Multimodal Diffusion Transformer for Learning from Play

Moritz Reuss, Rudolf Lioutikov
Intuitive Robots Lab, Karlsruhe Institute of Technology
{moritz.reuss, lioutikov}@kit.edu

Abstract: Diffusion models have emerged as a powerful class of generative models with wide-spread adoption in many areas. They have shown surprising effectiveness, as a conditional policy representation in the context of robotic learning. This performance has led to the popularity of various frameworks, that use diffusion models to predict trajectories, action sequences or videos. Despite their prowess, existing methodologies do not adequately address learning from multimodal goal specifications, a frequent occurrence in Learning from Play (LfP) with sparse language labels. Addressing this gap, we present Multimodal Diffusion Transformer (MDT), a novel diffusion policy framework. MDT integrates multimodal transformers, pretrained foundation models, and latent token alignment to master long-horizon manipulation based on multimodal goal specifications. Tested on the challenging CALVIN benchmark, MDT not only sets a new performance benchmark for end-to-end policies but also achieves this with less than 10% of the training time of preceding approaches. Our experiments and ablations further validate the effectiveness and strategic choices behind MDT.

1 Introduction

Achieving generalizable robot agents is a core challenge in robotics. Such agents must not only exhibit versatility across diverse tasks and environments but also provide an intuitive interface for non-expert users. They should understand and act on user commands by using natural language as an intuitive abstraction, whereas they currently rely on task-specific models with manually specified rewards or impractical goal designations.

Despite its potential, training robots using language instructions remains a significant challenge. Multi-Task Imitation Learning has emerged as a promising methodology, teaching robots a wide range of skills via learning from recorded human demonstrations [1, 2]. However, the effort and cost associated with curating demonstrations for every task are prohibitive. One way to circumvent these challenges is Learning from Play (LfP) [3], that leverages large uncurated datasets to teach robots in a self-supervised manner to reach arbitrary future states. As shown in prior work, robot agents can effectively learn language conditioned behavior, if as little as 1% of the collected play data contains language labels [1, 4]. However, multimodality, sub-optimal behavior and variance from various non-experts are still major challenges for learning from play. Effective policy representations need to address these challenges while being able to scale efficiently with large play datasets.

Recently, Diffusion Generative Models have gained traction as an effective robot learning policy representations [5, 6]. Diffusion Policies are able to learn expressive, versatile behavior conditioned on language-goals [7, 8]. However, none of these methods deal with the problem of learning from multimodal goal specifications, commonly used in LfP, where language labels are sparse. Leveraging large datasets with limited language labels requires policy representations to learn efficiently from these task specifications of different sensor modalities e.g. vision and language. To this end, we introduce a novel continuous-time diffusion policy framework, that can learn language-guided behavior from uncurated play data with different goal definitions. Our novel framework, Multimodal Diffusion Transformer (MDT), combines the strengths of multimodal transformers and pre-

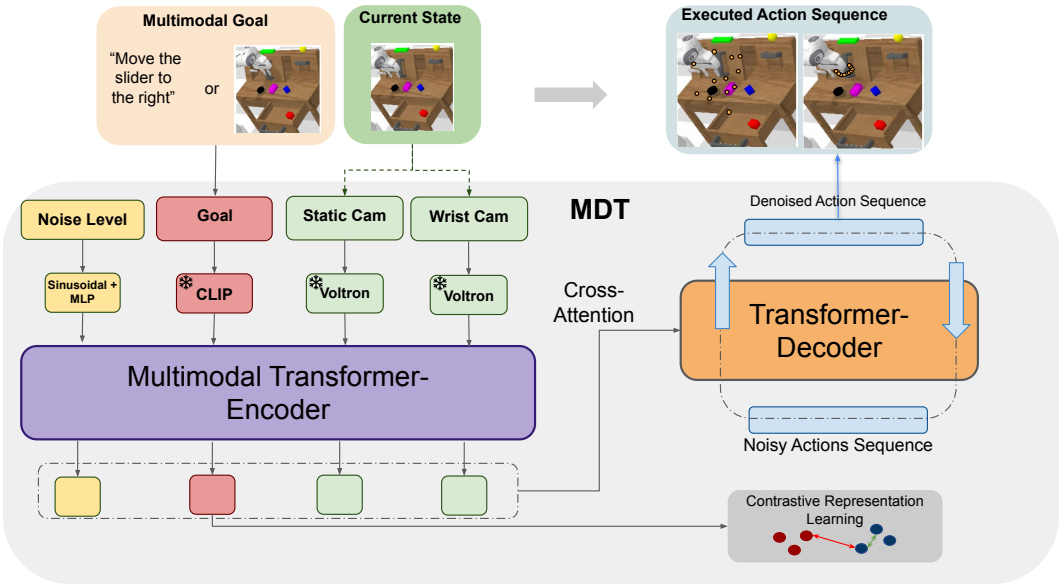

Figure 1: Overview of the proposed multimodal Transformer-Encoder-Decoder Diffusion Policy Framework. MDT learns a joint latent sequence of our noise level, goal, and observations and iteratively denoises an action sequence of 10 steps with a Transformer Decoder with causal Attention. In addition, we align latent goal tokens of images and goals using self-supervised contrastive learning.

trained foundation models with latent token alignment to perform long-horizon manipulation from goal specifications in vision and language. We test MDT in the CALVIN challenge, an established benchmark for language-guided learning from pure play data collected by human demonstrations. Our framework establishes a new state-of-the-art on two CALVIN challenges D→D and ABCD→D, while requiring less than 10% of the training time than the prior best methods. Notably, MDT achieves a 23.5% absolute performance increase on the ABCD→D over prior state-of-the-art. Several experiments and ablations demonstrate the effectiveness of the proposed method and design choices.

