What Do Latent Action Models Actually Learn?

Chuheng Zhang[1,*], Tim Pearce[1,*], Pushi Zhang[1], Kaixin Wang[1], Xiaoyu Chen[2], Wei Shen[3], Li Zhao[1], Jiang Bian[1]
[1] Microsoft Research [2] Tsinghua University [3] Independent Researcher
[*] Equal contribution: zhangchuheng123@live.com, tim.daniel.pearce@gmail.com

Abstract

Latent action models (LAMs) aim to learn action-relevant changes from unlabeled videos by compressing changes between frames as *latents*. However, differences between video frames can be caused by *controllable changes* as well as *exogenous noise*, leading to an important concern – do latents capture the changes caused by actions or irrelevant noise? This paper studies this issue analytically, presenting a linear model that encapsulates the essence of LAM learning, while being tractable. This provides several insights, including connections between LAM and principal component analysis (PCA), desiderata of the data-generating policy, and justification of strategies to encourage learning controllable changes using data augmentation, data cleaning, and auxiliary action-prediction. We also provide illustrative results based on numerical simulation, shedding light on the specific structure of observations, actions, and noise in data that influence LAM learning.

1 Introduction

Latent action models (LAMs) aim to infer controllable action changes from streams of image observations in an unsupervised manner (Rybkin et al., 2018; Menapace et al., 2021). This is valuable because action-labeled data is typically expensive to source, while unlabeled videos are abundant. Hence, it offers a route for embodied AI systems to learn from large unlabeled datasets, for instance using the inferred latent actions as targets for pre-training a policy, while a small amount of labeled data can be used to learn a mapping from latent to real action controls (Ye et al., 2024). This has proven effective in learning from videos of 2D video games, robotics, and even broadcast tennis footage (Menapace et al., 2021; Schmidt and Jiang, 2023; Bruce et al., 2024; Chen et al., 2024b; Ye et al., 2024; Sun et al., 2024; Cui et al., 2025; Gao et al., 2025).

The success of such LAM-based recipes relies on the inferred latent action labels mapping to the real control action signals of interest. However, there is concern that this may not always be the case. For example, there is an intuition that LAM's inferred latent 'actions' simply compress differences between consecutive frames, even when control actions are not the cause of those differences (McCarthy et al., 2024). As such, LAMs may only succeed in domains where the cause of changes between observations can be fully attributed to the control action. A further point of dispute is whether a bottleneck is required (Schmidt and Jiang, 2023).

To study these issues and more, this paper conducts a theoretical analysis of a linear version of LAM. By retaining the architecture of recent LAM models, but swapping deep neural network components with simpler linear layers, we preserve the fundamental challenge of LAM training in an analytically tractable form. Our analysis of linear LAM firstly provides precise insights into when inferred latent actions capture true control signals compared to noise, as well as what information is captured by different components within LAM. Surprisingly, our analysis also reveals additional issues not currently known to the LAM community – related to the over-parametrization property of LAM and the randomness of data-generation policy. Finally, we propose and study potential solutions to ameliorate these issues – data augmentation and predicting action as an auxiliary task.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

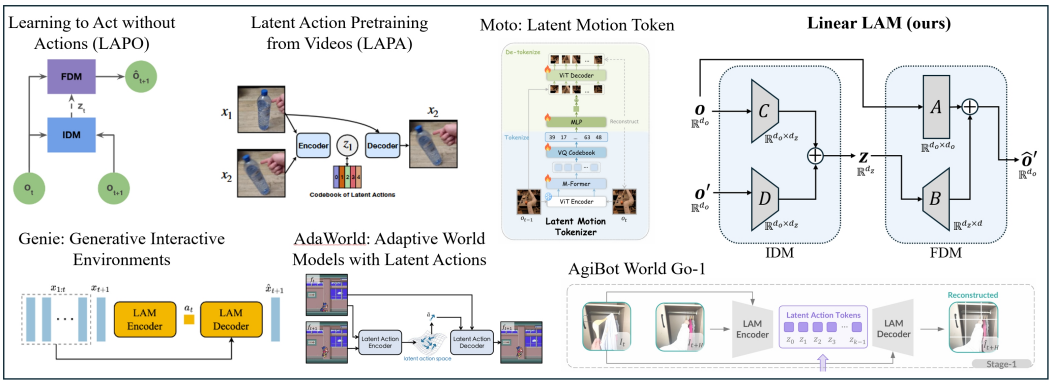

Figure 1: Linear LAM is an abstraction of the LAMs used in previous work. Inputting consecutive observation pairs $(\boldsymbol{o}, \boldsymbol{o}')$, the LAMs output the second observation via a reconstruction loss, $\|\hat{\boldsymbol{o}}' - \boldsymbol{o}'\|_2^2$. An information bottleneck tries to stop the direct copying of $\boldsymbol{o}'$, with the expectation the latent $\boldsymbol{z}$ will correspond to the control action $\mathbf{a}$. Linear LAM captures the essence of LAM training whilst being analytically tractable. The diagrams of previous LAMs are copied from their original papers: LAPO (Schmidt and Jiang, 2023), LAPA (Ye et al., 2024), Moto (Chen et al., 2024c), Genie (Bruce et al., 2024), AdaWorld (Gao et al., 2025), and Go-1 (AgiBot-World, 2025).

Concretely, this paper makes several key contributions.

1. Section 3 presents linear LAM, a tractable model preserving the essence of LAMs used in practice.
2. Section 4.1 shows linear LAM reduces to principal component analysis (PCA) on a mixture of controllable changes and exogenous noise, under certain assumptions. Our analysis justifies the practical use of LAM when the controllable action signals cause larger changes to observations than the exogenous noise.
3. Section 4.2 shows correlation between observations and actions decreases LAM's focus on learning controllable changes. This suggests that higher randomness in data-generating policies benefits LAM's learning.
4. Section 4.3 validates that performing data augmentation during LAM training can mitigate the over-parametrization issue and thus improve the semantics of the latent.
5. Section 4.4 finds that adding an action-prediction head encourages LAM to prioritize the learning of controllable changes for the latent.
6. Section 5 verifies that the main findings based on linear LAM still hold on LAMs in practice.

2 Related Work

The study of learning representations of real actions in reinforcement learning (RL) has a long history. For instance, PG-RA (Chandak et al., 2019) clusters actions based on the similarity of their impact on the state to improve generalization in the action space. LASER (Allshire et al., 2021) learns latent actions through an encoder-decoder architecture trained to reconstruct real actions, resulting in higher learning efficiency in RL. TAP (Jiang et al., 2022) learns the latent action that can help to reconstruct the full trajectory (with state, action, and reward) condition on the state. EAR (Hua et al., 2022) finds that the latent task embedding resulting from the training of multi-task policies turn out to be good action representations with a geometrically and semantically meaningful structure. AD3 (Wang et al., 2024) adopts an inverse dynamics model and a forward dynamics model to extract latent action, similar to popular LAMs, but conditions these models on the real actions.

Different from the above papers that learn action representation based on real actions, our paper focuses on the model where the latent action is learned *without access to the real action labels*. Removing the need for action labels during training is advantageous as it allows leveraging internet-scale video datasets in the pre-training stage (Miech et al., 2019; Chen et al., 2024a; Pei et al., 2025; Wang et al., 2023) – unlocking the value of diverse expert demonstrations without action labels. While high-quality robotic datasets with action annotations exist (Vuong et al., 2023; Fang et al., 2023; Khazatsky et al., 2024; AgiBot-World, 2025), they remain limited in scale. Such datasets can be

integrated into the semi-supervised learning framework to extract latent actions (Nikulin et al., 2025) or the policy fine-tuning stage (Schmidt and Jiang, 2023; Ye et al., 2024). An alternative approach aims to extract pre-defined actions from observations using computer vision techniques (Mendonca et al., 2023).

Beginning from Rybkin et al. (2018), LAMs have featured an information bottleneck or auto-encoder to allow learning in an unsupervised manner. ILPO (Edwards et al., 2019) learns latent actions along with the training of a policy that outputs the latent action and a world model that conditions on the latent action. Menapace et al. (2021) proposes a probabilistic action network that extracts a discrete action label and a continuous action variability embedding from consecutive observations. This network is trained jointly with an action decoder to generate video controlled by extracted actions. While these works adopt a bottleneck in the learning of latent actions, their training losses are complicated by involving policy learning or recurrent networks. LAPO (Schmidt and Jiang, 2023) proposes an LAM design with a inverse dynamics model that extracts the latent action and a forward dynamics model that reconstructs the next observation based on the latent action. This marks the template of the modern LAM. Many papers follow a similar architectures to LAPO with a discrete latent action space such as FICC (Ye et al., 2022), Genie (Bruce et al., 2024), LAPA (Ye et al., 2024), Moto (Chen et al., 2024c), IGOR (Chen et al., 2024b), Go-1 (AgiBot-World, 2025), and GR001T N1 (Nvidia, 2025). However, there is a debate whether discrete latent actions are better than continuous latent actions. For example, Nikulin et al. (2025) finds that using continuous latent actions with a larger bottleneck can not only result in better predictability on real actions but also lead to better performance for downstream policies. We consider continuous latent actions in our paper and leave the analysis of vector quantized latent action to future work.

