Learning Factorized Diffusion Policies for Conditional Action Diffusion

Omkar Patil[1], Eric Rosen , and Nakul Gopalan[2]

[1,2]SCAI, Arizona State University

***Abstract*—Diffusion models have emerged as a promising choice for learning robot skills from demonstrations. However, they face three problems: diffusion models are not sample-efficient, data is expensive to collect in robotics, and the space of tasks is combinatorially large. The established way of learning diffusion policies has little room to accommodate solutions for the aforementioned challenges, in addition to scaling the model size and paired observation-action data. In this work, we propose a novel method for training diffusion models termed 'Composable Diffusion Guidance' CoDiG to compositionally learn diffusion policies for robot skills with respect to different observation modalities, such as proprioception, vision and tactile. CoDiG provides more flexibility to deal with such observational modalities, leading to sample-efficiency gains over 20% in applicable tasks. CoDiG opens up more avenues for research on foundation models as it ameliorates the requirement of scaling all the observational modalities together during data collection.**

I. Introduction

Diffusion models have emerged as a promising choice for learning robot skills from demonstrations [4]. However, diffusion models are not sample efficient and require the collection of large amounts of demonstration data. Unfortunately, data collection in robotics is not easy due to the coupling of various modalities such as robot actions, vision, tactile, text etc. Previous work [4] has used diffusion models to learn the conditional distribution of action with respect to visual, proprioceptive, and other observations. However, this forces the collection of paired action, visual, tactile, and other modality data as demonstrations, which can be restrictive and difficult to scale. Instead, we propose a novel method CoDiG 'Composable Diffusion Guidance' utilizing Bayes' theorem to split the distribution of the robot actions conditioned on different observation modalities. For instance, the existing way to train visuomotor diffusion policies is to learn a conditional distribution of actions with respect to the visual and proprioceptive observations. Instead, CoDiG learns a diffusion 'motion-model' on the action distribution of skills and a visual-'guidance model', that when composed with the motion-model results in generation of the action conditioned on the visual observations.

opatil3@asu.edu

Critically, CoDiG enables the training of models conditioned on an additional observational modality with respect to the rest, so that data collection efforts can be concentrated on the cheaper modality. CoDiG does not require manual adjustment of compositional weights such as Wang et al. [25]. Our preliminary experiments on learning visuomotor policies for selected tasks show that CoDiG is highly sample efficient and robust to visual changes in the environment, where current diffusion policies drastically fail. Our contributions are as follows.

- We present a novel framework for training diffusion models on robot demonstration data called CoDiG, abbreviated for 'Composable Diffusion Guidance'. CoDiG decouples the observational modalities, enabling residual learning of an additional modality with respect to the rest.
- Our results show that visuomotor CoDiG performs favorably with respect to the conventional diffusion models on Adroit hand manipulation environments [10] and several RLBench [14] tasks. Further experimentation is required to validate the efficacy and robustness of CoDiG for long horizon tasks and precise manipulation.
- Existing diffusion policies drastically fail with small variations in the visual scene from the training data distribution. Visuomotor CoDiG is relatively robust to visual adaptation and responds stronger to additional demonstrations collected in the new visual scene.

II. Methodology

A. Theoretical Results

Assume that we have robot demonstrations $D = \{(\boldsymbol{x}, \boldsymbol{y})_i\}$ where $i = 1..N$, consisting of actions $\boldsymbol{x}$ and different observation modalities $\boldsymbol{y}^k$ such as camera images, task description and proprioception data. We are interested in learning $p(\boldsymbol{x}|\boldsymbol{y})$ from the data, such that given a task description, current camera images, state of the robot and other observations, we can sample an action $\boldsymbol{x}$ with a high likelihood in the demonstration data distribution.

Most treatments of diffusion models have considered distributions of a single entity such as images [13, 16, 20]. This formulation has been directly adopted by the robotics community [4] leading to the popular optimization objective shown in Equation 1.

$$\mathcal{L}_t(\boldsymbol{\theta}) = \mathbb{E}_{q(\boldsymbol{x}_0,\boldsymbol{y})\mathcal{N}(\boldsymbol{\epsilon}_0;0,\mathcal{I})}\left[[||\boldsymbol{\epsilon}_0 - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t,\boldsymbol{y},t)||_2^2]\right] \quad (1)$$

$$\boldsymbol{x}_{t-1} \sim \mathcal{N}\left(\boldsymbol{x}_t; \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t,\boldsymbol{y},t)\right), \sqrt{1-\alpha_t}\mathcal{I}\right) \quad (2)$$

Here, the network ϵ_θ is also parametrized with $\boldsymbol{y}$ for the conditional prediction of the noise added to the action $\boldsymbol{x}$. Once the network $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t,\boldsymbol{y},t)$ is trained, ancestral sampling shown in Equation 2 can be used to sample actions given the observations $\boldsymbol{y}$. However, while prior work assumes that ϵ_θ learns the score of the conditional distribution $p(\boldsymbol{x}|\boldsymbol{y})$, there is no formal proof provided for the same. Hence, we present our first result, where we show that a conditional diffusion process as defined by [5] does indeed result in the loss of Equation 1 [4] being a maximizer of the evidence lower bound (ELBO) of the log-likelihood of the conditional data distribution $\log q(\boldsymbol{x}|\boldsymbol{y})$.

Theorem II.1. *The diffusion loss function $\mathcal{L}_t(\boldsymbol{\theta})$ as defined in Equation 1, in expectation over the time-steps $1 \leq t \leq T$, is equivalent to maximizing the ELBO of the log-likelihood of the conditional data distribution $\log q(\boldsymbol{x}|\boldsymbol{y})$, under a conditional Markovian noising process $\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$ and the reverse transition kernel as $\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y})$.*

The proof for Theorem II.1 can be found in the Appendix B-A. We now proceed to our main result. We would like to decouple different observational modalities $\boldsymbol{y}^k$, $1 \leq k \leq M$, so that we can prioritize their collection based on cost. Assume that we have trained a diffusion policy with k observational modalities and we collect N additional demonstrations with the $(k+1)^{th}$ modality in conjunction with the existing k. The score of the new conditional distribution $p(\boldsymbol{x}|\boldsymbol{y}^{1:k},\boldsymbol{y}^{k+1})$, where $\boldsymbol{y}^{1:k} \equiv \boldsymbol{y}^1,..,\boldsymbol{y}^k$, can be written using Bayes's theorem along the lines of Equation 18 at diffusion time-step t as follows:

$$\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{y}^{1:k},\boldsymbol{y}^{k+1};\boldsymbol{\theta},\boldsymbol{\phi}) \quad (3)$$

$$= \nabla_{x_t} \log p(\boldsymbol{y}^{k+1}|\boldsymbol{x}_t,\boldsymbol{y}^{1:k};\boldsymbol{\phi}) + \nabla_{x_t} \log p(\boldsymbol{x}_t|\boldsymbol{y}^{1:k};\boldsymbol{\theta}) \quad (4)$$

Here $\nabla_{x_t} \log p(\boldsymbol{x}_t|\boldsymbol{y}^{1:k};\boldsymbol{\theta})$ is the score of the model trained on the original k observational modalities, while $\nabla_{x_t} \log p(\boldsymbol{y}^{k+1}|\boldsymbol{x}_t,\boldsymbol{y}^{1:k};\boldsymbol{\phi})$ corresponds to the score of the classifier for the modality $\boldsymbol{y}^{k+1}$. The method proposed by classifier guided-diffusion [5] to explicitly train a classifier $p(\boldsymbol{y}^{k+1}|\boldsymbol{x}_t,\boldsymbol{y}^{1:k})$ on the noisy samples of $\boldsymbol{x}_t$ and $\boldsymbol{y}^{1:k}$ does not apply due to continuous and high-dimensional observation modalities such as images, rather than simple classes such as cats or dogs.

