Skill Acquisition with Switching Latent PI Controllers

Juyan Zhang[1], Dana Kulić[1] and Michael Burke[1]

***Abstract*— Recent work [1] recasts neural network mixture heads as a library of latent-space feedback skills, with each skill behaving as a proportional (P) controller and achieving improved robustness compared to behavior cloning baselines. We extend this framework to proportional–integral (PI) control in latent space and evaluate our model on FetchPush and robot-trajectory tasks and compare it with P controller and other Behaviour cloning baselines. Results show that the integral path accumulates latent tracking error to cancel slowly varying disturbances, which empirically enhances robustness and performance at the cost of modestly lower sample efficiency in the lowest data setting. We also find that the introduction of the PI controller empirically bounds the deployment trajectory closer to the training trajectory than the P controller**

I. INTRODUCTION

A proficient robot should not only perform diverse tasks but also reuse its skills across scenarios. A common strategy for tackling long-horizon problems is to decompose a task into shorter stages with intermediate goals [2], yielding a library of reusable skills. Consider coffee preparation: reaching for the cup, pouring, stirring, and carrying are distinct subtasks. Each subtask requires bringing the robot and relevant objects toward designated target states, and transitions between subtasks are triggered by the *joint state* of the robot and scene. This naturally suggests representing each skill as a *feedback control law* with respect to a set point (goal), and composing complex behavior by *switching* among such controllers.

Mixture Density Networks (MDNs) are widely used to model multimodal action distributions in robotics, from autonomous driving [3]–[5] to manipulation [6]–[12]. However, MDNs can overfit due to flexible mixtures [13], [14]. In recent work [1], we reinterpreted the one-layer linear layer as a *classical feedback controller* acting on a learned latent state. This yields an MDN view as a *switching library of skill policies* that share a latent state, but differ in goals and gains. Building on this, we propose a probabilistic graphical model of skill acquisition as segmentation in latent space, where each segment is governed by a feedback proportional integral control law.

In this work, we extend latent proportional (P) control to a *latent proportional–integral (PI)* formulation by augmenting the action head with a short history of goal-relative latent error. Concretely, each skill uses a finite-window (or leaky) integrator that is reset on skill switches, providing bias rejection (sensor drift, mild actuation errors) while preserving tractability. We derive an ELBO that explicitly encodes this switching PI structure, which can be end-to-end trained without an explicit dynamics model. Importantly, results show that this addition helps keep latent states closer to the training data manifold at execution time, leading to greater robustness.

*This work is supported by Monash University

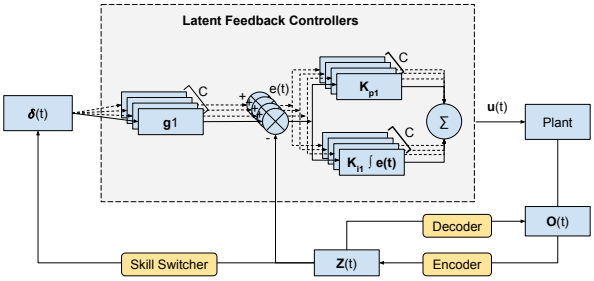

Fig. 1: Skills are modelled as PI feedback laws in latent space. The encoder maps observation o_t to latent state z_t; a switcher activates skill δ with the goal g_δ and the proportional gain $K_{p,\delta}$ and the integral gains $K_{i,\delta}$. Yellow blocks denote neural networks; the active controller outputs u_t to the robot for execution.

The key contributions of this work are summarized as follows:

- A probabilistic approach for learning a sequence of latent-space *feedback laws*, with a new ELBO that accounts for a switching *PI* control structure via a finite-history integrator.
- Empirical evaluation in FetchPush and robot letter writing, showing improved task success and robustness to both observation and plant noise.
- Empirically, switching latent *PI* feedback yields deployment trajectories that stay closer to the training-data manifold than latent *P* control.

II. RELATED WORK

As highlighted by [2], efficient skill acquisition hinges on autonomously discovering a hierarchy of skills, and jointly learning when to switch and how to control. Thus, segmentation (where/when to apply a skill) and the underlying feedback policy for each skill should be learned together.

