SafeDMPs: Integrating Formal Safety with DMPs for Adaptive HRI

Soumyodipta Nath*, Pranav Tiwari*, and Ravi Prakash
Cyber Physical Systems, Indian Institute of Science, Bengaluru, India
{soumyodiptan, pranavtiwari, ravipr}@iisc.ac.in
* denotes equal contribution.

Abstract: Robots operating in human-centered environments must generate motions that are not only adaptive and responsive, but also provably safe in real time. While temporal logic-based planners enable structured high-level task specification, executing these plans safely at the motion level remains challenging. Existing approaches based on Control Barrier Functions (CBFs) guarantee safety but incur significant computational overhead due to online constrained optimization. We propose a modular planning and control framework that combines Dynamic Movement Primitives (DMPs) for smooth, generalizable trajectory generation with Spatio-Temporal Tubes (STTs) for safety enforcement. Unlike CBFs, STTs avoid online optimization and provide closed-form feedback laws, ensuring real-time, collision-free execution. We validate our approach on a Franka Emika robot performing collaborative tasks such as whiteboard writing and adaptive recovery under human intervention in simulation. Compared to CBF and Neural ODE baselines, our method achieves up to 99.97% faster execution, and 48% lower memory usage, while maintaining formal safety guarantees.

Keywords: Spatio Temporal Tubes, Dynamic Movement Primitives, Human Robot Interaction, Reactive Task Transition.

1 Introduction

Robots are increasingly being deployed in real-world, human-centered environments homes, hospitals, factories where they are expected to collaborate with people in close physical proximity. In such settings, robots must do more than execute predefined tasks; they must adapt to dynamic environments, interpret ambiguous instructions, and respond in real time to human behavior. This requires the integration of long-horizon task planning, reactive decision-making, and safe motion execution.

Recent work has made remarkable progress toward end-to-end robotic frameworks that aim to unify perception, planning, and control. For instance, Mendez-Mendez et al. [1] introduced a lifelong learning approach to task and motion planning, enabling robots to improve their planning capabilities over time by reusing modular knowledge. Similarly, Ahn et al. [2] and Liang et al. [3] demonstrated how large language models (LLMs) can be grounded in physical actions to generate task plans or robot policies directly from natural language instructions. These systems offer impressive generalization and semantic understanding, pushing the boundaries of what autonomous systems can do in unstructured environments.

Building on this trend, Tu et al. [4] proposed a framework for language-embedded 6D pose estimation that enables robust tool manipulation from natural language commands and point cloud data. In parallel, Su et al. [5] introduced GSCE, a structured prompting approach that enhances LLM-based reasoning for drone control tasks, improving reliability and success rates under complex conditions. These works demonstrate the value of embedding semantic understanding into robotic systems, especially when interpreting ambiguous, human-issued commands.

Workshop on Safe and Robust Robot Learning for Operation in the Real World at CoRL 2025.

In addition to semantic and task-level reasoning, safety in motion execution remains a critical challenge. Traditional methods like control barrier functions offer strong theoretical guarantees but often suffer from computational bottlenecks due to online optimization. To address this, Das et al. [6] introduced Spatio-Temporal Tubes (STTs) for synthesizing safety-aware controllers that satisfy temporal reach-avoid-stay (T-RAS) constraints in unknown systems. Related efforts in autonomous driving, such as those by Pan et al. [7] and Magdici and Althoff [8], have developed planning frameworks that coordinate motion and control layers, account for tracking errors, and maintain emergency avoidance strategies to ensure safety under uncertainty and actuator faults.

Complementing these high-level task interpretation strategies, Nawaz et al. [9] proposed a modular framework that integrates reactive temporal logic-based task specifications with continuous, safe motion generation. Their system enables robots to interpret structured task goals while ensuring safe execution using control Lyapunov functions and control barrier functions, validated on interactive robotic tasks such as whiteboard wiping with human intervention. This highlights the importance of bridging symbolic task planning and reactive motion-level safety in dynamic, human-in-the-loop settings.

In dynamic shared spaces, robots must guarantee collision-free, smooth, and stable behaviors, even as they adapt to changing tasks or human interventions. Dynamic Movement Primitives (DMPs) [10] provide a way to learn and generalize robot motion from demonstrations, enabling trajectory generation through a closed-form solution. However, traditional DMP coupling based obstacle avoidance [11] lack formal safety guarantees and typically rely on heuristics. In contrast, Spatio-Temporal Tubes (STTs) provide a principled and provably safe mechanism for deviation and recovery, while retaining the computational efficiency of closed-form solutions.