In contrast to the popularity of LAMs in recent foundation models for embodied AI, issues around the objective, learnability, and robustness of LAM have received limited attention. Consequently, our paper aims to investigate these issues.

3 Setup

This section broadly introduces the problem setting, goal and model used in recent LAM work in practice. We then more formally detail these for the linear model to be used in subsequent analysis. Finally, we outline the details of our simulation setup to be used in support of our later analysis.

3.1 LAMs in Practice

Setting. The setting tackled by recent LAM work assumes access to a large dataset of pairs of observations and next observations $\mathcal{D} = \{(\boldsymbol{o}_i, \boldsymbol{o}'_i)\}_{i=1}^{N}$ and only a small subset of it has action labels $\mathcal{D}_a = \{(\boldsymbol{o}_i, \boldsymbol{o}'_i, \boldsymbol{a}_i)\}_{i=1}^{N_a}$, with $\lambda := |\mathcal{D}_a|/|\mathcal{D}| \ll 1$. Note we drop the i subscript when we do not need to refer to specific samples. In practice, $\boldsymbol{o}$ and $\boldsymbol{o}'$ could be feature vectors extracted from the original observations (Cui et al., 2025), and $\boldsymbol{o}$ could be a stack of historical observations to account for partial observability (Bruce et al., 2024). The two datasets are assumed to come from the same distribution in most LAM work (Menapace et al., 2021; Bruce et al., 2024; Ye et al., 2024)[1].

Model. Recent LAM designs (Schmidt and Jiang, 2023; Bruce et al., 2024; Ye et al., 2024; Chen et al., 2024b; AgiBot-World, 2025) (see Figure 1) take a pair of consecutive observations $(\boldsymbol{o}, \boldsymbol{o}')$ as input, and output a latent $\boldsymbol{z}$ as well as a prediction of the next observation $\hat{\boldsymbol{o}}'$. (We subsequently avoid calling $\boldsymbol{z}$ 'latent action' to avoid confusion with the real action.)

LAMs are typically decomposed into an inverse dynamics model (IDM) and forward dynamics model (FDM), implemented as deep neural networks, and trained via a reconstruction loss,

$$\boldsymbol{z} = \psi_{\text{IDM}}(\boldsymbol{o}, \boldsymbol{o}'), \quad \hat{\boldsymbol{o}}' = \psi_{\text{FDM}}(\boldsymbol{o}, \boldsymbol{z}), \quad \mathcal{L} := \min_{\psi_{\text{IDM}}, \psi_{\text{FDM}}} \mathbb{E}_{\mathcal{D}}\big[\|\hat{\boldsymbol{o}}' - \boldsymbol{o}'\|_2^2\big]. \tag{1}$$

where ψ_{IDM} contains a bottleneck so the dimension of $\boldsymbol{z}$ is smaller than that of $\boldsymbol{o}$.

Use cases. We identify two primary downstream use cases of LAMs.

- The LAM latents can be used as *input* to a world model $\hat{T}$ trained to generate future frames, $\hat{T}(\boldsymbol{o}'|\boldsymbol{o}, \boldsymbol{z})$ (Sun et al., 2024; Chen et al., 2024b; Bruce et al., 2024; Menapace et al., 2021). These

[1]Certain work explores mismatch case, e.g., $\mathcal{D}$ is human videos and $\mathcal{D}_a$ is robotics data (Chen et al., 2024b).

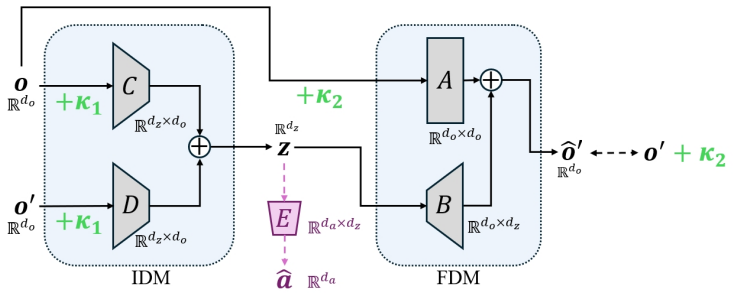

Figure 2: Overview of linear LAM. Grey blocks represent learnable parameter matrices, giving rise to the predictive model $\hat{\boldsymbol{o}}' = A\boldsymbol{o} + B(C\boldsymbol{o} + D\boldsymbol{o}')$. The green parts illustrate linear LAM with data augmentation to reduce the amount of information the latent contains about the observation (in Section 4.3). The red parts illustrate linear LAM with auxiliary action prediction head to encourage the latent to focus on the controllable actions and suppress the noise signal (in Section 4.4).

world models can generate higher-quality frames than the FDM in LAM, but have been used to only qualitatively interpret the meaning of the learned latents.

- The LAM latents can be used as *labels* in the pre-training of a latent policy $\pi_{\text{latent}}(\hat{\boldsymbol{z}}|\boldsymbol{o})$ (similar to behavior cloning on actions). This policy may later be mapped to real actions, $\pi_{\text{map}}(\hat{\boldsymbol{a}}|\hat{\boldsymbol{z}})$ (Bruce et al., 2024; Schmidt and Jiang, 2023), or be followed by a fine-tuning phase on real actions (Schmidt and Jiang, 2023; Ye et al., 2024; Chen et al., 2024b).

In both use cases, it is hoped that the learned latents can be aligned with the true actions as closely as possible. As in Schmidt and Jiang (2023), "*our hypothesis is ... [this] may allow us to learn a latent action representation with a structure closely corresponding to the true action space*".

3.2 Linear LAM

Setting. We conduct analysis in the controlled Markov process (CMP) framework (Puterman, 2014) with $(\mathcal{O}, \mathcal{A}, T)$. At a given timestep, the agent receives observation $\boldsymbol{o} \in \mathcal{O}$ and takes action $\boldsymbol{a} \in \mathcal{A}$. The transition function describes the probability distribution over the next observation, $T(\boldsymbol{o}'|\boldsymbol{o}, \boldsymbol{a})$.

We consider vector observations, $\mathcal{O} = \mathbb{R}^{d_o}$ (which could be thought of as image observations processed with a pre-trained image encoder as in Cui et al. (2025); Chen et al. (2024b)). The action space $\mathcal{A} = \mathbb{R}^{d_a}$ has lower dimensionality than $\mathcal{O}$, i.e., $d_a \ll {d_o}^2$. These actions are mapped to create *controllable changes* in observation space via an *action effect matrix* $X \in \mathbb{R}^{d_o \times d_a}$, $\boldsymbol{q} = X\boldsymbol{a}$.

We generally assume linear LAM cannot access the action $\boldsymbol{a}$ during the training, but we will make use of it to evaluate the learned LAM in simulations. We choose the transition function to be an additive combination of *controllable changes* $\boldsymbol{q} \in \mathbb{R}^d$ (the state changes caused by the ego agent's actions) and *exogenous noise* $\boldsymbol{\epsilon} \in \mathbb{R}^d$ (representing environmental stochasticity or other agent's actions), i.e. with the next state generated via, $\boldsymbol{o}' = \boldsymbol{o} + \boldsymbol{q} + \boldsymbol{\epsilon}$.

Model. For linear LAM, the IDM and FDM consist of linear mappings. Summarized in Figure 2, the IDM and FDM are given by,

$$\boldsymbol{z} = \psi_{\text{IDM}}^{\text{Linear}}(\boldsymbol{o}, \boldsymbol{o}') \coloneqq C\boldsymbol{o} + D\boldsymbol{o}' \tag{2}$$

$$\hat{\boldsymbol{o}}' = \psi_{\text{FDM}}^{\text{Linear}}(\boldsymbol{o}, \boldsymbol{z}) \coloneqq A\boldsymbol{o} + B\boldsymbol{z} = A\boldsymbol{o} + B(C\boldsymbol{o} + D\boldsymbol{o}'), \tag{3}$$

where $\boldsymbol{z} \in \mathcal{Z} = \mathbb{R}^{d_z}$ with $d_z \ll d_o$ is the latent. All matrices are learnable parameters, including FDM parameters $A \in \mathbb{R}^{d_o \times d_o}$, $B \in \mathbb{R}^{d_o \times d_z}$, and IDM parameters $C, D \in \mathbb{R}^{d_z \times d_o}$.

