Latent Adaptive Planner for Dynamic Manipulation

Donghun Noh[1♠], **Deqian Kong**[1,2♠], **Minglu Zhao**[1], **Andrew Lizarraga**[1], **Jianwen Xie**[2], **Ying Nian Wu**[1♣], **Dennis Hong**[1♣]
[1]UCLA [2]Lambda, Inc.
♠Equal contribution ♣Equal advising

Abstract: We present the Latent Adaptive Planner (LAP), a trajectory-level latent-variable policy for dynamic nonprehensile manipulation (e.g., box catching) that formulates planning as inference in a low-dimensional latent space and is learned effectively from human demonstration videos. During execution, LAP achieves real-time adaptation by maintaining a posterior over the latent plan and performing variational replanning as new observations arrive. To bridge the embodiment gap between humans and robots, we introduce a model-based proportional mapping that regenerates accurate kinematic-dynamic joint states and object positions from human demonstrations. Through challenging box catching experiments with varying object properties, LAP demonstrates superior success rates, trajectory smoothness, and energy efficiency by learning human-like compliant motions and adaptive behaviors. Overall, LAP enables dynamic manipulation with real-time adaptation and successfully transfer accross heterogeneous robot platforms using the same human demonstration videos.

Keywords: Imitation Learning, Dynamic Nonprehensile Manipulation, Latent Space Planning, Classical Variational Bayes, Test-time Adaptation

1 Introduction

Dynamic manipulation, which involves controlling objects through rapid contact changes and complex physical interactions [1, 2], remains a fundamental challenge in robotics. While humans naturally perform throwing, catching, and rapid transfers, robots remain confined to slow, conservative movements. Achieving true dynamic manipulation requires real-time consideration of diverse physical properties of objects (mass, friction, elasticity) [3, 4], changes in contact states, and joint torques [5, 6, 7], yet these dynamic characteristics are difficult to model analytically and vary significantly across different objects [8, 9]. Recent advances like Tossingbot [10], FlingBot [11], and Chi et al. [12] have shown progress in specific scenarios, yet predominantly rely on self-supervised learning requiring thousands of robot trials. For this reason, this approach lacks scalability since exploration with heavier objects or mobile-based platforms like humanoids become unsafe and leads to significant hardware damage upon failure. While simulation offers safer training, accurately modeling dynamic manipulation physics remains challenging, limiting sim-to-real transfer.

Recently, imitation learning has shown promising results on static or quasi-static manipulation tasks. However, recent approaches such as diffusion policy are limited by slow inference speeds that cannot meet the real-time requirements of dynamic manipulation. Moreover, most datasets for policy training consider only end-effector positions while overlooking critical dynamic elements such as contact information, joint torques, and grasping forces [13, 14, 15]. While more accurate data can be collected through teleoperation, this approach presents practical barriers including high implementation costs, logistical challenges for achieving scale and diversity, and technical difficulties in capturing complete physical interaction data [16, 17, 18]. Additionally, demonstration-based learning approaches struggle with temporal consistency and trajectory smoothness when transferring skills from human demonstrations to robotic systems [19]. Although recent generative modeling approaches such as sequence modeling [20] and diffusion models [13] have addressed some of these challenges, they still exhibit limitations in handling the long-term dependencies and environmental contingencies inherent in dynamic manipulation tasks [18].

In this work, we introduce Latent Adaptive Planner (LAP), which formulates planning as latent space inference. Our approach addresses key challenges in visuomotor policy learning through a principled variational replanning framework that maintains temporal consistency while efficiently adapting to environmental changes. By leveraging a latent variable that functions as an abstract plan, LAP enables more coherent long-horizon planning while maintaining the flexibility needed for real-time adaptation. The system employs Bayesian updating in latent space to incrementally refine plans as new observations become available, balancing computational efficiency with real-time adaptability. Furthermore, by model-based proportional mapping to regenerate accurate kinematic-dynamic joint states and object positions from human demonstration videos, our approach bridges the gap between human motion and robotic execution, capturing the adaptive and energy-efficient qualities inherent in human manipulation skills. Importantly, this regeneration methodology allows us to create versatile datasets suitable for training diverse robotic platforms on the same manipulation tasks without requiring direct robot data collection.

We demonstrate our approach through box catching, one of the most challenging reactive manipulation tasks. Unlike controlled release scenarios such as throwing and fling, catching requires real-time adaptation to unpredictable trajectories while managing objects without secure grasps [21, 22]. Boxes which have non-trivial weight and volumes tumble chaotically with asymmetric drag, forcing robots to predict contact locations and orientations while coordinating dual-arm movements with millisecond-scale precision and determining appropriate contact forces to absorb impact without dropping the object [21, 23]. Through this task, we show that imitation learning can achieve energy-efficient dynamic manipulation while bypassing the intractable formulations required by optimization methods, demonstrating the potential for broader dynamic manipulation challenges.

2 Related Works

Human Demonstration-based Robot Data Generation Those approaches that utilize human videos offer greater scalability but face embodiment gaps between humans and robots. Some methods address this by requiring hybrid datasets with both human and robot demonstrations [24, 25]. Others extract action labels from videos through object tracking [26, 27, 28, 29] or inverse modeling techniques [30, 31, 32]. Cross-embodiment techniques like RoviAug [33] but when applied to human-to-robot transfer [34], typically still require robot data.

Recent work has explored more efficient ways to leverage human demonstrations. Phantom [35] enables zero-shot transfer of policies trained solely on human demonstrations to robot embodiments through simple data editing techniques, eliminating the need for robot data collection. Similarly, DROID [36] demonstrates that collecting diverse manipulation data across multiple environments creates more robust policies capable of generalizing to new scenarios. DexMV [37] introduced a particularly relevant approach by establishing a framework for learning dexterous manipulation from human videos through demonstration translation techniques. This approach extracts 3D hand and object poses from videos and maps human hand trajectories to robot joint torques via inverse dynamics, closely aligning with our method of processing video data to generate robot demonstrations. Our specific data generation process, which involves human pose estimation, object pose extraction, and robot-specific scaling and kinematic retargeting, is detailed in Section 3.1.

Imitation Learning and Latent Space Planning Imitation learning enables robots to acquire skills by mimicking demonstrations, transforming complex control into supervised learning problems that map observations to actions [38], though traditional approaches like Behavior Cloning struggle with distribution shift during deployment [39]. Recent approaches have reframed decision-making as sequence modeling [20, 40] to capture temporal dependencies in state-action sequences. Diffusion models offer an alternative paradigm for imitation learning, with Chi et al. [13] generating multi-modal action distributions through iterative denoising processes and Janner et al. [41] leveraging classifier-guided diffusion models.

Latent variable models address temporal consistency challenges by encoding trajectory-level information that captures long-term dependencies. Yang et al. [42], Paster et al. [43] investigate Decision Transformer's overfitting to environment contingencies and propose latent variable solutions that encode trajectory-level information. Latent Plan Transformer [44] formulates planning as latent space inference and decouples trajectory generation from return estimation, enabling more coherent long-horizon planning. Earlier works [45, 46] propose VAE-based models [47] for temporally extended policies, but lack the adaptive replanning capabilities central to our framework. Our work builds upon these foundations but introduces classical variational Bayes learning [48, 49] for precise plan inference and principled Bayesian updating in latent space, enabling efficient plan refinement as new observations become available as detailed in Section 3.2.

3 Method

The Latent Adaptive Planner (LAP) presents a novel methodology for dynamic manipulation tasks, addressing the inherent challenges in human demonstration learning and real-time environmental adaptation. Within this framework, effective data regeneration from human demonstrations is achieved, and adaptive responses to dynamic environments in real-time are enabled. The integration of latent space representation with variational inference allows for precise mapping between demonstration data and robot execution parameters while maintaining adaptability to unpredicted environmental changes.

3.1 Robot Model-Based Data Regeneration from Human Demonstration Videos

A comprehensive framework is developed to regenerate robot training data from human demonstration videos, enabling the extraction of dynamic information without direct robot interaction. As illustrated in Figure 1, the approach consists of three main components: scene state estimation, object-robot proportional mapping, and kinematic-dynamic joint state reconstruction. This data regeneration pipeline systematically transforms human demonstrations into robot execution parameters while preserving essential dynamic characteristics.