The central idea of this work is to instead directly parametrize the gradient of the classifier $\nabla_{x_t} \log p(\boldsymbol{y}^{k+1}|\boldsymbol{x}_t,\boldsymbol{y}^{1:k};\boldsymbol{\phi})$ using a neural network, which we refer to as the guidance model π_g, rather than to learn a classifier and then obtain its gradients. To learn the guidance model, we minimize the Fisher divergence between the guidance model and the true score of the classifier.

$$D_F(p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})||p_{\alpha,\tau}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})) = \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}$$
$$\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}) - \nabla_{\boldsymbol{x}_t} log\ p_{\alpha,\tau}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})||_2^2\right] \quad (5)$$

Here, observation modalities $\boldsymbol{y}^{1:k+1}$ can be noised with a Gaussian kernel $\mathcal{N}(\tilde{\boldsymbol{y}};\boldsymbol{y},\tau^2 I)$ of variance τ^2 that is small enough such that $p_\tau(\tilde{\boldsymbol{y}}^i) \approx p(\boldsymbol{y}^i)$. Robot action $\boldsymbol{x}$ is noised with the diffusion transition kernel of $\mathcal{N}(\boldsymbol{x}_t;\sqrt{\alpha_t}\boldsymbol{x},(1-\alpha_t)\mathcal{I})$. However, Equation 5 is difficult to use in it's current form as estimation of the true score is difficult for large datasets. Chao et al. [3] derive the denoising likelihood score matching (DLSM) objective for conditional distributions, which forms the basis of our next result.

Theorem II.2. *Denoising score matching for the guidance model π_g: $\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})$, as expressed in Equation 5 is equal up to a constant to the following loss:*

$$L^t_{DLSM}(\boldsymbol{\phi}) = \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x},\boldsymbol{x}_t,\boldsymbol{y}^{1:k+1},\tilde{\boldsymbol{y}}^{1:k+1})}\Big[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})$$
$$+ \nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\tilde{\boldsymbol{y}}^{1:k}) - \nabla_{\boldsymbol{x}_t} log\ p_\alpha(\boldsymbol{x}_t|\boldsymbol{x})||_2^2\Big] \quad (6)$$

For diffusion models, we take the forward transition kernel as $p_\alpha(\boldsymbol{x}_t|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}_t;\sqrt{\alpha_t}\boldsymbol{x},(1-\alpha_t)\mathcal{I})$. We noise the observations $\boldsymbol{y}^{1:k+1}$ with a Gaussian kernel $\mathcal{N}(\tilde{\boldsymbol{y}};\boldsymbol{y},\tau^2 I)$ of variance τ^2 that is small enough such that $p_\tau(\tilde{\boldsymbol{y}}^i) \approx p(\boldsymbol{y}^i)$.

The proof for Theorem II.2 is a conditional variant of the one derived by [3], and is presented in Appendix B-B. Equation 6 can now be used in practice to learn the guidance model. Simplifying $\nabla_{\boldsymbol{x}_t} log\ p_\alpha(\boldsymbol{x}_t|\boldsymbol{x})$ to $-\epsilon_0/\sqrt{1-\bar{\alpha}_t}$, where $\epsilon_0 \sim \mathcal{N}(0,\mathcal{I})$, we obtain:

$$L_{DLSM}(\boldsymbol{\phi}) = \mathbb{E}_{p_\tau(\boldsymbol{x},\boldsymbol{y}^{1:k+1},\tilde{\boldsymbol{y}}^{1:k+1})}\mathbb{E}_{\epsilon_0\sim\mathcal{N}(0,\mathcal{I})}\Big[\frac{1}{2}||\hat{\epsilon}_\phi(\tilde{\boldsymbol{y}}^{k+1},\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})$$
$$+ \epsilon_\theta(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}) - \epsilon_0||_2^2\Big] \quad (7)$$

Here, ϵ_θ is frozen and known from prior training with observational modalities $\boldsymbol{y}^{1:k}$, while ϵ_ϕ is learned in expectation over data collected in conjugation with modality $\boldsymbol{y}^{k+1}$. Equation 7 is very similar to the diffusion loss Equation 1, except the model is expected to learn a residual of the noise added to the action sample x_t taken as the input. Once, ϵ_ϕ is learned, actions can be

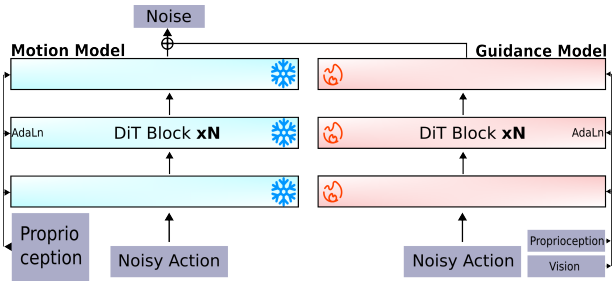

Fig. 1. The simple architecture adds the noise predictions from the two models at each denoising time-step. The motion model is frozen while training the guidance model, which takes an additional observational modality as vision.

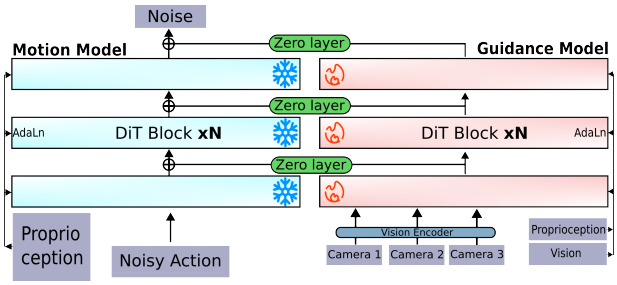

Fig. 2. The improved architecture **CoDiG**, inspired from Controlnet, where the instead of adding the model outputs at the end, we add interrim block outputs from DiT. We pass the visual embeddings into the transformer and condition them on the noisy action using adaLn.

sampled from the conditional distribution $p(\boldsymbol{x}|\boldsymbol{y}^{1:k+1})$ using ancestral sampling and Equation 4:

$$\boldsymbol{x}_{t-1} \sim \mathcal{N}\left(\boldsymbol{x}_t; \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}(\boldsymbol{x}_t, \boldsymbol{y}^{1:k+1}, t)\right), \sqrt{1-\alpha_t}\mathcal{I}\right) \tag{8}$$

$$\boldsymbol{\epsilon}(\boldsymbol{x}_t, \boldsymbol{y}^{1:k+1}, t) = \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\phi}}(\boldsymbol{x}_t, \boldsymbol{y}^{1:k+1}, t) + \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, \boldsymbol{y}^{1:k}, t) \tag{9}$$

Note that the score of the prior distribution $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, \boldsymbol{y}^{1:k}, t)$ or $\nabla_{x_t} \log p(\boldsymbol{x}_t|\boldsymbol{y}^{1:k}; \boldsymbol{\theta})$ can further be decomposed with respect to observational modalities as shown above. However, data collection for subsequent modalities $\boldsymbol{y}^{k+1}$ must be done in conjunction with existing ones $\boldsymbol{y}^{1:k}$.

B. Visual CoDiG with Motion-model

We now provide a concrete implementation of the method proposed in Section II-A. Existing diffusion models trained as visuo-motor polices learn the score for the conditional distribution $p(\boldsymbol{x}|\boldsymbol{y}^r, \boldsymbol{y}^c)$, where $\boldsymbol{y}^r$ corresponds to the proprioceptive observations and $\boldsymbol{y}^c$ correspond to the visual observations from cameras. We propose to decouple these observational modalities, and instead learn the scores for a 'motion-model' $p(\boldsymbol{x}|\boldsymbol{y}^r)$ and a vision 'guidance-model' $p(\boldsymbol{y}^c|\boldsymbol{x}, \boldsymbol{y}^r)$. The equations of diffusion loss for the motion model 1 and the guidance model 7 can be written as follows, and is showcased in Figure 1.

$$\mathcal{L}_{MM}(\boldsymbol{\theta}) = \mathbb{E}_{q(\boldsymbol{x}_0, \boldsymbol{y}^r)\mathcal{N}(\boldsymbol{\epsilon}_0; 0, \mathcal{I})}\left[[||\boldsymbol{\epsilon}_0 - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, \boldsymbol{y}^r, t)||_2^2]\right] \tag{10}$$

$$L_G(\boldsymbol{\phi}) = \mathbb{E}_{p_\tau(\boldsymbol{x}, \boldsymbol{y}^r, \boldsymbol{y}^c, \tilde{\boldsymbol{y}}^r, \tilde{\boldsymbol{y}}^c)}\mathbb{E}_{\epsilon_0 \sim \mathcal{N}(0,\mathcal{I})}\left[\frac{1}{2}||\epsilon_0 - \hat{\epsilon}_\phi(\tilde{\boldsymbol{y}}^c, \boldsymbol{x}_t, \tilde{\boldsymbol{y}}^r, t) - \epsilon_\theta(\boldsymbol{x}_t, \tilde{\boldsymbol{y}}^r, t)||_2^2\right] \tag{11}$$

Instead of having two learned components in Equation 11, we can simplify the learning as follows.