As for practical LAM (Eq. 1), the linear LAM is trained via a reconstruction loss,

$$\mathcal{L}(A, B, C, D) = \mathbb{E}\left[\|\hat{\boldsymbol{o}}_i' - \boldsymbol{o}_i'\|_2^2\right]. \tag{4}$$

Appendix A summarizes the gap between linear and practical LAM.

[2]Discrete actions can be represented as one-hot vectors within $\mathbb{R}^{d_a}$. While most of analysis (except that on bottleneck) should also apply to discrete actions, we confine our analysis in continuous actions.

Goal. Following our discussion of use-cases of LAM cases in Section 3.1, we interpret the desired latents to contain as much information as possible about the real action, while minimizing the amount of information about the exogenous noise and the observation. This can be formalized as,

$$\text{LAM Objective} \coloneqq \max_{\boldsymbol{z}} \left[\mathcal{I}(\boldsymbol{z};\boldsymbol{a}) - \mathcal{I}(\boldsymbol{z};\boldsymbol{\epsilon}) - \mathcal{I}(\boldsymbol{z};\boldsymbol{o})\right], \tag{5}$$

where $\mathcal{I}$ is mutual information. Hence, an ideal latent should contain all information about the action label, and no other parts of the environment, consistent with intuitions of previous work – *"biasing [the latent] towards simpler representations is likely preferrable"* (Schmidt and Jiang, 2023).

Whilst mutual information is typically a challenging quantity to measure, in our linear model we can approximate this straightforwardly. After training the linear LAM via Eq. 4 and freezing the parameters, we fit three additional linear layers predicting $(\boldsymbol{q}, \boldsymbol{\epsilon}, \boldsymbol{o})$ from $\boldsymbol{z}$, to give $(\hat{\boldsymbol{q}}, \hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{o}})$ respectively. Note that, under the assumption that the variables $\mathbf{x}$ and $\mathbf{y}$ are independent multivariate Gaussian variables, the mutual information $\mathcal{I}(\mathbf{x}, \mathbf{y}) = -\frac{1}{2}\log\left(\frac{\|\hat{\mathbf{y}}-\mathbf{y}\|_2^2}{\mathbb{V}\text{ar}(\mathbf{y})}\right)$ where $\hat{\mathbf{y}}$ is the least squares estimate (LSE) of $\mathbf{y}$ based on $\mathbf{x}$, and $\mathbb{V}\text{ar}(\cdot)$ indicates the total variance (see e.g., Chapter 8 of (Johnson et al., 2002)) of a multivariate random variable. Hence, we can define an evaluation metric that captures the objective of training linear LAM,

$$\text{Linear LAM Objective (LLO)} \coloneqq \max_{\boldsymbol{z}} \left[-\frac{\|\hat{\boldsymbol{q}}-\boldsymbol{q}\|_2^2}{\mathbb{V}\text{ar}(\boldsymbol{q})} + \frac{\|\hat{\boldsymbol{\epsilon}}-\boldsymbol{\epsilon}\|_2^2}{\mathbb{V}\text{ar}(\boldsymbol{\epsilon})} + \frac{\|\hat{\boldsymbol{o}}-\boldsymbol{o}\|_2^2}{\mathbb{V}\text{ar}(\boldsymbol{o})}\right]. \tag{6}$$

Note that we get rid of the $\log(\cdot)$ function wrapped around the MSE loss to better calculate the values for this objective since the value range for the mutual information $[0, +\infty)$ is unbounded. This objective is maximized when $\boldsymbol{z}$ perfectly predicts $\boldsymbol{q}$ while containing no information about $\boldsymbol{\epsilon}$ and $\boldsymbol{o}$, resulting in the optimal value for LLO equal to $0 + 1 + 1 = 2$.

4 Analysis

At times, we will present numerical simulations of linear LAM to visually communicate later analysis. See the details of simulation in Appendix B and the code in supplementary.

4.1 Analysis 1: Linear LAM is PCA

This section first shows that training linear LAM is equivalent to performing PCA on the mixture of controllable changes and exogenous noise. This requires that the controllable changes and exogenous noise $\boldsymbol{q}, \boldsymbol{\epsilon}$ are uncorrelated with the observation $\boldsymbol{o}$ (Section 4.2 relaxes this assumption).

We discuss the insight that this connection provides, followed by an analysis of several important cases covering specific settings of controllable changes and exogenous noise

Proposition 4.1 (Linear LAM is PCA)**.** *Under the linear LAM model and setup defined in Section 3.2, and additionally assuming $\mathbb{E}[\boldsymbol{o}(\boldsymbol{q}+\boldsymbol{\epsilon})^T] = \mathbf{0}$, the objective of linear LAM is equivalent to performing PCA on a mixture of controllable changes $\boldsymbol{q}$ and exogenous noise $\boldsymbol{\epsilon}$,*

$$\mathcal{L} = \mathbb{E}\left[\|(BD - I)(\boldsymbol{q}+\boldsymbol{\epsilon})\|_2^2\right] \tag{7}$$

Note, BD is a low-rank matrix to capture the main components of $\boldsymbol{q} + \boldsymbol{\epsilon}$. See proof in Appendix C.1.

Given that we have transformed the optimization problem of linear LAM into a PCA problem (which can also be viewed as a linear auto-encoder), PCA's property applies to linear LAM.

Proposition 4.2 (Linear LAM tries to capture $\boldsymbol{q} + \boldsymbol{\epsilon}$)**.** *Denote the covariance matrix of $\boldsymbol{q} + \boldsymbol{\epsilon}$ as $\Sigma_{\boldsymbol{q}+\boldsymbol{\epsilon}} = \mathbb{E}[(\boldsymbol{q}+\boldsymbol{\epsilon})(\boldsymbol{q}+\boldsymbol{\epsilon})^T]$ and its eigenvalue decomposition $\Sigma_{\boldsymbol{q}+\boldsymbol{\epsilon}} = U\Lambda U^T$ with eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_{d_o}$. Under the same conditions of Proposition 4.1, $\mathcal{L}$ is optimized when*

1. *$B = U_{d_z}$ spans the subspace of the top d_z principal components of $\Sigma_{\boldsymbol{q}+\boldsymbol{\epsilon}}$ where U_{d_z} contains the first d_z columns of U;*
2. *$D = U_{d_z}^T$ such that the reconstruction $BD(\boldsymbol{q}+\boldsymbol{\epsilon}) = U_{d_z}U_{d_z}^T(\boldsymbol{q}+\boldsymbol{\epsilon})$ projects $\boldsymbol{q}+\boldsymbol{\epsilon}$ onto the subspace spanned by U_{d_z}.*

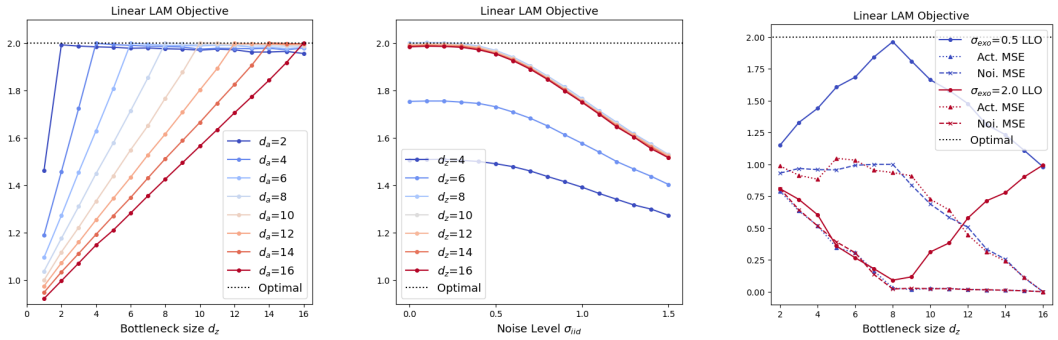

Figure 3: LLO (Linear LAM objective (6), higher better) measured in three noise settings. (Left) $\boldsymbol{\epsilon} = 0$. (Middle) $\boldsymbol{\epsilon}$ is i.i.d. noise. (Right) $\boldsymbol{\epsilon}$ contains the effect of other agents. Action MSE, noise MSE, and observation MSE are the three terms in (6). We set real action dimension $d_a = 8$ and exogenous action dimension $d_b = 8$ unless otherwise stated, and ensure $\boldsymbol{q}$ has unit variance.

The minimum loss is given by the sum of the eigenvalues of corresponding to the discarded principal components $\mathcal{L}^* = \sum_{i=d_z+1}^{d_o} \lambda_i$.

This proposition is equivalent to the Eckart–Young–Mirsky theorem (Eckart and Young, 1936) whose proof can be found in Chapter 2.4 of Golub and Van Loan (2013).

