems in extreme environments, but their approach focused on offline analysis without runtime adaptation. Similarly, Gleirscher et al. [10, 11] verified safety controllers for collaborative robots but did not incorporate runtime feedback or symbolic reasoning in human-interactive scenarios.

Bayesian Inference for Uncertainty Quantification Bayesian learning has been used to adapt robot behaviour based on noisy perception and dynamic environments. Zhao et al. [21] proposed a framework that combines Bayesian inference with formal verification to improve robustness in autonomous systems. Similarly, Calinescu et al. present a tool for the verification and Bayesian-learned parameter inference of probabilistic world models [3]. However, they did not consider symbolic runtime evaluation or human-in-the-loop feedback.

Runtime Monitoring and Adaptive Control Runtime monitoring techniques are often used for anomaly detection or logging, whereas runtime verification aims to formally check execution traces against temporal specifications. Kirca et al. [15] proposed a runtime monitoring framework for anomaly detection using logs, but without safety property verification. Li et al. [18] introduced a probabilistic motion planning approach that models human uncertainty, but it lacks continuous verification of safety requirements during task execution.

Our Contribution in Context: In contrast to the above, our work integrates probabilistic model checking, runtime Bayesian inference, and hazard-driven safety property verification in a human-in-the-loop context. We introduce a pDTMC model whose transition probabilities are updated at runtime using Bayesian learning. Precomputed symbolic expressions for PCTL properties are evaluated on-the-fly through parameter substitution, enabling lightweight runtime verification without re-invoking the model checker. Our framework bridges the gap between high-assurance probabilistic verification and adaptive human-aware control. It enables formal safety reasoning in real time while responding to user input, physical interaction events and task-level uncertainty during robot-assisted dressing.

3 System Overview and Modelling Framework

3.1 Robot-Assisted Dressing Context

Robot-Assisted Dressing (RAD) involves a robotic manipulator physically assisting a user in donning garments (e.g., jacket). The task is inherently collaborative, requiring safe human-robot interaction, real-time trajectory execution, and continuous hazard monitoring to ensure comfort and prevent failure. Dynamic factors such as user movement, garment deformation, and variable force profiles can lead to hazards including garment snagging or user discomfort. A robust RAD system must detect such events and respond appropriately.

In our setup, a validated low-level control system is responsible for executing motion trajectories, monitoring contact forces, and responding to user-issued verbal feedback (e.g., reports of pain). These behaviours will be detailed in a companion paper on reactive control strategies.

This paper focuses on the high-level control framework that operates above the low-level controller. It models the dressing task as a probabilistic discrete-time Markov chain (pDTMC), enabling symbolic runtime verification of safety

and performance requirements. The high-level system continuously updates beliefs about task state using Bayesian inference, substitutes these into precomputed symbolic expressions, and adaptively refines robot decision-making.

Together, this layered architecture supports both proactive and reactive responses to runtime uncertainty, advancing safe and adaptive human-robot collaboration in dressing assistance.

3.2 Low-Level Control Strategy overview

The low-level control strategies implemented in our RAD system serve as the foundational layer for real-time hazard mitigation. These strategies were developed and validated through physical dressing trials involving simulated garment snags and user-reported discomfort. Full implementation and evaluation details are provided in our comparison study [19] The two primary strategies are:

- **Garment Snagging Control:** When excessive interaction forces are detected via integrated force sensors, the system triggers one of three responses: (a) prompt the user for assistance through a Rasa-based chatbot interface, (b) initiate autonomous recovery by adjusting the robot's trajectory, or (c) abort the task safely if mitigation fails.
- **User Discomfort Mitigation:** Enables users to express pain or discomfort through natural language commands. The robot responds by progressively reducing speed and, if needed, aborting the task.

These low-level mechanisms are essential for reactive safety during dressing and were experimentally validated across multiple dressing trials, showing effective mitigation of hazards in both user-assisted and autonomous recovery modes.

The high-level control strategies introduced in this work are designed to complement these reactive behaviours by offering predictive, symbolic reasoning capabilities through pDTMC-based runtime verification. Together, the combined architecture ensures both proactive and reactive safety handling in robot-assisted dressing.

The high-level control strategy presented in this paper focuses on symbolic reasoning and runtime verification based on the pDTMC abstraction. While the low-level controller plays a critical role in reactive execution (e.g., force compliance and trajectory recovery), it is validated through extensive experimental dressing trials but not formally verified here. A detailed account of the low-level control mechanisms and their empirical evaluation is provided in our companion study [19]. Formal verification of continuous low-level dynamics is out of scope for this work, which concentrates on symbolic guarantees and adaptive decision-making at the task level.

3.3 High-Level Control Architecture

Figure 1 illustrates the runtime-adaptive control architecture for the RAD system. The architecture integrates low-level sensor processing with high-level symbolic reasoning to support safe and responsive operation under uncertainty.

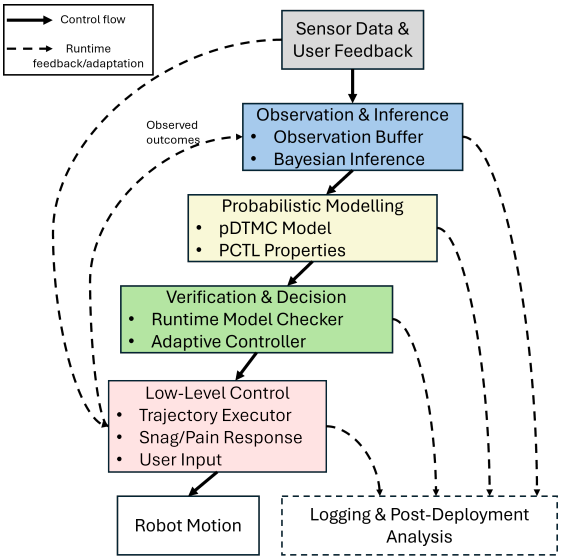

Fig. 1: High-level control architecture showing runtime adaptation via symbolic verification. Sensor inputs inform both high-level inference and low-level control actions.