Numerous approaches have been proposed to segment demonstrations into skills. One line of work is based on similarity. *Similarity-based* methods segment by clustering in different spaces, such as state [15], policy [16], reward [17], [18], or recently language spaces [19], including hierarchical/options RL [20], [21]. They are insightful yet fragile on complex tasks where similarities do not generalize, and RL can be sample-inefficient. Another line of work focuses on Change-point approaches to detect regime shifts in dynamics

TABLE I: Average success rate on test seed based on 5 trained models given each setting in FetchPush. For the evaluation of baselines, we reuse the metrics from our previous work [1]

(a) FetchPush: Average Success Rate given different numbers of skills

Number of Skills	**5**	**10**	**20**	**100**
MDN	**79.08%±2.70%**	**84.09%±1.84%**	77.25%±3.54%	69.00%±3.30%
Latent P controllers	74.40%±3.19%	82.00%±2.61%	84.40%±2.86%	82.80%±1.96%
Latent PI controllers	77.00%±3.17%	78.67%±2.04%	**87.20%±1.02%**	**87.67%±2.39%**

(b) FetchPush: Average Success Rate given different training data size with the optimal skill number

Sample Size	**25%**	**50%**	**75%**	**100%**
BC	59.09%±2.55%	74.45%±4.61%	79.25%±3.10%	83.25%±1.89%
MDN (10)	62.40%±4.45%	74.80%±2.42%	76.80%±2.73%	84.09%±1.84%
Latent P controllers (20)	**74.40%±4.79%**	74.40%±2.93%	82.40%±2.14%	84.40%±2.86%
Latent PI controllers (20)	71.60%±3.19%	**78.00%±2.61%**	**88.40%±1.72%**	**87.20%±1.02%**

[22]. These work for simple proprioception but degrade with high-dimensional observations, geodesic motions, or subtle events, and are sensitive to noise. We instead assess *skill similarity via state–action causality* in a probabilistic model with latent states, yielding more robust segmentation on multimodal data.

While the criteria above specify the segment target to distinguish skills, a common approach to learn these skills is fitting models, such as neural networks, to these demonstrations. Mixture models [6], especially MDNs, remain standard for switching among C modes:

$$q(\boldsymbol{u}_t \mid \boldsymbol{z}_t) = \sum_{i=1}^{C} \pi_i(\boldsymbol{z}_t)\, \mathcal{N}\big(\boldsymbol{u}_t \mid \mu_i(\boldsymbol{z}_t), \Sigma_i(\boldsymbol{z}_t)\big). \quad (1)$$

Switching density networks [23] add PID-structured heads to identify goals/gains, but require true end-effector states and are not scalable to multivariable control. Newtonian VAEs [24] learn locally linear latent dynamics from broad data and then infer proportional goals/gains from demonstrations.

Our previous work [1] reinterpret MDN heads as a library of latent feedback skills: a linear head $u_t = Wz_t + b$ is equivalent to $u_t = K(g - z_t)$ with $K = -W$ and $g = -W^{\dagger}b$. It introduces a probabilistic model that segments demonstrations, switches skills with a learned selector, and trains via a tailored ELBO. In this work, we extend each skill from P to PI by adding a short-horizon integrator, further enhancing robustness while preserving the same switching framework.

III. METHODOLOGY

A. Decomposition of sequential data with goals

We model the demonstrated sequence as observation–action pairs (o_t, u_t) with a hidden skill index $\delta_t \in \{0, 1, \ldots, C-1\}$, a latent state $z_t \in \mathbb{R}^d$, and a goal embedding matrix $G \in \mathbb{R}^{C\times d}$. Given δ_t, the latent goal is $g_t = G_{\delta_t}$. Unlike a purely proportional controller, our *latent PI* controller uses a *finite history window* of length W to form an integrator feature $i_t \in \mathbb{R}^d$:

$$i_t = \sum_{w=0}^{W-1} \frac{1}{W}\big(g_t - z_{t-w}\big)$$

The action is then produced by a per-skill PI controller

$$u_t = K_{P,\delta_t}\,(g_t - z_t) \;+\; K_{I,\delta_t}\, i_t$$

With the presence of (g_t, i_t), (o_t, u_t) are independent across t. In addition, we use the indicator function $\mathbb{I}$ to represent the deterministic relationship between goal index δ_t and g_t:

$$p(o_{1:T}, u_{1:T}, \delta_{1:T}) = \prod_{t=1}^{T} \prod_{c=1}^{C} \Big[p\big(o_t, u_t, g_t, i_t\big)\Big]^{\mathbb{I}(\delta_t = c)} \quad (2)$$

where $\mathbb{I}(\delta_t = c)$ is 1 if $\delta_t = c$ 0 otherwise

Compared to the model with latent P controller where u_t depends only on z_t and g_t, Eq. (2) captures the required temporal coupling through the variable i_t rather than the entire trajectory, preserving a tractable, switching-skill decomposition while enabling integral action over a bounded history.

