# LEARNING COMPOSABLE DIFFUSION GUIDANCE FOR MOTION PRIORS

**Omkar Patil**
Arizona State University
opatil3@asu.edu

**Eric Rosen**
Robotics and AI Institute
eric_rosen@brown.edu

**Nakul Gopalan**
Arizona State University
ng@asu.edu

## 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 method to train diffusion models on skill demonstrations borrow from the literature on image generation, and results in a conditional distribution of robot actions given the visual, proprioceptive and other observations. However, they have 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. CoDiG decouples the observation modalities allowing the residual learning of one modality with respect to the others. While presenting a more intuitive modeling paradigm, CoDiG also enables the scaling of modalities such as robot motions independently. Our preliminary results show that visual CoDiG with motion-priors outperforms the conventional way of learning visuomotor policies using diffusion models on skills with relatively low-diversity of robot motion. Further experimentation is needed to evaluate the performance and robustness of CoDiG for different observation modalities, and on different classes of skills, such as long-horizon and precise manipulation.

## 1 INTRODUCTION

Diffusion models have emerged as a promising choice for learning robot skills from demonstrations Chi et al. (2023). 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. Prior work Chi et al. (2023) 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 visuo-motor 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.

Recent efforts to exploit compositionality through diffusion models have yielded promising results Du et al. (2023); Wang et al. (2024). For instance, POCO Wang et al. (2024) composes multi-task conditional and unconditional policies to show better generalization over using a single multi-task policy for tool use. Wang et al. (2024) emphasize the heterogeneous nature of data collection in robotics and showcase composition of policies trained on different data modalities such as vision-action and tactile-action. However, the candidate policies are learned independently and require manual tuning of their compositional weights. Moreover, these methods are limited to the combination of pre-learned policies, requiring a careful consideration of the candidate tasks and data collection.

Critically, CoDiG enables the training of models conditioned on one observational modality with respect to the rest so that data collection efforts can be concentrated on the cheaper modality and consequent models learn on the residual rather than in an isolated manner. Consequently, CoDiG does not require manual tuning of the compositional weights. Specifically, we showcase this by training a 'motion-model' on the action distribution of a task, and learning a vision 'guidance-model' relative to it on vision-action coupled demonstration data. Through our experiments, we show that not all tasks are created equal, and policy learning using our method can benefit from further collection of only the robot motion data corresponding to the task, while reducing the requirement of collecting coupled vision-action data. Ultimately, CoDiG is a general policy learning method that can be flexibly used with different modalities of data

and shows better scaling properties for decoupled task-specific data collection. Further experiments are needed to validate the efficacy of CoDiG, especially for long-horizon tasks and for its scaling properties. Our contributions are as follows-

- We present a novel method to train diffusion models on robot demonstration data called CoDiG, abbreviated for 'Composable Diffusion Guidance'. Conventional diffusion policies Chi et al. (2023) learn a conditional distribution of actions with respect to the all observational modalities. In contrast, CoDiG decouples the observational modalities allowing the residual learning of one modality with respect to the others.
- CoDiG allows data collection to focus on the cheapest modality, removing the necessity of collecting paired modality data together. Our experiments show that not all skills are created equal, and some skills are less reliant on visual observations for their motions. We use this insight to show that CoDiG benefits from motion-only data collection for such skills, while limiting the collection of paired visual-action data.
- Our results show that visual CoDiG with a motion-model performs favorably with respect to the conventional diffusion models on several RLBench tasks. Further experimentation is required to validate the efficacy and robustness of CoDiG for long horizon tasks and precise manipulation. Our initial results also show that CoDiG scales better with additional motion data.

## 2 BACKGROUND

### 2.1 DIFFUSION MODELS

Gaussian diffusion models Sohl-Dickstein et al. (2015) 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 $\alpha_t$ 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 1 Ho et al. (2020).