At the top of the pipeline, the system receives continuous inputs from sensors (e.g., force, trajectory deviation) and verbal user feedback. These inputs are processed by an *Observation and Inference* module that maintains a buffer of recent events and uses Bayesian inference with observation ageing [1,2] to update runtime estimates of key transition probabilities (e.g., snag detection, recovery success).

These updated probabilities are injected into a symbolic *Probabilistic Model* comprising a parametric Discrete-Time Markov Chain (pDTMC) and a set of PCTL-specified safety and reliability requirements. The model is evaluated by the *Verification and Decision* layer using closed-form expressions obtained via `PRISM+PARAM` [13], enabling rapid runtime checks without re-invoking the model checker.

The output of this layer informs the *Adaptive Controller*, which adjusts system behaviour based on the verification results. If a safety threshold is violated (e.g., escalation risk exceeds 0.5), the system proactively transitions to compliant or abortive behaviour. Otherwise, it proceeds with recovery or continues execution.

Finally, the *Low-Level Control* layer executes trajectory plans, responds to snag and pain feedback, and integrates real-time user input. These behaviours are conditionally triggered based on the decisions verified in the upper layers. Together, this architecture ensures that the RAD system adapts at runtime to user-specific conditions while maintaining conformance with formal safety con-

straints. The pDTMC abstraction enables runtime introspection and supports explainability in decision-making.

3.4 Hazard Analysis and Safety Constraints

The pDTMC model and associated evaluation requirements were informed by a structured hazard analysis conducted during system design [8]. This analysis identified the following safety requirements related to task failure, user discomfort, and mitigation breakdowns, and derived constraints that the system must satisfy to maintain safe and reliable operation:

- **Limit the risk of task abortion.** The system should minimise the likelihood of entering an abort state due to unresolved snags or escalation failures.
- **Ensure reliable task completion.** The dressing task should succeed in a high proportion of executions, even under varying user behaviour and environmental noise.
- **Bound the expected cost of silent failures.** Escalations that go undetected should not incur high expected penalties, motivating prompt detection and mitigation.
- **Avoid prolonged or delayed recovery.** Mitigation pathways should resolve issues efficiently, avoiding excessive retries or delays in user or autonomous responses.

These constraints are later formalised in PCTL, and evaluated via symbolic expressions over transition parameters. This ensures that the RAD system maintains compliance with safety and comfort requirements—even in uncertain and dynamic execution contexts.

4 Snagging Model and Formal Verification

To verify safety-critical behaviour in robot-assisted dressing, we formalise the high-level control strategy as a pDTMC. This model captures task progression, garment-snag escalation, and recovery pathways as probabilistic transitions, abstracting from the robot's low-level motion control.

Model description. The pDTMC models 10 abstract task states (Fig. 2). The initial dressing task begins in state s_0 (`dressingProcess`), progressing towards s_3 (`dressingComplete`). Potential snags are detected in s_1 (`potentialSnag`) or escalated undetected to s_2 (`undetectedEscalation`). If mitigation is required, the system enters s_4 (`mitigationStrategy`), choosing between human assistance s_5 (`requestHRI`) or autonomous resolution s_6 (`autonomousResolution`). Successful mitigation reaches s_7 (`snagMitigated`); failure results in task abortion s_8 (`abortTask`). All outcomes eventually transition to s_9 (`moveHome`) before resetting.

Formal definition. The pDTMC is defined as a tuple $\mathcal{M} = (S, s_0, P, \mathcal{L}, \rho)$, where:

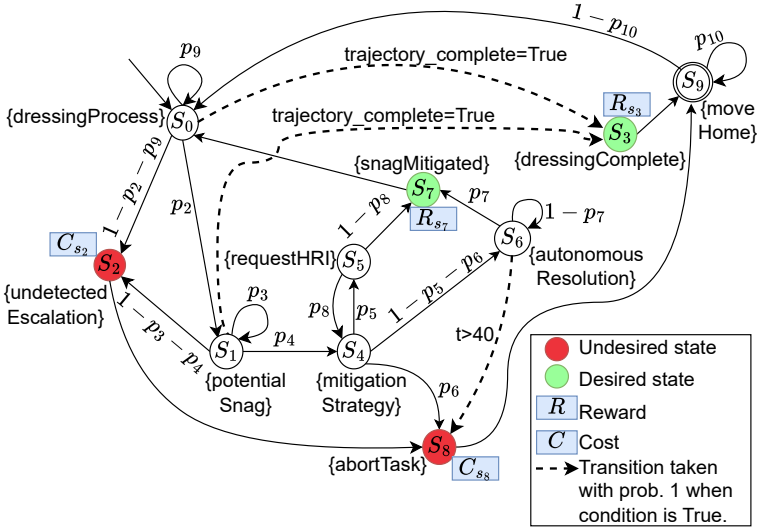

Fig. 2: Abstraction of the pDTMC RAD model. Transition probabilities ($p_{[.]}$) govern state transitions. Timeout conditions (e.g., $t = \texttt{MAX\_TIME}$ in s_6) ensure escalation resolution. Rewards ($R_{[.]}$) and costs ($C_{[.]}$) are linked to outcome states.