This paper addresses that gap by proposing a framework that integrates reactive temporal logic-based task planning with Dynamic Movement Primitives (DMPs) and Spatio-Temporal Tubes (STTs) for safe, smooth, and adaptive motion execution in real time.

2 Methodology

This section presents our proposed two-stage framework for generating robust, adaptive, and safe robot motion in dynamic environments. The framework consists of: (i) learning nominal motion plans via Dynamic Movement Primitives (DMPs), and (ii) ensuring robustness and safety at execution-time using Spatio-Temporal Tubes (STTs). Our approach operates in the end-effector space to directly encode Cartesian trajectories, ensuring better generalization and safety margins.

2.1 DMPs as a Nominal Motion Plan

Learning from demonstration (LfD) enables robots to acquire complex motor behaviors through kinesthetic teaching. Given a set of demonstrations $\mathcal{D} = \{\theta_{i1}(t_k), \ldots, \theta_{in}(t_k), \mathcal{X}_i(t_k), \mathcal{Y}_i(t_k), \mathcal{Z}_i(t_k)\}$, we extract end-effector trajectories $(\mathcal{X}_i, \mathcal{Y}_i, \mathcal{Z}_i)$ as our training data.

Among various LfD models such as TP-GMM [12], TP-GPT [13], ProMP [14, 15], and NODE [9], we adopt Dynamic Movement Primitives (DMPs) [10] as explained below.

DMPs model trajectories as a second-order dynamical system with a nonlinear forcing term that captures the motion's shape:

$$\tau^2 \ddot{x} = \alpha_z\big(\beta_z(g - x) - \tau\dot{x}\big) + f(z), \tag{1}$$

where $y(t) \in \mathbb{R}^d$ is the trajectory in d-dimensional space, $g \in \mathbb{R}^d$ is the goal, $\tau > 0$ is a temporal scaling parameter, α_z, β_z are system gains (typically with $\beta_z = \alpha_z/4$ for critical damping), and $f(x)$ is a learned nonlinear forcing function.

DMPs offer several critical properties that make them particularly attractive for dynamic and safe motion generation. First, their attractor dynamics guarantee globally stable goal convergence, regardless of perturbations. Second, they provide temporal scalability via the parameter τ, enabling

motions to be slowed down or sped up without retraining. Third, DMPs support spatial generalization: trajectories can adapt to different start or goal configurations through simple linear shifts.

Crucially, DMPs support feedback-driven modulation through an internal coupling mechanism:

$$\dot{e} = \alpha_e(x_a - x - e), \quad \tau = 1 + k_c e^2, \tag{2}$$

where e is the deviation between the actual y_a and nominal y. This mechanism enables smooth recovery from disturbances, allowing the system to respond adaptively to changes in the environment (Figure- 1(a)). Unlike many other LfD models that require external coupling controllers to handle deviations or perturbations, DMPs inherently support such behaviors within their structure. This internal feedback structure becomes especially important when integrating DMPs with safety-critical control strategies like Spatio-Temporal Tubes (STTs), as discussed in the subsequent sections. Additional details on the canonical system and weight learning procedure are provided in Appendix A.

2.2 Safe-DMPs: Enforcing Robustness and Safety via Spatio-Temporal Tubes

While DMPs offer convergence and generalizability, they do not ensure formal safety in the presence of dynamic obstacles. Traditional extensions like artificial potential fields [11] introduce reactive control but suffer from lack of guarantees and may yield jerky or unsafe behaviors.

We address this limitation by integrating Spatio-Temporal Tubes (STTs) [6], into the DMP execution phase. STTs define time-varying safety envelopes for each state dimension (around the nominal trajectory):

$$\rho_L(t) < x(t) < \rho_U(t), \quad \forall t \geq 0, \tag{3}$$

where $\rho_L(t)$ and $\rho_U(t)$ are trajectory-specific lower and upper bounds, computed from prior data or reachability analysis, and define an admissible region within deviations are permitted. Crucially, unlike control barrier functions (CBFs) or optimization-based methods, STTs allow safe tracking using a closed-form feedback control law, with no need for online optimization.