2 Related Work

Language Conditioned Robot Learning Language is an intuitive and understandable interface for human-robot interactions, hence there has been a growing interest for language-guided learning methods in the robotics community. The recent advancement of available foundation models, including large-language models (LLMs) [9, 10] and vision-and-language models (VLAs) [11, 12, 13] have further accelerated the progress of language-guided robot learning. Several recent approaches use frozen foundation models to encode images and language abstractions for down-stream policy learning [14, 15, 16] and improved language expression-grounding [17, 18]. Another body of research focuses on end-to-end learning methods that try to learn manipulation directly from pixel observations and text instructions [4, 19]. Hierarchical skill learning methods for language-guided manipulation are commonly used in LfP [20, 3, 15, 21, 22]. By disentangling low-level control from high-level plans, these policies are able to deal with the multimodality in human demonstrations. In addition, the low-level policy can focus on executing the proposed plan for precise low-level manipulation. This hierarchical setup enables policies to learn high-level plans from human videos and low-level control from robot demonstrations [23]. Other approaches focus on using transformer-based methods without any hierarchy [24, 6].

Diffusion Policies in Robotics Denoising Diffusion Generative Models (DDPM) [25, 26] are used in the context of imitation learning and offline reinforcement learning. Trajecory-Diffusion policies are commonly used to denoise a state-based trajectory [27, 28, 29], latent-trajectories [29] or videos

[30] in the context of offline Reinforcement Learning. Other approaches leverage diffusion models as a conditional policies in imitation learning to denoising actions directly [6, 5, 31]. Most related to our work is Play-Fusion [19], Distill-Down [8] and ChainedDiffuser [7], that also use a Diffusion-based Policy for Language-guided Robot learning. However, these methods are limited to language-conditioning only and do not consider the challenge of learning from multimodal goals with few language labels. MDT can effectively learn language-guided behavior with a dataset of little as 1% of language annotations and we show improved performance over the U-net baselines. LAD [29] uses the learned latent skill embedding of HULC [4], as a planning abstraction for its diffusion planning model combined with HULC as the low-level policy. UniPy [30] directly plans in the image space using a two stage hierarchical video diffusion model and executes the plan with an inverse dynamics model.

3 Method

In this section, we detail our MDT framework designed for learning language-guided manipulation from Play using Score-based Diffusion Policies. We start with the problem definition to provide context. Next, we discuss the continuous-time diffusion formulation, essential for understanding action sequence learning. This is followed by an overview of our proposed transformer architecture of MDT. We conclude the section by introducing the latent token alignment, which aids in optimizing learning from play.

3.1 Problem Formulation

A goal-conditioned policy $\pi_\theta(\bar{\boldsymbol{a}}_i|\boldsymbol{s}_i, \boldsymbol{g}_i)$ predicts a sequence of actions $\bar{\boldsymbol{a}}_i = (\boldsymbol{a}_i, \dots, \boldsymbol{a}_{i+k-1})$ of length k, conditioned on both the current state embedding $\boldsymbol{s}_i$ and a latent goal $\boldsymbol{g}_i$. The latent goal $\boldsymbol{g}_i$ encapsulates either an encoded free-form language instruction $\mathbf{l}_g$ or a goal-image $\mathbf{o}_g$, which illustrates the terminal state of a completed task. MDT learns such policies from a set of task-agnostic play trajectories $\mathcal{T}$. Each individual trajectory $\tau \in \mathcal{T}$ represents a series of triplets $(\boldsymbol{s}_i, \boldsymbol{a}_i, \boldsymbol{g}_i)_{i=1:N_\tau}$, with observation $\boldsymbol{s}_i$, action $\boldsymbol{a}_i$, and goal $\boldsymbol{g}_i$. Several subsequent steps $i, i+1$ of the trajectory can have the same goal, e.g $\boldsymbol{g}_i = \boldsymbol{g}_{i+1}$. The final play dataset is now defined as $\mathcal{D} = \{(\boldsymbol{s}_i, \bar{\boldsymbol{a}}_i, \boldsymbol{g}_i)|\bar{\boldsymbol{a}}_i = (\boldsymbol{a}_i, \dots, \boldsymbol{a}_{i+k-1}), (\boldsymbol{s}_i, \boldsymbol{a}_i, \boldsymbol{g}_i) \in \tau, \tau \in \mathcal{T}\}$, where $\bar{\boldsymbol{a}}_i$ is the action chunk starting at step i of trajectory τ. The methodology for sampling these goals is outlined in Section 4.5, elucidating MDT to handle the label-less multi-task nature of play data. MDT maximizes the log-likelihood across the play dataset,

$$\mathcal{L}_{\text{play}} = \mathbb{E}\left[\sum_{(\boldsymbol{s}_i, \bar{\boldsymbol{a}}_i, \boldsymbol{g}_i)\in\mathcal{D}} \log \pi_\theta\left(\bar{\boldsymbol{a}}_i|\boldsymbol{s}_i, \boldsymbol{g}_i\right)\right]. \tag{1}$$

Given the diversity in human behavior evident in the demonstrations, with multiple trajectories possibly converging towards an identical goal state, a successful realization of this objective mandates a policy adept at encoding such diverse behavior [32].