– S is the finite set of states. Each state is represented by:

$$s' = (s,\ t,\ \texttt{time\_step},\ \texttt{trajectory\_complete})$$

where:
- $s \in [0, 9]$ is the current task stage.
- $t \in [0, \texttt{MAX\_TIME}]$ tracks time in s_6 (autonomous retries).
- $\texttt{time\_step}$ tracks progress in s_0, s_1 until trajectory completion.
- $\texttt{trajectory\_complete}$ is a Boolean flag signalling task end.

– s_0 is the initial state: the dressing process starts with all timers at zero.
– $P : S \times S \to [0, 1]$ is the parametric transition matrix governed by symbolic parameters p_1 to p_{10}, representing task observations and control uncertainties. All model transitions are defined by the PRISM model in Appendix A of the full version of this paper on arXiv [20].[5]
– $\mathcal{L} : S \to \mathcal{AP}$ labels each state with atomic propositions (e.g., $\texttt{dressingComplete}$), enabling property specification such as $P_{\leq 0.1}[F\ \texttt{abortTask}]$.
– $\rho : S \to \mathbb{R}_{\geq 0}$ assigns rewards and costs to terminal states, capturing success and failure outcomes. These are:
- $R_{s_3} = BASE\_REWARD\_S3 = 20$ - base reward constant for successful dressing (used in symbolic analysis).

[5] For each state $s \in S$, the outgoing transition probabilities are either mutually exclusive or explicitly normalized to ensure that the total probability of leaving a state does not exceed 1. This guarantees that $\sum_{s' \in S} P(s, s') \leq 1$, preserving the validity of the DTMC semantics.

- $R_{s_7} = R\_S7 = 10$ for snag mitigation.
- $C_{s_2} = C\_S2 = 10$ for undetected escalation.
- $C_{s_8} = C\_S8 = 5$ for task abortion.

In our symbolic analysis, R_{s_3} is set to 20 as previously described; however, in the simulation, we further examine the effect of time by modelling this reward as a decayed value, as detailed below.

Reward decay. Although cumulative reward properties are not symbolically supported by `PRISM+PARAM`, we model a decaying reward in the PRISM simulation model to discourage long delays in dressing completion:

$$R_{s_3} = e^{-\texttt{DECAY\_RATE}\cdot\texttt{time\_step}} \cdot BASE\_REWARD\_S3$$

with `DECAY_RATE` set to 0.5. This decay reward is applied during simulation-based evaluation only and is excluded from the symbolic analysis in Section 5, which uses the constant `BASE_REWARD_S3`.

Model features. The pDTMC supports runtime verification and adaptive reasoning through:

- **Symbolic transitions:** Parametric probabilities capture runtime uncertainty in detection, recovery, and escalation.
- **Progress tracking:** `trajectory_complete` and `time_step` govern dressing task duration and allow symbolic evaluation of task timing.
- **Reward/cost abstraction:** Terminal states encode outcomes for reward-guided or risk-sensitive control.
- **Time-bounded retries:** Timeout in s_6 prevents infinite loops in autonomous recovery.

This model provides the formal basis for the symbolic evaluation in Section 5, and is structured to support runtime adaptation via parameter updates (see Discussion, Section 6).

4.1 Specification of Safety and Reliability Requirements

The safety and performance requirements identified through hazard analysis (Section 3.4) are formalised using Probabilistic Computation Tree Logic (PCTL) [6]. These requirements are used to evaluate the symbolic pDTMC model and guide runtime decision-making.

PCTL enables the specification of probabilistic reachability, time-bounded, and reward-bounded behaviours in discrete-time systems. For example, $P_{\leq 0.1}[F\, s = 8]$ specifies that the probability of eventually aborting the task should be no greater than 10%. Although `PRISM+PARAM` currently supports only reachability requirements in symbolic form, we approximate cost-based constraints using reachability proxies (see Section 5.3).

Finally, the following requirements formalise the core constraints derived from the hazard analysis [8]:

– **(H1) Limit the risk of task abortion**

$$P_{\leq 0.1}[F\, s = 8] \tag{1}$$

"The probability of reaching the task abort state $s = 8$ must remain below 10%."

– **(H2) Ensure reliable task completion**

$$P_{\geq 0.9}[F\, s = 3] \tag{2}$$

"The dressing task should complete successfully in at least 90% of executions."

– **(H3) Bound undetected escalation cost**

$$ExpectedCost_{s2} = C_{S2} \cdot P_{=?}[F\, s = 2] \leq MAX\_C2 \tag{3}$$

"The expected cost of undetected escalation (reaching $s = 2$) must remain within limit set by the maximum cost allowed MAX_C2." This is evaluated symbolically via a proxy expression.

– **(H4) Timely mitigation success reward**

$$ExpectedReward_{s7} = R_{S7} \cdot Pmax_{=?}[F\, s = 7] \tag{4}$$

"The system should maximise the probability of successfully mitigating snags." The expected reward is derived symbolically based on p_7 and p_8.

– **(H5) Encourage time-bounded completion**

$$P_{\geq 0.95}[F_{\leq \texttt{MAX\_TIME\_TRAJ}}\, s = 3] \tag{5}$$

"The dressing task should complete within the designated trajectory time in at least 95% of cases." This is used in simulation.