This feedback controller for each dimension i is given by:

$$u_i(x_i, t) = -k\,\xi_i(x_i, t)\,\epsilon_i(x_i, t), \quad k > 0, \tag{4}$$

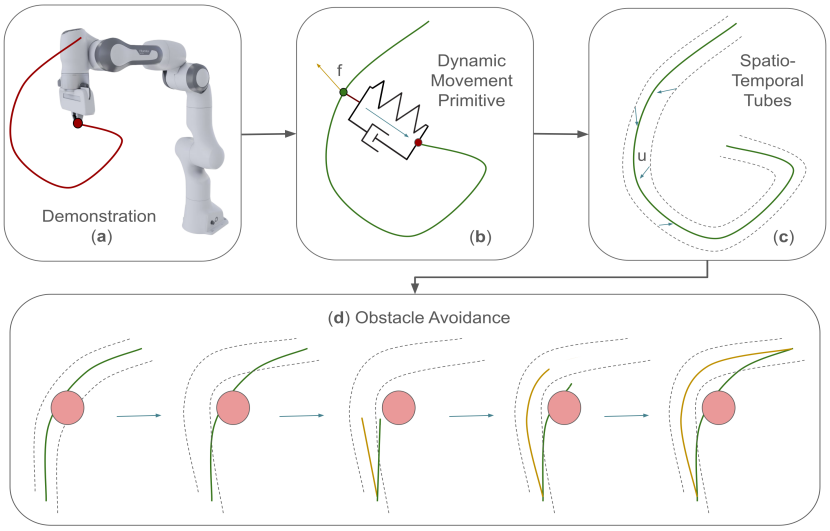

Figure 1: Proposed framework pipeline. (a) Demonstrations are recorded in end-effector space. (b) DMPs encode the nominal motion plan. (c) Spatio-Temporal Tubes (STTs) define a safe envelope around the DMP. (d) During execution, trajectory deformation is triggered when obstacles are encountered, and the system safely reroutes motion within the STT while ensuring convergence.

where $\epsilon_i(x_i, t)$ grows unbounded near the boundaries, enforcing constraint satisfaction, while $\xi_i(x_i, t)$ modulates the control effort. The result is finite-time convergence to the nominal trajectory set, while guaranteeing the state remains inside the defined safe envelope. We combine the expressiveness of DMPs with the safety guarantees of STTs to build a closed-loop motion controller that is both robust and formally safe. During execution, DMPs provide the nominal trajectory, while the STT mechanism modulates deviations when the robot encounters perturbations or obstacles as shown in Figure- 1.

The resulting control law incorporates both terms:

$$\tau^2 \ddot{x} = \alpha_z\big(\beta_z(g - x) - \tau\dot{x}\big) + f(z) + f_{\text{STT}}(z, x, \mathcal{O}), \tag{5}$$

where f_{STT} is an obstacle-aware modulation term informed by proximity to dynamic obstacles $\mathcal{O}$ and the the limits defined by the STT boundaries. This addition exploits the coupling structure of DMPs to provide smooth safety-aware deviation.

To respond to large deviations from the nominal path, an adaptive timing mechanism is triggered:

$$\dot{\mathbf{e}} = \alpha_e(\mathbf{x}_{\text{actual}} - \mathbf{x}_{\text{DMP}} - \mathbf{e}), \tag{6}$$

$$\tau = \tau_{\text{nominal}} + k_c\|\mathbf{e}\|^2, \tag{7}$$

where increasing deviation slows down DMP execution to allow recovery within the safe tube.

When the nominal path intersects with an obstacle, a rerouting mechanism is activated to ensure safe deviation within the STT region:

$$\mathbf{x}_{\text{rerouted}} = \begin{cases} \mathbf{x}_{\text{DMP}}, & \text{if } d(\mathbf{x}_{\text{DMP}}, \mathcal{O}_{\text{center}}) > r_{\text{safe}}, \\ \mathcal{O}_{\text{center}} + \frac{(\mathbf{x}_{\text{DMP}} - \mathcal{O}_{\text{center}})}{d(\mathbf{x}_{\text{DMP}}, \mathcal{O}_{\text{center}})} \cdot r_{\text{clearance}}, & \text{otherwise}, \end{cases} \tag{8}$$

where $\mathcal{O}_{\text{center}}$ represents the center of obstacle, r_{safe} defines the buffer zone around the obstacle, and $r_{\text{clearance}}$ enforces a minimal clearance margin. This rerouting strategy keeps the end-effector trajectory within the safe tube, while still progressing toward the task goal.

3 Experimental Setup

3.1 Simulation Environment and Data

Experiments are conducted in a PyBullet simulation [16] of the Franka Emika Panda 7-DOF manipulator [17]. The robot is controlled via end-effector position commands, with inverse kinematics translating these into joint torques. The simulation runs at 200 Hz ($\Delta t = 1/200$ s), and the workspace is constrained to a volume of $0.8 \times 0.8 \times 0.4$ meters.