3.2 Score-based Diffusion Policy

In this section, we introduce our Language-guided Diffusion Policy for Learning Long-Horizon Manipulation form Play with limited language annotation. MDT leverages the continuous time diffusion model [33, 26]. Diffusion models are a type of generative models that learn to generate new data by iteratively adding data to noise via Gaussian perturbations. After training, they generate new samples via sequential denoising steps starting with random Gaussian noise. This forward and inverse process can be described as a continuous diffusion process with a stochastic-differential equation (SDE) [26]. MDT leverages the EDM SDE formulation

$$d\mathbf{a} = \left(\beta_t\sigma_t - \dot{\sigma}_t\right)\sigma_t\nabla_a \log p_t(\bar{\boldsymbol{a}}_i|\boldsymbol{s}_i, \boldsymbol{g}_i)dt + \sqrt{2\beta_t}\sigma_t d\omega_t, \tag{2}$$

commonly used in image generation [33, 34]. The score-function $\nabla_{\bar{\boldsymbol{a}}_i} \log p_t(\bar{\boldsymbol{a}}_i|\boldsymbol{s}_i, \boldsymbol{g}_i)$ is parameterized by the continuous diffusion variable $t \in [0, T]$, with constant horizon $T > 0$. This formulation

reduces the stochasticity to the Wiener process ω_t, which can be interpreted as infinitesimal Gaussian noise that is added to the action sample. The term σ_t is the noise scheduler that defines the rate of added Gaussian noise depending on the current time t of the diffusion process. Following best practices of prior work [33, 6, 34], MDT uses $\sigma_t = t$ for the policy. The range of noise perturbations is set to $\sigma_t \in [0.001, 80]$ and the action range is rescaled to $[-1, 1]$. The function β_t describes the replacement of existing through injected new noise [33]. This SDE is notable for having an associated ordinary differential equation (ODE), termed the Probability Flow (PF) ODE [26], which can be recovered from the SDE. When action chunks of this ODE are sampled at time t of the diffusion, they align with the distribution $p_t(\bar{\boldsymbol{a}}_i|\boldsymbol{s}_i, \boldsymbol{g}_i)$,

$$d\bar{\boldsymbol{a}}_i = -\dot{\sigma}_t \sigma_t \nabla_{\bar{\boldsymbol{a}}_i} \log p_t(\bar{\boldsymbol{a}}_i|\boldsymbol{s}_i, \boldsymbol{g}_i) dt. \tag{3}$$

The diffusion model learns the score-model $\nabla_{\bar{\boldsymbol{a}}_i} \log p_t(\bar{\boldsymbol{a}}_i|\boldsymbol{s}_i, \boldsymbol{g}_i)$ via Score matching (SM) [35]

$$\mathcal{L}_{\text{SM}} = \mathbb{E}_{\sigma, \bar{\boldsymbol{a}}_i, \boldsymbol{\epsilon}} \big[\alpha(\sigma_t) \| D_\theta(\bar{\boldsymbol{a}}_i + \boldsymbol{\epsilon}, \boldsymbol{s}_i, \boldsymbol{g}_i, \sigma_t) - \bar{\boldsymbol{a}}_i \|_2^2 \big], \tag{4}$$

where $D_\theta(\bar{\boldsymbol{a}}_i + \boldsymbol{\epsilon}, \boldsymbol{s}_i, \boldsymbol{g}_i, \sigma_t)$ is our learned neural network. During training, we randomly sample noise levels from our noise distribution and predict the denoised action sequence. During inference, we then substitute our learned score-model into the reverse SDE and iteratively denoise our actions. By setting $\beta_t = 0$, we are able to recover the deterministic inverse process that allows for fast sampling in a few denoising steps without injecting additional noise into the forward process [26]. The modular continuous time diffusion perspective enables the use of any specialized sampler during inference, e.g., DDIM [36] or different time step discretizations [37, 6]. Detailed training and inference description can be found in the Section A of the Appendix. For our experiments, we use the Euler&Ancestral sampler [26] with 10 denoising steps [36] and exponential time steps.

3.3 Model Architecture

MDT uses a multi-modal transformer encoder-decoder architecture to learn the conditional score of the action sequence. An overview of the architecture is given in Figure 1. We encode the current image observation of the static camera and the wrist-camera with two pre-trained Voltron models [11]. Voltron is a vision-language foundation model, consisting of a vision-transformer with additional language tokens, trained on masked reconstruction to learn good visiual features. The sampled goal images or language annotations are encoded with a frozen CLIP model [12]. The current noise level σ_t is further embedded using a Sinusoidal Embedding with an additional Multi-Layer Perceptron (MLP). All inputs are then processed in 4 encoder layers with self-attention to predict a set of latent embedded tokens. The additional transformer-encoder is key to aligning goal-images with language ones. The decoder part of the model takes the sequence of noisy action tokens and predicts the denoised ones with causal masking. The conditioning information is fused into the denoising prediction via cross-attention in every decoder layer with all latent tokens of the encoder. MDT predicts 10 actions in all our experiments.