Scene State Estimation Scene state estimation is performed to extract spatial information from video frames. This process involves detecting and tracking box objects to determine their positions and dimensions within the scene coordinate system. Additionally, human pose estimation is conducted to track the demonstrator's joint positions, providing the motion data necessary for the subsequent mapping process.

Object-Robot Proportional Mapping To adapt the human demonstration environment to the robot's workspace, object-robot proportional mapping is implemented. This process transforms detected objects' positions and dimensions to the robot's base coordinate frame:

$$^{R}\mathbf{p}_{\text{obj}} = \mathbf{T}_S^R \cdot {}^{S}\mathbf{p}_{\text{obj}} \tag{1}$$

$$^{R}\mathbf{d}_{\text{obj}} = s \cdot {}^{S}\mathbf{d}_{\text{obj}} \tag{2}$$

where $^{R}\mathbf{p}_{\text{obj}}$ represents the object position expressed in robot frame R, $^{S}\mathbf{p}_{\text{obj}}$ represents the object position expressed in scene frame S, $\mathbf{T}_S^R$ is the transformation matrix from scene frame to robot frame, $^{R}\mathbf{d}_{\text{obj}}$ and $^{S}\mathbf{d}_{\text{obj}}$ denote the object dimensions expressed in the robot and scene frames respectively, and s is the scaling factor determined by the ratio between the robot arm length obtained from the robot model and the human arm length measured in pixels from the video.

Kinematic-Dynamic Joint State Reconstruction The robot's kinematic and dynamic states are reconstructed from the human demonstration. The joint positions are obtained through direct joint position mapping between human and robot:

$$\mathbf{q} = f_{\text{map}}(\mathbf{q}_{\text{human}}) \tag{3}$$

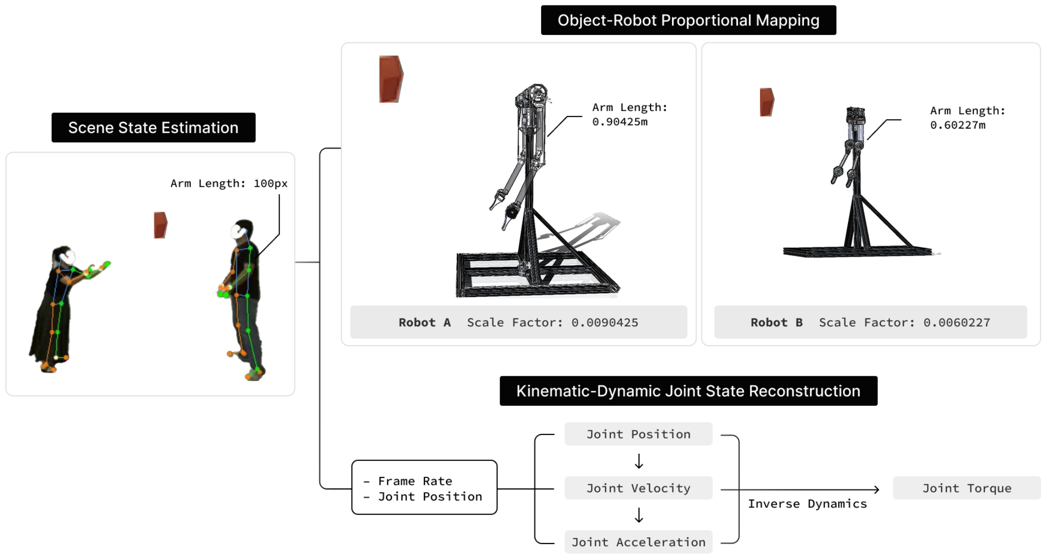

Figure 1: Robot Model-Based Data Regeneration Pipeline from Human Demonstration Videos. The pipeline consists of three main stages: (1) Scene State Estimation, which detects and tracks box objects and human pose from demonstration videos; (2) Object-Robot Proportional Mapping, which scales object dimensions and positions relative to the robot base frame; and (3) Kinematic-Dynamic Joint State Reconstruction, which maps human joint positions to robot configurations, differentiates to obtain velocities and accelerations, and computes required joint torques through inverse dynamics including external forces.

where $\mathbf{q}$ represents the robot joint positions and f_{map} is the mapping function that accounts for the different kinematic structures between human and robot.

Joint velocities and accelerations are calculated by differentiating the joint positions with respect to time using the video's frame rate:

$$\dot{\mathbf{q}} = \frac{d\mathbf{q}}{dt} \approx \frac{\mathbf{q}_{t+\Delta t} - \mathbf{q}_t}{\Delta t} \tag{4}$$

$$\ddot{\mathbf{q}} = \frac{d^2\mathbf{q}}{dt^2} \approx \frac{\dot{\mathbf{q}}_{t+\Delta t} - \dot{\mathbf{q}}_t}{\Delta t} \tag{5}$$

where Δt is the time interval between consecutive frames.

With the complete kinematic state ($\mathbf{q}$, $\dot{\mathbf{q}}$, $\ddot{\mathbf{q}}$), inverse dynamics is employed to compute the required joint torques for task execution:

$$\boldsymbol{\tau} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) + \mathbf{J}^T(\mathbf{q})\mathbf{F}_{\text{ext}} \tag{6}$$

where $\mathbf{M}(\mathbf{q})$ is the inertia matrix, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ accounts for Coriolis and centrifugal effects, $\mathbf{G}(\mathbf{q})$ represents gravitational forces, $\mathbf{J}(\mathbf{q})$ is the Jacobian matrix, and $\mathbf{F}_{\text{ext}}$ denotes external forces encountered during interaction with objects. The inclusion of external forces is crucial for accurately modeling tasks involving object manipulation and environmental contact.

This comprehensive data regeneration pipeline enables the transformation of human demonstrations into robot execution data with full kinematic and dynamic information, facilitating more efficient transfer of human skills to robotic systems.

3.2 Latent Adaptive Planner (LAP)

Our Latent Adaptive Planner (LAP) introduces a powerful framework for modeling and executing complex dynamic manipulation tasks based on human demonstrations.

Model Given a trajectory $\mathbf{x} = \{(o_t, a_t)\}_{t=1}^T$ consisting of observations and actions from demonstrations, LAP defines a joint probability distribution:

$$p_\theta(\mathbf{x}, \mathbf{z}) = p(\mathbf{z})p_\theta(\mathbf{x}|\mathbf{z}) \tag{7}$$

where $\mathbf{z} \in \mathbb{R}^d$ is the latent plan vector, $p(\mathbf{z})$ is the prior model, and $p_\theta(\mathbf{x}|\mathbf{z})$ is the trajectory generator. The prior model $p(\mathbf{z})$ is an isotropic Gaussian $\mathbf{z} \sim \mathcal{N}(0, I_d)$.

The trajectory generator $p_\theta(\mathbf{x}|\mathbf{z})$ is a conditional autoregressive model that produces actions based on the current state, historical context, and the latent plan vector:

$$p_\theta(\mathbf{x}|\mathbf{z}) = \prod_{t=1}^{T} p_\theta(\mathbf{x}_{(t)}|\mathbf{x}_{(t-K)}, ..., \mathbf{x}_{(t-1)}, \mathbf{z}) \tag{8}$$

where $\mathbf{x}_{(t)} = (o_t, a_t)$, and K is the context length. The observation o_t includes box position, contact state, and previous timestep joint positions, velocities, and torques ($\mathbf{q}_{t-1}$, $\dot{\mathbf{q}}_{t-1}$, $\boldsymbol{\tau}_{t-1}$). Action a_t consists of current timestep joint positions, velocities, and torques ($\mathbf{q}_t$, $\dot{\mathbf{q}}_t$, $\boldsymbol{\tau}_t$). $\mathbf{x}_{(0)}$ is a special learnable token. θ is implemented using a causal Transformer model [50]. The latent plan vector $\mathbf{z}$ controls each step of action generation through cross-attention mechanisms as $p_\theta(a_t|o_{t-K:t-1}, a_{t-K:t-1}, \mathbf{z})$. The action is assumed to follow a unimodal Gaussian distribution with fixed variance, i.e. $a_t \sim \mathcal{N}(g_\theta(o_{t-K:t}, a_{t-K:t-1}, \mathbf{z}), I_{|a|})$.