$$L_G(\boldsymbol{\phi}) = \mathbb{E}_{p_\tau(\boldsymbol{x}, \boldsymbol{y}^r, \boldsymbol{y}^c, \tilde{\boldsymbol{y}}^r, \tilde{\boldsymbol{y}}^c)}\mathbb{E}_{\epsilon_0 \sim \mathcal{N}(0,\mathcal{I})}\left[\frac{1}{2}||\epsilon_0 - \hat{\epsilon}_\gamma(\tilde{\boldsymbol{y}}^c, \boldsymbol{x}_t, \tilde{\boldsymbol{y}}^r, t)||_2^2\right]$$

$$\hat{\epsilon}_\gamma(\phi, \boldsymbol{x}_t, \tilde{\boldsymbol{y}}^r, t) = \epsilon_\theta(\boldsymbol{x}_t, \tilde{\boldsymbol{y}}^r, t) \tag{12}$$

We implement this model using a ControlNet [26] inspired architecture, as shown in Figure 2. Unlike the simple architecture shown in Figure 1, this variant adds the outputs of each DiT block before passing the sum to the consequent block in the motion model. To prevent harmful updates at the start of training, a zero-initialized layer is applied to the guidance model outputs, similar to ControlNet. Moreover, our architecture benefits from attention being applied to the image embeddings. which are in-turn conditioned on the noisy actions in the guidance model.

Note that we expect this approach to have a distinct advantage for visuomotor skills that are heavily dependent on proprioception, as we learn a residual for vision. We hypothesize that the existing way to train visuomotor policies using diffusion models do not learn the right dependencies when presented with all the observational modalities together, leading to lower sample-efficiency.

III. Results

A. Environments and Baselines

Since we propose an alternative to the existing way of training diffusion models, our baseline is a model that takes in all the observations and predicts the robot action, learning the conditional distribution. We choose DiT-small (∼90M) [17] as our model architecture, which is kept the same for both the baseline and CoDiG. For comparison, we also include UNet [18] implemented by [4] as a part of the baselines. We also include POCO [25], where we compose a motion policy and pre-learned visuomotor policy using a compositional weight of 0.2, as suggested in the paper. Finally, we also showcase the results for classifier-free guidance proposed by Ho and Salimans [11], where we switch out the vision condition with a probability of 0.1. Further details on the baselines, experiments, and the environments used are provided in Appendix C.

TABLE I
PERFORMANCE RESULTS ACROSS DIFFERENT RLBENCH TASKS FOR 10, 50 AND 100 DEMONSTRATIONS.

Task	Number of Demonstrations	DiT		UNet		DiT-cfg		POCO		CoDiG	
		Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
OpenBox	10	21.0	± 2.0	7.5	± 0.7	20.3	± 4.0	27.7	± 4.9	**59.0**	± 4.0
	50	86.0	± 1.7	86.0	± 1.0	86.0	± 2.6	87.3	± 0.6	**90.0**	± 2.6
	100	**98.3**	± 1.5	93.0	± 1.0	92.3	± 2.5	**98.3**	± 1.5	91.3	± 4.5
CloseBox	10	25.7	± 5.9	30.0	± 6.2	19.7	± 4.7	29.7	± 3.5	**71.7**	± 6.8
	50	79.7	± 1.2	70.0	± 3.6	75.0	± 3.5	79.7	± 1.2	**87.7**	± 3.1
	100	85.7	± 1.5	83.7	± 3.8	**88.7**	± 1.5	**88.3**	± 1.5	**88.3**	± 2.9
OpenDoor	10	24.7	± 4.2	7.7	± 1.5	12.7	± 1.5	13.0	± 4.4	**42.0**	± 5.2
	50	44.0	± 1.7	11.3	± 3.5	22.3	± 9.3	38.0	± 5.0	**54.7**	± 5.0
	100	44.7	± 4.2	23.3	± 3.5	60.0	± 0.0	42.0	± 2.6	**65.0**	± 2.6
CloseDoor	10	2.0	± 1.0	3.0	± 1.0	0.0	± 0.0	4.3	± 0.6	**9.0**	± 1.7
	50	0.0	± 0.0	**7.3**	± 1.5	1.7	± 0.6	1.7	± 0.6	**6.7**	± 3.5
	100	2.7	± 1.5	**8.0**	± 0.0	5.0	± 0.0	5.0	± 2.6	**9.0**	± 1.0
Basketball in Hoop	10	1.7	± 1.5	2.7	± 1.5	1.7	± 1.2	2.0	± 1.0	**10.7**	± 2.5
	50	18.0	± 3.5	17.0	± 1.0	13.7	± 5.5	20.7	± 4.2	**47.3**	± 3.2
	100	38.7	± 4.2	34.3	± 4.2	46.0	± 4.0	42.7	± 4.0	**63.3**	± 3.8

B. Learning Visuomotor Policies

The results for RLBench are shown in Table I. CoDiG generally outperforms all baselines in the presented tasks across different number of demonstrations. CoDiG achieves 20% higher performance on average with 10 demonstrations, and 10% higher performance on average with 100 demonstrations. CoDiG is very sample efficient with low number of demonstrations as the model learns the motion of the task and then learns a residual on the visual observations. On the Adroit dataset, CoDiG does better by 3% on average, and the results can be found in Appendix D.

C. Robustness to Visual Changes

We introduce modifications in tasks from RLBench to test how learned models adapt to distribution shifts in the visual observations. We introduce two types of modifications- change in the color of the object being manipulated, and addition of visual distractors. Further, we also collect 5 additional demonstrations in the new modified environment to understand how models benefit from few-shot training on the out-of-distribution data. The results are shown in Table II. CoDiG is more robust to changes in the visual scene of the tasks, while the conventionally trained diffusion models fail miserably. Note that, CoDiG is not limited to the performance of the motion-model in zero-shot setting, which is 17%, 4% and 19% for *OpenBox*, *BasketballInHoop* and *OpenDoor* respectively, with 100 demonstrations. Further, CoDiG also responds better to additional demonstrations in the modified environment. This is expected, as we only update the guidance model with the additional demonstrations, in-effect, only updating the conditional distribution of the visual observation modality $p(y^c|x, y^r)$, rather than updating the conditional action distribution $p(x|y^r, y^c)$.

Task	Number of Demonstrations	DiT	DiT GMod
OpenBox	100 demos original	**98.3** ± 1.5	91.3 ± 4.5
	zero-shot blue box	43.3 ± 2.5	**46.7**± 1.5
	5 demos blue box	34.7 ± 3.5	**76.7** ± 0.6
	zero-shot distractors	1.7 ± 2.1	**16.7**± 2.1
	5 demos distractors	42.3 ± 4.0	**53.3** ± 2.3
Basketball in Hoop	100 demos original	38.7 ± 4.2	**63.3** ± 3.8
	zero-shot blue ball	29.0 ± 8.7	**63.0** ± 1.0
	5 demos blue ball	13.0 ± 0.0	**45.0** ± 2.6
	zero-shot distractors	0.7 ± 1.2	**56.3** ± 3.2
	5 demos distractors	2.7 ± 1.2	**39.7** ± 4.9
Open Door	100 demos original	44.7 ± 4.2	**65.0** ± 2.6
	zero-shot blue door	0.0 ± 0.0	**14.3** ± 3.1
	5 demos blue door	17.7 ± 3.1	**52.0** ± 7.2
	zero-shot red door	0.3 ± 0.6	**30.7** ± 2.5
	5 demos red door	20.0 ± 5.2	**53.7** ± 3.8

TABLE II
PERFORMANCE COMPARISON ACROSS TASKS WITH VISUAL ADAPTATIONS FOR DIT AND CODIG.

IV. CONCLUSION

We present a novel compositional method to train diffusion models by decoupling different observational modalities such as proprioception, vision and tactile. Our factorized approach yields a loss function that is simple to use, while also presenting a more intuitive modeling paradigm. We also showcase CoDiG, an architectural implementation that makes the factorized training simpler and more efficient. We find that visuomotor CoDiG performs strongly over its baselines for tasks with a narrow distribution of robot motion. We also show that CoDiG is more robust to distribution shift in its visual observations than current diffusion policies. Several limitations exist, such as the choice to split the observation conditionals must be undertaken based on the task at hand. More experiments are needed to validate the efficacy, robustness and scaling properties of CoDiG.

REFERENCES

[1] Brian DO Anderson. Reverse-time diffusion equation models. Stochastic Processes and their Applications, 12(3):313–326, 1982.