3.4 Latent Alignment of Goal Tokens

In order to effectively learn from multimodal goal specifications, MDT must align visual goals with their linguistic counterparts. To this end, MDT employs a pre-trained CLIP model, which has been trained on paired language and text samples from a substantial internet dataset [12]. However, as highlighted in various studies [38, 13, 4], CLIP exhibits a tendency toward static images and struggles to interpret spatial relationships. Furthermore, goal specifications in robotics are inherently linked to the contextual dynamics between the current state $\boldsymbol{s}_i$ and the desired goal $\boldsymbol{g}_i$. Due to its inherent design, CLIP embeds singular images or sentences and cannot deal with dynamic scenes. To improve the multimodal alignment, while keeping all input models frozen, MDT leverages a contrastive loss on the transformer encoded latent goal token. Thanks to this shift in focus MDT can efficiently use CLIP without the need to fine-tune the 149 million weights of the model. Instead, we solely optimize the relatively small transformer-encoder, which is notably more resource-efficient. Crucially, our latent goal token $\mathbf{z}_g$ incorporated additional information from the current context by

Train→Test	Method	No. Instructions in a Row (1000 chains)					
		1	2	3	4	5	**Avg. Len.**
D→D	MCIL	48.9%	12.9%	2.6%	0.5%	0.08%	0.64
	HULC	82.5%	66.8%	52%	39.34%	27.5%	2.68
	Diffuser-2D	37.4%	9.3%	1.3%	0.2%	0.07%	0.48
	LAD	88.7%	69.9%	54.5%	42.7%	32.2%	2.88
	Distill-D	85.0%	67.5%	54.4%	42.1%	33.3%	2.81
	MT-ACT	89.1%	74.8%	61.0%	48.8%	38.7%	3.12
	MDT (ours)	**91.1%**	**78.5%**	**67.0%**	**56.3%**	**47.4%**	**3.40**
ABCD→D	MCIL	37.3%	2.7%	0.2%	0%	0%	0.40
	HULC	88.9%	73.3%	58.7%	47.5%	38.3%	3.06
	MDT (ours)	**93.0%**	**84.7%**	**75.9%**	**66.7%**	**58.2%**	**3.78**

Table 1: Performance comparison of various policies learned end-to-end on the CALVIN D environment within the CALVIN benchmark. We show the average rollout length to solve 5 instructions in a row (Avg. Len.) of 1000 chains. MDT outperforms all reported baselines on this task averaged over 3 seeds.

using attention with the image embeddings of the current state $\boldsymbol{s}_i$, offering a context-rich latent goal embedding. We combine our Score Matching loss from Eq. (4) with the contrastive self-supervised loss:

$$\mathcal{L}_{\text{CLIP}} = -\frac{1}{2N}\sum_{i=1}^{N}\left(\log\left(\frac{\exp\left(\frac{V_i\cdot T_i}{\tau}\right)}{\sum_{k=1}^{N}\exp\left(\frac{V_i\cdot T_k}{\tau}\right)}\right)+\log\left(\frac{\exp\left(\frac{T_i\cdot V_i}{\tau}\right)}{\sum_{k=1}^{N}\exp\left(\frac{T_i\cdot V_k}{\tau}\right)}\right)\right), \tag{5}$$

where $V = \frac{\mathbf{o}_g}{\|\mathbf{o}_g\|_2}$ refers to the normalized image goal embedding, $T = \frac{\mathbf{l}_g}{\|\mathbf{l}_g\|_2}$ is the normalized language goal embedding, τ is a tunable temperature parameter and N the batch size. The full loss of MDT is defined as

$$\mathcal{L}_{\text{MDT}} = \mathcal{L}_{\text{SM}} + \lambda\mathcal{L}_{\text{CLIP}}, \tag{6}$$

where we chose $\lambda = 1$ for MDT.

4 Evaluation

In our experiments we try to answer the following questions: (I) Is MDT able to learn long-horizon language guided manipulation from play data with little language annotation? (II) How time-effective is MDT training compared to other state-of-the-art methods on CALVIN? (III) Which key-components of MDT enable the effective language understanding with limited annotations?

4.1 Baselines

We compare our proposed policy against the following state-of-the-art language-conditioned multi-task policies:

- **MCIL:** A hierarchical play policy, that learns an abstract latent skill space using a variational autoencoder (VAE) combined with a low-level action policy [3]
- **HULC:** Another hierarchical play policy, that uses discrete VAE skill space with an improved low-level action policy and additional allignment losses [4].
- **Diffuser-2D** Inspired by the Diffuser [28, 27], we use a baseline that plans in a image-based latent space from Stable Diffusion [39]. We use the results reported in Zhang et al. [29].
- **LAD:** A hierarchical diffusion policy, that extends the HULC policy by substituting the high-level planner with a U-Net Diffusion model [29].

MDT:	default	no $\mathcal{L}_{\text{CLIP}}$	Random	Key States	ResNet	Lang-only	Vision-only
Avg. Len.	3.40	3.22	3.30	0.48	3.13	2.95	0.21

Table 2: MDT Ablation studies on different goal sampling strategies and language annotations. We compare the performance all ablations on the CALVIN benchmark to solve 5 free-form language instructions in a row across 1000 chains. The "$\mathcal{L}_{CLIP}$" configuration indicates MDT trained without the inclusion of the latent alignment loss. In the "*Random*" setting, random sampling is employed for goal index selection as opposed to conventional random sampling. The "*Key States*" configuration makes use of key-states as goals, detected using Alg. 3. "*ResNet*" denotes MDT with ResNets trained from scratch instead of pre-trained Voltron. The "*Lang-only*" setup denotes an MDT model trained solely on the 1% of the dataset that contains language labels, while "*Vision-only*" represents MDT trained entirely without any language labels.