These formal specifications provide a rigorous basis for symbolic evaluation and runtime adaptation. Requirements (H1)–(H4) are fully captured by symbolic expressions derived via `PRISM+PARAM`, enabling lightweight verification through parameter substitution. Time-bounded and cumulative reward constraints are approximated or evaluated using proxy expressions where symbolic support is unavailable.

Parameter value ranges. The symbolic expressions for reachability and reward properties are defined over symbolic parameters p2–p10, which represent key transition probabilities within the pDTMC. For the purpose of symbolic analysis and heatmap visualisation, these parameters were evaluated over plausible ranges informed by system design, empirical dressing trials, and prior work [19]. These ranges are summarised in Table 1, and provide a safe operating envelope for runtime adaptation. During execution, these probabilities are dynamically updated using Bayesian inference, allowing the system to respond to real-time sensor data and user feedback.

Table 1: Parameter Ranges Used in Symbolic Evaluation

Parameter	Description and Range
p_2	Probability of detecting a potential snag. Range: $[0.0, 1.0]$
p_3	Probability of remaining in escalation monitoring without triggering mitigation. Range: $[0.0, 1.0]$
p_4	Probability of escalation after a detected snag. Range: $[0.4, 0.9]$
p_5	Probability of selecting human-assisted recovery. Range: $[0.5, 1.0]$
p_6	Probability of selecting autonomous recovery. Range: $[0.0, 0.5]$
p_7	Success probability of autonomous recovery. Range: $[0.0, 1.0]$
p_8	Failure probability of human intervention. Range: $[0.0, 1.0]$
p_9	Probability of escalation failure leading to task abort. Range: $[0.0, 0.3]$
p_{10}	Probability of returning to idle/home state. Range: $[0.9, 1.0]$

4.2 Symbolic Model Checking using PRISM+PARAM

We perform symbolic verification of the pDTMC model using the `PRISM+PARAM` toolchain [13]. This enables closed-form algebraic expressions to be extracted for PCTL reachability requirements such as $P_{=?}[F\, s = 2]$ and $P_{=?}[F\, s = 7]$. These symbolic expressions are parametrised over key transition probabilities (e.g., p_2, p_3, p_7, p_8), enabling analysis of system behaviour under uncertainty and variable conditions. Each expression captures the probability of reaching a safety-critical or goal state as a function of the system's current configuration. In our evaluation (Section 5), these expressions are visualised over bounded parameter ranges to assess risk and performance.

While the symbolic engine supports reachability queries, cumulative reward properties of the form $P_{=?}[C \leq T]$ are not currently supported. To address this, we reformulate cost and reward analysis using proxy expressions—multiplying the reachability probability by a fixed reward or cost scalar (e.g., see Equation 3).

4.3 Runtime Verification using Bayesian Learning

To enable adaptive decision-making, the RAD system performs runtime verification by updating the transition probabilities of the pDTMC using Bayesian learning. This approach applies *observation ageing* [1, 2], giving more weight to recent observations while discounting older data, allowing the system to respond to changes in user behaviour or task context.

The updated estimate of transition probability $p_{ij}^{(k)}$ after k observations is computed as:

$$p_{ij}^{(k)} = \frac{c_0}{c_0 + k} p_{ij}^{(0)} + \frac{k}{c_0 + k} \cdot \frac{\sum_{l=1}^{k} w_l \cdot x_{ij}^{(l)}}{\sum_{l=1}^{k} w_l} \tag{6}$$

Here, $p_{ij}^{(0)}$ is the prior estimate, c_0 controls the influence of the prior, and $w_l = \alpha^{-(t_k - t_l)}$ applies exponential ageing to past observations. The decay factor $\alpha \geq 1$ controls how quickly older data are discounted. The variable $x_{ij}^{(l)}$ denotes

the observed occurrence (typically binary: 1 for a transition observed from state i to j at time step l, and 0 otherwise). The symbolic expressions derived offline for reachability and reward properties (e.g., $P_{=?}[F\, s = 7]$) are evaluated at runtime by substituting dynamically updated probabilities (e.g., p_7, p_8). This enables lightweight, real-time runtime verification to guide decision-making during dressing without invoking a model checker.

4.4 How the Adaptation Mechanism Operates

During execution, the RAD framework monitors real-time sensor data (e.g., garment force feedback, joint velocities) and user responses to update transition probabilities within the symbolic pDTMC model. These updated probabilities are substituted into precomputed symbolic expressions for key safety and performance requirements, allowing the framework to assess evolving risk and make informed decisions at runtime.

If a safety threshold is violated—e.g., the probability of task abortion exceeds a bound—the framework responds on two levels:

1. **Low-Level Control Actions**: The controller immediately enters a compliant mode, reducing speed and applied force or halting movement to mitigate user discomfort or mechanical risks.
2. **High-Level Adaptation**: The symbolic pDTMC model is used to track evolving task context. Bayesian learning updates transition probabilities (e.g., likelihood of snag, success of user or autonomous recovery), enabling dynamic substitution into symbolic expressions (e.g., $P_{=?}[F\, s = 2]$). This supports real-time risk evaluation without rechecking the model.

For example, if repeated snagging is observed under specific motion trajectories, the estimated probability of transitioning from s_0 (`dressingProcess`) to s_1 (`potentialSnag`) increases. Similarly, success rates of user-assisted or autonomous mitigation are reflected in the transition probabilities to s_5 and s_6. Rather than storing full execution histories, the Bayesian update loop maintains a compact belief over transition likelihoods. This integration of symbolic runtime evaluation and adaptive control ensures that the framework responds promptly to immediate safety threats while gradually improving high-level decision-making in dynamic, user-specific contexts.