[2] Christophe Andrieu, Nando De Freitas, Arnaud Doucet, and Michael I Jordan. An introduction to mcmc for machine learning. Machine learning, 50: 5–43, 2003.

[3] Chen-Hao Chao, Wei-Fang Sun, Bo-Wun Cheng, Yi-Chen Lo, Chia-Che Chang, Yu-Lun Liu, Yu-Lin Chang, Chia-Ping Chen, and Chun-Yi Lee. Denoising likelihood score matching for conditional score-based data generation, 2022. URL https://arxiv.org/abs/2203.14206.

[4] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. The International Journal of Robotics Research, page 02783649241273668, 2023.

[5] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis, 2021. URL https://arxiv.org/abs/2105.05233.

[6] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. Advances in neural information processing systems, 34:8780–8794, 2021.

[7] Carl Doersch. Tutorial on variational autoencoders. arXiv preprint arXiv:1606.05908, 2016.

[8] Yilun Du, Conor Durkan, Robin Strudel, Joshua B. Tenenbaum, Sander Dieleman, Rob Fergus, Jascha Sohl-Dickstein, Arnaud Doucet, and Will Grathwohl. Reduce, reuse, recycle: Compositional generation with energy-based diffusion models and mcmc, 2023.

[9] Bradley Efron. Tweedie's formula and selection bias. Journal of the American Statistical Association, 106(496):1602–1614, 2011.

[10] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning, 2020.

[11] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. arXiv preprint arXiv:2207.12598, 2022.

[12] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance, 2022. URL https://arxiv.org/abs/2207.12598.

[13] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020. URL https://arxiv.org/abs/2006.11239.

[14] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J. Davison. Rlbench: The robot learning benchmark & learning environment. CoRR, abs/1909.12271, 2019. URL http://arxiv.org/abs/1909.12271.

[15] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL https://arxiv.org/abs/1412.6980.

[16] Calvin Luo. Understanding diffusion models: A unified perspective, 2022. URL https://arxiv.org/abs/2208.11970.

[17] William Peebles and Saining Xie. Scalable diffusion models with transformers, 2023. URL https://arxiv.org/abs/2212.09748.

[18] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18, pages 234–241. Springer, 2015.

[19] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics, 2015.

[20] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models, 2022. URL https://arxiv.org/abs/2010.02502.

[21] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution, 2020.

[22] Yang Song and Diederik P. Kingma. How to train your energy-based models, 2021.

[23] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. arXiv preprint arXiv:2011.13456, 2020.

[24] Pascal Vincent. A connection between score matching and denoising autoencoders. Neural computation, 23(7):1661–1674, 2011.

[25] Lirui Wang, Jialiang Zhao, Yilun Du, Edward H Adelson, and Russ Tedrake. Poco: Policy composition from and for heterogeneous robot learning. arXiv preprint arXiv:2402.02511, 2024.

[26] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models, 2023. URL https://arxiv.org/abs/2302.05543.

APPENDIX A
BACKGROUND

A. Diffusion Models

Gaussian diffusion models [19] learn the reverse diffusion kernel $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ for a fixed forward kernel that adds Gaussian noise at each step $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{\alpha_t}\boldsymbol{x}_{t-1}, (1-\alpha_t)\mathcal{I})$, such that $q(\boldsymbol{x}_T) \approx \mathcal{N}(0, \mathcal{I})$. Here, $t <= T$ represents the diffusion time-step and α_t the noise schedule. To generate trajectories from the learned data distribution $p_{\boldsymbol{\theta}}(\boldsymbol{x}_0)$, we sample at time step T from $\mathcal{N}(0, \mathcal{I})$ and apply the reverse diffusion kernel $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ at each time step. For training the model, maximization of the evidence lower bound derived from the log-likelihood of the data distribution $\log q(\boldsymbol{x}_0)$ yields the commonly used loss function in Equation 13 [13].

$$\mathcal{L}_t(\boldsymbol{\theta}) = \mathbb{E}_{q(\boldsymbol{x}_0)\mathcal{N}(\boldsymbol{\epsilon}_0;0,\mathcal{I})}\left[\lambda_t[||\boldsymbol{\epsilon}_0 - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)||_2^2]\right] \tag{13}$$

Here, λ_t, a function of α_t is the weighting parameter for different time-steps, usually taken as 1. We train our model to predict the noise $\boldsymbol{\epsilon}_0$ added to the data sample $\boldsymbol{x}_0$ to generate the noisy sample $\boldsymbol{x}_t$ taken as input to the network. The Tweedie formula [9] can be used to show that $\boldsymbol{\epsilon}_0$, and consequently $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}$ are proportional to the score of the diffused data distribution $q(\boldsymbol{x}_t) = \int q(\boldsymbol{x}_t|\boldsymbol{x}_0)q(\boldsymbol{x}_0)d\boldsymbol{x}_0$ [16].

$$\frac{-1}{\sqrt{1-\bar{\alpha}_t}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) \approx \frac{-1}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_0 = \nabla_{\boldsymbol{x}} log\ q(\boldsymbol{x}_t) \tag{14}$$

B. Energy Based Models (EBMs)

EBMs are a class of probabilistic models of the form $p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{e^{f_{\boldsymbol{\theta}}(\boldsymbol{x})}}{Z}$ where $Z(\boldsymbol{\theta}) = \int e^{f_{\boldsymbol{\theta}}(\boldsymbol{x})}d\boldsymbol{x}$ is the normalizing constant. Denoising score matching (DSM) [24] used to train EBMs [22] minimizes the Fisher divergence between the model $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ and the Gaussian-smoothed data distribution $q(\tilde{\boldsymbol{x}}) = \int q(\boldsymbol{x})\mathcal{N}(\tilde{\boldsymbol{x}}; \boldsymbol{x}, \sigma_t^2 I)d\boldsymbol{x}$ at various noise scales σ_t.

$$\mathcal{J}_{\sigma_t}(\boldsymbol{\theta}) = \mathbb{E}_{q(\boldsymbol{x},\tilde{\boldsymbol{x}})}\left[\frac{1}{2}||\nabla_{\tilde{\boldsymbol{x}}} log\ q(\tilde{\boldsymbol{x}}|\boldsymbol{x}) - \nabla_{\tilde{\boldsymbol{x}}} log\ p_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}})||_2^2\right] \tag{15}$$

This circumvents the normalizing constant by evaluating the gradient of the log-probability of the model $\nabla_{\boldsymbol{x}} p_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}) = \nabla_{\boldsymbol{x}} f_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}})$ at different noise scales. Equation 15 simplifies to the following when $q(\boldsymbol{x}|\tilde{\boldsymbol{x}}) = \mathcal{N}(\tilde{\boldsymbol{x}}; \boldsymbol{x}, \sigma_t^2 I)$ [24]:

$$\mathcal{J}_{\sigma_t}(\boldsymbol{\theta}) = \mathbb{E}_{q(\boldsymbol{x})\mathcal{N}(\boldsymbol{\epsilon};0,\mathcal{I})}\left[||\frac{\boldsymbol{\epsilon}}{\sigma_t} + \nabla_{\boldsymbol{x}} f_{\boldsymbol{\theta}}(\boldsymbol{x} + \sigma_t\boldsymbol{\epsilon})||_2^2\right] \tag{16}$$

Once trained, MCMC methods such as Langevin [2] can be used to sample from EBMs since they only depend on the score of the data distribution. This approach is also known in the literature as score-based modeling [21].

C. Sampling from the Policy

Song et al. [23] show that score-based and denoising diffusion models can be considered as discretizations of a family of stochastic differential equations (SDE) that slowly add noise to the data distribution. For the generation process, the time-reversal of this SDE was given by Anderson [1] $d\boldsymbol{x} = \left[\boldsymbol{f}(\boldsymbol{x}, t) - g(t)^2\nabla_{\boldsymbol{x}} \log q_t(\boldsymbol{x})\right] dt + g(t)d\bar{\boldsymbol{w}}$, where $\bar{\boldsymbol{w}}$ is a standard Wiener process for reverse time, $g(t)$ is the scalar diffusion coefficient, and $f(\cdot, t)$ is the drift coefficient. Some manifestations of the reverse SDE equation are ancestral sampling [23] proposed by Ho et al. [13], shown in Equation 17, and Langevin dynamics [2].

$$\boldsymbol{x}_{t-1} \sim \mathcal{N}\left(\boldsymbol{x}_t; \frac{1}{\sqrt{\alpha_t}}\left[\boldsymbol{x}_t + (1-\alpha_t)\nabla_{\boldsymbol{x}_t} \log q(\boldsymbol{x}_t)\right], \sqrt{1-\alpha_t}\mathcal{I}\right) \tag{17}$$

Equation 15 is then used to obtain an estimate of the score of the perturbed data distribution $\nabla_{\boldsymbol{x}} \log q_t(\boldsymbol{x})$, where the transition kernel $q(\tilde{\boldsymbol{x}}|\boldsymbol{x})$ varies between approaches. Diffusion models use a forward transition kernel of $\mathcal{N}(\boldsymbol{x}_t; \sqrt{\bar{\alpha}_t}\boldsymbol{x}_{t-1}, (1-\bar{\alpha}_t)I)$, yielding the same loss as Equation 13, while score-based model typically use $\mathcal{N}(\boldsymbol{x}_t; \boldsymbol{x}_{t-1}, \sigma_t^2 I)$, where α_t and σ_t are respective noise scales.