Over-parameterization issue. We show that linear LAM is over-parameterized, with multiple solutions of (A, B, C, D) able to minimize the reconstruction loss objective. Specifically, the latent $\boldsymbol{z} = C\boldsymbol{o} + D\boldsymbol{o}'$, in addition to capturing information about $\boldsymbol{q}$ and $\boldsymbol{\epsilon}$, may further contain information about $\boldsymbol{o}$. There is no detrimental effect provided the A matrix compensates to 'knock out' $\boldsymbol{o}$'s information in $\boldsymbol{z}$. Concretely, there exists a family of solutions $B(C\boldsymbol{o} + D\boldsymbol{o}') = (\boldsymbol{q} + \boldsymbol{\epsilon}) + \alpha\boldsymbol{o}$ and $A = (1-\alpha)I$ for any $\alpha \in \mathbb{R}$ such that $\hat{\boldsymbol{o}}' = B(C\boldsymbol{o} + D\boldsymbol{o}') + A\boldsymbol{o} = (\boldsymbol{q} + \boldsymbol{\epsilon}) + (1-\alpha)\boldsymbol{o} + \alpha\boldsymbol{o} = \boldsymbol{o}'$.

We will revisit this issue in Section 4.3, showing that data augmentation handles this over-parameterization issue. For the purpose of our immediate analysis, we predict $(\hat{\boldsymbol{q}}, \hat{\boldsymbol{\epsilon}}, \hat{\boldsymbol{o}})$ from a surrogate latent $\tilde{\boldsymbol{z}} := B^{-1}(\hat{\boldsymbol{o}}' - \boldsymbol{o})$ when calculating LLO to get around this issue. This surrogate latent is the same as the original one when $A = I$, which is the case when data augmentation is adopted. Hence, we have $C = -D$ which indicates that the semantic meaning of the latent $\boldsymbol{z} = C\boldsymbol{o} + D\boldsymbol{o}'$ remains the same (*no movement*) when $\boldsymbol{o}' = \boldsymbol{o}$ across different observations.

Case 1: $\boldsymbol{\epsilon} = 0$. In the absence of exogenous noise $\boldsymbol{\epsilon}$, linear LAM does capture the true action $\boldsymbol{a}$ in the latent $\boldsymbol{z}$, when the bottleneck dimension is set equal or larger than the action dimension.

When $\boldsymbol{\epsilon} = 0$ and $\boldsymbol{q} = X\boldsymbol{a}$, the covariance matrix of $\Sigma_{\boldsymbol{q}+\boldsymbol{\epsilon}} = \Sigma_{\boldsymbol{q}} = \mathbb{E}[X\boldsymbol{a}\boldsymbol{a}^T X]$ only has d_a non-zero eigenvalues. Following Proposition 4.2, we can make the following conclusions.

- When $d_z \geq d_a$, the capacity of the latent is large enough to capture all the information about the controllable change $\boldsymbol{q}$, the minimum loss $\mathcal{L}^* = 0$, and LLO achieves the optimal.
- Specifically, when $d_z = d_a$, the subspace spanned by the columns of $B = U_{d_a}$ is the same to the subspace spanned by the columns of action effect matrix X (by noting that $U_{d_a}\Lambda U_{d_a}^T = \Sigma_{\boldsymbol{q}+\boldsymbol{\epsilon}} = X\mathbb{E}[\boldsymbol{a}\boldsymbol{a}^T]X^T$). In this case, the learned latent $\boldsymbol{z}$ (whose effect is interpreted by B) fully captures the information of $\boldsymbol{a}$ (whose effect is X), and LLO is maximized. In other words, for this ideal case linear LAM's latent perfectly captures the information of the true action $\boldsymbol{a}$ without access to it.
- When $d_z < d_a$, $\mathcal{L}^* > 0$ and this linear auto-encoder captures the first d_z components in $\boldsymbol{q}$.

We illustrate how LLO varies across different d_a and d_z through numerical simulation in Figure 3 (left). The simulation validates that linear LAM is optimized in terms of LLO when $d_z \geq d_a$.

Case 2: $\boldsymbol{\epsilon}$ is i.i.d. noise. We consider i.i.d. noise (independent and identically distributed), which may be a realistic assumption in the case of sensors or image encoders. We assume $\boldsymbol{\epsilon}$ is i.i.d. with zero-mean $\mathbb{E}[\boldsymbol{\epsilon}] = 0$ and isotropic covariance $\mathbb{E}[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^T] = \sigma_{iid}^2 I$. Considering that $\boldsymbol{q}$ and $\boldsymbol{\epsilon}$ are independent, the covariance matrix $\Sigma_{\boldsymbol{q}+\boldsymbol{\epsilon}}$ can be eigenvalue decomposed as $\Sigma_{\boldsymbol{q}+\boldsymbol{\epsilon}} = U\Gamma_0 U^T = U_0(\Lambda_0 + \sigma_{iid}^2 I)U_0^T$ where U_0 and Λ_0 are the eigenvectors and eigenvalues of $\Sigma_{\boldsymbol{q}}$ respectively (i.e., when $\boldsymbol{\epsilon} = 0$). Combining the results in Proposition 4.2, we conclude that, when there is i.i.d. noise, 1) the FDM

parameter B (and therefore the semantics of the latent since B interpret the latent) remains the same, and 2) the loss increases since the eigenvalues increase. This conclusion is consistent with the conclusion on the robustness of PCA (Anderson, 1963; Johnstone, 2001).

Figure 3 (middle) shows how LLO changes for Gaussian noise of differing variance. We observe that linear LAM is robust to i.i.d. noise up to around $\sigma_{iid} = 0.5$ (when the noise intensity is half that of the signal), linear LAM still succeeds with negligible gap to the optimal case.

Case 3: ϵ contains the effect of other agents. In many real-world datasets (such as Ego4d (Grauman et al., 2022)), the change between two observations may not only be caused by the control action of the ego-agent (e.g. joints of a robot arm), but additionally can be effected by exogenous noise, such as other agents (e.g. a person walking in the background, camera shake). Compared with i.i.d. noise, this noise is structured and partially predictable.

Here, we assume $\boldsymbol{\epsilon} = Y\mathbf{b}$ where $Y \in \mathbb{R}^{d_o \times d_b}$ is the exogenous effect matrix and $\mathbf{b}$ represents the action taken by other agents. In this case, analysis similar to that of Case 1 concludes that *linear LAM will learn to capture effects with largest variances, no matter if they result from the controllable action or other agents.* Moreover, when the columns in Y are not orthogonal to that of X, the FDM parameter B will be impacted by the exogenous noise.

Simulation results in Figure 3 (right) illustrate which part of information (the controllable $\boldsymbol{q}$ or the noise $\boldsymbol{\epsilon}$) enters the latent when the latent dimension d_z is increased. We observe that: 1) When the variance of exogenous noise is smaller than $\boldsymbol{q}$ (i.e., $\sigma_{exo} = 0.5$), linear LAM learns to fit the controllable part first, and $d_z = d_a$ is still the optimal configuration. 2) When the variance of noise exceeds that the signal $\boldsymbol{q}$ (i.e., $\sigma_{exo} = 2.0$), linear LAM will first fit the noise, and the latent fails to exclude the information of the noise no matter how we set d_z. To alleviate this issue, an obvious solution in practice is to preprocess training data to reduce significant noise components (such as stabilizing camera shake).

4.2 Analysis 2: Effects of Data Collection Policy

Till now, we consider the cases where both the controllable changes and the exogenous noise are independent of the observation, i.e., $\mathbb{E}[\boldsymbol{o}(\boldsymbol{q} + \boldsymbol{\epsilon})^T] = \mathbf{0}$. Unfortunately, this is usually not true for practical cases where LAM is trained using data collected by expert policies, e.g., many robot datasets are collected via teleoperation by humans. Within such datasets, the observation is correlated with the action $\boldsymbol{a}$, and thus the controllable change $\boldsymbol{q}$, resulting in $\mathbb{E}[\boldsymbol{o}(\boldsymbol{q} + \boldsymbol{\epsilon})^T] \neq \mathbf{0}$. Therefore, we dive into this scenario to see what is the effect of the data collection policy on linear LAM in this part.

Specifically, we assume the data is collected by a policy that generate action via $\boldsymbol{a} = \Pi_d \boldsymbol{o} + \boldsymbol{\pi}_s$ where $\Pi_d \in \mathbb{R}^{d_a \times d_o}$ represents the deterministic part of the policy and $\pi_s \in \mathbb{R}^{d_a}$ represents the stochastic part of the policy. We assume that $\boldsymbol{o}$, $\boldsymbol{\epsilon}$, and $\boldsymbol{\pi}_s$ are uncorrelated.