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

Here, $\lambda_t$, a function of $\alpha_t$ 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 Efron (2011) 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 $q(\boldsymbol{x}_t) = \int q(\boldsymbol{x}_t|\boldsymbol{x}_0)q(\boldsymbol{x}_0)d\boldsymbol{x}_0$ Luo (2022).

$$\frac{-1}{\sqrt{1-\bar{\alpha}_t}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) \approx \frac{-1}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_0 = \nabla_{\boldsymbol{x}} log\ q(\boldsymbol{x}_t) \tag{2}$$

### 2.2 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) Vincent (2011) used to train EBMs Song and Kingma (2021) 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 $\sigma_t$.

$$\mathcal{J}_{\sigma_t}(\boldsymbol{\theta}) = \mathbb{E}_{q(\boldsymbol{x},\tilde{\boldsymbol{x}})}\left[\frac{1}{2}||\nabla_{\tilde{\boldsymbol{x}}} log\ q(\tilde{\boldsymbol{x}}|\boldsymbol{x}) - \nabla_{\tilde{\boldsymbol{x}}} log\ p_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}})||_2^2\right] \tag{3}$$

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 3 simplifies to the following when $q(\boldsymbol{x}|\tilde{\boldsymbol{x}}) = \mathcal{N}(\tilde{\boldsymbol{x}}; \boldsymbol{x}, \sigma_t^2 I)$ Vincent (2011):

$$\mathcal{J}_{\sigma_t}(\boldsymbol{\theta}) = \mathbb{E}_{q_{(\boldsymbol{x})}\mathcal{N}(\boldsymbol{\epsilon};0,\mathcal{I})}\left[||\frac{\boldsymbol{\epsilon}}{\sigma_t} + \nabla_{\boldsymbol{x}} f_{\boldsymbol{\theta}}(\boldsymbol{x} + \sigma_t\boldsymbol{\epsilon})||_2^2\right] \tag{4}$$

Once trained, MCMC methods such as Langevin Andrieu et al. (2003) 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 Song and Ermon (2020).

### 2.3 Sampling from the Policy

Song et al. (2020) 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 (1982) $d\boldsymbol{x} = \left[\boldsymbol{f}(\boldsymbol{x}, t) - g(t)^2 \nabla_{\boldsymbol{x}} \log q_t(\boldsymbol{x})\right] dt + g(t) d\bar{\boldsymbol{w}}$, 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 Song et al. (2020) proposed by Ho et al. (2020), shown in Equation 5, and Langevin dynamics Andrieu et al. (2003).

$$\boldsymbol{x}_{t-1} \sim \mathcal{N}\left(\boldsymbol{x}_t; \frac{1}{\sqrt{\alpha_t}}\left[\boldsymbol{x}_t + (1-\alpha_t)\nabla_{\boldsymbol{x}_t} \log q(\boldsymbol{x}_t)\right], \sqrt{1-\alpha_t}\mathcal{I}\right) \tag{5}$$

Equation 3 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 1, while score-based model typically use $\mathcal{N}(\boldsymbol{x}_t; \boldsymbol{x}_{t-1}, \sigma_t^2 I)$, where $\alpha_t$ and $\sigma_t$ are respective noise scales.

### 2.4 Classifier Guided Diffusion

Classifier guided diffusion Dhariwal and Nichol (2021a) 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 6 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.

$$\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{y}; \boldsymbol{\theta}, \boldsymbol{\phi}) = \nabla_{x_t} \log p(\boldsymbol{y}|\boldsymbol{x}_t; \boldsymbol{\phi}) + \nabla_{x_t} \log p(\boldsymbol{x}_t; \boldsymbol{\theta}) \tag{6}$$

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 re-weighted version of Equation 6 for $\nabla_{\boldsymbol{x}_t} \log q(\boldsymbol{x}_t)$ in the ancestral sampling Equation 5.

## 3 Methodology

### 3.1 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 Ho et al. (2020); Luo (2022); Song et al. (2022). This formulation has been directly adopted by the robotics community Chi et al. (2023) leading to the popular optimization objective shown in Equation 7.

$$\mathcal{L}_t(\boldsymbol{\theta}) = \mathbb{E}_{q(\boldsymbol{x}_0, \boldsymbol{y})\mathcal{N}(\boldsymbol{\epsilon}_0; 0, \mathcal{I})}\left[[||\boldsymbol{\epsilon}_0 - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, \boldsymbol{y}, t)||_2^2]\right] \tag{7}$$