5 Evaluation

To assess the robot-assisted dressing system under uncertainty, we adopt a symbolic evaluation approach using the `PRISM+PARAM` toolchain. Our evaluation focuses on three core aspects: (i) the reachability of critical failure or successful mitigation states, (ii) the expected cost of failure, and (iii) the expected reward of mitigation success. Rather than relying on simulation-based numerical experiments, we symbolically evaluated the parametric DTMC model across key

parameters that influence the system's decision-making behaviour—such as the reliability of snag detection, human and autonomous recovery, and escalation outcomes. These symbolic expressions allow for offline design analysis and enable runtime reasoning via parameter substitution. Each symbolic property is visualised using annotated heatmaps and contour plots.

5.1 Symbolic Analysis of Snag Mitigation

We analysed the probability of successfully mitigating a snag (i.e., reaching state $s = 7$) in the pDTMC model (Figure 2) using `PRISM+PARAM`. The property of interest is the symbolic reachability expression $P =?[F\, s = 7]$, which quantifies the probability of eventually reaching the successful mitigation state under varying system conditions.

A closed-form symbolic expression was derived, capturing the joint influence of snag detection, escalation, and recovery through both autonomous and human intervention. To aid interpretability, we introduce the following terms:

- $\alpha = p_2.p_4$ — snag detection and escalation,
- $\beta = p_5.p_8$ — failed recovery attempts via human mitigation,
- $\delta_k = p_7^k$ — recursive retries of autonomous recovery (limited to 40 steps).

A simplified excerpt of the symbolic expression extracted using `PRISM+PARAM` is shown below. This highlights the compound effects of detection ($\alpha = p_2 \cdot p_4$), human failure ($\beta = p_5 \cdot p_8$), and recursive autonomous retries ($\delta_k = p_7^k$):

$$P_{s=7} = \frac{\begin{array}{l} 10000\,\beta^2 \cdot \alpha^2 - 20000\,\beta \cdot p_5 \cdot \alpha^2 + \cdots + \\ 100\,\beta \cdot \delta_2 \cdot \alpha^2 + \cdots + 10\,\delta_1 \cdot \alpha + \cdots \end{array}}{\begin{array}{l} 10000\,\beta^2 \cdot \alpha^2 - 20000\,\beta \cdot p_5 \cdot \alpha^2 + \cdots + \\ 200\,\beta^2 \cdot p_5^2 \cdot p_2 + \cdots + 40\,p_2 + 1 \end{array}} \tag{7}$$

The symbolic expression captures nested recovery paths and enables comparison of mitigation effectiveness. The full expression is provided in Appendix B of the arXiv version [20].

Figure 3 shows the evaluated symbolic expression over a range of human and autonomous recovery capabilities. A safety threshold contour at $P = 0.5$ distinguishes high-risk scenarios (top-left) from safer operating regions (bottom-right). The *Safe Zone* is defined where the probability of reaching sate $s = 7$ (snag mitigation) exceeds 0.6. This typically occurs when the autonomous mitigation success probability (p_7) is high, or the human intervention failure probability (p_8) is low.

The results highlight an important compensation effect: strong performance in one recovery pathway (human or autonomous) can offset weaknesses in the other. For example, even when p_7 is low, maintaining a low p_8 (i.e., reliable human intervention) preserves high overall success rates. Conversely, increasing p_7 improves robustness against human failure.

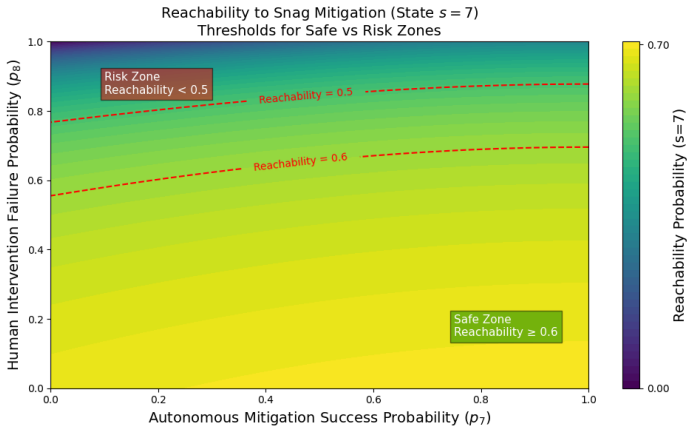

Fig. 3: Probability of successful mitigation (s_7) varying autonomous mitigation success probability (p_7) and human failure probability (p_8). Lighter regions indicate higher success.

Our findings reinforce the benefit of hybrid, adaptive recovery strategies in which human-in-the-loop support and autonomous recovery dynamically compensate for each other. Our symbolic reachability analysis provides both an interpretable quantitative basis for design decisions and an analytical foundation for runtime adaptation.

5.2 Symbolic Analysis of Snag Escalation Failure

We now analyse the second key reachability property of interest $P_{=?}[F\ s = 2]$. This property quantifies the probability of reaching state s_2, representing an undetected escalation of a detected snag—a critical failure scenario in the robot-assisted dressing task.

Using `PRISM+PARAM`, we symbolically evaluated this property while varying two key parameters: (1) the probability of detecting a potential snag, p_2, and (2) the probability of remaining in the monitoring state without triggering mitigation, p_3. These directly govern the likelihood of silent escalation, with all other transitions held constant to isolate their effect.

The symbolic expression extracted from `PRISM+PARAM` was algebraically simplified and normalised. It captures both direct and indirect contributions to s_2 reachability via recursive monitoring cycles. A normalised version was evaluated over a grid of p_2 and p_3 values, revealing the influence of detection and monitoring performance.