B. ELBO with a finite-window integrator

Our derivation closely follows [1] but augments the action likelihood with a finite-window integrator. Let $\delta_t \in \{0, \ldots, C-1\}$ be the skill index, $z_t \in \mathbb{R}^d$ the latent state, and $G \in \mathbb{R}^{C\times d}$ the goal embedding with $g_t = G_{\delta_t}$.

The joint distribution factors as

$$p(o_t, u_t, g_t, i_t) = \quad (3)$$

$$\int p(o_t|z_t)p(u_t|z_t, i_t, g_t)p(\delta_t \mid z_t)p(i_t)p(z_t)dz_t \quad (4)$$

We use a mean-field variational family that mirrors the local dependencies:

$$q(\delta_t, z_t|o_t, u_t, i_t) = \prod_{t=1}^{T} q(z_t|o_t)q\left(\delta_t \mid z_t, u_t, i_t\right),$$

where $q(\delta_t|z_t, u_t, i_t)$ is the exact posterior given z_t and i_t are known.

TABLE II: Robustness calculated on the AUC of the Success Rate given different factors.

(a) FetchPush: Robustness given different skill number

Number of Skills	**5**	**10**	**20**	**100**
BC	55.26%±3.61%	55.26%±3.61%	55.26%±3.61%	55.26%±3.61%
MDN	57.47%±3.05%	57.83%±2.21%	56.46%±5.86%	53.49%±4.20%
Latent P controllers	**58.57%±4.30%**	**62.17%±5.64%**	64.51%±3.74%	63.60%±4.72%
Latent PI controllers	58.19%±4.52%	56.67%±4.18%	**65.24%±4.22%**	**66.52%±2.87%**

(b) FetchPush: Robustness given different training data size

Sample Size	**25%**	**50%**	**75%**	**100%**
BC	40.46%±2.56%	49.77%±7.85%	54.34%±5.19%	55.26%±3.61%
MDN (10)	43.94%±5.01%	50.69%±2.63%	55.09%±4.47%	55.09%±5.77%
Latent P controllers (20)	**53.66%±3.55%**	54.51%±4.32%	60.34%±3.16%	64.51%±3.74%
Latent PI controllers (20)	52.69%±5.39%	**55.43%±3.33%**	**62.97%±2.96%**	**65.24%±4.22%**

Then the ELBO is formulated as

$$\begin{aligned}
&\log p(o_{1:T}, u_{1:T}) \geq \\
&\quad \mathbb{E}_q \left[\log \frac{p(o_{1:T}, u_{1:T}, z_{1:T}, \delta_{1:T})}{q(z_{1:T}, \delta_{1:T} \mid o_{1:T}, u_{1:T})} \right] \\
&= \sum_{t=1}^{T} \mathbb{E}_{q(z_t|o_t)} \big[\log p(o_t \mid z_t) \big] + \\
&\quad \sum_{t=1}^{T} \mathbb{E}_{q(z_{t-w:t},\, \delta_t | o_{1:T}, u_{1:T})} \big[\log p(u_t | z_t, i_t, \delta_t) \big] - \\
&\quad \sum_{t=1}^{T} D_{\mathrm{KL}} \big(q(z_t|o_t) \| p(z_t) \big) - \\
&\quad \sum_{t=1}^{T} \mathbb{E}_{q(z_t|o_t)} D_{\mathrm{KL}} \big(q(\delta_t | z_t, u_t, i_t) \| p(\delta_t | z_t) \big) \Big\}
\end{aligned} \tag{5}$$

Compared to the P-only case (where $p(u_t|z_t, \delta_t)$ is local in t), the PI term in (5) introduces an expectation over a *window* $z_{t-w:t}, \delta_t$ through i_t, retaining tractability while capturing integral action over recent history.

C. Latent Proportional Integral Controller

Compared with [1], the control signals are modelled as a library of PI controllers with different goals, gains, but shared latent states z_t.