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

Here, the network $\epsilon_\theta$ 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 8 can be used to sample actions given the observations $\boldsymbol{y}$. However, while prior work assumes that $\epsilon_\theta$ 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 Dhariwal and Nichol (2021a) does indeed result in the loss of Equation 7 Chi et al. (2023) 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 3.1.** *The diffusion loss function $\mathcal{L}_t(\boldsymbol{\theta})$ as defined in Equation 7, 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 3.1 can be found in the Appendix A.1. 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 6 at diffusion time-step $t$ as follows:

$$\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{y}^{1:k}, \boldsymbol{y}^{k+1}; \boldsymbol{\theta}, \boldsymbol{\phi}) = \nabla_{x_t} \log p(\boldsymbol{y}^{k+1}|\boldsymbol{x}_t, \boldsymbol{y}^{1:k}; \boldsymbol{\phi}) + \nabla_{x_t} \log p(\boldsymbol{x}_t|\boldsymbol{y}^{1:k}; \boldsymbol{\theta}) \tag{9}$$

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 Dhariwal and Nichol (2021a) 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 $\pi_g$, rather than to learn a classifier and then obtain its gradients. To learn the guidance model, following the suit of Equation 3, we minimize the Fisher divergence between the guidance model and the true score of the classifier.

$$D_F(p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t, \tilde{\boldsymbol{y}}^{1:k})||p_{\alpha,\tau}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t, \tilde{\boldsymbol{y}}^{1:k})) = \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t, \tilde{\boldsymbol{y}}^{1:k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\, p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t, \tilde{\boldsymbol{y}}^{1:k}) - \nabla_{\boldsymbol{x}_t} log\, p_{\alpha,\tau}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t, \tilde{\boldsymbol{y}}^{1:k})||_2^2\right] \tag{10}$$

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 $\tau^2$ 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 10 is difficult to use in it's current form as estimation of the true score is difficult for large datasets. Chao et al. (2022) derive the denoising likelihood score matching (DLSM) objective from the denoising score matching of Equation 3 for conditional distributions, which forms the basis of our next result.

**Theorem 3.2.** *Denoising score matching for the guidance model $\pi_g$: $\nabla_{\boldsymbol{x}_t} log\, p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t, \tilde{\boldsymbol{y}}^{1:k})$, as expressed in Equation 10 is equal up to a constant to the following loss:*

$$L^t_{DLSM}(\boldsymbol{\phi}) = \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}, \boldsymbol{x}_t, \boldsymbol{y}^{1:k+1}, \tilde{\boldsymbol{y}}^{1:k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\, p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t, \tilde{\boldsymbol{y}}^{1:k}) + \nabla_{\boldsymbol{x}_t} log\, p_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\tilde{\boldsymbol{y}}^{1:k}) - \nabla_{\boldsymbol{x}_t} log\, p_\alpha(\boldsymbol{x}_t|\boldsymbol{x})||_2^2\right] \tag{11}$$

*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 $\tau^2$ that is small enough such that $p_\tau(\tilde{\boldsymbol{y}}^i) \approx p(\boldsymbol{y}^i)$.*

The proof for Theorem 3.2 is a conditional variant of the one derived by Chao et al. (2022), and is presented in Appendix A.2. Equation 53 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 $\epsilon_0 \sim \mathcal{N}(0, \mathcal{I})$, and using Equation 2, we obtain:

$$L_{DLSM}(\boldsymbol{\phi}) = \mathbb{E}_{p_\tau(\boldsymbol{x}, \boldsymbol{y}^{1:k+1}, \tilde{\boldsymbol{y}}^{1:k+1})}\mathbb{E}_{\epsilon_0 \sim \mathcal{N}(0,\mathcal{I})}\left[\frac{1}{2}||\hat{\epsilon}_\phi(\tilde{\boldsymbol{y}}^{k+1}, \boldsymbol{x}_t, \tilde{\boldsymbol{y}}^{1:k}) + \epsilon_\theta(\boldsymbol{x}_t, \tilde{\boldsymbol{y}}^{1:k}) - \epsilon_0||_2^2\right] \tag{12}$$

Here, $\epsilon_\theta$ is frozen and known from prior training with observational modalities $\boldsymbol{y}^{1:k}$, while $\epsilon_\phi$ is learned in expectation over the new data collected in conjugation with modality $\boldsymbol{y}^{k+1}$. Equation 12 is very similar to the diffusion loss Equation 7, except the model is expected to learn a residual of the noise added to the action sample $x_t$ taken as the input. Once, $\epsilon_\phi$ is learned, actions can be sampled from the conditional distribution $p(\boldsymbol{x}|\boldsymbol{y}^{1:k+1})$ using ancestral sampling and Equation 9:

$$\boldsymbol{x}_{t-1} \sim \mathcal{N}\left(\boldsymbol{x}_t; \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}(\boldsymbol{x}_t, \boldsymbol{y}^{1:k+1}, t)\right), \sqrt{1-\alpha_t}\mathcal{I}\right) \tag{13}$$

$$\boldsymbol{\epsilon}(\boldsymbol{x}_t, \boldsymbol{y}^{1:k+1}, t) = \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\phi}}(\boldsymbol{x}_t, \boldsymbol{y}^{1:k+1}, t) + \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, \boldsymbol{y}^{1:k}, t) \tag{14}$$

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 $\boldsymbol{y}^{1:k}$.