In this case, since $\mathbb{R}[\boldsymbol{o}\boldsymbol{q}^T] \neq 0$, the cross term in the expansion of loss (7) remains. Letting $\Sigma_{\boldsymbol{o}} := \mathbb{E}[\boldsymbol{o}\boldsymbol{o}^T]$ to be the covariance matrix of the observation, we can further solve for A by setting the partial derivative of the loss function w.r.t. A to zero, obtaining

$$A = I - (BC + BD) - \Sigma_{\boldsymbol{o}} \Pi_d^T X^T (BD - I)^T \Sigma_{\boldsymbol{o}}^{-1}. \tag{8}$$

This differs from the previous result for the random policy where $A = I - (BC + BD)$ by the last term $\Sigma_{\boldsymbol{o}} \Pi_d^T X^T (BD - I)^T \Sigma_{\boldsymbol{o}}^{-1}$. The intuitive interpretation for this term is that A *can capture the changes caused by the deterministic part of the policy if these changes are not captured by the latent* (noting that $BD - I$ can project the vectors onto the orthogonal complement of the column space of B). In other words, the latent may not capture the effect of each real action. Instead, the FDM would capture the effect of the deterministic part of the policy, whereas the latent would capture the remaining stochastic part.

In our simulation, we control the randomness of the policy using a coefficient χ, and let $\boldsymbol{a} = \chi \Pi_d \boldsymbol{o} + (1 - \chi)\boldsymbol{\pi}_s$ where Π_d is a random projection matrix and π_s is a standard Gaussian vector. The simulation results in Figure 4 (left) indicate that the more deterministic the data collection policy is, the less information about $\boldsymbol{a}$ is captured by $\boldsymbol{z}$. Therefore, the analysis in this section suggest us to use the data collected by the policies with higher randomness whenever possible. This may also indicate that, for LAM training, the trajectories collected by expert policies may not be enough, and more exploratory trajectories are needed.

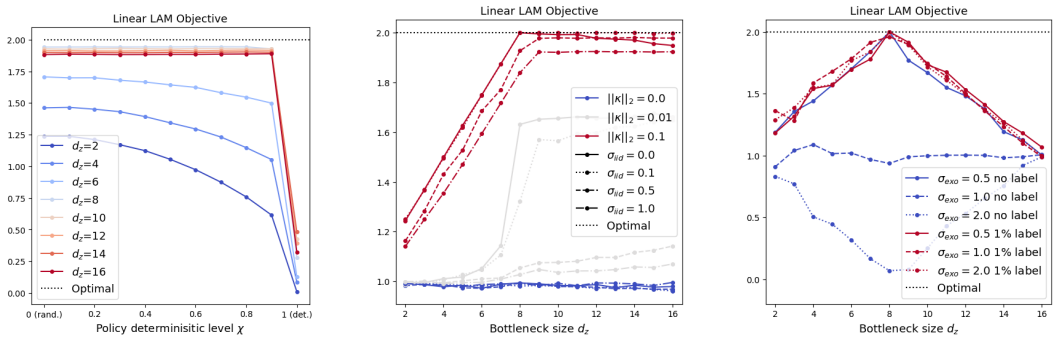

Figure 4: LLO (Linear LAM objective) defined (6) measured when 1) data is collected by policies of different deterministic levels (left), 2) linear LAM is trained w/ and w/o data augmentation (middle), and 3) linear LAM is trained w/ and w/o action prediction (right). We set $d_a = 8$ in the experiments.

4.3 Analysis 3: Improvements via Data Augmentation

This section analyzes the augmentation scheme used in Chen et al. (2024b), which will turn out to resolve a key issue encountered in linear LAM (see the discussion on over-parameterization in Section 4.1), where information about the observation $\boldsymbol{o}$ enters into the latent $\boldsymbol{z}$.

The IGOR model (Chen et al., 2024b) applies one shared random crop to both inputs of the IDM, and a second shared random crop applied to the FDM and reconstruction target. The intuition is that "*by using different croppings [for IDM and FDM], the model is encouraged to learn a more semantically invariant latent action*". Our subsequent analysis shows this intuition is provably correct in linear LAM, resulting in new terms to the loss that encourage removal of the latent of any semantic information about the observation. Sun et al. (2024) also apply a similar scheme, with a single action preserving crop applied to the IDM which makes it harder "*to copy the appearance information directly from the [IDM]*".

Data augmentation in linear LAM. We extend our linear LAM setup to include a data augmentation operator $\mathrm{Aug}_i[\boldsymbol{o}] \coloneqq \boldsymbol{o} + \boldsymbol{\kappa}_i$ that takes an observation as input and adding some random vector $\boldsymbol{\kappa} \in \mathbb{R}^d_o$. $\mathrm{Aug}_i[\cdot]$ applies the same i-th operator to different variables, say $\mathrm{Aug}_1[\boldsymbol{o}]$ and $\mathrm{Aug}_1[\boldsymbol{o}']$ will apply a consistent random variable $\boldsymbol{\kappa}_1$ to each observation.

Proposition 4.3 (Data augmentation addresses over-parameterization)**.** *With data augmentation,*

$$\boldsymbol{z} = \psi^{Linear}_{IDM}(\mathrm{Aug}_1[\boldsymbol{o}], \mathrm{Aug}_1[\boldsymbol{o}']_1) \coloneqq C(\boldsymbol{o} + \boldsymbol{\kappa}_1) + D(\boldsymbol{o}' + \boldsymbol{\kappa}_1) \tag{9}$$

$$\hat{\boldsymbol{o}}' = \psi^{Linear}_{FDM}(\mathrm{Aug}_2[\boldsymbol{o}], \boldsymbol{z}) \coloneqq A(\boldsymbol{o} + \boldsymbol{\kappa}_2) + B\boldsymbol{z}. \tag{10}$$

and assuming $\mathbb{E}[\boldsymbol{o}(\boldsymbol{q} + \boldsymbol{\epsilon})^T] = \mathbf{0}$*, optimizing the loss defined in* (1) *results in* $A = I$ *and* $C + D = 0$.

We provide the proof in Appendix C.2. What does it mean for these terms to encourage $A = I$ and $C + D = 0$? If, A is the identity, all observation information flows directly through this matrix, and $\boldsymbol{z}$ need not carry any additional information about $\boldsymbol{o}$, under the additive assumption on the dynamics. With similar effect, setting $C + D = 0$ immediately cancels out $\boldsymbol{o}$ information in $\boldsymbol{z}$.

$$\boldsymbol{z} = C(\boldsymbol{o} + \boldsymbol{\kappa}_1) + D(\boldsymbol{o}' + \boldsymbol{\kappa}_1) = -D(\boldsymbol{o} + \boldsymbol{\kappa}_1) + D(\boldsymbol{o} + \boldsymbol{q} + \boldsymbol{\epsilon} + \boldsymbol{\kappa}_1) = D(\boldsymbol{q} + \boldsymbol{\epsilon}) \tag{11}$$

This condition also hints semantic consistency – the semantic meaning of the latent *the frame does not change from* $\boldsymbol{o}$ *to* $\boldsymbol{o}'$ will be consistent across different observations. Hence, this augmentation scheme is a mechanism to remove information about $\boldsymbol{o}$ from $\boldsymbol{z}$, explicitly minimizing $\mathcal{I}(\boldsymbol{z}; \boldsymbol{o})$ in the original objective (5). In this way, data augmentation results in latents with better semantics.

Simulation. We show the effect of data augmentation under different noise level in Figure 4 (middle) The results indicate the by data augmentation can improve the semantics of the latent $\boldsymbol{z}$ by addressing the over-parameterization issue. Note that, starting here, the simulation calculate LLO based on the true latent $\boldsymbol{z}$ instead of the pseudo-latent. In this way, even when there is no noise $\sigma_{iid} = 0$, the learned latent $\boldsymbol{z}$ cannot perfectly predict the real action $\boldsymbol{a}$ without involving other information. However, adding in the augmentation vector with 10% variance (compared with that of observation $\boldsymbol{o}$), can greatly improve the learning of linear LAM.

Designing data augmentation. In our linear setting, a natural design for data augmentation is to have $\boldsymbol{\kappa}$ i.i.d. across its elements. This design is based on our knowledge that the semantic meaning for the frame change should be *invariant* to this data augmentation, i.e., $\boldsymbol{o}' - \boldsymbol{o} = (\boldsymbol{o}' + \boldsymbol{\kappa}) - (\boldsymbol{o} + \boldsymbol{\kappa})$, since we know the dynamics is additive. For real images, Chen et al. (Chen et al., 2024b) adopts random crops, reflecting the prior that the semantic meaning about frame change is *invariant* to this way of augmentation. Further, the variance of different augmentations determines how important this term is – a larger augmentation variance enforces this constraint more strictly. This indicates that we may design a weighted mix of data augmentation that best capture the invariance property to improve the semantics of the learned latent.