- **Distill-D** A language-guided Diffusion policy from [8], that extends the initial U-Net diffusion policy from [5] with additional Clip Encoder for language-prompts. We use our continuous time diffusion variant instead of the discrete one for a direct comparison.
- **MT-ACT** A multitask transformer policy [16, 40], that uses a VAE encoder for action sequences and also predicts action chunks instead of single actions.

We adopt the recommend hyperparameters for all baselines to guarantee a fair comparison. Further, we directly report the results for HULC, LAD and Diffuser-2D from their paper [29, 4]. We focus our comparison of MDT with policies that are predict actions at each timestep. Consequently, we exclude HULC++ [15], which combines motion planning with a trained HULC policy.

4.2 Simulation Environment

We conduct our experiments on the CALVIN Benchmark [20], that has been designed to assess the performance of robot manipulation policies that are conditioned on language, particularly for tasks requiring extended planning and execution horizons. The benchmark consists of four different environments A,B,C,D in a similar setting. The four setups vary in desk shades and the layout of items as visualized in Figure 3. Our main experiments are conducted on the sub-environment D, where we train and test all policies on D→D. This setting contains 6 hours of uncurated teleoperated play data with multiple sensor modalities and 34 different tasks. Further, only 1% of data is annotated by language. All methods are evaluated on the long-horizon benchmark, that consists of 1000 unique sequences consisting of 5 tasks described natural language. During the rollouts, the agent gets a reward of 1 for solving a pre-defined task with a maximum of 5 for every rollout. However, the agents only receives the next task, if the prior one has been successfully completed. We additionally perform experiments on the full benchmark ABCD→D consisting of 24 hours of play data in all four environments to investigate the scaling efficiency of MDT. To be successful on the CALVIN environment, policies require long-term planning, vision-language goal grounding, and expressive learning of multi-modal behavior.

4.3 Evaluation Results

The results of our experiments are summarized in Table 1. Our proposed MDT policy sets a new state-of-the-art performance on the CALVIN challenge with an absolute performance increase of 18% over the prior SOTA LAD. Further, the results show that MDT improves upon the U-net based Distill-D policy, indicating that our proposed architecture is key to success. We also test MDT on the full CALVIN dataset and test its performance on the split ABCD→D. The results are shown in Table 1. MDT outperforms the prior state-of-the-art method HULC with an absolute performance boost of 23.5%, while training over ten time faster. Thus, we can answer Question (I) in the affirmative.

4.4 Average Training and Inference Time

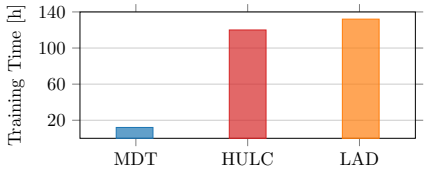

Figure 2: The average Training time of MDT compared to the two state-of-the-art methods on CALVIN on our SLURM Cluster in hours.

Next, we investigate Question (II) by comparing the training efficiency of various methods by measuring the average training time of MDT against LAD and HULC. We train all methods on a SLURM Cluster with 4 RTX 3090 GPUs. The average training time of all tested models of this experiment are summarized in Figure 2. Comparing the training time shows that MDT outperforms both baselines with considerably less training time, demonstrating the efficiency of our method.

Diffusion Policies require numerous denoising steps to predict a single action or action chunks. Thus, they commonly have significantly slower inference compared the other methods. MDT only requires 10 denoising steps every 10 rollout steps to create an action chunk. On average MDT requires the same number of forward passes compared to other methods such as HULC, that predict a single action every time-step.

4.5 Ablation Studies

We further analyse the importance of various design choices for our proposed framework to learn effectively with limited language annotation to answer question (III).

Goal-State Sampling Strategies. During training we need to chose a random future state for learning goal-conditioned behavior in a self-supervised manner. This is required, since the play data contains trajectories of flexible length, where the start and end-states of individual tasks are unknown. Hence, we require a specific strategy to sample goal-images during training. We asses three sampling strategies: geometric distribution with $p = 0.1$, random sampling in goal-window range, and key-states based sampling inspired by an algorithm commonly used in RLBench [41] and summarized in Alg. 3 in the Appendix. We train MDT with all these sampling strategies to determine the best one and report the results in Table 2. The key-state strategy performs by far the worst and lowers the performance, while the geometric strategy and the random sampling perform similarly. We hypothesise that the human play demonstrations make it harder to detect key states. The CALVIN dataset was collected by various humans that can act and control the arm differently, while the RLBench data consists of non-human demonstrations. The results demonstrate the importance of the goal-sampling strategy for MDT to succeed on CALVIN. On average, the geometric strategy works best and random sampling being second. Thus, we adopt the geometric sampling strategy as our default method for MDT.

Multimodal Token Alignment. Next, we evaluate the significance of our latent token alignment technique. We train MDT without the additional contrastive loss and compare the performance on CALVIN. The findings, detailed in Table 2, reveal that in the absence of latent alignment, MDT's average rollout length reduces to 3.22. Introducing the contrastive loss not only enhances overall performance but also empowers MDT to more effectively utilize play data devoid of language annotations.

The Importance of Goal-Image Data. Further, we test if having additional training data without the language-annotations helps the model to improve its performance and generalization. We train MDT with only the 1% language-annotated data of the CALVIN benchmark and compare the results against the performance reported in Table 1. The results of the language-only model are summarized in Table 2. Without the additional data, the performance of MDT drops from 3.40 to 2.95. The performance decrease compared to the model trained on image-data demonstrates that the additional data helps the model to generalize better.