The simplified symbolic expression is given by:

$$P = \frac{100\,P_3 \cdot P_2 + 98\,P_2 - 99}{88\,P_2 - 100} \tag{8}$$

This captures the trade-off between detection (p_2) and monitoring (p_3). While the numerator increases linearly with p_2 and p_3 the denominator introduces a non-linear effect governed solely by p_2. As p_2 improves, the denominator approaches zero from below, causing a steep drop in failure probability—a clearly reflected in the heatmap gradient (Figure 4). The PRISM+PARAM-generated expression is valid for $88p_2 - 100 > 0$; outside this range, results are clipped to $[0, 1]$ for validity.

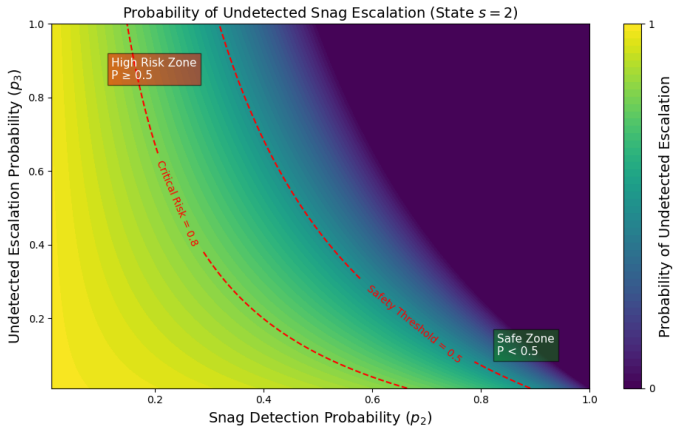

Fig. 4: Reachability probability of undetected snag escalation (s_2) as a function of detection probability p_2 and undetected escalation probability p_3. Lighter regions indicate higher failure probability. Dashed contour lines highlight key risk thresholds: $P = 0.5$ (Safety Threshold) and $P = 0.8$ (Critical Risk). Annotated zones mark high-risk and safe regions.

Figure 4 illustrates the reachability landscape for undetected snag escalation, based on the symbolic property $P_{=?}[F\ s = 2]$. The heatmap shows how the failure probability varies over a grid of snag detection probabilities (p_2) and escalation persistence probabilities (p_3).

The *"Safe Zone"* ($P < 0.5$) occurs in the lower-right corner, where detection is strong ($p_2 \to 1$) and escalation is unlikely ($p_3 \to 0$). The *"High Risk Zone"* ($P \geq 0.5$) appears in the upper-left corner, where detection is weak ($p_2 \to 0$) and escalation is persistent ($p_3 \to 1$).

The red dashed contours highlight two thresholds: the safety threshold at $P = 0.5$, and the critical risk boundary at $P = 0.8$, making conditions where the likelihood of silent failure is dangerously high. Notably, the steep gradient along the p_2 axis reveals that even modest gains in detection capability can substantially reduce failure risk.

These findings highlight the system's high sensitivity to snag detection accuracy and provide actionable insights for both formal verification and runtime adaptation. Prioritising high p_2 values can significantly lower the probability of silent escalation, thereby improving overall safety.

5.3 Proxy Cost Analysis of Undetected Snag Escalation

To evaluate the cost implications of undetected snag escalation, we adopt a proxy formulation based on the symbolic reachability expression for state $s = 2$ (undetected escalation). Specifically, we compute Equation 3, where C_{S2} is a fixed penalty associated with entering the failure state. This formulation provides a lightweight approximation of the impact by scaling the symbolic reachability with the defined failure cost. It enables efficient evaluation during runtime and captures the increasing cost of failure under unsafe parameter regimes.

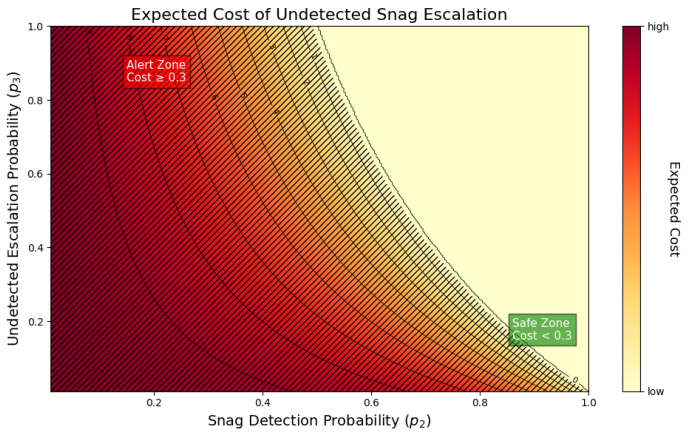

Fig. 5: Expected cost of undetected snag escalation, computed as $C_{S2} \cdot P_{=?}[F\, s = 2]$, over varying detection probability (p_2) and escalation persistence (p_3). The *Alert Zone* (Cost ≥ 0.3) highlights failure-prone conditions; the *Safe Zone* (Cost < 0.3) reflects robust detection and recovery.

Figure 5 visualises this cost landscape. The *"Alert Zone"* (Cost ≥ 0.3) arises when snag detection is weak ($p_2 \to 0$) and the likelihood of persistent, undetected escalation is high ($p_3 \to 1$). The hatched area clearly marks this high-risk region.

Conversely, the *"Safe Zone"* (Cost < 0.3) occupies the lower-right corner of the heatmap, where effective detection significantly reduces escalation risk and associated penalty. Compared to the raw reachability view in Figure 4, this proxy formulation adds interpretability by contextualising risk in cost terms.