D. Classifier Guided Diffusion

Classifier guided diffusion [5] has received considerable attention due to its ability to reuse existing diffusion models to sample images conforming to specific classes. To sample from a class $\boldsymbol{y}$, Equation 18 as a result of Bayes' theorem allows us to decompose the conditional score at time-step t into the gradient of the classifier and the unconditional score.

$$\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{y}; \boldsymbol{\theta}, \boldsymbol{\phi}) = \nabla_{x_t} \log p(\boldsymbol{y}|\boldsymbol{x}_t; \boldsymbol{\phi}) + \nabla_{x_t} \log p(\boldsymbol{x}_t; \boldsymbol{\theta}) \tag{18}$$

Classifier guided-diffusion requires a classifier trained on noisy samples to get accurate estimate of the gradients $\nabla_{x_t} \log p(\boldsymbol{y}|\boldsymbol{x}_t; \boldsymbol{\phi})$. Sampling from the class $\boldsymbol{y}$ can then be achieved for diffusion models by substituting a reweighted version of Equation 18 for $\nabla_{\boldsymbol{x}_t} \log q(\boldsymbol{x}_t)$ in the ancestral sampling Equation 17.

APPENDIX B
METHODOLOGY

A. Proof for Theorem II.1

Theorem B.1. *The diffusion loss function $\mathcal{L}_t(\boldsymbol{\theta})$ as defined in Equation 1, in expectation over the time-steps $1 \leq t \leq T$, is equivalent to maximizing the ELBO of the log-likelihood of the conditional data distribution $\log q(\boldsymbol{x}|\boldsymbol{y})$, under a conditional Markovian noising process $\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$ and the reverse transition kernel as $\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{y})$.*

Here, we derive the diffusion loss function for the conditional distribution $p(\boldsymbol{x}|\boldsymbol{y})$ instead of only $p(\boldsymbol{x})$. A parallel derivation for conditional variational auto-encoders can be found in Doersch [7]. Following Dhariwal and Nichol [6], we start with a conditional Markovian noising forward process $\hat{q}$ similar to $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{\alpha_t}\boldsymbol{x}_{t-1}, (1-\alpha_t)\mathcal{I})$, and define the following:

$$\hat{q}(\boldsymbol{x}_0) := q(\boldsymbol{x}_0) \tag{19}$$

$$\hat{q}(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{y}) := q(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t) \tag{20}$$

$$\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0, \boldsymbol{y}) := \prod_{t=1}^{T} \hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{y}) \tag{21}$$

We now reproduce some results that will be used later in the derivation of diffusion loss for conditional distributions. Dhariwal and Nichol [6] also show that

$$\hat{q}(\boldsymbol{y}|\boldsymbol{x}_t, \boldsymbol{x}_{t+1}) = \hat{q}(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{y})\frac{\hat{q}(\boldsymbol{y}|\boldsymbol{x}_t)}{\hat{q}(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t)} \tag{22}$$

$$= \hat{q}(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t)\frac{\hat{q}(\boldsymbol{y}|\boldsymbol{x}_t)}{\hat{q}(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t)} \tag{23}$$

$$= \hat{q}(\boldsymbol{y}|\boldsymbol{x}_t) \tag{24}$$

Moreover, the unconditional reverse transition kernels can be shown to be equal using Bayes theorem, given Equations 19 and 20: $\hat{\boldsymbol{q}}(\boldsymbol{x}_t|\boldsymbol{x}_{t+1}) = \boldsymbol{q}(\boldsymbol{x}_t|\boldsymbol{x}_{t+1})$. Dhariwal and Nichol [6] use the result from Equation 24 to show the following for conditional reverse transition kernels.

$$\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_{t+1}, \boldsymbol{y}) = \frac{\hat{q}(\boldsymbol{x}_t, \boldsymbol{x}_{t+1}, \boldsymbol{y})}{\hat{q}(\boldsymbol{x}_{t+1}, \boldsymbol{y})} \tag{25}$$

$$= \frac{\hat{q}(\boldsymbol{x}_t, \boldsymbol{x}_{t+1}, \boldsymbol{y})}{\hat{q}(\boldsymbol{y}|\boldsymbol{x}_{t+1})\hat{q}(\boldsymbol{x}_{t+1})} \tag{26}$$

$$= \frac{\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_{t+1})\hat{q}(\boldsymbol{y}|\boldsymbol{x}_t, \boldsymbol{x}_{t+1})\hat{q}(\boldsymbol{x}_{t+1})}{\hat{q}(\boldsymbol{y}|\boldsymbol{x}_{t+1})\hat{q}(\boldsymbol{x}_{t+1})} \tag{27}$$

$$= \frac{\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_{t+1})\hat{q}(\boldsymbol{y}|\boldsymbol{x}_t, \boldsymbol{x}_{t+1})}{\hat{q}(\boldsymbol{y}|\boldsymbol{x}_{t+1})} \tag{28}$$

$$= \frac{q(\boldsymbol{x}_t|\boldsymbol{x}_{t+1})\hat{q}(\boldsymbol{y}|\boldsymbol{x}_t)}{\hat{q}(\boldsymbol{y}|\boldsymbol{x}_{t+1})} \tag{29}$$

Further, we can show the following using Equations 20 and 21 and the Markovian noising process. It states that the joint distribution of the noised samples conditioned on $\boldsymbol{y}$ and $\boldsymbol{x}_0$ are the same for both $\hat{q}$ and q.

$$\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0, \boldsymbol{y}) = \prod_{t=1}^{T} \hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{y}) \tag{30}$$

$$= \prod_{t=1}^{T} q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) \tag{31}$$

$$= q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0) \tag{32}$$

We adapt the derivation of diffusion loss from Luo [16] to work with conditional distributions by maximizing the log-likelihood of the conditional data distribution $log\ p(\boldsymbol{x}|\boldsymbol{y})$ leading to evidence lower bound (ELBO).

$$\log p(\boldsymbol{x}|\boldsymbol{y}) = \log \int p(\boldsymbol{x}_{0:T}|\boldsymbol{y}) d\boldsymbol{x}_{1:T} \tag{33}$$

$$= \log \int \frac{p(\boldsymbol{x}_{0:T}|\boldsymbol{y})\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0, \boldsymbol{y})}{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0, \boldsymbol{y})} d\boldsymbol{x}_{1:T} \tag{34}$$

$$= \log \mathbb{E}_{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0, \boldsymbol{y})} \left[\frac{p(\boldsymbol{x}_{0:T}|\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0, \boldsymbol{y})} \right] \tag{35}$$

$$\geq \mathbb{E}_{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log \frac{p(\boldsymbol{x}_{0:T}|\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0,\boldsymbol{y})}\right] \tag{36}$$