For motion data, we use the LASA handwriting dataset [18], which provides diverse 2D trajectory demonstrations. These are lifted into 3D by assigning a fixed z-height, resampled for temporal consistency, smoothed using a low-pass filter, and encoded as DMPs using 25 basis functions. These DMPs serve as nominal motion plans across both static and dynamic obstacle avoidance tasks.

3.2 Parameters and Implementation

DMP parameters are tuned to ensure stable convergence and smooth trajectory reproduction. Specifically, the primary gain α is set to 25.0 to promote exponential convergence, while the damping gain $\beta = 6.25$ achieves critical damping. The phase decay constant $\alpha_z = 4.17$ regulates trajectory progression over time. A total of 25 basis functions are used to capture complex motion shapes. To support online adaptation, the coupling mechanism within DMPs uses a recovery gain $\alpha_e = 2.5$ and curvature-based slowdown gain $k_c = 1250$, allowing effective modulation in response to perturbations.

For the STT module, the feedback control gain K is set to 10.0, ensuring prompt yet stable deviation correction. The tube width δ_γ is fixed at 0.1 meters, defining a soft safety boundary around the

nominal trajectory. Error saturation bounds are limited to $[-0.99,\ 0.99]$ within activation functions to prevent instability in high-deviation regimes.

All simulations are run on a workstation with an Intel Core i7-12700K CPU and 16 GB RAM. The implementation is publicly available at the project repository[1].

4 Comparison and Results

We evaluate SafeDMP against two standard baselines: NODE + CBF [9] and DMP + APF [11]. The results are presented in three stages: qualitative comparisons of safety and robustness, quantitative benchmarking of efficiency and accuracy, and scenario-based demonstrations of trajectory execution.

4.1 Qualitative Comparison

Table- 1 summarizes the strengths and weaknesses of all methods along two criterion's.

Table 1: Qualitative comparison of motion generation methods.

Method	Safety and Robustness	Computational Efficiency
NODE + CBF [9]	Provides formal safety guarantees but requires solving an optimization at each timestep, making recovery from fast perturbations sluggish.	Computationally expensive due to repeated quadratic programming, unsuitable for real-time control.
DMP + APF [11]	Lightweight approach but prone to oscillations, local minima, and unsafe transients near obstacles.	Efficient to compute, but requires additional tuning of potentials and does not scale well to cluttered environments.
SafeDMP (ours)	Ensures provable safety through Spatio-Temporal Tubes. Avoids oscillations, guarantees bounded deviation, and responds smoothly to disturbances.	Extremely lightweight closed-form updates enable real-time performance without optimization overhead.

Figure 2 highlights the qualitative differences between SafeDMP and the APF-based baseline. In subplot (a), when subjected to identical perturbations, DMP+APF (a.1) undergoes abrupt deviations and slower convergence, while SafeDMP (a.2) achieves smooth, bounded recovery. In subplot (b), during obstacle interaction, DMP+APF (b.1) exhibits oscillatory and unsafe trajectories near the constraint, whereas SafeDMP (b.2) stays strictly inside the safe tube while preserving task-directed motion. These results confirm the qualitative claims of Table 1.

We further benchmark our integrated SafeDMP approach against NODE + CBF [9], which offers formal safety guarantees but relies on online optimization. For fairness, we also include **JIT-accelerated** variants of NODE + CBF and SafeDMP. Here, JIT (Just-In-Time) compilation is applied via the `jax.jit` backend to pre-compile repeated computations into optimized machine code. This reduces interpreter overhead but does not change the underlying algorithmic complexity, making it useful for comparing raw method efficiency.

Compared to optimization-based NODE–CBF baselines, our SafeDMP framework provides significant efficiency and accuracy gains:

[1] https://github.com/Tiwari-Pranav/ghost-In-the-Arm

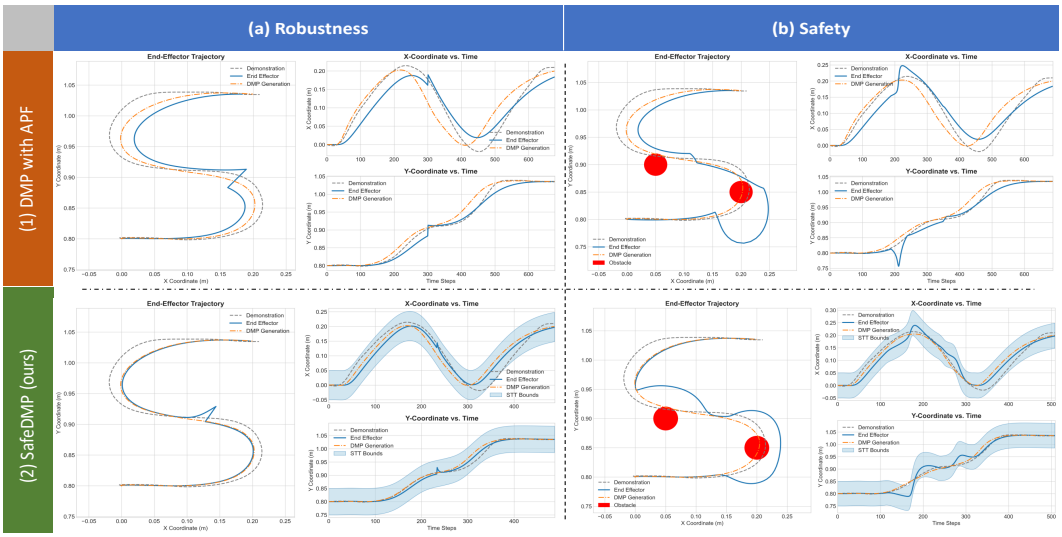

Figure 2: (a) Perturbation response: (1) DMP+APF shows abrupt deviations with erroneous tracking and slow recovery; (2) SafeDMP reroutes smoothly with bounded error within minimal time. (b) Obstacle avoidance: (1) DMP+APF oscillates near obstacles; (2) SafeDMP stays within the safety tube while maintaining task-directed motion.

Table 2: Comparative evaluation of motion generation methods in terms of computational efficiency and trajectory accuracy.

Method	**Computational Efficiency**		**Accuracy**
	Exec. Time (s)	Mem. (KB)	MAE
Baseline 1 (NODE + CBF) [9]	0.3207	–	3.804
Baseline 2 (NODE + CBF, JIT) [9]	1.8e-3	0.311	3.804
SafeDMP	**1.03e-4**	**0.109**	**0.0426**
SafeDMP (JIT)	**3.7e-5**	**0.161**	**0.0426**

- **Execution Time (s):** Average wall clock time per rollout, capturing real-time feasibility. Our method achieves a **99.97% reduction** in runtime relative to Baseline 1 (NODE+CBF), and up to **97.9% faster** execution compared to the JIT-accelerated Baseline 2.
- **Trajectory accuracy:** Mean absolute error between the demonstrated and generated trajectories. The mean absolute error (MAE) is reduced by **98.9%**, preserving demonstrated motion shapes far more faithfully than Neural ODE tracking.
- **Memory Footprint (KB):** Overall RAM usage during task execution. Memory consumption is reduced by **65%** (non-JIT) and **48%** (JIT), enabling lightweight deployment in resource-constrained settings.

In all metrics, our method not only guarantees safety but also delivers real-time performance with a large advantage in both speed and accuracy.

All methods are evaluated in PyBullet [16] with identical LASA trajectories, obstacle configurations, and robot model (Franka Panda) [17].

We evaluated the effectiveness of our integrated motion generation framework, SafeDMP, in ensuring real-time trajectory generation, safe obstacle avoidance, and reactive task switching in dynamic environments. The results are organized into three core scenarios: nominal trajectory execution, obstacle avoidance with recovery, and dynamic task transitions.

4.2 Nominal Trajectory Execution

In the absence of obstacles, the robot follows the planned trajectory purely based on DMP generation. As shown in Figure 3, the robot smoothly executes the desired path and converges to the goal without external interventions. The motion profiles in position and velocity confirm the expected stability and convergence, with no abrupt changes or oscillations. A demonstration video of this execution is available: here.

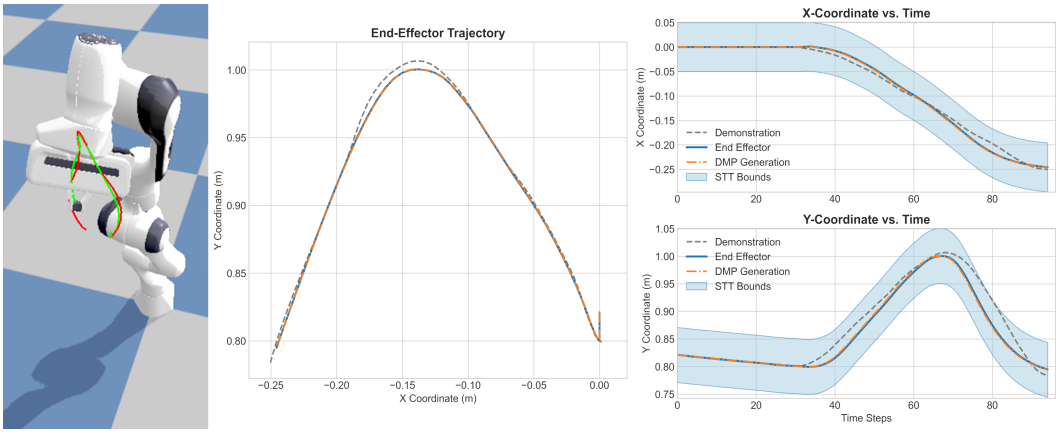

Figure 3: Nominal execution scenario using DMPs. The robot follows a smooth, stable path and reaches the goal without intervention in the absence of obstacles.