The Impact of Language Annotations. We examined how including a small amount of language annotations—just 1% of the dataset—affected the performance of MDT on the CALVIN Benchmark. Therefore, we train MDT with language labels and without on the CALVIN Benchmark and

report the respective performance on the Long Horizon Benchmark. Without the 1% language annotation, the model performance drops from 3.4 to 0.2. This highlights the crucial role even a small number of language labels can play in improving model performance and points to the limited ability of the pretrained CLIP model to generalize, a finding supported by previous research [38, 42].

Pre-trained Visual Embeddings vs End-to-End Visual Embeddings Finally, we examine the performance of MDT combined with visual embeddings trained from scratch. Therefore, we train MDT with the standard ResNet-18 used in Diffusion Policy [5]. The peformance of this variant achieves 3.13 underlining the effectiveness of Voltron Embeddings. Intriguingly, MDT is the only model, that benefits from pre-trained Voltron Embeddings. We also evaluated Distill-D and MT-ACT with Voltron, but noticed significant performance drops compared to standard ResNet-18 in both model variants.

5 Conclusion

In this work we present MDT, a novel language-guided continuous-time diffusion policy, that learns long-horizon manipulation form play with as little as 1% language labels. We show that our approach outperforms current state-of-the-art methods on the challenging CALVIN Benchmark, while training 10 times faster. For future work, we will explore fine-tuned CLIP embeddings, as they have shown to improve performance [42, 13] and efficient methods for fine-tuning MDT after training [43] to further increase its performance. MDT presents a promising foundation for further investigations into Diffusion Policies tailored for long-horizon manipulation with sparse language annotation.

6 Acknowledgements

This work is supported by the German Research Foundation (DFG) – 448648559. The authors acknowledge support by the state of Baden-Württemberg through HoreKa supercomputer funded by the Ministry of Science, Research and the Arts Baden-Württemberg and by the German Federal Ministry of Education and Research.

References

[1] C. Lynch and P. Sermanet. Language conditioned imitation learning over unstructured data. *arXiv preprint arXiv:2005.07648*, 2020.

[2] C. Lynch, A. Wahid, J. Tompson, T. Ding, J. Betker, R. Baruch, T. Armstrong, and P. Florence. Interactive language: Talking to robots in real time. *IEEE Robotics and Automation Letters*, 2023.

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

[4] O. Mees, L. Hermann, and W. Burgard. What matters in language conditioned robotic imitation learning over unstructured data. *IEEE Robotics and Automation Letters (RA-L)*, 7(4):11205–11212, 2022.

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

[6] M. Reuss, M. Li, X. Jia, and R. Lioutikov. Goal-conditioned imitation learning using score-based diffusion policies. *arXiv preprint arXiv:2304.02532*, 2023.

[7] Z. Xian, N. Gkanatsios, T. Gervet, and K. Fragkiadaki. Unifying diffusion models with action detection transformers for multi-task robotic manipulation. In *7th Annual Conference on Robot Learning*, 2023.

[8] H. Ha, P. Florence, and S. Song. Scaling up and distilling down: Language-guided robot skill acquisition. In *7th Annual Conference on Robot Learning*, 2023.

[9] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, Q. Vuong, V. Vanhoucke, H. Tran, R. Soricut, A. Singh, J. Singh, P. Sermanet, P. R. Sanketi, G. Salazar, M. S. Ryoo, K. Reymann, K. Rao, K. Pertsch, I. Mordatch, H. Michalewski, Y. Lu, S. Levine, L. Lee, T.-W. E. Lee, I. Leal, Y. Kuang, D. Kalashnikov, R. Julian, N. J. Joshi, A. Irpan, brian ichter, J. Hsu, A. Herzog, K. Hausman, K. Gopalakrishnan, C. Fu, P. Florence, C. Finn, K. A. Dubey, D. Driess, T. Ding, K. M. Choromanski, X. Chen, Y. Chebotar, J. Carbajal, N. Brown, A. Brohan, M. G. Arenas, and K. Han. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *7th Annual Conference on Robot Learning*, 2023. URL https://openreview.net/forum?id=XMQgwiJ7KSX.

[10] K. Rana, J. Haviland, S. Garg, J. Abou-Chakra, I. Reid, and N. Suenderhauf. Sayplan: Grounding large language models using 3d scene graphs for scalable robot task planning. In *7th Annual Conference on Robot Learning*, 2023. URL https://openreview.net/forum?id=wMpOMO0Ss7a.

[11] S. Karamcheti, S. Nair, A. S. Chen, T. Kollar, C. Finn, D. Sadigh, and P. Liang. Language-driven representation learning for robotics. *arXiv preprint arXiv:2302.12766*, 2023.

[12] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[13] V. Myers, A. W. He, K. Fang, H. R. Walke, P. Hansen-Estruch, A. Kolobov, A. Dragan, and S. Levine. Goal representations for instruction following: A semi-supervised language interface to control. In *7th Annual Conference on Robot Learning*, 2023. URL https://openreview.net/forum?id=0bZaUfELuW.

[14] M. Shridhar, L. Manuelli, and D. Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning*, pages 785–799. PMLR, 2023.

[15] O. Mees, J. Borja-Diaz, and W. Burgard. Grounding language with visual affordances over unstructured data. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11576–11582. IEEE, 2023.

[16] H. Bharadhwaj, J. Vakil, M. Sharma, A. Gupta, S. Tulsiani, and V. Kumar. Roboagent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking, 2023.