For a particular demonstration, the sequence of control signals is represented as a sequence of controllers with the continuous shared latent states z_t and different goals g_t. Therefore, the probability of the control signals is modeled as

$$p(u_t|z_t, \delta_t) = \mathcal{N}(u_t; K_P(g_{\delta_t} - z_t) + WK_I i_t, \Sigma_{\delta_t}(z_t)) \tag{6}$$

Noticed that δ_t is the goal index at time t, given δ_t is known, the goal g_t is known, which is the δ_t-th row in the goal embedding G.

Similar to the PI controller, at time t, we only consider the steady error between the current goal g_t and historical states z_{t-w} and ignore the error between the historical goals g_{t-w} and their corresponding states z_{t-w}.

IV. EVALUATION

The performance of the model is evaluated in the FetchPush task and a Letter writing task. We also did an ablation to compare the difference between the latent PI controllers and the latent P controllers.

A. Metrics

1) Success Rate: We use the success rate as the performance metric for the models, which measures the number of successful trials in the test set defined by each of the simulated environments.

2) Robustness AUC: To measure the robustness, we add different levels of noise to the observations and monitor the success rates. We use the AUC (Area Under the Curve) of those success rates given different levels of noise as the robustness metric.

B. FetchPush

1) Dataset: The FetchPush task requires a manipulator to move a block to a target position on top of a table by pushing with its gripper. The robot is controlled by the displacement of the end effector position in Cartesian coordinates, and the inverse kinematics are computed by the MuJoCo framework. The gripper is locked for this task. The input and output are sensory, which are detailed in [25].

2) Baselines: The model is compared with Behaviour Cloning (BC), Mixture Density Network (MDN), and Latent switching P controller model [1]. For BC, we use the same structure as the behaviour cloning baseline from [26]. For the MDN and Latent switching P controller model, we keep the encoder and skill switching network the same as for our model.

3) Analysis: The analysis focuses on two perspectives, performance and robustness, quantified by success rate and robustness AUC, respectively, across varying C and data regimes.

a) Model Performance: Table I shows that latent PI increasingly outperforms latent P as the skill library grows: with $C = 20$ and $C = 100$, PI attains the best success rates, respectively, while at small C, PI is on par or slightly

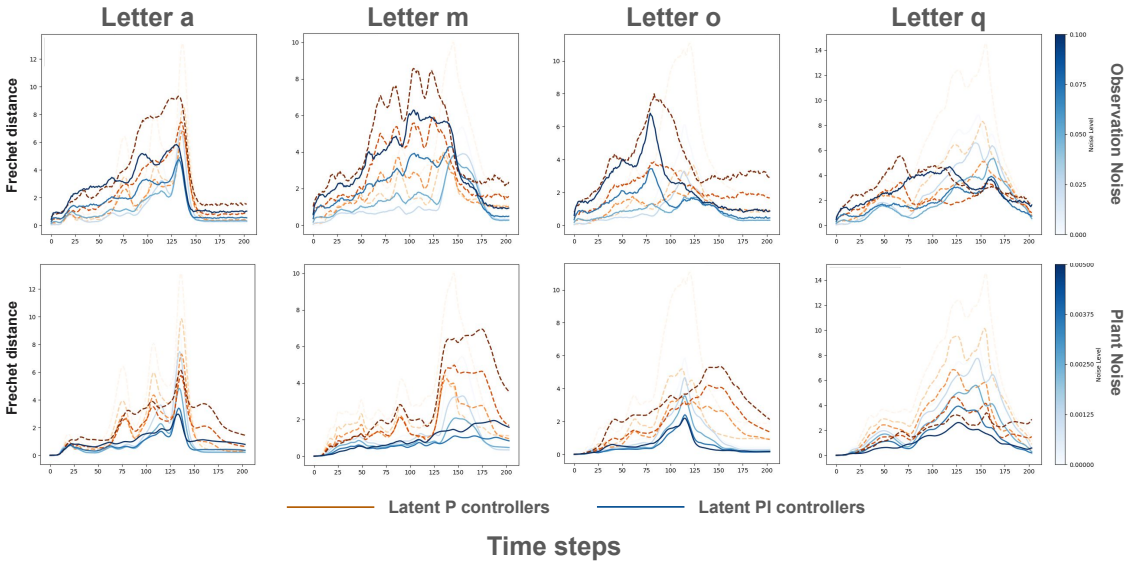

Fig. 2: Fréchet distance between latent spaces of generated trajectories and training dataset trajectories under varying levels of observation noise (top row) and plant noise (bottom row) for four different tasks (a, m, o, q). Lower distances indicate better trajectory distribution matching. Models with latent PI controllers consistently achieve lower distances compared to the latent P controllers, indicating improved robustness under noisy conditions.

below latent P Controller and MDN. This pattern matches the control intuition: integral action removes steady-state bias once each skill covers a relatively stationary slice of the dynamics (larger C), but can add variance when a single skill must explain heterogeneous behavior (small C).