### 3.2 Visual CoDiG with Motion-model

We now provide a concrete implementation of the method proposed in Section 3.1. 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' $p(\boldsymbol{y}^c|\boldsymbol{x}, \boldsymbol{y}^r)$. This implies that for certain tasks, we can collect a larger set of pure motions of the robot performing the specific skill, and fewer visual demonstrations of the robot actually performing the task. We learn the motion-model using diffusion loss of Equation 7 and the guidance model using DLSM loss of Equation 12.

$$\mathcal{L}_{MM}(\boldsymbol{\theta}) = \mathbb{E}_{q(\boldsymbol{x}_0, \boldsymbol{y}^r)\mathcal{N}(\boldsymbol{\epsilon}_0; 0, \mathcal{I})}\left[[||\boldsymbol{\epsilon}_0 - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, \boldsymbol{y}^r, t)||_2^2]\right] \tag{15}$$

$$L_G(\boldsymbol{\phi}) = \mathbb{E}_{p_\tau(\boldsymbol{x},\boldsymbol{y}^r,\boldsymbol{y}^c,\tilde{\boldsymbol{y}}^r,\tilde{\boldsymbol{y}}^c)}\mathbb{E}_{\epsilon_0\sim\mathcal{N}(0,\mathcal{I})}\left[\frac{1}{2}||\epsilon_0 - \hat{\epsilon}_\phi(\tilde{\boldsymbol{y}}^c, \boldsymbol{x}_t, \tilde{\boldsymbol{y}}^r, t) - \epsilon_\theta(\boldsymbol{x}_t, \tilde{\boldsymbol{y}}^r, t)||_2^2\right] \tag{16}$$

In Section 4, we showcase that this approach has a distinct advantage for skills that are heavily dependent on proprioception, while performing comparably on others. 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. Moreover, CoDiG allows us to tailor the collection of data to suit the task and costs.

# 4 RESULTS

## 4.1 ENVIRONMENTS AND BASELINES

We use 9 randomly-picked tasks from RLBench James et al. (2019) and the low-dimensional Franka-Kitchen environment Gupta et al. (2019) 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. Franka-Kitchen uses joint velocity as the action modality with a 30-dimensional observational state that includes the position of the robot joints and the objects in the kitchen, along with the goal states encoding the target objects to manipulate. There are three versions of the dataset-complete, partial and mixed. The complete version is a smaller part of the dataset where all the demonstrations correspond to a fixed set of objects (microwave, kettle, light and slider) that are manipulated in the same order.

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 ($\sim$90M) Peebles and Xie (2023) as our model architecture, which is kept the same for both the baseline and CoDiG. Since CoDiG uses two separate models for motion and guidance, we also include DiT-base ($\sim$190M) in the baseline. All models are trained for 2000 epochs for RLBench tasks and 3000 epochs for Franka-Kitchen. We use Adam optimizer Kingma and Ba (2017) and a learning rate of $1e-4$, in accordance to Peebles and Xie (2023).

In this work, we present two types of results. In the first, we train both the baseline and our method using only 10 visual demonstrations from RLBench to compare how CoDiG fares relative to the existing way of learning visuomotor policies. For Franka-Kitchen we train all models, including the motion-model only on the 'complete' version of the dataset. In the second, we collect an additional 'motions' of the robot performing the skill and record the evolution of the action and the proprioceptive states. In simulation, the motions are collected using the same planner used to collect visual demonstration in the environment. Our objective for the second set of results is to understand how collection of additional motion data can benefit CoDiG in comparison to directly learning the conditional distribution. In addition to these, we also show results for the motion-model that does not use any visual information. All our results for RLBench are reported as the mean of the success rates over 50 rollouts for 3 different seeds (total of 150 rollouts). For Franka-Kitchen we report the mean task-completion rate out of 4 tasks. We leave real-world experiments and evaluation on play-motion data for future work.