Classical Variational Bayes Learning We employ classical variational Bayes (VB) [48, 49, 51, 52] to learn LAP. Instead of learning an inference network (as in VAEs [47]), we directly optimize the latent plan vectors (local parameters in classical VB) for each trajectory using gradient descent. This iterative instance-level optimization offers greater precision in plan inference and greater flexibility that enables efficient adaptive (re)planning.

For each trajectory $\mathbf{x}$, we approximate the posterior distribution by $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$, where $\boldsymbol{\mu}$ is the posterior mean vector and $\boldsymbol{\sigma}^2$ is the variance-covariance matrix, assumed to be diagonal for computational efficiency. These are local parameters specific to each trajectory, while θ are global parameters shared by all samples. LAP can be learned by maximizing the evidence lower bound (ELBO),

$$\mathcal{L}(\theta, \boldsymbol{\mu}, \boldsymbol{\sigma}) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - D_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})), \tag{9}$$

where $\mathbf{z} \sim q(\mathbf{z}|\mathbf{x})$ is sampled using reparameterization trick [47].

The training procedure alternates between the optimization of local and global parameters:

(a) For each trajectory $\{\mathbf{x}_i\}_{i=1}^N$ in the mini-batch, optimize the corresponding local parameters $(\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i)$ to maximize $\mathcal{L}(\theta, \boldsymbol{\mu}_i, \boldsymbol{\sigma}_i)$ with T_{local} steps of gradient descent (we set $T_{\text{local}} = 16$ in experiments).

(b) Update the global parameters θ to maximize ${}^1\!/_N \sum_{i=1}^N \mathcal{L}(\theta, \boldsymbol{\mu}_i, \boldsymbol{\sigma}_i)$.

Detailed algorithms can be found in Appendix E. This learning procedure also shares the similar intuition of fast-slow learning discussed in [52] with fast learning of local parameters and slower updates of global parameters. This approach allows our model to efficiently learn an adaptive policy specific to manipulation scenarios while maintaining general knowledge about dynamics and control strategies across various tasks.

Variational Replanning for Test-time Adaptation We propose the variational replanning method that implements a principled Bayesian updating in the latent space, realizing the concept of *planning as latent space inference* [44], which significantly improves the model efficiency and stability.

With a learned LAP and initial observation o_1 during test time, we first sample $\mathbf{z} \sim p(\mathbf{z}|o_1) \propto p(\mathbf{z})p(\mathbf{x}_{0:1}|\mathbf{z})$ using T_{local} steps of gradient descent as the initial plan. As new observations become available, we aim to adaptively update the latent plan $\mathbf{z}$ within the Bayesian framework. With replanning horizon Δ, new observations $\mathbf{x}_{t+1:t+\Delta}$, and previously inferred posterior $q(\mathbf{z}|\mathbf{x}_{0:t}) = \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\sigma}_t^2)$, we update our belief through Bayesian updating:

$$q(\mathbf{z}|\mathbf{x}_{0:t+\Delta}) = \mathcal{N}(\boldsymbol{\mu}_{t+\Delta}, \boldsymbol{\sigma}^2_{t+\Delta}) \propto q(\mathbf{z}|\mathbf{x}_{0:t})p(\mathbf{x}_{t+1:t+\Delta}|\mathbf{x}_{0:t}, \mathbf{z}) \tag{10}$$

Specifically, we use classical VB to optimize the local parameters $\boldsymbol{\mu}_{t+\Delta}$ and $\boldsymbol{\sigma}_{t+\Delta}$ as:

$$\boldsymbol{\mu}_{t+\Delta}, \boldsymbol{\sigma}_{t+\Delta} = \arg\max_{\boldsymbol{\mu},\boldsymbol{\sigma}} \mathbb{E}_{q_{t+\Delta}}[\log p_\theta(\mathbf{x}_{t+1:t+\Delta}|\mathbf{x}_{0:t}, \mathbf{z})] - D_{\mathrm{KL}}(q_{t+\Delta}\|q_t), \tag{11}$$

where we define $q_{t+\Delta} \triangleq q(\mathbf{z}|\mathbf{x}_{0:t+\Delta})$ and $q_t \triangleq q(\mathbf{z}|\mathbf{x}_{0:t})$ for notational simplicity. This amortized computation enables efficient replanning with just $T_{\text{local}} = 1$ gradient step rather than the 16 steps required for initial planning.

The essence of our approach is treating the previously inferred distribution $q(\mathbf{z}|\mathbf{x}_{0:t})$ as the prior belief and updating it with a small number of gradient steps to obtain $q(\mathbf{z}|\mathbf{x}_{0:t+\Delta})$. As $t + \Delta \to T$ approaches the full trajectory length, the variational posterior $\mathcal{N}(\boldsymbol{\mu}_{t+\Delta}, \boldsymbol{\sigma}^2_{t+\Delta})$ converges to the posterior distribution of $\mathcal{N}(\boldsymbol{\mu}_T, \boldsymbol{\sigma}^2_T)$ that would be inferred given the complete trajectory. This establishes theoretical consistency between our incremental replanning approach and the optimal plan that would be determined with complete information. Detailed algorithms can be found in Appendix E. The integrated pipeline can be found in Figure 2.

Discussion LAP with variational replanning offers a principled middle ground between traditional planning paradigms. Open-loop planning ($\mathbf{z} \sim p(\mathbf{z}|o_1)$ inferred once) provides computational efficiency but suffers from error accumulation over time. Closed-loop planning (resampling $\mathbf{z} \sim p(\mathbf{z}|\mathbf{x}_{0:t})$ at each step) allows adaptation but at a high computational cost. Our approach incrementally updates the latent plan through Bayesian inference, treating previous distributions as prior beliefs. This maintains the adaptive benefits of closed-loop planning while preserving computational efficiency, essentially performing online learning of the latent plan as new observations arrive.

Meanwhile, the objective in Equation (11) performs a KL-regularized proximal step around $q_t(\mathbf{z})$: it increases the likelihood of the newly observed segment while staying close to q_t. Its optimizer is the exponential-tilting update in Equation (10), i.e., q_t reweighted by the new-segment likelihood. With q constrained to Gaussians, our single VB step implements a projected version of this tilt, acting as a small "trust-region" move in latent space that yields stable, real-time adaptation.

4 Experimental Results

We evaluate our LAP on a nonprehensile dynamic manipulation task of box catching, comparing its performance against several baseline methods including rule-based planning, behavior cloning, and Diffusion Policy approaches. This task represents a challenging example of nonprehensile dynamic manipulation, where the robot must initially interact with and control objects without stable grasping, using the dynamics of the arm movement to stabilize and then secure the incoming object.

4.1 Experimental Setup

Task Description Box catching constitutes the primary focus of this study, a challenging nonprehensile dynamic manipulation task where the robotic system must intercept, temporarily control, and securely grasp boxes of varying sizes, weights, and shapes handed or tossed by human operators. This task requires real-time trajectory adaptation, impact anticipation, and force modulation as the robot first manages dynamics using arm surfaces before transitioning to a stable grasp.

Hardware Setup To evaluate generalizability, experiments were conducted on two distinct robot platforms with different sizes and arm configurations. The experimental procedure involved data regeneration processes tailored to each robot's specific characteristics, followed by training and validation phases. Both platforms were equipped with a ZED2 stereo camera that continuously tracks object position and trajectory through real-time segmentation algorithms. Human demonstrators introduced variability by throwing boxes with diverse velocities, heights, and release angles, ensuring comprehensive training and testing conditions. Detailed specifications of the robotic platforms are provided in the Appendix B.