Our proxy formulation allows the system to perform low-overhead runtime checks against unsafe conditions using a single symbolic expression and supports proactive adaptation in safety-critical human-robot interactions.

5.4 Symbolic Analysis of Mitigation Success

This analysis focuses on the expected reward associated with successful snag mitigation, represented by reaching state s_7 in the pDTMC model. Since cumulative reward properties cannot currently be symbolically evaluated in `PRISM+PARAM`, we compute the expected reward using a scaled reachability formulation from Equation 4, where R_{S7} is a fixed reward assigned to successful mitigation. The symbolic reachability expression $P_{=?}[F\, s = 7]$ was derived in Section 5.1 and captures the combined effect of detection, escalation, and mitigation strategies (both human and autonomous).

To examine reward trade-offs, we fix parameters $p_2 = 0.9$, $p_4 = 0.7$, and $p_5 = 0.65$, and vary:

- p_8: the failure probability of human intervention,
- p_7: the success probability of autonomous recovery.

The symbolic expression is evaluated over this 2D parameter space, and the expected reward is computed accordingly.

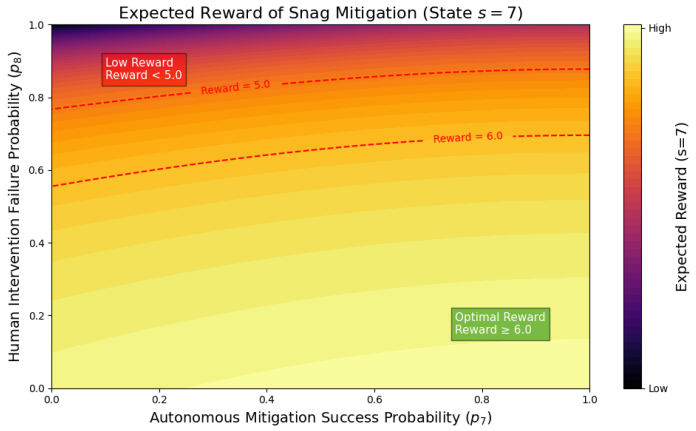

Fig. 6: Expected reward for mitigation success ($s = 7$) evaluated from the symbolic expression over human intervention failure (p_8) and autonomous recovery success (p_7). The *Optimal Reward* zone arises when both modalities are reliable; *Low Reward* emerges when both pathways are likely to fail.

Figure 6 illustrates the resulting expected reward surface. A clear *"Optimal Reward"* zone emerges in the bottom-right corner ($p_8 \to 0$, $p_7 \to 1$), where both

mitigation pathways are highly effective. In contrast, the *"Low Reward"* region appears when both autonomous and human strategies are likely to fail ($p_7 \to 0$, $p_8 \to 1$). Contour lines at reward thresholds $R = 5.0$ (Lower Bound) and $R = 6.0$ (Optimal Reward) delineate these regions.

These findings reinforce the hybrid design principle: strong performance in either the human-in-the-loop or autonomous recovery pathway can compensate for the other's limitations, enabling robust, reward-aware adaptation in safety-critical scenarios.

6 Discussion

Our approach demonstrates how hazard-informed modelling, symbolic verification, and Bayesian adaptation can be combined to support safe, adaptive decision-making in human-robot collaboration. The symbolic evaluation presented in this work offers several critical insights into the safety and adaptability of RAD systems operating under uncertainty. It also shows how requirements like snag escalation and mitigation success can be interpreted as algebraic functions of runtime-updated parameters such as p_2, p_3, p_7, and p_8, from our model example. Analysis of such functions allow the controller to identify and avoid unsafe regions of the parameter space proactively. Our approach addresses a fundamental challenge in pHRI: managing runtime uncertainty with provable safety assurances.

A key benefit is that runtime verification does not require re-running a model checker. Once symbolic expressions are precomputed, parameter substitution enables real-time evaluation with minimal computational overhead. This facilitates explainable decision-making and supports runtime assurance. One current limitation is that symbolic evaluation is based on an abstract model that does not fully capture the continuous dynamics of low-level controllers. Further work will integrate these physical dynamics to bridge the gap between symbolic guarantees and physical execution.

Future validation will adopt a hybrid pipeline. Simulators such as Gazebo or Isaac Sim will enable safe parameter tuning and support transfer learning for physical deployment. Our current symbolic analysis provides a feasibility baseline.

7 Conclusion and Future Work

This paper introduced a high-level control framework for safety-critical human-robot interaction that integrates symbolic verification with runtime adaptation. Although we demonstrated the framework using a Robot-Assisted Dressing (RAD) scenario, the contribution is methodological in nature and applicable to a wide range of human-in-the-loop systems with probabilistic behaviour and runtime uncertainty.

Our pDTMC-based framework enables fast runtime evaluation of PCTL properties using precomputed symbolic expressions and dynamically updated transition probabilities via Bayesian inference. This supports real-time adaptation and safety assurance. Results identify safe parameter regimes and alert zones requiring mitigation.

Future work includes closer integration of the symbolic model with low-level control. Due to RAD's complexity, we focus on snagging mitigation, but full assurance requires broader scenario modelling. We will explore a multi-model RAD world [3] to support system-level verification. The framework can extend to robot-assisted undressing (RAUD), which may involve different constraints (e.g., constrained-to-unconstrained transitions) and hazards (e.g., visibility or balance loss), requiring adapted models and PCTL properties. Finally, our approach can generalise to pHRI systems with inherent uncertainty and variability. Future work will focus on validating its generality and effectiveness through extensive case studies and experimental evaluations.