4.4 Analysis 4: Improvements via Auxiliary Action Prediction

This section analyzes the setting when a small dataset of action labeled data $\mathcal{D}_a$ is available during training of the LAM. This can be used to help guide the latents to represent controllable changes rather than exogenous noise. Specifically we consider the a simple auxiliary loss, with latents as input, predicting the action labels when available. Nikulin et al. (2025) also provide empirically evidence to show that this is a promising strategy to avoiding focus on 'distractors' present in the observations.

Action prediction in linear LAM. Consider a linear prediction head $E \in \mathbb{R}^{d_a \times d_z}$ at the latent bottleneck, $\hat{\boldsymbol{a}} = E\boldsymbol{z}$, and a corresponding action reconstruction loss $\|\hat{\boldsymbol{a}} - \boldsymbol{a}\|_2^2$, we optimize this objective, $\mathcal{L}_a := \mathbb{E}_{\mathcal{D}} \left[\|\hat{\boldsymbol{o}}' - \boldsymbol{o}'\|_2^2\right] + \mathbb{E}_{\mathcal{D}_a} \left[\|\hat{\boldsymbol{a}} - \boldsymbol{a}\|_2^2\right]$ and define $\lambda := |\mathcal{D}_a|/|\mathcal{D}|$.

Proposition 4.4 (Action prediction can denoise)**.** *Following the conditions in Proposition 4.3 and assuming $\boldsymbol{\epsilon}^T X = 0$ and $d_z \geq d_a$, optimizing $\mathcal{L}_a$ biases the encoder parameter perpendicular to the noise. For an artificial case where $\lambda \to +\infty$, we obtain perfect LAM with $D\boldsymbol{\epsilon} = \mathbf{0}$ and $B\boldsymbol{z} = \boldsymbol{q}$.*

We provide the proof in Appendix C.3. Note that we assume that $\boldsymbol{\epsilon}^T X = 0$, which means that the signal $\boldsymbol{q} = X\boldsymbol{a}$ and the noise $\boldsymbol{\epsilon}$ are disentangled. This can correspond to the case, e.g., where $\boldsymbol{\epsilon}$ is background disturbance and $\boldsymbol{q}$ is the table-top manipulation (cf. Case 3 in Section 4.1). The conclusion $D \perp \boldsymbol{\epsilon}$ indicates that noise will not enter the latent (noting that $C = -D$ in this case).

Although Proposition 4.3 considers an artificial case with learn λ, it indicates that auxiliary action prediction can bias linear LAM to denoise. In our simulation, we induce different levels of noise caused by other agents with the same setting as in Figure 3 (right), and present the result in Figure 4 (right). In contrast to previous results where σ_{exo} fails the learning of linear LAM, using only 1% of action label in training leads to successful learning. The simulation results indicate that only a small proportion of action label in training can significantly encourage the latent to encode the real actions and denoise.

5 Experiments on practical LAM

To understand how our insights from linear LAM analysis transfer to real settings, we provide a set of experiments using practical LAM. Specifically, we use image as the input (instead of vectors), non-linear CNNs in IDM and FDM (instead of linear layers), and vector quantization. We show that the main conclusions in our paper still hold in this complex setting.

Dataset. We designed a 4×4 grid-world style synthetic dataset. The top 3×4 grid of the observation contains a square (intensity=1.0) that can be controlled with 5 actions (up, down, left, right, and stay still). The bottom 1×4 grid of the observation contains random Bernoulli noise (with prob 0.5). An intensity parameter controls the noise magnitude (none=0.0, low=1.0, high=2.0).

Policy. By default, we use a uniform policy, where each action is equally probable. For one experiment, we also use a correlated policy, where state and action are correlated. With 95% prob, the action moves the square on a fixed snaking pattern through the grid, and with 5% chance a random action is selected.

Model. For the IDM, we use a small CNN to process o and o', followed by a VQ bottleneck with codebook size of 5 outputting the latent. Finally, for the FDM, a separate UNet takes the latent and previous observation o to output the predicted $\hat{o}'$. When predicting actions, codes are preassigned to actions, and latents are trained to minimize L2 distance to their true action code. For data augmentation, we shift the 4×4 image left/right for one grid with periodic padding. Models

	Controllable Loss	Stochastic Loss
No noise	0.624 ± 0.087	0.0 ± 0.0
Low noise	0.781 ± 0.079	0.739 ± 0.047
High noise	1.046 ± 0.017	0.607 ± 0.021
Uniform policy	0.781 ± 0.079	0.739 ± 0.047
Correlated policy	1.997 ± 0.022	0.599 ± 0.036
No data augmentation	0.781 ± 0.079	0.739 ± 0.047
Data augmentation	0.415 ± 0.020	0.898 ± 0.049
No action prediction	0.781 ± 0.079	0.739 ± 0.047
1% action prediction	0.295 ± 0.011	0.986 ± 0.002

Table 1: The performance of practical LAM on different scenarios. We present the mean and standard errors of controllable loss (measuring how well the latent captures the information, lower the better) and stochastic loss (measure how well the latent captures the noise, higher the better) over 5 seeds. Each group corresponds to the analysis of one sub-section in Section 4

.

were trained for 16k updates with the Adam optimizer. Unless specified, we use low stochastic noise, no action prediction, no data augmentation, and 5 codebook vectors.

Evaluation. Not being linear, it is not possible to measure the mutual information between latent and quantities of interest exactly. Instead, since by design the observation separates the controllable (top half) from stochastic (bottom half) changes, we measure the reconstruction loss on the relevant portion of the observation to assess how what information the latent has captured following training. A lower controllable loss means the latent contains more information about the action, and lower stochastic loss means more noise has been captured. Similar to LLO, this is normalized by the variance of the signal. Note there is unfortunately no straightforward way to measure the information about the observation o in the latent for this non-linear case.

Results. We present experiment results containing controllable loss and stochastic loss in Table 1. Firstly, we train vanilla LAM on different noise levels. Results show that as noise is increased from none to high, the action information in the latent decreases (increasing controllable loss). At the same time, the information about the stochastic noise increases (decreasing stochastic loss). This is consistent with our linear LAM theory in Section 4.1, that latents encode whatever leads to most variance. Secondly, we see that switching from a uniform to a correlated policy reduces the information about actions in the latents, while increasing the information about stochastic noise. This is predicted by our linear LAM theory in Section 4.2, that deterministic data collection policy can lead to degenerated performance. Thirdly, we see that using data augmentation can improve LAM learning. This is consistent with the conclusion in Section 4.3, that data augmentation can help LAM learning. Finally, we show that incorporating 1% of actions labels into the training process improves over vanilla LAM by increasing information about actions, and reducing information about noise. This is consistent with the conclusion in Section 4.4, that predicting true action can improve LAM learning.

6 Conclusion

Latent action models (LAMs) have become popular in the pre-training phase of many embodied AI models. In contrast to the popularity, detailed analysis into the learnability of LAMs is missing. This paper tries to bridge the gap by proposing a linear LAM model that captures the essence of LAM but remains mathematical tractability. By analyzing linear LAM, we have made several fruitful observations, including justifying the design of LAM training loss by showing that it corresponds to PCA and can succeed in recovering the real action, highlighting in the data collection policy's impact on the learning of LAM, and showing the functionality of data augmentation and auxiliary action prediction. We acknowledge that we should also notice the gap between linear LAM and practical LAM in terms of both the capacity of the FDM and IDM models and the linear dynamics assumed in linear LAM. Nevertheless, we believe linear LAM can be further used as a quick testbed for researchers to develop other tricks for LAM learning.

Funding Disclosure Statement

Microsoft supports this work.

References

AgiBot-World, T. Agibot world colosseo: Large-scale manipulation platform for scalable and intelligent embodied systems. *agibot-world.com*, 2025.

Allshire, A. et al. Laser: Learning a latent action space for efficient reinforcement learning. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6650–6656. IEEE, 2021.

Anderson, T.W. Asymptotic theory for principal component analysis. *The Annals of Mathematical Statistics*, 34(1):122–148, 1963.

Bruce, J. et al. Genie: Generative interactive environments. In *Forty-first International Conference on Machine Learning*, 2024.

Chandak, Y. et al. Learning action representations for reinforcement learning. In *International conference on machine learning*, pages 941–950. PMLR, 2019.

Chen, T.S. et al. Panda-70m: Captioning 70m videos with multiple cross-modality teachers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13320–13331, 2024a.

Chen, X. et al. Igor: Image-goal representations are the atomic control units for foundation models in embodied ai. *arXiv preprint arXiv:2411.00785*, 2024b.