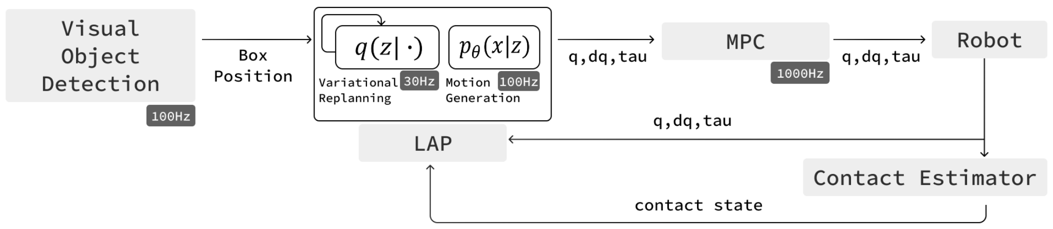

Figure 2: System Architecture for LAP Framework. The diagram illustrates our perception-planning-control pipeline. Camera input undergoes segmentation for object detection, providing box states to the Latent Adaptive Planner (LAP). LAP operates on a dual-rate hierarchy, performing updating of latent plan at 30Hz while generating motion commands at 100Hz. Reference joint positions, velocities, and torques from LAP are refined by Model Predictive Control before execution at the motor level. Joint states and contact state are fed back to the LAP for replanning. This architecture ensures smooth execution of dynamic nonprehensile manipulation tasks while respecting robot dynamics and physical constraints, enabling real-time adaptation to environmental changes.

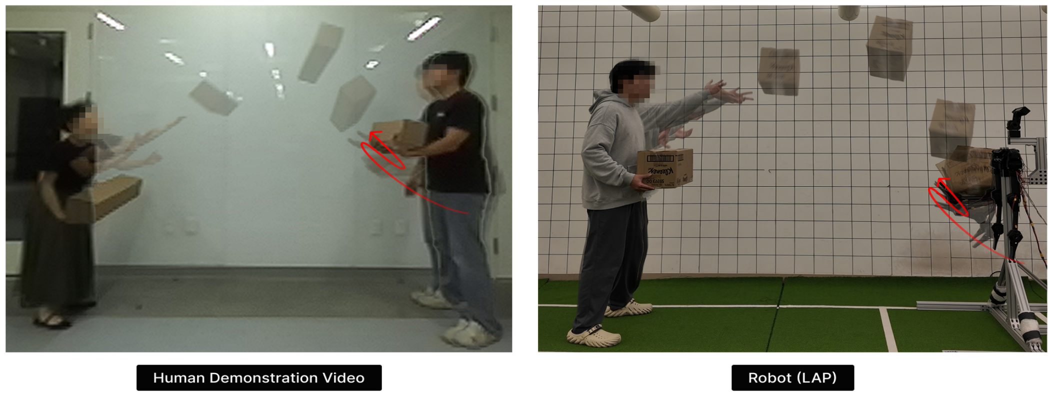

Figure 3: Impact-aware retreat trajectory for box catching motion learned with the Latent Adaptive Planner (LAP). The left panel shows a human demonstration where the subject absorbs impact by yielding their arm along the trajectory of the incoming box before returning to the nominal pose. The right panel shows the robot reproducing this compliant motion using LAP. The red curves indicate the impact-aware retreat trajectory that minimizes energy consumption during dynamic interaction.

Baselines We evaluated LAP against several baselines: (1) Model-based planner that determines optimal end-effector positions and arm configurations based on box dimensions and positions, using the inverse dynamics for superior tracking performance; (2) Transformer-based Behavior Cloning (BC) using a 3-layer decoder transformer (8 attention heads, batch size 12, hidden size 64, learning rate 2e-4, trained for 2500 epochs); (3) Diffusion Policy implemented with a 1D UNet architecture using 100 denoising steps based on the original repository [13]. Our LAP uses identical architecture and hyper-parameters as BC with an additional cross-attention layer, where the number of $\mathbf{z}$ is 16, each with dimension 64, with $T_{\text{local}} = 16$ for plan optimization. All learning-based methods were trained on the same dataset regenerated from human demonstrations.

4.2 Results and Analysis

Our experiments demonstrate that LAP consistently outperforms the baselines in all metrics and box types on both robot platforms. Table 1 summarizes the quantitative results for Robot A and Robot B, respectively. Figure 3 visualizes the impact-aware retreat trajectory for the the box catching motion learned by LAP using Robot B.

Model-based Planning The model-based approach achieved the highest success rate among all methods examined, however, this performance came at a significant energy cost due to its consis-

Table 1: Performance comparison of different methods on Robot A and B with various box types. Only trials where boxes were thrown within the reachable workspace of the fixed-base robots without excessive rotation were included in the success evaluation.

Robot	**Method**	**Success (count)**	**Energy (J)**	**Success (count)**	**Energy (J)**	**Success (count)**	**Energy (J)**
		Box A (66×16.5×14cm, 453g)		*Box B (61×30.5×30.5cm, 777g)*		*Box C (67.1×38.2×23cm, 660g)*	
Robot A	Model-based	**30/30**	74.99	**30/30**	57.31	**30/30**	70.46
	BC	26/30	31.39	28/30	38.50	26/30	37.40
	Diffusion	20/30	42.53	21/30	45.21	21/30	44.86
	LAP (Ours)	29/30	**11.47**	29/30	**12.12**	**30/30**	**11.41**
		Box A (66×16.5×14cm, 453g)		*Box D (48.3×22.9×24.8cm, 362g)*		*Box E (49.5×12.7×23.5cm, 365g)*	
Robot B	Model-based	**30/30**	33.12	**30/30**	33.17	**30/30**	26.09
	BC	27/30	15.82	28/30	15.40	**30/30**	13.78
	Diffusion	24/30	21.32	22/30	20.73	23/30	19.95
	LAP (Ours)	**30/30**	**7.14**	**30/30**	**6.64**	**30/30**	**6.86**

tent application of high torque values for ensuring secure grasps. This conservative control strategy, while effective for task completion, proved suboptimal from an efficiency perspective. The implementation utilized an inverse-dynamic controller to enhance tracking performance and response time. The impedance control implementation offered inherent advantages in energy efficiency through its mass-spring-damper system, which naturally produces compliant motions when receiving impact forces. However, this approach presented substantial adaptability challenges, requiring manual gain tuning for variations in box weight and impact force, a significant limitation in dynamic environments.

Learning-based Methods Behavioral Cloning (BC) and Diffusion-based methods demonstrated some capacity to capture human-like manipulation behaviors, but exhibited lower success rates and less stable action outputs compared to other approaches. While integration with Model Predictive Control (MPC) improved motion smoothness, these methods still failed to achieve optimal performance across the evaluation metrics.

LAP Performance LAP balanced high success rates with superior energy efficiency compared to other policy-based methods. This performance can be attributed to two key factors: efficient replanning capabilities and effective encoding of contact dynamics and human-like retreat motions in the latent variable space during training. By learning to implement retreat motions through motion planning rather than relying solely on control-based implementations, LAP achieved a balanced performance profile. Though not matching the perfect success rates of the model-based approach with MPC, LAP demonstrated significantly improved energy efficiency through learned human-like motions that effectively absorbed impact forces and minimized overall energy consumption.

5 Conclusion

We introduced the Latent Adaptive Planner (LAP), a trajectory-level latent-variable policy that treats planning as inference and is learned from human demonstration videos via a robot-model-based data regeneration pipeline. During execution , LAP performs variational replanning by updating a posterior over the latent plan as new observations arrive, yielding real-time adaptation in dynamic scenes. This test-time adaptation behaves like a small trust-region move in latent space, providing stability without solving a full optimization at every step.

Our experiments demonstrate LAP's effectiveness across different manipulation scenarios and validate its transferability to diverse robot configurations. By regenerating appropriate training data from the same human demonstration videos for each platform, LAP successfully adapts to distinct kinematic and dynamic properties of different robots. This cross-platform success confirms that the combination of data regeneration and latent planning provides a robust framework for dynamic manipulation tasks regardless of robot embodiment, opening promising directions for scalable visuomotor learning from human demonstrations.