## 4.2 LEARNING VISUOMOTOR POLICIES

For RLBench, we train the DiT-small and DiT-base baselines on 10 visual demonstrations using the loss defined in Equation 7. For CoDiG, we train the motion model $\pi_{mm}$ on only the proprioception-action data using the loss defined in Equation 15 and the vision guidance-model relative to the motion model using Equation 16. We also report the mean success rate for the motion-model $\pi_{mm}$. The results for RLBench and Franka-Kitchen (complete) are shown in Table 1.

CoDiG performs comparably with the baselines, and shows strong improvements when the motion of the robot is heavily dependent on the robot proprioception. An indicator of this is the performance of the motion model (MM), since it is only trained on the proprioception of the robot and does not use any visual information. CoDiG outperforms the baselines on the tasks of *OpenBox* and *CloseBox* by large margins, which can be corroborated by the better performance of the motion-model on them, as compared to the other tasks.

## 4.3 PERFORMANCE WITH ADDITIONAL MOTION DEMONSTRATIONS

To understand how the performance of the model scales with independent modalities, we collect 40 and 90 additional motions of the robot performing the task in simulation. The implication on training CoDiG is that the guidance-models learn (with only 10 visual demonstrations) relative to the motion-models trained on different number of motion demonstrations (40 and 90). The results are shown in Table 2.

Although pre-training DiT with additional motion data does help improve the performance for tasks such as *OpenBox* and *CloseBox*, it still falls short of CoDiG even without any additional motion demonstrations. We see performance gains for CoDiG with increasing the available task-specific robot motions, for skills that are not heavily dependent on

| **Tasks** | **DiT** | **DiT x2** | **CoDiG** | **MM** |
|---|---|---|---|---|
| Demonstrations | 10 | 10 | 10 | (50) |
| Open Box | 18 | 20 | **39.33** | 17.33 |
| Open Door | 24 | **27.33** | 20.67 | 7.33 |
| Open Microwave | 2 | 1.33 | 1.33 | 0 |
| Open Fridge | 0.66 | 1.33 | 2 | 0 |
| Close Box | 29.33 | 18 | **52** | 33.33 |
| Close Door | 2 | 1.33 | 0.67 | 0.67 |
| Close Microwave | **49.33** | **49.33** | 36 | 24 |
| Close Fridge | 53.33 | 52 | 54 | 46 |
| Basketball in Hoop | 0.66 | 2 | **10** | 2 |
| Kitchen (complete) | 1.2 | 1.12 | 1.22 | 1.06 |

Table 1: Performance comparison of different models across various tasks on RLBench and Franka-Kitchen (complete). All the models on RLBench are trained using 10 visual demonstrations, except the motion-model which is trained on 50 motions of the robot performing the task. DiT x2 denotes a DiT-base model that is twice the size of the smaller variant, and comparable to CoDiG.

| **Task** | **CoDiG** | **CoDiG** | **CoDiG** | **MM** | **DiT x2 Pt.** |
|---|---|---|---|---|---|
| Visual(+Motion) Demos | 10(10) | 10(50) | 10(100) | 0(50) | 10(50) |
| Open Box | 39.33 | 31.33 | **44.6** | 17.33 | 22 |
| Open Door | 20.67 | 18 | 18.67 | 7.33 | 3.33 |
| Open Microwave | 1.33 | 0.67 | 2 | 0 | 6 |
| Open Fridge | 2 | 0 | 0 | 0 | 2 |
| Close Box | 52 | 50.67 | **56.67** | 33.33 | 38.67 |
| Close Door | 0.67 | 2 | 2.67 | 0.67 | 2.67 |
| Close Microwave | 36 | 36.67 | 33.34 | 24 | 38.67 |
| Close Fridge | 54 | **58.67** | 44 | 46 | 54 |
| Basketball in Hoop | 10 | 9.33 | 9.33 | 2 | 3.33 |

Table 2: Performance comparison on scaling the collection of robot motions specific to the task. All the models except MM use only 10 visual demonstrations, and further robot motions as indicated in brackets. DiT x2 Pt. refers to DiT-base that has been pre-trained using the motion data and then fine-tuned on the visual demonstrations.

visual observations. However, the magnitude of improvement is small, indicating that there is scope to improve the interaction between the motion and guidance models.