[17] N. Gkanatsios, A. Jain, Z. Xian, Y. Zhang, C. Atkeson, and K. Fragkiadaki. Energy-based Models are Zero-Shot Planners for Compositional Scene Rearrangement. In *Robotics: Science and Systems*, 2023.

[18] A. Jain, N. Gkanatsios, I. Mediratta, and K. Fragkiadaki. Bottom up top down detection transformers for language grounding in images and point clouds. In *European Conference on Computer Vision*, pages 417–433. Springer, 2022.

[19] L. Chen, S. Bahl, and D. Pathak. Playfusion: Skill acquisition via diffusion from language-annotated play. In *7th Annual Conference on Robot Learning*, 2023.

[20] O. Mees, L. Hermann, E. Rosete-Beas, and W. Burgard. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, 2022.

[21] E. Rosete-Beas, O. Mees, G. Kalweit, J. Boedecker, and W. Burgard. Latent plans for task-agnostic offline reinforcement learning. In *6th Annual Conference on Robot Learning*, 2022. URL https://openreview.net/forum?id=ViYLaruFwN3.

[22] H. Zhou, Z. Bing, X. Yao, X. Su, C. Yang, K. Huang, and A. Knoll. Language-conditioned imitation learning with base skill priors under unstructured data. *arXiv preprint arXiv:2305.19075*, 2023.

[23] C. Wang, L. Fan, J. Sun, R. Zhang, L. Fei-Fei, D. Xu, Y. Zhu, and A. Anandkumar. Mimicplay: Long-horizon imitation learning by watching human play. In *7th Annual Conference on Robot Learning*, 2023. URL https://openreview.net/forum?id=hRZ1YjDZmTo.

[24] Z. J. Cui, Y. Wang, N. M. M. Shafiullah, and L. Pinto. From play to policy: Conditional behavior generation from uncurated robot data. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=c7rM7F7jQjN.

[25] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[26] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2020.

[27] A. Ajay, Y. Du, A. Gupta, J. B. Tenenbaum, T. S. Jaakkola, and P. Agrawal. Is conditional generative modeling all you need for decision making? In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=sP1fo2K9DFG.

[28] M. Janner, Y. Du, J. Tenenbaum, and S. Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, pages 9902–9915. PMLR, 2022.

[29] E. Zhang, Y. Lu, W. Wang, and A. Zhang. Language control diffusion: Efficiently scaling through space, time, and tasks. *arXiv*, 2023.

[30] Y. Dai, M. Yang, B. Dai, H. Dai, O. Nachum, J. Tenenbaum, D. Schuurmans, and P. Abbeel. Learning universal policies via text-guided video generation. *arXiv preprint arXiv:2302.00111*, 2023.

[31] T. Pearce, T. Rashid, A. Kanervisto, D. Bignell, M. Sun, R. Georgescu, S. V. Macua, S. Z. Tan, I. Momennejad, K. Hofmann, and S. Devlin. Imitating human behaviour with diffusion models. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Pv1GPQzRrC8.

[32] D. Blessing, O. Celik, X. Jia, M. Reuss, M. X. Li, R. Lioutikov, and G. Neumann. Information maximizing curriculum: A curriculum-based approach for training mixtures of experts. *arXiv preprint arXiv:2303.15349*, 2023.

[33] T. Karras, M. Aittala, T. Aila, and S. Laine. Elucidating the design space of diffusion-based generative models. In A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=k7FuTOWMOc7.

[34] Y. Song, P. Dhariwal, M. Chen, and I. Sutskever. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023.

[35] P. Vincent. A connection between score matching and denoising autoencoders. *Neural Computation*, 23(7):1661–1674, 2011. doi:10.1162/NECO_a_00142.

[36] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. In *ICLR*, 2021.

[37] A. Q. Nichol and P. Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021.

[38] T. Xiao, H. Chan, P. Sermanet, A. Wahid, A. Brohan, K. Hausman, S. Levine, and J. Tompson. Robotic skill acquisition via instruction augmentation with vision-language models. *arXiv preprint arXiv:2211.11736*, 2022.

[39] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[40] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

[41] S. James, Z. Ma, D. Rovick Arrojo, and A. J. Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 2020.

[42] Y. Ge, A. Macaluso, L. E. Li, P. Luo, and X. Wang. Policy adaptation from foundation model feedback. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19059–19069, 2023.

[43] J. Watson, S. H. Huang, and N. Heess. Coherent soft imitation learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

A Score Model Training and Inference

The training process of MDT is summarized in Alg. 1 and the reverse diffusion process used for generating action chunks during the sequence is summarized in Alg. 2. Further, an overview for the used hyperparameters is given in Table 3. To increase the performance, we deploy the preconditioning of Karras et al. [33]. This includes additional skip-connections and two pre-conditioning layers, which are conditioned on the current noise level σ_t for effective balancing of the high range of noise levels from 0.001 to 80

$$D_\theta(\bar{\boldsymbol{a}}_i|\boldsymbol{s}_i, \boldsymbol{g}_i, \sigma_t) = c_{\text{skip}}(\sigma_t)\bar{\boldsymbol{a}}_i + c_{\text{out}}(\sigma_t)F_\theta(c_{\text{in}}(\sigma_t)\bar{\boldsymbol{a}}_i, \boldsymbol{s}_i, \boldsymbol{g}_i, c_{\text{noise}}(\sigma_t)). \tag{7}$$

The utilized reconditioning functions are defined as:

- $c_{\text{skip}} = \sigma_{\text{data}}^2/(\sigma_{\text{data}}^2 + \sigma_t^2)$
- $c_{\text{out}} = \sigma_t\sigma_{\text{data}}/\sqrt{\sigma_{\text{data}}^2 + \sigma_t^2}$
- $c_{\text{in}} = 1/\sqrt{\sigma_{\text{data}}^2 + \sigma_t^2}$
- $c_{\text{noise}} = 0.25\ln(\sigma_t)$

They allow the model to decide, if it wants to predict the current noise, the denoised action sequence or something inbetweeen, depending on the current noise level [33].