Limitations

While our Latent Adaptive Planner demonstrates strong performance on dynamic manipulation tasks, several limitations remain and point to clear directions for future work.

Contact Modeling Our current approach employs a simplified binary contact representation and approximates contact locations at the wrist position. This simplification limits our ability to model the rich dynamics of multi-point contact interactions with varying force distributions, a critical aspect of dexterous manipulation where contact can occur across the entire arm surface with varying force distributions.

Kinematic Constraints To facilitate human-to-robot mapping, we restricted our demonstration analysis to three primary degrees of freedom (shoulder, elbow, and wrist pitch), excluding the full seven-DOF capability of human arms. This constraint, combined with our fixed-base manipulator assumption, prevents the system from leveraging whole-body coordination strategies that humans naturally employ during dynamic tasks, such as weight shifting and torso movements that contribute significantly to task performance and energy efficiency.

Perception Limitations Our reliance on 2D pose estimation from monocular video introduces fundamental constraints in spatial reasoning. The lack of depth information limits our ability to accurately reconstruct 3D trajectories and handle occlusions during dynamic movements, potentially affecting the fidelity of learned manipulation strategies in tasks requiring precise spatial awareness.

Prior Model Limitations From a generative modeling perspective, LAP employs a simple latent prior that may under-represent complex, multi-modal expert behaviors. More structured prior models, such as energy-based models [53, 54, 55, 56], diffusion models [57], and other expressive families [58, 59], could provide richer representations of the latent space, enabling more nuanced planning capabilities and better generalization to novel scenarios.

Future Work We aim to incorporate continuous contact models with multi-point force estimation, strengthen 3D perception for accurate spatial reasoning, extend to mobile platforms with whole-body coordination, and explore more expressive latent priors. These advances will be important for approaching human-level dexterity while maintaining the real-time performance required for dynamic manipulation.

Acknowledgments

Y. W. was partially supported by NSF DMS-2015577, NSF DMS-2415226, and a gift fund from Amazon. We gratefully acknowledge Lambda, Inc. for providing compute resources for this project. We thank Sirui Xie for early collaboration and conceptual discussions.

References

[1] M. T. Mason and K. M. Lynch. Dynamic manipulation. In *Proceedings of 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'93)*, volume 1, pages 152–159. IEEE, 1993.

[2] F. Ruggiero, V. Lippiello, and B. Siciliano. Nonprehensile dynamic manipulation: A survey. *IEEE Robotics and Automation Letters*, 3(3):1711–1718, 2018.

[3] D. Noh, H. Nam, M. S. Ahn, H. Chae, S. Lee, K. Gillespie, and D. Hong. Surface material dataset for robotics applications (smdra): A dataset with friction coefficient and rgb-d for surface segmentation. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 6275–6281. IEEE, 2021.

[4] D. Noh. *Control and Motion Planning for a Low-Inertia Multi-DoF Robotic Manipulator with Proprioceptive Actuators for Dynamic Manipulation*. University of California, Los Angeles, 2024.

[5] K. M. Lynch and M. T. Mason. Dynamic nonprehensile manipulation: Controllability, planning, and experiments. *The International Journal of Robotics Research*, 18(1):64–92, 1999.

[6] T. Tsuji, J. Ohkuma, and S. Sakaino. Dynamic object manipulation considering contact condition of robot with tool. *IEEE Transactions on Industrial Electronics*, 63(3):1972–1980, 2015.

[7] Y. Gong, G. Sun, A. Nair, A. Bidwai, R. CS, J. Grezmak, G. Sartoretti, and K. A. Daltorio. Legged robots for object manipulation: A review. *Frontiers in Mechanical Engineering*, 9: 1142421, 2023.

[8] K.-T. Yu, M. Bauza, N. Fazeli, and A. Rodriguez. More than a million ways to be pushed. a high-fidelity experimental dataset of planar pushing. In *2016 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 30–37. IEEE, 2016.

[9] N. Fazeli, R. Kolbert, R. Tedrake, and A. Rodriguez. Parameter and contact force estimation of planar rigid-bodies undergoing frictional contact. *The International Journal of Robotics Research*, 36(13-14):1437–1454, 2017.

[10] A. Zeng, S. Song, J. Lee, A. Rodriguez, and T. Funkhouser. Tossingbot: Learning to throw arbitrary objects with residual physics. *IEEE Transactions on Robotics*, 36(4):1307–1319, 2020.

[11] H. Ha and S. Song. Flingbot: The unreasonable effectiveness of dynamic manipulation for cloth unfolding. In *Conference on Robot Learning*, pages 24–33. PMLR, 2022.

[12] C. Chi, B. Burchfiel, E. Cousineau, S. Feng, and S. Song. Iterative residual policy: for goal-conditioned dynamic manipulation of deformable objects. *The International Journal of Robotics Research*, 43(4):389–404, 2024.

[13] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[14] M. Lepert, J. Fang, and J. Bohg. Phantom: Training robots without robots using only human videos. *arXiv preprint arXiv:2503.00779*, 2025.

[15] M. Shridhar, L. Manuelli, and D. Fox. Cliport: What and where pathways for robotic manipulation. In *Conference on robot learning*, pages 894–906. PMLR, 2022.

[16] A. Mandlekar, F. Ramos, B. Boots, S. Savarese, L. Fei-Fei, A. Garg, and D. Fox. Iris: Implicit reinforcement without interaction at scale for learning control from offline robot manipulation data. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4414–4420. IEEE, 2020.

[17] T. Zhang, Z. McCarthy, O. Jow, D. Lee, X. Chen, K. Goldberg, and P. Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 5628–5635. Ieee, 2018.

[18] A. Mandlekar, J. Booher, M. Spero, A. Tung, A. Gupta, Y. Zhu, A. Garg, S. Savarese, and L. Fei-Fei. Scaling robot supervision to hundreds of hours with roboturk: Robotic manipulation dataset through human reasoning and dexterity. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1048–1055. IEEE, 2019.

[19] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *arXiv preprint arXiv:1709.10087*, 2017.

[20] L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch. Decision transformer: Reinforcement learning via sequence modeling. In *Advances in Neural Information Processing Systems*, volume 34, pages 15084–15097, 2021.

[21] S. Kim, A. Shukla, and A. Billard. Catching objects in flight. *IEEE Transactions on Robotics*, 30(5):1049–1065, 2014.

[22] R. Lampariello, D. Nguyen-Tuong, C. Castellini, G. Hirzinger, and J. Peters. Trajectory planning for optimal robot catching in real-time. In *2011 IEEE International Conference on Robotics and Automation*, pages 3719–3726. IEEE, 2011.

[23] L. Yan, T. Stouraitis, J. Moura, W. Xu, M. Gienger, and S. Vijayakumar. Impact-aware bimanual catching of large-momentum objects. *IEEE Transactions on Robotics*, 2024.

[24] F. Ebert, Y. Yang, K. Schmeckpeper, B. Bucher, G. Georgakis, K. Daniilidis, C. Finn, and S. Levine. Bridge data: Boosting generalization of robotic skills with cross-domain datasets. *arXiv preprint arXiv:2109.13396*, 2021.

[25] P. Sharma, L. Mohan, L. Pinto, and A. Gupta. Multiple interactions made easy (mime): Large scale demonstrations data for imitation. In *Conference on robot learning*, pages 906–915. PMLR, 2018.

[26] S. Song, A. Zeng, J. Lee, and T. Funkhouser. Grasping in the wild: Learning 6dof closed-loop grasping from low-cost demonstrations. *IEEE Robotics and Automation Letters*, 5(3): 4978–4985, 2020.

[27] P. Mandikal and K. Grauman. Dexvip: Learning dexterous grasping with human hand pose priors from video. In *Conference on Robot Learning*, pages 651–661. PMLR, 2022.

[28] M. Cai, K. M. Kitani, and Y. Sato. Understanding hand-object manipulation with grasp types and object attributes. In *Robotics: science and systems*, volume 3, 2016.

[29] Y. Chen, Z. Tu, D. Kang, R. Chen, L. Bao, Z. Zhang, and J. Yuan. Joint hand-object 3d reconstruction from a single image with cross-branch feature fusion. *IEEE Transactions on Image Processing*, 30:4008–4021, 2021.