## 5 Conclusion

We present CoDiG, a novel compositional method to train diffusion models by decoupling different observational modalities such as proprioception, vision and tactile. CoDiG yields a loss function that is simple to use, while also presenting a more intuitive modeling paradigm. Our experiments show that not all tasks are equal, and some tasks depend heavily on certain observational modalities than others. CoDiG allows us to tailor the data collection for tasks based on their dependence on observational modalities and costs. We find that visual CoDiG with a motion-model performs strongly over its baselines for visuomotor tasks that rely more on the proprioception. More experiments are needed to validate the efficacy, robustness and scaling properties of CoDiG.

## REFERENCES


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

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

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.

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.

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

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

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

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.

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

Abhishek Gupta, Vikash Kumar, Corey Lynch, Sergey Levine, and Karol Hausman. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning, 2019. URL https://arxiv.org/abs/1910.11956.

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

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.

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

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

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

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

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

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

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

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.

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

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.

# A METHODOLOGY

## A.1 PROOF FOR THEOREM 3.1

**Theorem A.1.** *The diffusion loss function $\mathcal{L}_t(\boldsymbol{\theta})$ as defined in Equation 7, 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 (2016). Following Dhariwal and Nichol (2021b), 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:

$$\hat{q}(\boldsymbol{x}_0) := q(\boldsymbol{x}_0) \tag{17}$$

$$\hat{q}(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{y}) := q(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t) \tag{18}$$

$$\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0, \boldsymbol{y}) := \prod_{t=1}^{T} \hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{y}) \tag{19}$$

We now reproduce some results that will be used later in the derivation of diffusion loss for conditional distributions. Dhariwal and Nichol (2021b) also show that

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

$$= \hat{q}(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t) \frac{\hat{q}(\boldsymbol{y}|\boldsymbol{x}_t)}{\hat{q}(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t)} \tag{21}$$

$$= \hat{q}(\boldsymbol{y}|\boldsymbol{x}_t) \tag{22}$$

Moreover, the unconditional reverse transition kernels can be shown to be equal using Bayes theorem, given Equations 17 and 18: $\hat{\boldsymbol{q}}(\boldsymbol{x}_t|\boldsymbol{x}_{t+1}) = \boldsymbol{q}(\boldsymbol{x}_t|\boldsymbol{x}_{t+1})$. Dhariwal and Nichol (2021b) use the result from Equation 22 to show the following for conditional reverse transition kernels.

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

$$= \frac{\hat{q}(\boldsymbol{x}_t, \boldsymbol{x}_{t+1}, \boldsymbol{y})}{\hat{q}(\boldsymbol{y}|\boldsymbol{x}_{t+1})\hat{q}(\boldsymbol{x}_{t+1})} \tag{24}$$

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

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

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

Further, we can show the following using Equations 18 and 19 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 $q$.

$$\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0, \boldsymbol{y}) = \prod_{t=1}^{T} \hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{y}) \tag{28}$$

$$= \prod_{t=1}^{T} q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) \tag{29}$$

$$= q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0) \tag{30}$$

We adapt the derivation of diffusion loss from Luo (2022) 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).

$$\log p(\boldsymbol{x}|\boldsymbol{y}) = \log \int p(\boldsymbol{x}_{0:T}|\boldsymbol{y}) d\boldsymbol{x}_{1:T} \tag{31}$$

$$= \log \int \frac{p(\boldsymbol{x}_{0:T}|\boldsymbol{y})\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0, \boldsymbol{y})}{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0, \boldsymbol{y})} d\boldsymbol{x}_{1:T} \tag{32}$$

$$= \log \mathbb{E}_{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0, \boldsymbol{y})} \left[ \frac{p(\boldsymbol{x}_{0:T}|\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0, \boldsymbol{y})} \right] \tag{33}$$

$$\geq \mathbb{E}_{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log \frac{p(\boldsymbol{x}_{0:T}|\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0,\boldsymbol{y})}\right] \tag{34}$$