With the *optimal* skill size for each baseline, PI dominates as data increases. Therefore, PI trades a bit of low-data efficiency for markedly higher final performance and robustness as demonstrations and skill granularity grow.

b) Robustness: The robustness AUC improves with finer skill granularity, and latent PI overtakes P once the library is sufficiently large: at $C = 20$ and $C = 100$, PI attains the top AUCs, while at small libraries ($C = 5, 10$) P is equal or better, see Table. IIa.

With each method's best skill number, robustness increases with data, and PI is strictly the best from 50% onward; at 25% data, P is slightly higher. In short, PI trades a bit of low-data efficiency for stronger disturbance rejection at realistic data scales and moderate-to-large skill libraries; see Table. IIb.

C. LetterWritting Task

1) Dataset: The Character Trajectories dataset [27] consists of multiple labelled samples of human pen tip movements on a tablet, recorded while writing individual characters. All samples come from the same writer writing letters in different ways and are intended for primitive extraction. Each trajectory state is represented in 3D Cartesian space, and the actions correspond to the displacements in the same space.

2) Baselines: For the MDN baseline, we use a 3-layer MLP combined with an LSTM as the state encoder and action head as a single-layer MDN for the switching feedback controllers. Our model only differs in the action head and training objective. We deploy the model on a UR5 robot. During testing, we compute the immediate waypoint with the current pose from the sensor and the predicted displacement from the model.

3) Analysis: Figure 2 reports Fréchet distance between deployment trajectory distribution and training trajectory distribution over time for four letters (*a*, *m*, *o*, *q*). The top row sweeps *observation noise* and the bottom row sweeps *plant/actuation noise*. Across all letters and both noise types, the latent PI controller (blue) consistently lies below the latent P controller (orange), indicating lower tracking error throughout the rollout in a distributional sense.

V. CONCLUSION

We extend the switching latent *P* controller [1] to a *PI* controller. This improves the model performance and robustness, as well as reducing the distance to the training manifold further compared with the latent P controller, at the cost of modestly lower sample efficiency in the low data setting.

A potential limitation of the proposed approach is that the integral term can amplify noise and slow early-stage convergence when both the dataset and the skill library are small (low C). In these regimes, proportional-only heads often match or slightly outperform PI. In addition, PI introduces extra tunables, most notably the history-window length, which increases tuning effort and implementation complexity.

ACKNOWLEDGMENT

This research is supported by the Monash Graduate Scholarship. We are grateful to our colleagues in Monash Robotics for general suggestions and feedback.

REFERENCES

[1] J. Zhang, D. Kulic, and M. Burke, "A probabilistic model for skill acquisition with switching latent feedback controllers," *arXiv preprint arXiv:2410.14191*, 2024.

[2] O. Kroemer, C. Daniel, G. Neumann, H. Van Hoof, and J. Peters, "Towards learning hierarchical skills for multi-phase manipulation tasks," in *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2015, pp. 1503–1510.

[3] Y. Chai, B. Sapp, M. Bansal, and D. Anguelov, "Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction," in *Conference on Robot Learning*. PMLR, 2020, pp. 86–99.

[4] Y. Choi, K. Lee, and S. Oh, "Distributional deep reinforcement learning with a mixture of gaussians," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 9791–9797.

[5] Z. Huang, J. Wu, and C. Lv, "Efficient deep reinforcement learning with imitative expert priors for autonomous driving," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 10, pp. 7391–7403, 2022.

[6] C. M. Bishop, "Mixture density networks," 1994.

[7] D. Angelov, Y. Hristov, M. Burke, and S. Ramamoorthy, "Composing diverse policies for temporally extended tasks," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2658–2665, 2020.