Chen, Y. et al. Moto: Latent motion token as the bridging language for robot manipulation. *arXiv preprint arXiv:2412.04445*, 2024c.

Cui, Z. et al. Dynamo: In-domain dynamics pretraining for visuo-motor control. *Advances in Neural Information Processing Systems*, 37:33933–33961, 2025.

Eckart, C. and Young, G. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936. doi: 10.1007/BF02288367.

Edwards, A. et al. Imitating latent policies from observation. In *International conference on machine learning*, pages 1755–1763. PMLR, 2019.

Fang, H.S. et al. Rh20t: A comprehensive robotic dataset for learning diverse skills in one-shot. *arXiv preprint arXiv:2307.00595*, 2023.

Gao, S. et al. Adaworld: Learning adaptable world models with latent actions. *arXiv preprint arXiv:2503.18938*, 2025.

Golub, G.H. and Van Loan, C.F. *Matrix computations*. JHU press, 2013.

Grauman, K. et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18995–19012, 2022.

Hua, P., Chen, Y. and Xu, H. Simple emergent action representations from multi-task policy training. *arXiv preprint arXiv:2210.09566*, 2022.

Jiang, Z. et al. Efficient planning in a compact latent action space. *arXiv preprint arXiv:2208.10291*, 2022.

Johnson, R.A., Wichern, D.W. et al. *Applied multivariate statistical analysis*. Prentice hall Upper Saddle River, NJ, 2002.

Johnstone, I.M. On the distribution of the largest eigenvalue in principal components analysis. *The Annals of statistics*, 29(2):295–327, 2001.

Khazatsky, A. et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

McCarthy, R. et al. Towards generalist robot learning from internet video: A survey. *arXiv preprint arXiv:2404.19664*, 2024.

Menapace, W. et al. Playable video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10061–10070, 2021.

Mendonca, R., Bahl, S. and Pathak, D. Structured world models from human videos. *arXiv preprint arXiv:2308.10901*, 2023.

Miech, A. et al. Howto100m: Learning a text-video embedding by watching hundred million narrated video clips. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2630–2640, 2019.

Nikulin, A. et al. Latent action learning requires supervision in the presence of distractors. *arXiv preprint arXiv:2502.00379*, 2025.

Nvidia. Gr00t n1: An open foundation model for generalist humanoid robot. *Nvidia Release*, 2025.

Pei, B. et al. Modeling fine-grained hand-object dynamics for egocentric video representation learning. *arXiv preprint arXiv:2503.00986*, 2025.

Puterman, M.L. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

Rybkin, O. et al. Learning what you can do before doing anything. *arXiv preprint arXiv:1806.09655*, 2018.

Schmidt, D. and Jiang, M. Learning to act without actions. *arXiv preprint arXiv:2312.10812*, 2023.

Sun, Y. et al. Video creation by demonstration. *arXiv preprint arXiv:2412.09551*, 2024.

Van Den Oord, A., Vinyals, O. et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

Vuong, Q. et al. Open x-embodiment: Robotic learning datasets and rt-x models. In *Towards Generalist Robots: Learning Paradigms for Scalable Skill Acquisition@ CoRL2023*, 2023.

Wang, Y. et al. Internvid: A large-scale video-text dataset for multimodal understanding and generation. *arXiv preprint arXiv:2307.06942*, 2023.

Wang, Y. et al. Ad3: implicit action is the key for world models to distinguish the diverse visual distractors. *arXiv preprint arXiv:2403.09976*, 2024.

Ye, S. et al. Latent action pretraining from videos. *arXiv preprint arXiv:2410.11758*, 2024.

Ye, W. et al. Become a proficient player with limited data through watching pure videos. In *The Eleventh International Conference on Learning Representations*, 2022.

A From practical LAM to linear LAM.

Our goal in designing linear LAM was to preserve the key features of LAMs in practice, while making the resulting model as simple as possible. Here we remark on similarities and differences.

- *Function approximation.* Linear LAM uses linear layers, while practical LAM uses deep neural networks. However, linear LAM is compatible with input vectors processed by other non-linear pre-trained image encoders.
- *Bottleneck.* Both models have an information bottleneck between in the IDM but this is implemented in different ways. Practical LAMs usually adopt vector quantization Van Den Oord et al. (2017), while linear LAM uses a low continuous dimension $d_z \ll d_o$.
- *Additive changes.* We formulate changes between observations as additions of controllable changes and exogenous noise. This additive structure provides the simplest combination of two elements which we believe are a primary concern of whether LAM's latents actually represent actions.
- *Noise.* While our model constrains the noise as additive, it does not make further assumptions. We later analyze realistic cases corresponding to real world scenarios, such as when action and noise are correlated, and action and observations are correlated.

B Simulation Set-ups

We adopt the following setting unless otherwise stated. We provide the source code in the appendix.

- *Observations.* Observations are sampled from the standard normal distribution, $\boldsymbol{o} \sim \mathcal{N}(\mathbf{0}, I)$ with I as the identity matrix. Note that $\mathbb{V}\text{ar}(\boldsymbol{o}) = 1$ (i.e., each element in $\boldsymbol{o}$ has unit variance). We set the dimension of the observations $d_o = 128$.
- *Actions.* We use continuous actions also sampled from a standard normal distribution with dimensionality d_a, so $\boldsymbol{a} \sim \mathcal{N}(\mathbf{0}, I)$. Note that $\mathbb{V}\text{ar}(\boldsymbol{a}) = 1$. We set $d_a = 8$ unless otherwise stated.
- *Controllable changes.* The action effect matrix X that maps the action $\boldsymbol{a}$ to the controllable changes $\boldsymbol{q}$, is chosen as a random orthogonal matrix (using QR decomposition), subsequently normalized to ensure $\mathbb{V}\text{ar}(\boldsymbol{q}) = 1$.
- *i.i.d. noise.* In our simulation we use isotropic Gaussian $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_\epsilon^2 I)$ with variance σ_{iid}^2 as the i.i.d. noise. For this noise, $\mathbb{V}\text{ar}(\boldsymbol{\epsilon}) = \sigma_{iid}^2$.
- *Exogenous noise.* We also consider the noise induced by other agents $\boldsymbol{\epsilon} = Y\boldsymbol{b} = \sigma_{exo} Y_0 \boldsymbol{b}$ where $\boldsymbol{b}$ is other agents' action, Y is the action effect matrix of other agents, and Y_0 is the normalized matrix of Y_0. Similarly, Y_0 is chosen as a random orthogonal matrix using QR decomposition to ensure that $\mathbb{V}\text{ar}(\boldsymbol{\epsilon}) = \sigma_{exo}^2$.
- *Optimization.* We implement our system in PyTorch, optimizing trainable parameters via stochastic gradient descent with the Adam optimizer with batch size 128. We use the default learning rate and run for 4,000 steps to ensure convergence.
- *Evaluation.* Our use the quantity defined in (6) as the default evaluation metric. For the experiments that do not involve noise, we set normalized MSE for noise to 1.
- *Data augmentation.* Data augmentation is a trick that may improve the learnability of LAM mentioned in several previous papers (Chen et al., 2024b; Sun et al., 2024). We implement data augmentation by adding a Gaussian noise $\boldsymbol{\kappa} \sim \mathcal{N}(\mathbf{0}, |\boldsymbol{\kappa}|^2 \mathbf{I})$ for linear LAM. Note that $\mathbb{V}\text{ar}(\boldsymbol{\kappa}) = |\boldsymbol{\kappa}|^2$. By default data augmentation is turned off, except for Figure 4 (middle and right).
- *Action prediction.* Action prediction is another trick proposed in the previous paper (Nikulin et al., 2025). We implement action prediction by predicting the true action label based on the latent with a learnable linear transformation for a small proportion of the data samples. We denote the ratio of the samples that we access their action labels as λ. We find that setting $\lambda = 1\%$ is enough. By default action prediction is turned off, except for Figure 4 (right).