To understand the benefit of this integrated approach, we compare it to a common alternative: DMPs combined with artificial potential fields (APFs). Although APF-based methods can guide the robot around obstacles, they often result in abrupt accelerations and lack formal guarantees of safety or smooth recovery. In contrast, the proposed SafeDMP framework ensures that all deviations remain within bounded regions, and recovery is handled smoothly through formally defined dynamics.

4.3 Obstacle Avoidance and Safe Recovery

In dynamic environments, obstacles are introduced during motion execution. The STT controller reroutes the path in real time to maintain safety. As shown in Figure 4, the robot performs safe deviations, avoids collisions, and realigns with the goal trajectory after clearing the obstacle. The position and velocity profiles further confirm that the recovery is smooth, with no sudden discontinuities or overshoot, validating the effectiveness of the coupling-based damping and STT blending. A demonstration video of this scenario is available: here.

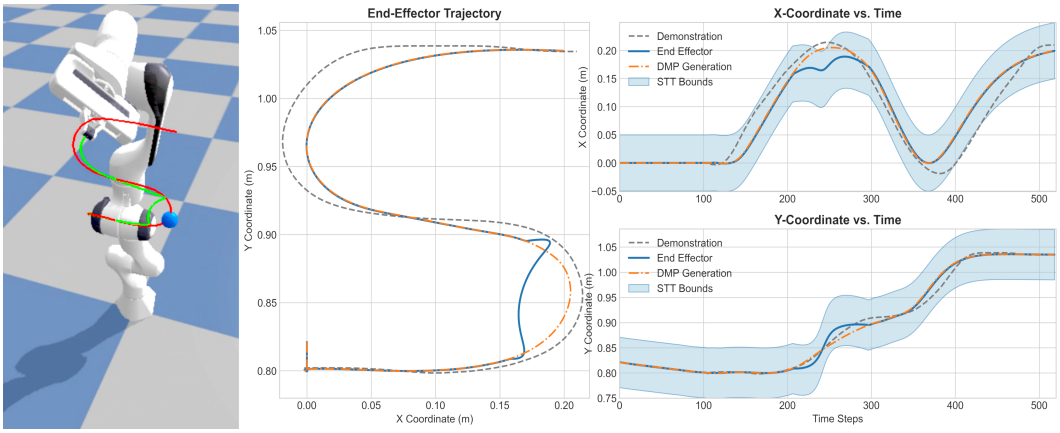

Figure 4: Trajectory during dynamic obstacle (blue ball) avoidance.

4.4 Reactive Task Transition

To evaluate adaptability to new task goals, the robot is dynamically commanded to switch from one nominal trajectory to another mid-execution. Figure 5 shows that the system effectively reconfigures the DMP trajectory and continues execution without interruption. The motion profiles reveal smooth transitions with no instability or control discontinuity, thanks to the continuous blending and reparameterization of the trajectory. A demonstration video of reactive switching is available: here.

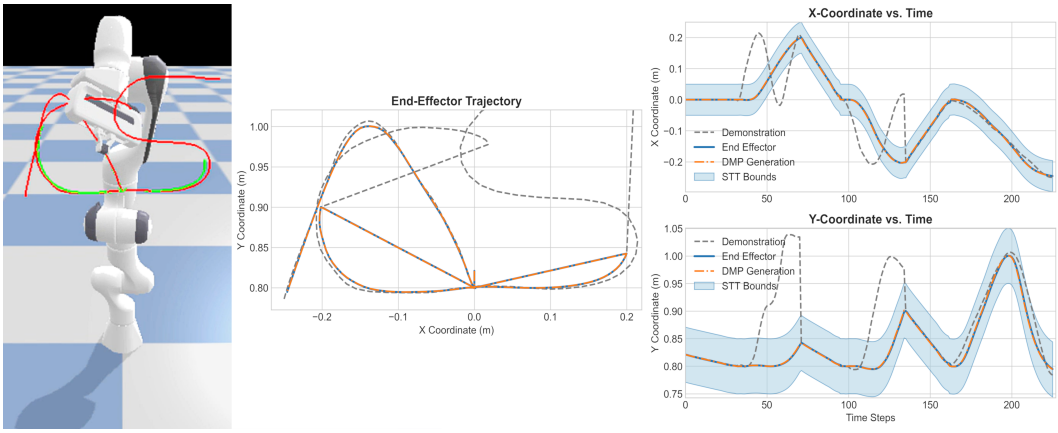

Figure 5: Reactive task transition scenario. The robot adapts in real time to changing goals, maintaining safe and stable motion.