[8] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone, "Libero: Benchmarking knowledge transfer for lifelong robot learning," *Advances in Neural Information Processing Systems*, vol. 36, pp. 44 776–44 791, 2023.

[9] E. Pignat and S. Calinon, "Bayesian gaussian mixture model for robotic policy imitation," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4452–4458, 2019.

[10] M. Wilson and T. Hermans, "Learning to manipulate object collections using grounded state representations," in *Conference on Robot Learning*. PMLR, 2020, pp. 490–502.

[11] R. Rahmatizadeh, P. Abolghasemi, A. Behal, and L. Bölöni, "From virtual demonstration to real-world manipulation using lstm and mdn," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[12] V. Prasad, A. Kshirsagar, D. K. R. Stock-Homburg, J. Peters, and G. Chalvatzaki, "Moveint: Mixture of variational experts for learning human-robot interactions from demonstrations," *IEEE Robotics and Automation Letters*, 2024.

[13] H. Cui, V. Radosavljevic, F.-C. Chou, T.-H. Lin, T. Nguyen, T.-K. Huang, J. Schneider, and N. Djuric, "Multimodal trajectory predictions for autonomous driving using deep convolutional networks," in *2019 international conference on robotics and automation (icra)*. IEEE, 2019, pp. 2090–2096.

[14] O. Makansi, E. Ilg, O. Cicek, and T. Brox, "Overcoming limitations of mixture density networks: A sampling and fitting framework for multimodal future prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7144–7153.

[15] D. Tanneberg, K. Ploeger, E. Rueckert, and J. Peters, "Skid raw: Skill discovery from raw trajectories," *IEEE robotics and automation letters*, vol. 6, no. 3, pp. 4696–4703, 2021.

[16] C. Daniel, H. Van Hoof, J. Peters, and G. Neumann, "Probabilistic inference for determining options in reinforcement learning," *Machine Learning*, vol. 104, no. 2, pp. 337–357, 2016.

[17] S. Krishnan, A. Garg, R. Liaw, B. Thananjeyan, L. Miller, F. T. Pokorny, and K. Goldberg, "Swirl: A sequential windowed inverse reinforcement learning algorithm for robot tasks with delayed rewards," *The international journal of robotics research*, vol. 38, no. 2-3, pp. 126–145, 2019.

[18] P. Ranchod, B. Rosman, and G. Konidaris, "Nonparametric bayesian reward segmentation for skill discovery using inverse reinforcement learning," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 471–477.

[19] Z. Liang, Y. Mu, H. Ma, M. Tomizuka, M. Ding, and P. Luo, "Skilldiffuser: Interpretable hierarchical planning via skill abstractions in diffusion-based task execution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 16 467–16 476.

[20] R. S. Sutton, D. Precup, and S. Singh, "Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning," *Artificial intelligence*, vol. 112, no. 1-2, pp. 181–211, 1999.

[21] M. Stolle and D. Precup, "Learning options in reinforcement learning," in *Abstraction, Reformulation, and Approximation: 5th International Symposium, SARA 2002 Kananaskis, Alberta, Canada August 2–4, 2002 Proceedings 5*. Springer, 2002, pp. 212–223.

[22] P. Fitzpatrick, A. Arsenio, and E. R. Torres-Jara, "Reinforcing robot perception of multi-modal events through repetition and redundancy and repetition and redundancy," *Interaction Studies*, vol. 7, no. 2, pp. 171–196, 2006.

[23] M. Burke, Y. Hristov, and S. Ramamoorthy, "Hybrid system identification using switching density networks," in *Conference on Robot Learning*. PMLR, 2020, pp. 172–181.

[24] M. Jaques, M. Burke, and T. M. Hospedales, "Newtonianvae: Proportional control and goal identification from pixels via physical latent spaces," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 4454–4463.

[25] R. de Lazcano, K. Andreas, J. J. Tai, S. R. Lee, and J. Terry, "Gymnasium robotics," 2024. [Online]. Available: http://github.com/Farama-Foundation/Gymnasium-Robotics

[26] Y. Lee, A. Szot, S.-H. Sun, and J. J. Lim, "Generalizable imitation learning from observation via inferring goal proximity," *Advances in neural information processing systems*, vol. 34, pp. 16 118–16 130, 2021.

[27] B. Williams, "Character Trajectories," UCI Machine Learning Repository, 2006, DOI: https://doi.org/10.24432/C58G7V.