The ELBO can be further simplified as follows

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

$$= \mathbb{E}_{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log \frac{p(\boldsymbol{x}_T|\boldsymbol{y})\prod_{t=1}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y})}{\prod_{t=1}^{T}\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1},\boldsymbol{y})}\right] \tag{36}$$

$$= \mathbb{E}_{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log \frac{p(\boldsymbol{x}_T|\boldsymbol{y})p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1,\boldsymbol{y})\prod_{t=2}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_1|\boldsymbol{x}_0,\boldsymbol{y})\prod_{t=2}^{T}\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1},\boldsymbol{y})}\right] \tag{37}$$

$$= \mathbb{E}_{\hat{q}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log \frac{p(\boldsymbol{x}_T|\boldsymbol{y})p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1,\boldsymbol{y})\prod_{t=2}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y})}{\hat{q}(\boldsymbol{x}_1|\boldsymbol{x}_0,\boldsymbol{y})\prod_{t=2}^{T}\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1},\boldsymbol{x}_0,\boldsymbol{y})}\right] \tag{38}$$

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

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

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

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

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

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

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

$$= \underbrace{\mathbb{E}_{\hat{q}(\boldsymbol{x}_{1}|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1,\boldsymbol{y})\right]}_{\text{reconstruction term}} - \underbrace{D_{\mathrm{KL}}(\hat{q}(\boldsymbol{x}_T|\boldsymbol{x}_0,\boldsymbol{y}) \parallel p(\boldsymbol{x}_T|\boldsymbol{y}))}_{\text{prior matching term}} - \sum_{t=2}^{T}\underbrace{\mathbb{E}_{\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_0,\boldsymbol{y})}\left[D_{\mathrm{KL}}(\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0,\boldsymbol{y}) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y}))\right]}_{\text{denoising matching term}} \tag{46}$$

The prior matching term does not contain any trainable parameters. We further simplify the denoising matching term using Equation 27 further conditioned on $\boldsymbol{x}_0$.

$$-\sum_{t=2}^{T}\mathbb{E}_{\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_0,\boldsymbol{y})}\left[D_{\mathrm{KL}}(\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0,\boldsymbol{y}) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y}))\right] \tag{47}$$

$$= -\sum_{t=2}^{T}\mathbb{E}_{\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_0,\boldsymbol{y})}\left[\mathbb{E}_{\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0,\boldsymbol{y})}\left[\log\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0,\boldsymbol{y}) - \log p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y})\right]\right] \tag{48}$$

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

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

Using the result of Equation 50, Equation 46 can be rewritten as

$$\mathbb{E}_{\hat{q}(\boldsymbol{x}_1|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1,\boldsymbol{y})\right] - D_{\mathrm{KL}}(\hat{q}(\boldsymbol{x}_T|\boldsymbol{x}_0,\boldsymbol{y}) \parallel p(\boldsymbol{x}_T|\boldsymbol{y})) - \sum_{t=2}^{T}\mathbb{E}_{\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_0,\boldsymbol{y})}\left[D_{\mathrm{KL}}(\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0,\boldsymbol{y}) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y}))\right] \tag{51}$$

$$= \underbrace{\mathbb{E}_{\hat{q}(\boldsymbol{x}_1|\boldsymbol{x}_0,\boldsymbol{y})}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1,\boldsymbol{y})\right]}_{\text{reconstruction term}} - \underbrace{D_{\mathrm{KL}}(\hat{q}(\boldsymbol{x}_T|\boldsymbol{x}_0,\boldsymbol{y}) \parallel p(\boldsymbol{x}_T|\boldsymbol{y}))}_{\text{prior matching term}} - \sum_{t=2}^{T}\underbrace{\mathbb{E}_{\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_0,\boldsymbol{y})}\left[D_{\mathrm{KL}}(\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y}))\right]}_{\text{denoising matching term}} \tag{52}$$

$$- \sum_{t=2}^{T}\underbrace{\mathbb{E}_{\hat{q}(\boldsymbol{x}_t|\boldsymbol{x}_0,\boldsymbol{y})}\left[\mathbb{E}_{\hat{q}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0,\boldsymbol{y})}\left[\log\frac{\hat{q}(\boldsymbol{y}|\boldsymbol{x}_{t-1},\boldsymbol{x}_0)}{\hat{q}(\boldsymbol{y}|\boldsymbol{x}_t,\boldsymbol{x}_0)}\right]\right]}_{\text{label consistency term}}$$

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 (2022)that the reconstruction term and the denoising matching term when developed further lead to the diffusion loss of Equation 7, 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 30.