5 Conclusion

We presented a modular framework SafeDMP, for safe and adaptive motion generation that integrates Dynamic Movement Primitives (DMPs) with Spatio-Temporal Tubes (STTs). By avoiding online constrained optimization and instead using closed-form safety guarantees, the approach enables real-time execution at high control frequencies. An adaptive coupling mechanism further enhances robustness, ensuring stable recovery under disturbances, obstacle interactions, and reactive task switching.

Extensive experiments on the Franka Emika Panda demonstrated that our framework not only preserves safety but also delivers substantial efficiency and accuracy gains over optimization-based Neural ODE + CBF baselines. In particular, execution time was reduced by up to **99.97%**, trajectory error decreased by **98.9%**, and memory footprint lowered by up to **65%**. These results highlight the suitability of the proposed method for deployment in collaborative, assistive, and industrial robotics, where real-time safety and adaptability are essential for human-centered environments.

6 Limitations and Future Work

While the proposed framework demonstrates strong performance in real-time, adaptive, and safe motion generation, several limitations remain. The current implementation assumes spherical obstacle geometries and relies on manual parameter tuning, which may limit scalability and generalization. Future work will aim to extend the framework to handle arbitrary-shaped obstacles using geometric approximations or learned representations, and to incorporate adaptive parameter tuning mechanisms. Another promising direction is the integration of a high-level spatio-temporal logic (STL) planner to guide motion planning over complex task specifications, enabling more expressive and globally informed decision-making.

References

[1] J. Mendez-Mendez, L. P. Kaelbling, and T. Lozano-Pérez. Embodied lifelong learning for task and motion planning. In *Conference on Robot Learning*, pages 2134–2150. PMLR, 2023.

[2] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[3] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500, 2023. doi:10.1109/ICRA48891.2023.10160591.

[4] Y. Tu, Y. Wang, H. Zhang, W. Chen, and J. Zhang. Language-embedded 6d pose estimation for tool manipulation. *IEEE Robotics and Automation Letters*, 10(9):8618–8625, 2025. doi:10.1109/LRA.2025.3587559.

[5] W. Su, R. Zhu, Z. Chen, W. Li, and G. S. Chirikjian. Put a lid on it! a learning-free method to cap a container via physical simulations. In *2025 22nd International Conference on Ubiquitous Robots (UR)*, pages 313–319, 2025. doi:10.1109/UR65550.2025.11078134.

[6] R. Das, A. Basu, and P. Jagtap. Spatiotemporal tubes for temporal reach-avoid-stay tasks in unknown systems. *IEEE Transactions on Automatic Control*, pages 1–8, 2025. doi:10.1109/TAC.2025.3592723.

[7] H. Pan, M. Luo, J. Wang, T. Huang, and W. Sun. A safe motion planning and reliable control framework for autonomous vehicles. *IEEE Transactions on Intelligent Vehicles*, 9(4):4780–4793, 2024. doi:10.1109/TIV.2024.3360418.

[8] S. Magdici and M. Althoff. Fail-safe motion planning of autonomous vehicles. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 452–458, 2016. doi:10.1109/ITSC.2016.7795594.

[9] F. Nawaz, S. Peng, L. Lindemann, N. Figueroa, and N. Matni. Reactive temporal logic-based planning and control for interactive robotic tasks. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 12108–12115. IEEE, 2024.

[10] A. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal. Dynamical movement primitives: Learning attractor models for motor behaviors. *Neural Computation*, 25:328–373, 02 2013. doi:10.1162/NECO_a_00393.

[11] H. Hoffmann, P. Pastor, D.-H. Park, and S. Schaal. Biologically-inspired dynamical systems for movement generation: Automatic real-time goal adaptation and obstacle avoidance. In *2009 IEEE International Conference on Robotics and Automation*, pages 2587–2592, 2009. doi:10.1109/ROBOT.2009.5152423.