The ELBO can be further simplified as follows

$$\log p(\boldsymbol{x}|\boldsymbol{y}) \geq \mathbb{E}_{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log \frac{p(\boldsymbol{x}_{0:T}|\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0,\boldsymbol{y})}\right] \tag{37}$$

$$= \mathbb{E}_{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log \frac{p(\boldsymbol{x}_T|\boldsymbol{y})\prod_{t=1}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y})}{\prod_{t=1}^{T}\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1},\boldsymbol{y})}\right] \tag{38}$$

$$= \mathbb{E}_{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log \frac{p(\boldsymbol{x}_T|\boldsymbol{y})p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1,\boldsymbol{y})\prod_{t=2}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_1|\boldsymbol{x}_0,\boldsymbol{y})\prod_{t=2}^{T}\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1},\boldsymbol{y})}\right] \tag{39}$$

$$= \mathbb{E}_{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log \frac{p(\boldsymbol{x}_T|\boldsymbol{y})p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1,\boldsymbol{y})\prod_{t=2}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_1|\boldsymbol{x}_0,\boldsymbol{y})\prod_{t=2}^{T}\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1},\boldsymbol{x}_0,\boldsymbol{y})}\right] \tag{40}$$

$$= \mathbb{E}_{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log \frac{p_(\boldsymbol{x}_T|\boldsymbol{y})p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1,\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_1|\boldsymbol{x}_0,\boldsymbol{y})} + \log\prod_{t=2}^{T}\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1},\boldsymbol{x}_0,\boldsymbol{y})}\right] \tag{41}$$

$$= \mathbb{E}_{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log \frac{p(\boldsymbol{x}_T|\boldsymbol{y})p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1,\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_1|\boldsymbol{x}_0,\boldsymbol{y})} + \log\prod_{t=2}^{T}\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y})}{\frac{\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0,\boldsymbol{y})\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_0,\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_0,\boldsymbol{y})}}\right] \tag{42}$$

$$= \mathbb{E}_{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log \frac{p(\boldsymbol{x}_T|\boldsymbol{y})p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1,\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_1|\boldsymbol{x}_0,\boldsymbol{y})} + \log\prod_{t=2}^{T}\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y})}{\frac{\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0,\boldsymbol{y})\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_0,\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_0,\boldsymbol{y})}}\right] \tag{43}$$

$$= \mathbb{E}_{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log \frac{p(\boldsymbol{x}_T|\boldsymbol{y})p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1,\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_1|\boldsymbol{x}_0,\boldsymbol{y})} + \log\frac{\hat{q}(\boldsymbol{x}_1|\boldsymbol{x}_0,\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_T|\boldsymbol{x}_0,\boldsymbol{y})} + \log\prod_{t=2}^{T}\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0,\boldsymbol{y})}\right] \tag{44}$$

$$= \mathbb{E}_{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log \frac{p(\boldsymbol{x}_T|\boldsymbol{y})p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1,\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_T|\boldsymbol{x}_0,\boldsymbol{y})} + \sum_{t=2}^{T}\log\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0,\boldsymbol{y})}\right] \tag{45}$$

$$= \mathbb{E}_{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1,\boldsymbol{y})\right] + \mathbb{E}_{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log \frac{p(\boldsymbol{x}_T|\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_T|\boldsymbol{x}_0,\boldsymbol{y})}\right] + \sum_{t=2}^{T}\mathbb{E}_{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0,\boldsymbol{y})}\right] \tag{46}$$

$$= \mathbb{E}_{\hat{q}(\boldsymbol{x}_1|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1,\boldsymbol{y})\right] + \mathbb{E}_{\hat{q}(\boldsymbol{x}_T|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log \frac{p(\boldsymbol{x}_T|\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_T|\boldsymbol{x}_0,\boldsymbol{y})}\right] + \sum_{t=2}^{T}\mathbb{E}_{\hat{q}(\boldsymbol{x}_t,\boldsymbol{x}_{t-1}|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0,\boldsymbol{y})}\right] \tag{47}$$

$$= \underbrace{\mathbb{E}_{\hat{q}(\boldsymbol{x}_1|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1,\boldsymbol{y})\right]}_{\text{reconstruction term}} - \underbrace{D_{\mathrm{KL}}(\hat{q}(\boldsymbol{x}_T|\boldsymbol{x}_0,\boldsymbol{y}) \parallel p(\boldsymbol{x}_T|\boldsymbol{y}))}_{\text{prior matching term}} - \sum_{t=2}^{T}\underbrace{\mathbb{E}_{\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_0,\boldsymbol{y})}\left[D_{\mathrm{KL}}(\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0,\boldsymbol{y}) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y}))\right]}_{\text{denoising matching term}} \tag{48}$$

The prior matching term does not contain any trainable parameters. We further simplify the denoising matching term using Equation 29 further conditioned on $\boldsymbol{x}_0$.

$$-\sum_{t=2}^{T}\mathbb{E}_{\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_0,\boldsymbol{y})}\left[D_{\mathrm{KL}}(\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0,\boldsymbol{y}) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y}))\right] \tag{49}$$

$$= -\sum_{t=2}^{T}\mathbb{E}_{\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_0,\boldsymbol{y})}\left[\mathbb{E}_{\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0,\boldsymbol{y})}\left[\log \hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0,\boldsymbol{y}) - \log p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y})\right]\right] \tag{50}$$

$$= -\sum_{t=2}^{T}\mathbb{E}_{\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_0,\boldsymbol{y})}\left[\mathbb{E}_{\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0,\boldsymbol{y})}\left[\log \hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0) + \log\frac{\hat{q}(\boldsymbol{y}|\boldsymbol{x}_{t-1},\boldsymbol{x}_0)}{\hat{q}(\boldsymbol{y}|\boldsymbol{x}_t,\boldsymbol{x}_0)} - \log p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y})\right]\right] \tag{51}$$

$$= -\sum_{t=2}^{T}\mathbb{E}_{\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_0,\boldsymbol{y})}\left[D_{\mathrm{KL}}(\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y}))\right] - \sum_{t=2}^{T}\mathbb{E}_{\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_0,\boldsymbol{y})}\left[\mathbb{E}_{\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0,\boldsymbol{y})}\left[\log\frac{\hat{q}(\boldsymbol{y}|\boldsymbol{x}_{t-1},\boldsymbol{x}_0)}{\hat{q}(\boldsymbol{y}|\boldsymbol{x}_t,\boldsymbol{x}_0)}\right]\right] \tag{52}$$

Using the result of Equation 52, Equation 48 can be rewritten as

$$\mathbb{E}_{\hat{q}(\boldsymbol{x}_1|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1,\boldsymbol{y})\right] - D_{\mathrm{KL}}(\hat{q}(\boldsymbol{x}_T|\boldsymbol{x}_0,\boldsymbol{y}) \parallel p(\boldsymbol{x}_T|\boldsymbol{y})) - \sum_{t=2}^{T}\mathbb{E}_{\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_0,\boldsymbol{y})}\left[D_{\mathrm{KL}}(\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0,\boldsymbol{y}) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y}))\right] \tag{53}$$

$$=\underbrace{\mathbb{E}_{\hat{q}(\boldsymbol{x}_1|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1,\boldsymbol{y})\right]}_{\text{reconstruction term}} - \underbrace{D_{\mathrm{KL}}(\hat{q}(\boldsymbol{x}_T|\boldsymbol{x}_0,\boldsymbol{y}) \parallel p(\boldsymbol{x}_T|\boldsymbol{y}))}_{\text{prior matching term}} - \sum_{t=2}^{T} \underbrace{\mathbb{E}_{\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_0,\boldsymbol{y})}\left[D_{\mathrm{KL}}(\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y}))\right]}_{\text{denoising matching term}} \tag{54}$$
$$- \sum_{t=2}^{T} \underbrace{\mathbb{E}_{\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_0,\boldsymbol{y})}\left[\mathbb{E}_{\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0,\boldsymbol{y})}\left[\log \frac{\hat{q}(\boldsymbol{y}|\boldsymbol{x}_{t-1},\boldsymbol{x}_0)}{\hat{q}(\boldsymbol{y}|\boldsymbol{x}_t,\boldsymbol{x}_0)}\right]\right]}_{\text{label consistency term}}$$

The expression derived from the ELBO for conditional distribution introduces an additional term for label consistency. This minimizes the difference in the likelihood of the labels between consecutive denoising steps. However, since it does not have trainable parameters along with the prior matching term, we will ignore it. Moreover, it is easy to see from Luo [16]that the reconstruction term and the denoising matching term when developed further lead to the diffusion loss of Equation 1, with the additional parametrization of the model with $\boldsymbol{y}$. Note that the expectation is calculated over the same distributions, since $\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0,\boldsymbol{y}) = q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)$, as shown in Equation 32.

B. Proof for theorem II.2

Theorem B.2. *Denoising score matching for the guidance model π_g: $\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})$, as expressed in Equation 5 is equal up to a constant to the following loss:*

$$L_{DLSM}(\boldsymbol{\phi}) = \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x},\boldsymbol{x}_t,\boldsymbol{y}^{1:k+1},\tilde{\boldsymbol{y}}^{1:k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}) + \nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\tilde{\boldsymbol{y}}^{1:k}) - \nabla_{\boldsymbol{x}_t} log\ p_{\alpha}(\boldsymbol{x}_t|\boldsymbol{x})||_2^2\right] \tag{55}$$

For diffusion models, we take the forward transition kernel as $p_\alpha(\boldsymbol{x}_t|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}_t;\sqrt{\alpha_t}\boldsymbol{x},(1-\alpha_t)\mathcal{I})$. We noise the observations $\boldsymbol{y}^{1:k+1}$ with a Gaussian kernel $\mathcal{N}(\tilde{\boldsymbol{y}};\boldsymbol{y},\tau^2 I)$ of variance τ^2 that is small enough such that $p_\tau(\tilde{\boldsymbol{y}}^i) \approx p(\boldsymbol{y}^i)$.