### A.2 Proof for theorem 3.2

**Theorem A.2.** *Denoising score matching for the guidance model $\pi_g$: $\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})$, as expressed in Equation 10 is equal up to a constant to the following loss:*

$$L_{DLSM}(\boldsymbol{\phi}) = \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x},\boldsymbol{x}_t,\boldsymbol{y}^{1:k+1},\tilde{\boldsymbol{y}}^{1:k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}) + \nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\tilde{\boldsymbol{y}}^{1:k}) - \nabla_{\boldsymbol{x}_t} log\ p_{\alpha}(\boldsymbol{x}_t|\boldsymbol{x})||_2^2\right] \tag{53}$$

*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 $\tau^2$ that is small enough such that $p_\tau(\tilde{\boldsymbol{y}}^i) \approx p(\boldsymbol{y}^i)$.*

Chao et al. (2022) in their insightful work show that the following two losses differ only by a constant.-

$$D_F(p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t)||p_{\alpha,\tau}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t)) = \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\ p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}_t} log\ p_{\alpha,\tau}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t)||_2^2\right] \tag{54}$$

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

We extend their proof for multiple conditionals below. The Fisher divergence between the guidance model and the true score of the classifier (Equation 10) can be further expanded as:

$$D_F(p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})||p_{\alpha,\tau}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})) \tag{56}$$

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

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

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

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

Simplifying the Term 1 further:

$$
\begin{aligned}
&- \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]
\end{aligned}
$$

Plugging this back into Equation 60, we get-

$$
\begin{aligned}
&D_F(p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})||p_{\alpha,\tau}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})) \\
&= \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\, p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})||_2^2\right] + \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\, p_{\alpha,\tau}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})||_2^2\right] \\
&+ \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\langle \nabla_{\boldsymbol{x}_t} log\, p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}), \nabla_{\boldsymbol{x}_t} log\, p_{\alpha,\tau}(\boldsymbol{x}_t|\tilde{\boldsymbol{y}}^{1:k})\rangle\right] \\
&- \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_0,\boldsymbol{x}_t,\boldsymbol{y}^{1:k+1},\tilde{\boldsymbol{y}}^{1:k+1})}\left[\langle \nabla_{\boldsymbol{x}_t} log\, p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}), \nabla_{\boldsymbol{x}_t} \log p_\alpha(\boldsymbol{x}_t|\boldsymbol{x}_0)\rangle\right]
\end{aligned} \tag{61}
$$

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 61, we get:

$$
\begin{aligned}
&D_F(p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})||p_{\alpha,\tau}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})) \\
&= \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\, p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})||_2^2\right] + \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k+1})}\left[\langle \nabla_{\boldsymbol{x}_t} log\, p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}), \nabla_{\boldsymbol{x}_t} log\, p_{\alpha,\tau}(\boldsymbol{x}_t|\tilde{\boldsymbol{y}}^{1:k})\rangle\right] + C \\
&- \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_0,\boldsymbol{x}_t,\boldsymbol{y}^{1:k+1},\tilde{\boldsymbol{y}}^{1:k+1})}\left[\langle \nabla_{\boldsymbol{x}_t} log\, p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}), \nabla_{\boldsymbol{x}_t} \log p_\alpha(\boldsymbol{x}_t|\boldsymbol{x}_0)\rangle\right] + \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\, p_{\alpha,\tau}(\boldsymbol{x}_t|\tilde{\boldsymbol{y}}^{1:k}) - \nabla_{\boldsymbol{x}_t} log\, p_\alpha(\boldsymbol{x}_t|\boldsymbol{x}_0)||_2^2\right]
\end{aligned} \tag{62}
$$

$$
= \mathbb{E}_{p_{\alpha,\tau}(\boldsymbol{x},\boldsymbol{x}_t,\boldsymbol{y}^{1:k+1},\tilde{\boldsymbol{y}}^{1:k+1})}\left[\frac{1}{2}||\nabla_{\boldsymbol{x}_t} log\, p_{\boldsymbol{\phi}}(\tilde{\boldsymbol{y}}^{k+1}|\boldsymbol{x}_t,\tilde{\boldsymbol{y}}^{1:k}) + \nabla_{\boldsymbol{x}_t} log\, p_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\tilde{\boldsymbol{y}}^{1:k}) - \nabla_{\boldsymbol{x}_t} log\, p_\alpha(\boldsymbol{x}_t|\boldsymbol{x})||_2^2\right] + C \tag{63}
$$