[12] S. Calinon, D. Bruno, and D. G. Caldwell. A task-parameterized probabilistic model with minimal intervention control. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3339–3344, 2014. doi:10.1109/ICRA.2014.6907339.

[13] G. Franzese, R. Prakash, C. D. Santina, and J. Kober. Generalizable motion policies through keypoint parameterization and transportation maps. *IEEE Transactions on Robotics*, 41:4557–4573, 2025. doi:10.1109/TRO.2025.3582821.

[14] A. Paraschos, C. Daniel, J. Peters, and G. Neumann. Probabilistic movement primitives. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 2616–2624, Red Hook, NY, USA, 2013. Curran Associates Inc.

[15] A. Paraschos, G. Neumann, and J. Peters. A probabilistic approach to robot trajectory generation. In *2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pages 477–483, 2013. doi:10.1109/HUMANOIDS.2013.7030017.

[16] E. Coumans and Y. Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning, 2016.

[17] Franka Emika GmbH. Panda robot arm. `https://www.franka.de/`, 2017. Accessed: 2025-08-20.

[18] S. M. Khansari-Zadeh and A. Billard. Learning stable nonlinear dynamical systems with gaussian mixture models. *IEEE Transactions on Robotics*, 27(5):943–957, 2011. doi:10.1109/TRO.2011.2159412.

Appendix

A Dynamic Movement Primitive Details

A.1 Canonical System

The canonical system provides a monotonically decreasing phase variable:

$$\tau \dot{z} = -\alpha_z z, \quad z(0) = 1, \tag{9}$$

where $\alpha_z > 0$ ensures convergence $z(t) \to 0$ as $t \to \infty$.

A.2 Learning from Demonstrations

Given a demonstrated trajectory $\{x_d(t), \dot{x}_d(t), \ddot{x}_d(t)\}_{t=0}^{T}$, the target forcing function is computed by rearranging (1):

$$f_{\text{target}}(z) = \tau \ddot{x}_d(t) - \alpha_z(\beta_z(g - x_d(t)) - \tau \dot{x}_d(t)). \tag{10}$$

A.3 Forcing Function Parameterization

The forcing function is parameterized using a weighted combination of Gaussian basis functions:

$$f(z) = \frac{\sum_{i=1}^{N} w_i \psi_i(z) z}{\sum_{i=1}^{N} \psi_i(z)} (g - x_0), \tag{11}$$

where:

- $w_i \in \mathbb{R}^d$ are learnable weight parameters
- $\psi_i(z) = \exp(-h_i(z - c_i)^2)$ are Gaussian basis functions
- c_i and h_i are the centers and widths of the basis functions
- $(g - z_0)$ provides spatial scaling invariance

The weights $\{w_i\}_{i=1}^{N}$ are learned via locally weighted regression

B Spatio-Temporal Tube Details

B.1 Error Formulation and Control Law

The tube-based control system uses a normalized error representation. Define the sum and difference of bounds:

$$\rho_{i,s}(t) = \rho_{i,U}(t) + \rho_{i,L}(t) \tag{12}$$

$$\rho_{i,d}(t) = \rho_{i,U}(t) - \rho_{i,L}(t) \tag{13}$$

The normalized error for dimension i is:

$$e_i(x_i, t) = 2\rho_{i,d}^{-1}(t) \left(x_i(t) - \frac{1}{2}\rho_{i,s}(t) \right), \tag{14}$$

which maps the tube interior to $(-1, 1)$ and centers the error at zero when $x_i(t)$ is at the tube center.

The control function that enforces tube constraints is:

$$\xi_i(x_i, t) = 4\rho_{i,d}^{-1}(t) \left(1 - e_i(x_i, t)^2\right), \tag{15}$$

which provides maximum control authority at the tube boundaries and minimum at the center.

B.2 Finite-Time Convergence Control

The STT control law ensures finite-time convergence to the target set while maintaining tube constraints:

$$u_i(x_i, t) = -k\, \xi_i(x_i, t)\, \epsilon_i(x_i, t), \quad k > 0, \tag{16}$$

where the error term $\epsilon_i(x_i, t)$ is defined as:

$$\epsilon_i(x_i, t) = \ln\left(\frac{1 + e_i(x_i, t)}{1 - e_i(x_i, t)}\right). \tag{17}$$

This logarithmic error term ensures:

- $\epsilon_i \to -\infty$ as $e_i \to -1$ (approaching lower bound)
- $\epsilon_i = 0$ when $e_i = 0$ (at tube center)
- $\epsilon_i \to +\infty$ as $e_i \to +1$ (approaching upper bound)