Acknowledgment

This work was funded by the EPSRC projects EP/V026747/1 'UKRI Trustworthy Autonomous Systems Node in Resilience' and EP/V026801/2 'UKRI Trustworthy Autonomous Systems Node in Verifiability'.

Disclosure of Interests

The authors declare that they have no conflict of interest relevant to the content of this work.

References

1. Calinescu, R., Johnson, K., Rafiq, Y.: Using observation ageing to improve markovian model learning in qos engineering. In: Proceedings of the 2nd ACM/SPEC International Conference on Performance engineering. pp. 505–510 (2011)
2. Calinescu, R., Rafiq, Y., Johnson, K., Bakır, M.E.: Adaptive model learning for continual verification of non-functional properties. In: Proceedings of the 5th ACM/SPEC international conference on Performance engineering. pp. 87–98 (2014)
3. Calinescu, R., Yaman, S.G., Gerasimou, S., Vázquez, G., Bassett, M.: Verification and external parameter inference for stochastic world models. arXiv preprint arXiv:2503.16034 (2025)
4. Chance, G., Jevtić, A., Caleb-Solly, P., Dogramadzi, S.: A quantitative analysis of dressing dynamics for robotic dressing assistance. Frontiers in Robotics and AI **4**, 13 (2017)
5. Christoforou, E.G., Avgousti, S., Ramdani, N., Novales, C., Panayides, A.S.: The upcoming role for nursing and assistive robotics: Opportunities and challenges ahead. Frontiers in Digital Health **2**, 585656 (2020)

6. Ciesinski, F., Größer, M.: On probabilistic computation tree logic. Validation of Stochastic Systems: A Guide to Current Research pp. 147–188 (2004)
7. Cooper, S., Di Fava, A., Vivas, C., Marchionni, L., Ferro, F.: Ari: The social assistive robot and companion. In: 2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN). pp. 745–751. IEEE (2020)
8. Delgado Bellamy, D., Chance, G., Caleb-Solly, P., Dogramadzi, S.: Safety assessment review of a dressing assistance robot. Frontiers in Robotics and AI **8**, 667316 (2021)
9. Erickson, Z., Gangaram, V., Kapusta, A., Liu, C.K., Kemp, C.C.: Assistive gym: A physics simulation framework for assistive robotics. In: 2020 IEEE International Conference on Robotics and Automation (ICRA). pp. 10169–10176. IEEE (2020)
10. Gleirscher, M., Calinescu, R.: Safety controller synthesis for collaborative robots. In: 2020 25th International Conference on Engineering of Complex Computer Systems (ICECCS). pp. 83–92. IEEE (2020)
11. Gleirscher, M., Calinescu, R., Douthwaite, J., Lesage, B., Paterson, C., Aitken, J., Alexander, R., Law, J.: Verified synthesis of optimal safety controllers for human-robot collaboration. Science of Computer Programming **218**, 102809 (2022)
12. Gu, D., Andreev, K., Dupre, M.E.: Major trends in population growth around the world. China CDC weekly **3**(28), 604 (2021)
13. Hahn, E.M., Hermanns, H., Wachter, B., Zhang, L.: PARAM: A model checker for parametric Markov models. In: Computer Aided Verification: 22nd International Conference, CAV 2010, Edinburgh, UK, July 15-19, 2010. Proceedings 22. pp. 660–664. Springer (2010)
14. Jevtić, A., Valle, A.F., Alenyà, G., Chance, G., Caleb-Solly, P., Dogramadzi, S., Torras, C.: Personalized robot assistant for support in dressing. IEEE transactions on cognitive and developmental systems **11**(3), 363–374 (2018)
15. Kirca, Y.S., Degirmenci, E., Demirci, Z., Yazici, A., Ozkan, M., Ergun, S., Kanak, A.: Runtime verification for anomaly detection of robotic systems security. Machines **11**(2), 166 (2023)
16. Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: Verification of probabilistic real-time systems. In: 23rd International Conference on Computer Aided Verification (CAV'11). pp. 585–591 (2011)
17. Kwiatkowska, M., Norman, G., Parker, D.: Probabilistic symbolic model checking with prism: A hybrid approach. International journal on software tools for technology transfer **6**(2), 128–142 (2004)
18. Li, S., Figueroa, N., Shah, A., Shah, J.A.: Provably Safe and Efficient Motion Planning with Uncertain Human Dynamics. In: Proceedings of Robotics: Science and Systems. Virtual (July 2021). https://doi.org/10.15607/RSS.2021.XVII.050
19. Rafiq, Y., James, B.A., Xu, K., Hierons, R.M., Dogramadzi, S.: Hybrid control strategies for safe and adaptive robot-assisted dressing (2025), https://arxiv.org/abs/2505.07710
20. Rafiq, Y., Vázquez, G., Calinescu, R., Dogramadzi, S., Hierons, R.M.: Symbolic runtime verification and adaptive decision-making for robot-assisted dressing (2025), https://arxiv.org/abs/2504.15666
21. Zhao, X., Gerasimou, S., Calinescu, R., Imrie, C., Robu, V., Flynn, D.: Bayesian learning for the robust verification of autonomous robots. Communications Engineering **3**(1), 18 (2024)
22. Zhao, X., Robu, V., Flynn, D., Dinmohammadi, F., Fisher, M., Webster, M.: Probabilistic model checking of robots deployed in extreme environments. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 8066–8074 (2019)