[30] S. Young, D. Gandhi, S. Tulsiani, A. Gupta, P. Abbeel, and L. Pinto. Visual imitation made easy. In *Conference on Robot learning*, pages 1992–2005. PMLR, 2021.

[31] K. Schmeckpeper, A. Xie, O. Rybkin, S. Tian, K. Daniilidis, S. Levine, and C. Finn. Learning predictive models from observation and interaction. In *European Conference on Computer Vision*, pages 708–725. Springer, 2020.

[32] K. Schmeckpeper, O. Rybkin, K. Daniilidis, S. Levine, and C. Finn. Reinforcement learning with videos: Combining offline observations with interaction. *arXiv preprint arXiv:2011.06507*, 2020.

[33] L. Y. Chen, C. Xu, K. Dharmarajan, M. Z. Irshad, R. Cheng, K. Keutzer, M. Tomizuka, Q. Vuong, and K. Goldberg. Rovi-aug: Robot and viewpoint augmentation for cross-embodiment robot learning. *arXiv preprint arXiv:2409.03403*, 2024.

[34] P. Sharma, D. Pathak, and A. Gupta. Egomimic: Imitation learning for robotic manipulation from third-person videos. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

[35] M. Lepert, V. Kumar, A. Zhan, H. Wang, S. Dasari, and S. Levine. Phantom: Learning generalizable mobile manipulation from human videos with motion guidance. In *Conference on Robot Learning (CoRL)*, 2023.

[36] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. In *arXiv preprint arXiv:2403.12945*, 2024.

[37] Y. Qin, Y.-H. Wu, S. Liu, H. Jiang, R. Yang, Y. Fu, and X. Wang. Dexmv: Imitation learning for dexterous manipulation from human videos. In *Robotics: Science and Systems (RSS)*, 2022.

[38] S. Schaal. Is imitation learning the route to humanoid robots? *Trends in cognitive sciences*, 3 (6):233–242, 1999.

[39] B. D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009.

[40] M. Janner, Q. Li, and S. Levine. Offline reinforcement learning as one big sequence modeling problem. In *Advances in Neural Information Processing Systems*, volume 34, pages 1273–1286, 2021.

[41] M. Janner, Y. Du, J. B. Tenenbaum, and S. Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022.

[42] S. Yang, D. Schuurmans, P. Abbeel, and O. Nachum. Dichotomy of control: Separating what you can control from what you cannot. In *International Conference on Learning Representations (ICLR)*, 2022.

[43] K. Paster, S. McIlraith, and J. Ba. You can't count on luck: Why decision transformers and rvs fail in stochastic environments. In *Advances in Neural Information Processing Systems*, volume 35, pages 38966–38979, 2022.

[44] D. Kong, D. Xu, M. Zhao, B. Pang, J. Xie, A. Lizarraga, Y. Huang, S. Xie, and Y. N. Wu. Latent plan transformer for trajectory abstraction: Planning as latent space inference. In *Conference on Neural Information Processing Systems*, 2024.

[45] A. Ajay, A. Kumar, P. Agrawal, S. Levine, and O. Nachum. Opal: Offline primitive discovery for accelerating offline reinforcement learning. In *International Conference on Learning Representations*, 2021.

[46] C. Lynch, M. Khansari, T. Xiao, V. Kumar, J. Tompson, S. Levine, and P. Sermanet. Learning latent plans from play. In *Conference on robot learning (CoRL)*, pages 1113–1132, 2020.

[47] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[48] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

[49] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

[50] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008, 2017.

[51] K. P. Murphy. *Machine Learning: A Probabilistic Perspective*. Adaptive Computation and Machine Learning series. MIT Press, Cambridge, MA, 2012.

[52] D. Kong, M. Zhao, D. Xu, B. Pang, S. Wang, E. Honig, Z. Si, C. Li, J. Xie, S. Xie, and Y. N. Wu. Latent thought models with variational bayes inference-time computation. In *Forty-second International Conference on Machine Learning*, 2025.

[53] B. Pang, T. Han, E. Nijkamp, S.-C. Zhu, and Y. N. Wu. Learning latent space energy-based prior model. *Advances in Neural Information Processing Systems*, 33:21994–22008, 2020.

[54] D. Kong, B. Pang, T. Han, and Y. N. Wu. Molecule design by latent space energy-based modeling and gradual distribution shifting. In R. J. Evans and I. Shpitser, editors, *Proceedings of the Thirty-Ninth Conference on Uncertainty in Artificial Intelligence*, volume 216 of *Proceedings of Machine Learning Research*, pages 1109–1120. PMLR, 31 Jul–04 Aug 2023.

[55] D. Kong, B. Pang, and Y. N. Wu. Unsupervised meta-learning via latent space energy-based model of symbol vector coupling. In *Fifth Workshop on Meta-Learning at the Conference on Neural Information Processing Systems*, 2021.

[56] S. Cheng, D. Kong, J. Xie, K. Lee, Y. N. Wu, and Y. Yang. Latent space energy-based neural ODEs. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856.

[57] P. Yu, S. Xie, X. Ma, B. Jia, B. Pang, R. Gao, Y. Zhu, S.-C. Zhu, and Y. N. Wu. Latent diffusion energy-based model for interpretable text modeling. *arXiv preprint arXiv:2206.05895*, 2022.

[58] D. Kong, Y. Huang, J. Xie, E. Honig, M. Xu, S. Xue, P. Lin, S. Zhou, S. Zhong, N. Zheng, and Y. N. Wu. Molecule design by latent prompt transformer. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[59] Y. Xu, D. Kong, D. Xu, Z. Ji, B. Pang, P. Fung, and Y. N. Wu. Diverse and faithful knowledge-grounded dialogue generation via sequential posterior inference. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 38518–38534. PMLR, 23–29 Jul 2023.

A Model Predictive Control (MPC)

Dynamic Model Predictive Control (MPC) provides an effective framework for safely tracking joint-space references $\hat{q}_k, \hat{\dot{q}}_k, \hat{\tau}_k$ generated by learning policies. When implemented, this approach significantly mitigates the inherently noisy characteristics of learning-based control policies while maintaining high tracking performance. By incorporating the robot's dynamics and physical constraints into the optimization problem, the controller ensures smooth and secure motions even when the underlying learning policy produces potentially suboptimal commands.

The formulated cost function penalizes deviations between desired and actual joint states while simultaneously limiting control effort, allowing the robot to operate efficiently within specified joint position, velocity, and torque limits. This balance between precise reference tracking and constraint satisfaction ultimately leads to more natural robot movements, enhancing the overall execution quality of learned behaviors while preventing excessive energy consumption and vibrational behaviors that might otherwise occur with direct policy execution.

QP Formulation Define the robot state at time step k as

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{q}_k \\ \dot{\mathbf{q}}_k \end{bmatrix},$$

where $\mathbf{q}_k$ and $\dot{\mathbf{q}}_k$ represent the joint angles and joint velocities, respectively. Let $\mathbf{u}_k$ be the control input (torque). Over a prediction horizon of length N_p, the following cost function is minimized:

$$\sum_{k=i}^{i+N_p} \Big[(\hat{\mathbf{q}}_k - \mathbf{q}_k)^T Q_q \, (\hat{\mathbf{q}}_k - \mathbf{q}_k) \; + \; (\hat{\dot{\mathbf{q}}}_k - \dot{\mathbf{q}}_k)^T Q_{\dot{q}} \, (\hat{\dot{\mathbf{q}}}_k - \dot{\mathbf{q}}_k) \; + \; \mathbf{u}_k^T Q_u \, \mathbf{u}_k \Big],$$

subject to the robot dynamics and constraints. The linearized (or identified) system model is given by

$$\mathbf{x}_{k+1} = A_k \, \mathbf{x}_k + B_k \, \mathbf{u}_k + \mathbf{r}_k,$$

where A_k, B_k, and $\mathbf{r}_k$ approximate the robot's dynamics around a nominal operating point. The following constraints are imposed to ensure feasibility and safety:

$$\begin{aligned} \mathbf{u}_{\min} &\leqslant \mathbf{u}_k \leqslant \mathbf{u}_{\max}, \\ \mathbf{q}_{\min} &\leqslant \mathbf{q}_k \leqslant \mathbf{q}_{\max}, \\ \dot{\mathbf{q}}_{\min} &\leqslant \dot{\mathbf{q}}_k \leqslant \dot{\mathbf{q}}_{\max}, \\ \ddot{\mathbf{q}}_{\min} &\leqslant \ddot{\mathbf{q}}_k \leqslant \ddot{\mathbf{q}}_{\max}, \\ \mathbf{q}_0 &= \mathbf{q}_{\text{current}}, \quad \dot{\mathbf{q}}_0 = \dot{\mathbf{q}}_{\text{current}}, \end{aligned}$$

where $\mathbf{q}_{\min}, \mathbf{q}_{\max}, \dot{\mathbf{q}}_{\min}, \dot{\mathbf{q}}_{\max}$, and so forth denote predefined joint and actuator limits. Collecting the state and input variables into a decision vector

$$\mathbf{z}_k = \begin{bmatrix} \mathbf{x}_k \\ \mathbf{u}_k \end{bmatrix},$$

the QP can be expressed in a compact form as

$$\begin{aligned} \min_{\{\mathbf{z}_k\}_{k=i}^{i+N_p}} \quad & \sum_{k=i}^{i+N_p} \big(\mathbf{z}_k^T W_k \, \mathbf{z}_k + 2 \, \mathbf{w}_k^T \, \mathbf{z}_k \big), \\ \text{subject to} \quad & \mathbf{x}_{k+1} = A_k \, \mathbf{x}_k + B_k \, \mathbf{u}_k + \mathbf{r}_k, \\ & \mathbf{u}_{\min} \leqslant \mathbf{u}_k \leqslant \mathbf{u}_{\max}, \\ & \mathbf{q}_{\min} \leqslant \mathbf{q}_k \leqslant \mathbf{q}_{\max}, \\ & \dot{\mathbf{q}}_{\min} \leqslant \dot{\mathbf{q}}_k \leqslant \dot{\mathbf{q}}_{\max}, \\ & \ddot{\mathbf{q}}_{\min} \leqslant \ddot{\mathbf{q}}_k \leqslant \ddot{\mathbf{q}}_{\max}, \\ & \mathbf{q}_0 = \mathbf{q}_{\text{current}}, \quad \dot{\mathbf{q}}_0 = \dot{\mathbf{q}}_{\text{current}}. \end{aligned}$$

The matrices W_k and vectors $\mathbf{w}_k$ encapsulate the quadratic and linear terms derived from expanding the cost function and incorporating the linearized dynamics. Solving this QP at each time step allows the robot to effectively track the learning policy's references $\hat{\mathbf{q}}_k, \hat{\dot{\mathbf{q}}}_k$ while honoring physical constraints and preserving safe operation.

B Robot Configurations

B.1 Actuators

Table 2: Specifications of Different Actuator Models

Specification	**Koala BEAR**	**Koala BEAR Muscle**	**Panda BEAR**	**Panda BEAR Plus**
Dimensions (mm)	$63.5 \times 62 \times 37$	$75 \times 67 \times 37.5$	$113 \times 113 \times 49.7$	$113 \times 113 \times 49.7$
Weight (g)	250	285	650	925
Peak Torque (15 sec)	4.2 Nm	8 Nm	16.8 Nm	33 Nm
Peak Torque (1.5 sec)	10.5 Nm	20 Nm	33.5 Nm	67 Nm

Table 3: Actuator Configuration for Robot A

Joint	**Actuator Model**
Body	Panda BEAR Plus
Shoulder Pitch	Panda BEAR Plus
Shoulder Yaw	Panda BEAR
Elbow Pitch	Panda BEAR Plus
Wrist Pitch	Koala BEAR
Wrist Roll	Koala BEAR

Table 4: Actuator Configuration for Robot B

Joint	**Actuator Model**
Body	Koala BEAR Muscle
Shoulder Pitch	Koala BEAR Muscle
Shoulder Yaw	Koala BEAR Muscle
Elbow Pitch	Koala BEAR Muscle
Wrist Pitch	Koala BEAR Muscle
Wrist Roll	Koala BEAR Muscle

B.2 Robot A

Table 5: Robot A Configuration

Joint	**Frame Coordination**	**Coordinate (m)**	**Joint Limit (rad)**
Body Yaw	z-axis rotation	(0.0, 0.0, 1.1276)	$[-1.57, 1.57]$
Shoulder Yaw	z-axis rotation	(0.0, $\pm$0.21, 0.117)	$[-2.27, 2.27]$
Shoulder Pitch	y-axis rotation	(0.0, $\pm$0.15, 0.0)	$[-1.57, 1.57]$
Elbow Pitch	y-axis rotation	(0.0, 0.4, 0.0)	$[-1.57, 2.53]$
Wrist Pitch	y-axis rotation	(0.0, 0.375, 0.0)	$[-2.44, 2.44]$
Wrist Roll	x-axis rotation	(0.01925, 0.0, 0.0)	$[-1.57, 1.57]$

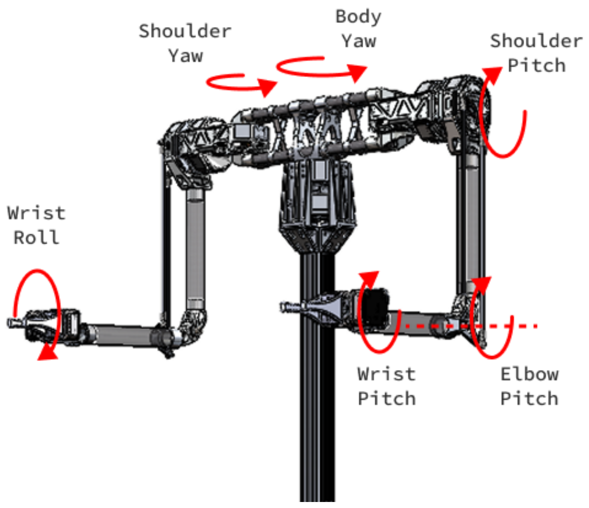

Figure 4: Coordinate system of Robot A

B.3 Robot B

Table 6: Robot B Configuration

Joint	Frame Coordination	Coordinate (m)	Joint Limit (rad)
Body Yaw	z-axis rotation	(0.0, 0.0, 1.1327)	$[-1.57, 1.57]$
Shoulder Pitch	y-axis rotation	(0.0, ±0.19475, 0.0)	$[-1.57, 1.57]$
Shoulder Roll	x-axis rotation	(0.0, 0.0, 0.0)	$[-1.57, 0.25]$
Elbow Yaw	z-axis rotation	(0.0, 0.0, 0.0)	$[-0.79, 1.57]$
Elbow Pitch	y-axis rotation	(0.02, 0.0, -0.27585)	$[-0.3, 2.9]$
Wrist Pitch	y-axis rotation	(0.23226, 0.019701, 0.0)	$[-1.15, 2.2]$

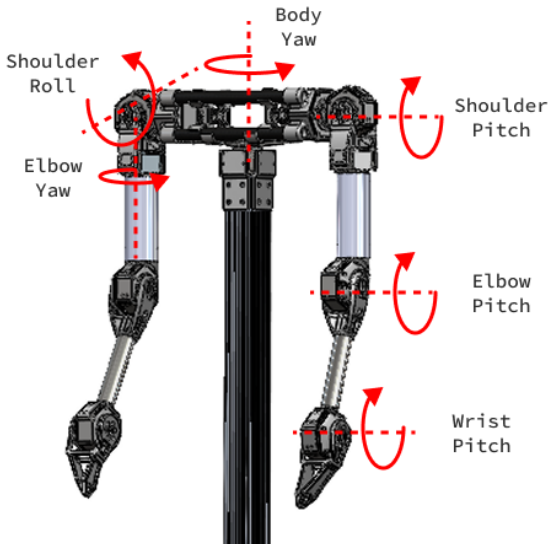

Figure 5: Coordinate system of Robot B

C 2D Human Pose Estimation

For 2D human pose estimation, we utilized the OpenMMLab library. In the detection phase, we used the RTMDet-M model (rtmdet_m_8xb32-100e_coco-obj365-person-235e8209.pth). Following detection, we utilized HRNet-W48 (hrnet_w48_coco_wholebody_384x288_dark-f5726563_20200918.pth) for precise keypoint estimation. The HRNet architecture maintains high-resolution representations throughout the network, enabling accurate localization of body joints. This specific model was trained on the COCO-WholeBody dataset with a 384×288 input resolution and incorporates the DARK pose estimation technique for sub-pixel accuracy. The combination of these models provided reliable human pose tracking that was essential for evaluating the physical interaction capabilities of our robotic systems.