Chao et al. [3] in their insightful work show that the following two losses differ only by a constant.-

$$D_F(p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t)||p_{\alpha,\tau}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t)) = \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}_t} log\ p_{\alpha,\tau}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t)||_2^2\right] \tag{56}$$

$$L_{DLSM}(\boldsymbol{\phi}) = \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x},\boldsymbol{x}_t,\boldsymbol{y}^{k+1},\tilde{\boldsymbol{y}}^{k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t) + \nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\theta}}(\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}_t} log\ p_{\alpha}(\boldsymbol{x}_t|\boldsymbol{x})||_2^2\right] \tag{57}$$

We extend their proof for multiple conditionals below. The Fisher divergence between the guidance model and the true score of the classifier (Equation 5) can be further expanded as:

$$D_F(p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})||p_{\alpha,\tau}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})) \tag{58}$$
$$= \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}) - \nabla_{\boldsymbol{x}_t} log\ p_{\alpha,\tau}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})||_2^2\right] \tag{59}$$
$$= \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})||_2^2\right] + \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\ p_{\alpha,\tau}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})||_2^2\right]$$
$$- \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\langle\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}),\nabla_{\boldsymbol{x}_t} log\ p_{\alpha,\tau}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})\rangle\right] \tag{60}$$
$$= \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})||_2^2\right] + \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\ p_{\alpha,\tau}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})||_2^2\right]$$
$$- \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\langle\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}),\nabla_{\boldsymbol{x}_t} log\ p_{\alpha,\tau}(\boldsymbol{x}_t|\tilde{\boldsymbol{y}}^{1:k},\tilde{\boldsymbol{y}}^{k+1}) - \nabla_{\boldsymbol{x}_t} log\ p_{\alpha,\tau}(\boldsymbol{x}_t|\tilde{\boldsymbol{y}}^{1:k})\rangle\right] \tag{61}$$
$$= \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})||_2^2\right] + \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\ p_{\alpha,\tau}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})||_2^2\right]$$
$$+ \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\langle\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}),\nabla_{\boldsymbol{x}_t} log\ p_{\alpha,\tau}(\boldsymbol{x}_t|\tilde{\boldsymbol{y}}^{1:k})\rangle\right]$$
$$\underbrace{-\mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\langle\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}),\nabla_{\boldsymbol{x}_t} log\ p_{\alpha,\tau}(\boldsymbol{x}_t|\tilde{\boldsymbol{y}}^{1:k},\tilde{\boldsymbol{y}}^{k+1})\rangle\right]}_{\text{Term 1}} \tag{62}$$

Simplifying the Term 1 further:

$$- \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\langle\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}),\nabla_{\boldsymbol{x}_t} log\ p_{\alpha,\tau}(\boldsymbol{x}_t|\tilde{\boldsymbol{y}}^{1:k+1})\rangle\right]$$
$$= -\int_{\boldsymbol{x}_t}\int_{\tilde{\boldsymbol{y}}^{1:k+1}} p_\tau(\tilde{\boldsymbol{y}}^{1:k+1}) p_{\alpha,\tau}(\boldsymbol{x}_t|\tilde{\boldsymbol{y}}^{1:k+1})\langle\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}),\frac{\nabla_{\boldsymbol{x}_t} p_{\alpha,\tau}(\boldsymbol{x}_t|\tilde{\boldsymbol{y}}^{1:k+1})}{p_{\alpha,\tau}(\boldsymbol{x}_t|\tilde{\boldsymbol{y}}^{1:k+1})}\rangle d\tilde{\boldsymbol{y}}^{1:k+1} d\boldsymbol{x}_t$$
$$= -\int_{\boldsymbol{x}_t}\int_{\tilde{\boldsymbol{y}}^{1:k+1}} p_\tau(\tilde{\boldsymbol{y}}^{1:k+1})\langle\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}),\nabla_{\boldsymbol{x}_t}\int_{\boldsymbol{x}_0} p_{0,\tau}(\boldsymbol{x}_0|\tilde{\boldsymbol{y}}^{1:k+1}) p_{\alpha,\tau}(\boldsymbol{x}_t|\boldsymbol{x}_0,\tilde{\boldsymbol{y}}^{1:k+1}) d\boldsymbol{x}_0\rangle d\tilde{\boldsymbol{y}}^{1:k+1} d\boldsymbol{x}_t$$
$$= -\int_{\boldsymbol{x}_t}\int_{\tilde{\boldsymbol{y}}^{1:k+1}} p_\tau(\tilde{\boldsymbol{y}}^{1:k+1})\langle\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}),\nabla_{\boldsymbol{x}_t}\int_{\boldsymbol{x}_0}\int_{\boldsymbol{y}^{1:k+1}} p_{0,\tau}(\boldsymbol{x}_0|\tilde{\boldsymbol{y}}^{1:k+1}) p_{\alpha,\tau}(\boldsymbol{x}_t|\boldsymbol{x}_0,\tilde{\boldsymbol{y}}^{1:k+1},\boldsymbol{y}^{1:k+1}) p_(\boldsymbol{y}^{1:k+1}|\boldsymbol{x}_0,\tilde{\boldsymbol{y}}^{1:k+1}) d\boldsymbol{y}^{1:k+1} d\boldsymbol{x}_0\rangle d\tilde{\boldsymbol{y}}^{1:k+1} d\boldsymbol{x}_t$$

$$= -\int_{\boldsymbol{x}_t}\int_{\tilde{\boldsymbol{y}}^{1:k+1}}\int_{\boldsymbol{x}_0}\int_{\boldsymbol{y}^{1:k+1}} p_\tau(\boldsymbol{x}_0,\boldsymbol{x}_t,\boldsymbol{y}^{1:k+1},\tilde{\boldsymbol{y}}^{1:k+1})\langle\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}),\nabla_{\boldsymbol{x}_t}\log p_{\alpha,\tau}(\boldsymbol{x}_t|\boldsymbol{x}_0,\tilde{\boldsymbol{y}}^{1:k+1},\boldsymbol{y}^{1:k+1})\rangle d\boldsymbol{y}^{1:k+1}d\boldsymbol{x}_0 d\tilde{\boldsymbol{y}}^{1:k+1}d\boldsymbol{x}_t$$
$$= -\mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_0,\boldsymbol{x}_t,\boldsymbol{y}^{1:k+1},\tilde{\boldsymbol{y}}^{1:k+1})}\left[\langle\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}),\nabla_{\boldsymbol{x}_t}\log p_\alpha(\boldsymbol{x}_t|\boldsymbol{x}_0)\rangle\right]$$

Plugging this back into Equation 62, we get-

$$\begin{aligned}
&D_F(p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})||p_{\alpha,\tau}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}))\\
&= \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})||_2^2\right] + \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\ p_{\alpha,\tau}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})||_2^2\right]\\
&+ \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\langle\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}),\nabla_{\boldsymbol{x}_t} log\ p_{\alpha,\tau}(\boldsymbol{x}_t|\tilde{\boldsymbol{y}}^{1:k})\rangle\right]\\
&- \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_0,\boldsymbol{x}_t,\boldsymbol{y}^{1:k+1},\tilde{\boldsymbol{y}}^{1:k+1})}\left[\langle\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}),\nabla_{\boldsymbol{x}_t}\log p_\alpha(\boldsymbol{x}_t|\boldsymbol{x}_0)\rangle\right] \qquad (63)
\end{aligned}$$