C Proofs

C.1 Proof of Proposition 4.1

Proof. Expand the loss function in (4) with the definitions of $\hat{\boldsymbol{o}}'$ (3) and $\boldsymbol{o}'$ ($\boldsymbol{o}' = \boldsymbol{o} + \boldsymbol{q} + \boldsymbol{\epsilon}$), then rearrange (ignoring the expectation outside the RHS).

$$\begin{aligned}
\mathcal{L} =& \|\boldsymbol{o}' - \hat{\boldsymbol{o}}'\|_2^2 && (12)\\
=& \|\boldsymbol{o}' - (A\boldsymbol{o} + BC\boldsymbol{o} + BD\boldsymbol{o}')\|_2^2 && (13)\\
=& \|(\boldsymbol{o} + \boldsymbol{q} + \boldsymbol{\epsilon}) - (A\boldsymbol{o} + BC\boldsymbol{o} + BD(\boldsymbol{o} + \boldsymbol{q} + \boldsymbol{\epsilon}))\|_2^2 && (14)\\
=& \|(A + BC + BD - I)\boldsymbol{o} + (BD - I)(\boldsymbol{q} + \boldsymbol{\epsilon})\|_2^2 && (15)\\
=& \|(A + BC + BD - I)\boldsymbol{o}\|_2^2 + \|(BD - I)(\boldsymbol{q} + \boldsymbol{\epsilon})\|_2^2 \\
& \qquad + 2\boldsymbol{o}^T(A + BC + BD - I)^T(BD - I)(\boldsymbol{q} + \boldsymbol{\epsilon}) && (16)\\
=& \|(A + BC + BD - I)\boldsymbol{o}\|_2^2 + \|(BD - I)(\boldsymbol{q} + \boldsymbol{\epsilon})\|_2^2 \\
& \qquad + 2\,\mathrm{Tr}\left((A + BC + BD - I)\boldsymbol{o}(\boldsymbol{q} + \boldsymbol{\epsilon})^T(BD - I)\right) && (17)
\end{aligned}$$

Recall that this loss $\mathcal{L}$ is found within an expectation in (4). By assumption $\mathbb{E}[\boldsymbol{o}(\boldsymbol{q} + \boldsymbol{\epsilon})^T] = \mathbf{0}$ and the final term can be ignored (since both expectation and trace are additive).

$$\mathcal{L} = \mathbb{E}\left[\|(A + BC + BD - I)\boldsymbol{o}\|_2^2\right] + \mathbb{E}\left[\|(BD - I)(\boldsymbol{q} + \boldsymbol{\epsilon})\|_2^2\right] \tag{18}$$

Regarding the first term, note that A is full rank d_o while BC and BD are of rank d_z. Since A only appears in this term and it is of greater or equal rank than $BD + BD$, it's optimal value is $A = I - B(C + D)$, setting the first term is zero. Hence we are left with the middle term.

$$\mathcal{L} = \|(BD - I)(\boldsymbol{q} + \boldsymbol{\epsilon})\|_2^2 \tag{19}$$

□

Note that, under the condition of Proposition 4.1, (A, B, C, D) are over-parameterized. When we adopting data augmentation in Proposition 4.3, we obtain $A = I$ and $C + D = 0$, which refines how the condition $A = I - B(C + D)$ is satisfied.

C.2 Proof of Proposition 4.3

We start by expanding the loss defined in (1) with the data augmentation scheme, and unpack (ignoring the expectation outside the RHS).

$$\begin{aligned}
\mathcal{L} =& \|\mathrm{Aug}_2[\boldsymbol{o}'] - \hat{\boldsymbol{o}}'\|_2^2 && (20)\\
=& \|\boldsymbol{o}' + \boldsymbol{\kappa}_2 - (A(\boldsymbol{o} + \boldsymbol{\kappa}_2) + B(C(\boldsymbol{o} + \boldsymbol{\kappa}_1) + D(\boldsymbol{o}' + \boldsymbol{\kappa}_1)))\|_2^2 && (21)\\
=& \|(\boldsymbol{o} + \boldsymbol{q} + \boldsymbol{\epsilon}) + \boldsymbol{\kappa}_2 - (A(\boldsymbol{o} + \boldsymbol{\kappa}_2) + B(C(\boldsymbol{o} + \boldsymbol{\kappa}_1) + D(\boldsymbol{o} + \boldsymbol{q} + \boldsymbol{\epsilon} + \boldsymbol{\kappa}_1)))\|_2^2 && (22)\\
=& \|(A + BC + BD - I)\boldsymbol{o} + (BD - I)(\boldsymbol{q} + \boldsymbol{\epsilon}) + (BC + BD)\boldsymbol{\kappa}_1 + (I - A)\boldsymbol{\kappa}_2\|_2^2 && (23)\\
=& \|(BD - I)(\boldsymbol{q} + \boldsymbol{\epsilon})\|_2^2 + \|(A + BC + BD - I)\boldsymbol{o}\|_2^2 \\
& \qquad + \|B(C + D)\boldsymbol{\kappa}_1\|_2^2 + \|(I - A)\boldsymbol{\kappa}_2\|_2^2 && (24)
\end{aligned}$$

Where the last line follows since $\boldsymbol{\kappa}_1$ and $\boldsymbol{\kappa}_2$ are sampled independently of all other terms and $\mathbb{E}[\boldsymbol{o}^T(\boldsymbol{q} + \boldsymbol{\epsilon})] = 0$. Hence, we are left with the vanilla linear LAM loss in (15) plus two additional terms.

Minimizing this function can be achieved by exactly setting $A = I$ and $C = -D$, which zeros the last three terms simultaneously. In this case, the optimization problem again reduces to PCA $\|(BD - I)(\boldsymbol{q} + \boldsymbol{\epsilon})\|_2^2$.

Discussion. In the correlated case $\mathbb{E}[\boldsymbol{o}^T(\boldsymbol{q} + \boldsymbol{\epsilon})] \neq 0$, when the third term (the cross term) in (17) cannot be ignored, nevertheless there is encouragement to reach $A = I$ and $C = -D$.

C.3 Proof of Proposition 4.4

Based on the conclusion in Proposition 4.3, optimizing for the first term in $\mathcal{L}_a$ results in $A = I$ and $C = -D$. Since $\boldsymbol{\epsilon}^T X = 0$, we have $\boldsymbol{q}^T \boldsymbol{\epsilon} = 0$. We consider the case where $\mathcal{D}$ and $\mathcal{D}_a$ come from the same distribution and the availability of action labels are uniformly random. Therefore, we can ignore the expectation outside the RHS (for simplicity) and re-write the loss as,

$$\mathcal{L}_a = \|(BD - I)(\boldsymbol{q} + \boldsymbol{\epsilon})\|_2^2 + \lambda \|ED\,(\boldsymbol{q} + \boldsymbol{\epsilon}) - \boldsymbol{a}\|_2^2 \tag{25}$$

$$= \|(BD - I)\boldsymbol{q}\|_2^2 + \|(BD - I)\boldsymbol{\epsilon}\|_2^2 + \lambda \|ED\boldsymbol{q} - \boldsymbol{a}\|_2^2 + \lambda \|ED\boldsymbol{\epsilon}\|_2^2 \tag{26}$$

$$= \|(BDX - X)\boldsymbol{a}\|_2^2 + \|(BD - I)\boldsymbol{\epsilon}\|_2^2 + \lambda \|(EDX - I_{d_a})\boldsymbol{a}\|_2^2 + \lambda \|ED\boldsymbol{\epsilon}\|_2^2 \tag{27}$$

We decompose X as $X = U\Sigma V^T$ with $U \in \mathbb{R}^{d_o \times d_a}$ (which spans the column space of X), $\Sigma \in \mathbb{R}^{d_a \times d_a}$, and $V \in \mathbb{R}^{d_a \times d_a}$.

We will show that, when $\lambda \to +\infty$, $D = \begin{bmatrix} U^T \\ 0 \end{bmatrix}$, $B = [U, *]$, and $E = [V\Sigma^{-1}, *]$ minimizes $\mathcal{L}_a$ where $*$ indicates arbitrary entries and $*$ vanishes when $d_z = d_a$.

First, this solution zeros the last two terms. For the last term, $D\boldsymbol{\epsilon} = \begin{bmatrix} U^T \\ 0 \end{bmatrix} \boldsymbol{\epsilon} = 0$ since $\boldsymbol{\epsilon}^T X = 0$ and U spans the column space of X. For the third term, $EDX = [V\Sigma^{-1}, *] \begin{bmatrix} U^T \\ 0 \end{bmatrix} U\Sigma V^T = I_{d_a}$. Then, since we have determined D, the second term becomes $\|(BD - I)\boldsymbol{\epsilon}\|_2^2 = \|\boldsymbol{\epsilon}\|_2^2$. We can choose $B = [U, *]$ to zero the first term since $BDX = [U, *] \begin{bmatrix} U^T \\ 0 \end{bmatrix} X = X$. In this way, we obtain the minimal loss $\mathcal{L}_a^* = \|\boldsymbol{\epsilon}\|_2^2$. We can also verify that $B\boldsymbol{z} = B(C\boldsymbol{o} + D\boldsymbol{o}') = BD\boldsymbol{q} = UU^T\boldsymbol{q} = \boldsymbol{q}$.

Though the $\lambda \to +\infty$ setting is artificial, the analysis show that a positive λ will bias the encoder D to capture less about the noise $\boldsymbol{\epsilon}$ and more about the signal $\boldsymbol{q}$ (or $\boldsymbol{a}$).