D Box Pose Estimation

For real-time object detection in our system, we implemented YOLOv8. We created a custom dataset specifically tailored to our application environment, with all annotations performed using the Computer Vision Annotation Tool (CVAT). This approach enabled precise labeling of objects of interest while maintaining consistency across the training dataset. The segmentation model achieves an impressive inference rate of 100 Hz, allowing our system to process visual information at real-time speeds necessary for responsive robotic interaction. This high-frequency detection capability proved essential for tracking dynamic objects in the environment and facilitating accurate decision-making in our experimental scenarios.

E Latent Adaptive Planner Algorithms

In this section, we provide detailed algorithms for the Latent Adaptive Planner (LAP) framework. We describe both the training procedure and the variational replanning methodology used during inference.

E.1 Classical Variational Bayes Learning of LAP

Algorithm 1 outlines the training procedure for our Latent Adaptive Planner using Classical Variational Bayes. This approach optimizes both local parameters (latent plans for individual trajectories) and global parameters (shared decoder model) in an alternating fashion.

Algorithm 1 LAP: Classical Variational Bayes Learning

```
Require: Trajectory dataset $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^N$, where $\mathbf{x}_i = \{(o_t^i, a_t^i)\}_{t=1}^T$
Ensure: Trained model parameters $\theta$
 1: Initialize global parameters $\theta$ randomly
 2: repeat
 3:     Sample a mini-batch of trajectories $\{\mathbf{x}_i\}_{i=1}^B$ from $\mathcal{D}$
 4:     for each $\mathbf{x}_i$ in the mini-batch do
 5:         Initialize local parameters $(\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i)$ randomly
 6:         for $j = 1$ to $T_{\text{local}}$ do
 7:             Sample $\mathbf{z}_i \sim \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i^2)$ using reparameterization trick
 8:             Compute ELBO: $\mathcal{L}(\theta, \boldsymbol{\mu}_i, \boldsymbol{\sigma}_i) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x}_i)}[\log p_\theta(\mathbf{x}_i|\mathbf{z})] - D_{\text{KL}}(q(\mathbf{z}|\mathbf{x}_i) \| p(\mathbf{z}))$
 9:             Update $(\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i)$ with gradient ascent on $\mathcal{L}(\theta, \boldsymbol{\mu}_i, \boldsymbol{\sigma}_i)$
10:         end for
11:     end for
12:     Sample $\mathbf{z}_i \sim \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i^2)$ for each trajectory in mini-batch
13:     Update $\theta$ with batch gradient ascent on $\frac{1}{B}\sum_{i=1}^B \mathcal{L}(\theta, \boldsymbol{\mu}_i, \boldsymbol{\sigma}_i)$
14: until convergence
```

Algorithm 2 LAP: Variational Replanning

```
Require: Trained model parameters $\theta$, replanning horizon $\Delta$, initial observation $o_1$
Ensure: Robot actions $\{a_t\}_{t=1}^{T}$
 1: // Initial planning
 2: Initialize $\boldsymbol{\mu}_0$ and $\boldsymbol{\sigma}_0$ randomly
 3: for $j = 1$ to $T_{\text{local}}$ do
 4:     Sample $\mathbf{z} \sim \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\sigma}_0^2)$ using reparameterization trick
 5:     Compute objective: $\mathcal{L}_0 = \log p_\theta(\mathbf{x}_{0:1}|\mathbf{z}) - D_{\text{KL}}(\mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\sigma}_0^2)\|p(\mathbf{z}))$
 6:     Update $(\boldsymbol{\mu}_0, \boldsymbol{\sigma}_0)$ with gradient ascent on $\mathcal{L}_0$
 7: end for
 8: Sample $\mathbf{z}_0 \sim \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\sigma}_0^2)$
 9: Generate and execute action $a_1 \sim p_\theta(a_1|o_1, \mathbf{z}_0)$
10: $t \leftarrow 1$
11: // Adaptive replanning loop
12: while task not complete do
13:     Observe $o_{t+1}$
14:     if $t \bmod \Delta = 0$ then
15:         // Variational replanning
16:         Set $\boldsymbol{\mu}_t \leftarrow \boldsymbol{\mu}_{t-\Delta}$, $\boldsymbol{\sigma}_t \leftarrow \boldsymbol{\sigma}_{t-\Delta}$
17:         for $j = 1$ to $T_{\text{replan}}$ do
18:             Sample $\mathbf{z} \sim \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\sigma}_t^2)$ using reparameterization trick
19:             Compute objective:
                $\mathcal{L}_t = \log p_\theta(\mathbf{x}_{t-\Delta+1:t+1}|\mathbf{x}_{0:t-\Delta}, \mathbf{z}) - D_{\text{KL}}(\mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\sigma}_t^2)\|\mathcal{N}(\boldsymbol{\mu}_{t-\Delta}, \boldsymbol{\sigma}_{t-\Delta}^2))$
20:             Update $(\boldsymbol{\mu}_t, \boldsymbol{\sigma}_t)$ with gradient ascent on $\mathcal{L}_t$
21:         end for
22:         Sample $\mathbf{z}_t \sim \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\sigma}_t^2)$
23:     end if
24:     Generate action $a_{t+1} \sim p_\theta(a_{t+1}|\mathbf{x}_{0:t+1}, \mathbf{z}_t)$
25:     Execute action $a_{t+1}$
26:     $t \leftarrow t + 1$
27: end while
```

The key insight of this algorithm is the alternating optimization between local and global parameters. For each trajectory, we first optimize the latent plan distribution parameters $(\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i)$ using T_{local} steps of gradient ascent (typically 16 steps in our implementation). After optimizing all local parameters in the mini-batch, we then perform a single update of the global parameters θ based on the average ELBO across the mini-batch.

E.2 Variational Replanning Algorithm

Algorithm 2 presents our variational replanning approach, which enables efficient adaptation during test time as new observations become available. This approach uses Bayesian updating in the latent space, treating previous distributions as prior beliefs.

The variational replanning algorithm has two key phases. First, we perform initial planning to establish our initial belief about the latent plan. Then, during task execution, we periodically update our belief through Bayesian updating every Δ timesteps.

A crucial aspect of this approach is that we treat the previously inferred distribution $\mathcal{N}(\boldsymbol{\mu}_{t-\Delta}, \boldsymbol{\sigma}_{t-\Delta}^2)$ as a prior for the current update, as shown in the KL-divergence term of the objective function. This enables efficient adaptation while maintaining temporal consistency in the latent plan.

In practice, we set $T_{\text{replan}} = 1$ for computational efficiency, which is sufficient for incremental updates given that we're starting from a previously optimized distribution. This stands in contrast to the more expensive initial planning phase that uses $T_{\text{local}} = 16$ steps.

E.3 Implementation Details

For the variational inference optimization, we use AdamW optimizer with a learning rate of 1e-3 for optimizing local parameters and 2e-4 for global parameters. The latent dimension is set to 64, and we found that diagonal covariance matrices provide a good balance between expressiveness and computational efficiency.

The replanning horizon Δ is a tunable parameter that trades off computational cost against adaptability. In our experiments, we found $\Delta = 10$ to provide good results, allowing the system to adapt to changing conditions while maintaining real-time performance.