Here, $\mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\ p_{\alpha,\tau}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})||_2^2\right]$ is a constant. Further, adding the constant $\mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\ p_{\alpha,\tau}(\boldsymbol{x}_t|\tilde{\boldsymbol{y}}^{1:k}) - \nabla_{\boldsymbol{x}_t} log\ p_\alpha(\boldsymbol{x}_t|\boldsymbol{x}_0)||_2^2\right]$ to Equation 63, we get:

$$\begin{aligned}
&D_F(p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})||p_{\alpha,\tau}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}))\\
&= \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})||_2^2\right] + \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\langle\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}),\nabla_{\boldsymbol{x}_t} log\ p_{\alpha,\tau}(\boldsymbol{x}_t|\tilde{\boldsymbol{y}}^{1:k})\rangle\right] + C\\
&- \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_0,\boldsymbol{x}_t,\boldsymbol{y}^{1:k+1},\tilde{\boldsymbol{y}}^{1:k+1})}\left[\langle\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}),\nabla_{\boldsymbol{x}_t}\log p_\alpha(\boldsymbol{x}_t|\boldsymbol{x}_0)\rangle\right] + \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\ p_{\alpha,\tau}(\boldsymbol{x}_t|\tilde{\boldsymbol{y}}^{1:k}) - \nabla_{\boldsymbol{x}_t} log\ p_\alpha(\boldsymbol{x}_t|\boldsymbol{x}_0)||_2^2\right] \qquad (64)\\
&= \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x},\boldsymbol{x}_t,\boldsymbol{y}^{1:k+1},\tilde{\boldsymbol{y}}^{1:k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}) + \nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\tilde{\boldsymbol{y}}^{1:k}) - \nabla_{\boldsymbol{x}_t} log\ p_\alpha(\boldsymbol{x}_t|\boldsymbol{x})||_2^2\right] + C \qquad (65)
\end{aligned}$$

APPENDIX C
EXPERIMENTAL SETUP

A. Environments

We use 5 tasks from RLBench [14] and all the tasks from the dexterous hand-manipulation environment Adroit [10] to evaluate our performance. RLBench has a wide variety of tasks and an inbuilt planner that enables collection of demonstrations. We train visuomotor policies on RLBench with a 5 camera setup recording 96x96 RGB images that use joint position as the action modality. The tasks considered in this paper are shown in Figure 3. Further we also modify the visual scene of these tasks to evaluate the performance of models under visual distribution shift, as shown in Figure 4.

The Adroit dataset focuses on high-dimensional dexterous manipulation tasks using a 24-degree-of-freedom anthropomorphic hand. The dataset encompasses a suite of challenging manipulation tasks designed to test fine motor control and complex hand-object interactions. The environment state is specified as a low-dimensional vector tailored to each task.

Fig. 3. RLBench Tasks: *OpenBox*; *CloseBox*; *BasketballInHoop*; *OpenDoor*; *CloseDoor*

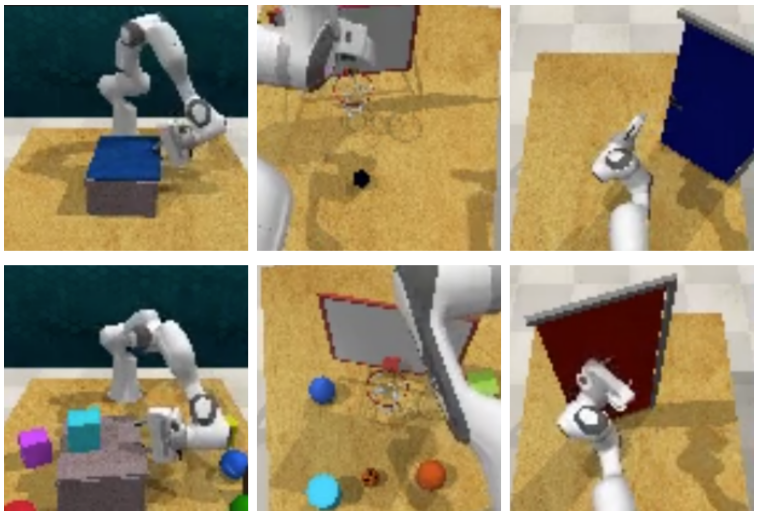

Fig. 4. **Top row from left to right**: *OpenBox* task with a blue lid; *BasketballInHoop* with a blue ball; *OpenDoor* with a blue door. **Bottom row frome left to right**: *OpenBox* task with distractors; *BasketballInHoop* with distractors; *OpenDoor* with a red door.

B. Baselines

POCO [25] can be formulated for composing a motion and a vision guidance model as follows. We train the motion and the vision guidance models separately prior to the compositional sampling proposed by Du et al. [8]. The motion model is only trained on proprioceptive observations, while the vision guidance model also uses the vision observations. We set $\lambda = 0.2$ in Equation 66.

$$\begin{aligned} \boldsymbol{x}_{t-1} &\sim \mathcal{N}\left(\boldsymbol{x}_t; \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}(\boldsymbol{x}_t, \boldsymbol{y}^r, \boldsymbol{y}^c, t)\right), \sqrt{1-\alpha_t}\mathcal{I}\right) \\ \boldsymbol{\epsilon}(\boldsymbol{x}_t, \boldsymbol{y}^r, \boldsymbol{y}^c, t) &= \hat{\boldsymbol{\epsilon}}_{\phi}(\boldsymbol{x}_t, \boldsymbol{y}^r, \boldsymbol{y}^c, t) + \lambda * \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, \boldsymbol{y}^r, t) \end{aligned} \tag{66}$$

For Classifier-free guidance [12], we train a single model and switch out the vision modality with a probability of 0.1. We then sample according to the Equation 67, where we set $\lambda_1 = 1.1$ and $\lambda_2 = 0.1$.

$$\begin{aligned} \boldsymbol{x}_{t-1} &\sim \mathcal{N}\left(\boldsymbol{x}_t; \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}(\boldsymbol{x}_t, \boldsymbol{y}^r, \boldsymbol{y}^c, t)\right), \sqrt{1-\alpha_t}\mathcal{I}\right) \\ \boldsymbol{\epsilon}(\boldsymbol{x}_t, \boldsymbol{y}^r, \boldsymbol{y}^c, t) &= \lambda_1 * \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, \boldsymbol{y}^r, \boldsymbol{y}^c, t) + \lambda_2 * \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, \boldsymbol{y}^r, \boldsymbol{\phi}, t) \end{aligned} \tag{67}$$

C. Training and Rollout

All models are trained for 2000 epochs for RLBench tasks and 3000 epochs for Adroit. We use Adam optimizer [15] and a learning rate of $1e-4$, in accordance to Peebles and Xie [17]. All our results are reported as the mean and standard deviation of the success rates over 100 rollouts for 3 different seeds (total of 300 rollouts) for 10, 50 and 100 demonstrations for each task. We rollout models for 300 and 475 timesteps for RLBench and Adroit tasks, respectively. We report results for the number of rollouts with cumulative success signals greater than 1 and 25 for all Adroit tasks.

APPENDIX D
RESULTS: ADROIT

We show the results in the Adroit environments in Table III. Note that we modify the *Hammer* environment to consider the distance between the nail and the board as greater than 0.2 to be successful, instead of 0.1. Here **DiT CMoD** refers to an architectural ablation, which resembles ControlNet closely. Here, the noisy actions are passed into the transformer and then conditioned upon the environment state using adaLn.

Task	Criteria	DiT	UNet	DiT-cfg	DiT CMod	CoDiG
Door Human	Success achieved > 1	58.3 ± 6.7	67.3 ± 4.7	42.7 ± 5.5	60.3 ± 6.7	**74.3** ± 3.1
	Success achieved > 25	34.3 ± 2.5	30.7 ± 0.6	22.3 ± 7.5	17.7 ± 3.1	**45.7** ± 5.8
Pen Human	Success achieved > 1	58.3 ± 1.5	63.3 ± 3.5	**67.7** ± 3.5	62.3 ± 5.9	62.3 ± 4.7
	Success achieved > 25	53.3 ± 1.2	57.3 ± 2.5	**62.3** ± 3.2	56.0 ± 4.4	56.0 ± 3.5
Hammer Human	Success achieved > 1	**54.0** ± 1.7	47.3 ± 6.0	40.7 ± 2.3	**56.7** ± 4.6	51.7 ± 6.4
	Success achieved > 25	**52.7** ± 2.3	47.0 ± 6.0	39.0 ± 2.0	**56.0** ± 5.2	**50.7** ± 6.4
Relocate Human	Success achieved > 1	**63.0** ± 4.6	56.0 ± 7.2	1.3 ± 0.6	43.3 ± 4.0	63.0 ± 6.9
	Success achieved > 25	**60.0** ± 3.6	52.0 ± 7.0	0.3 ± 0.6	38.7 ± 4.6	**58.7** ± 4.0

TABLE III

PERFORMANCE COMPARISON ACROSS ADROIT TASKS FOR DIFFERENT MODELS.