Algorithm 1 MDT Training

Require: Play Dataset $\mathcal{L}_{\text{play}}$
Require: Score Model $\mathbf{D}_\theta(\bar{\boldsymbol{a}}_i, \boldsymbol{s}_i, \boldsymbol{g}_i, \sigma_t)$
Require: Noise Distribution: LogLogistic(α, β)
for $i \in \{0, ..., N_{\text{train steps}}\}$ **do**
  Sample $(\boldsymbol{o}, \boldsymbol{g}_i) \sim \mathcal{L}_{\text{play}}$
  Sample $\sigma_t \sim$ LogLogistic(α, β)
  Sample $\epsilon \sim \mathcal{N}(0, \sigma_t)$
  $\mathcal{L}_{D_\theta} \leftarrow \mathbb{E}_{\sigma, \bar{\boldsymbol{a}}_i, \boldsymbol{\epsilon}}\big[\alpha(\sigma_t) \|D_\theta(\bar{\boldsymbol{a}}_i + \boldsymbol{\epsilon}, \boldsymbol{s}_i, \boldsymbol{g}_i, \sigma_t) - \bar{\boldsymbol{a}}_i\|_2^2\big]$
end for

Algorithm 2 MDT Action Generation using Deterministic 1st Order Euler Sampler [33]

Require: Current state $\boldsymbol{s}_i$, goal $\boldsymbol{g}_i$
Require: Score Model: $D_\theta(\mathbf{a}, \boldsymbol{s}_i, \boldsymbol{g}_i, \sigma)$
Require: Noise scheduler $\sigma_t = \sigma(t_i)$
Require: Discrete time steps $t_{i\in\{0,..,N\}}$
Draw sample $\mathbf{a}_{1:k,0} \sim \mathcal{N}(\mathbf{0}, \sigma_0^2\mathbf{I})$
for $i \in \{0, ..., N-1\}$ **do**
  $\mathbf{d}_i \leftarrow \big(\mathbf{a}_{1:k,i} - D_\theta(\mathbf{a}_i, \boldsymbol{s}_i, \mathbf{g}, \sigma_i)\big)/\sigma_i$
  $\mathbf{a}_{1:k,i+1} \leftarrow \mathbf{a}_{1:k,i} + (t_{i+1} - t_i)\mathbf{d}_i$
end for
return $\mathbf{a}_{1:k,N}$

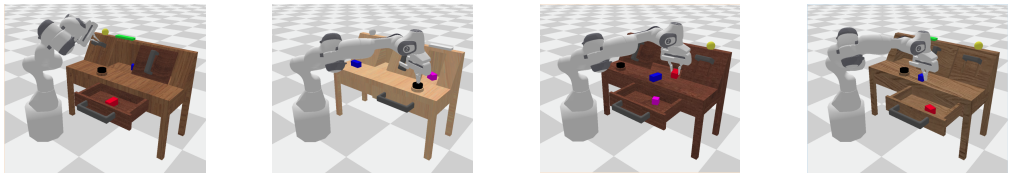

Figure 3: Different CALVIN benchmark environments, each with unique positions and textures for slider, drawer, LED, and lightbulb.

Algorithm 3 Key-Point Detection Algorithm adapted from RLBench [41]

function KEYPOINT_DISCOVERY($actions$, $stopping\_delta$)
  $max\_keypoints \leftarrow$ length of $actions$ ▷ or some other upper bound
  $keypoints \leftarrow$ empty array of size $max\_keypoints$ with data type int
  $k \leftarrow 0$ ▷ Counter for actual number of keypoints
  $prev\_gripper\_open \leftarrow actions[0, -1]$
  $stopped\_buffer \leftarrow 0$
  for $i = 0 \rightarrow$ length of $actions - 1$ **do**
    $stopped \leftarrow$ IS_STOPPED($actions, i, stopped\_buffer, stopping\_delta$)
    if $stopped$ **then**
      $stopped\_buffer \leftarrow 4$
    else
      $stopped\_buffer \leftarrow stopped\_buffer - 1$
    end if
    if $i \neq 0$ **and** ($actions[i, -1] \neq prev\_gripper\_open$ **or** $stopped$) **then**
      $keypoints[k] \leftarrow i$
      $k \leftarrow k + 1$
    end if
    $prev\_gripper\_open \leftarrow actions[i, -1]$
  end for
  return $keypoints[: k]$ ▷ Slice to the actual size
end function

	MDT	
Hyperparameter	D→D	ABCD→D
Number of Encoder Layers	4	4
Number of Decoder Layers	6	7
Attention Heads	4	12
Action Chunk Size	10	10
Goal Window Sampling Size	49	49
Hidden Dimension	768	768
Attention Dropout	0.3	0.3
Residual Dropout	0.1	0.1
MLP Dropout	0.05	0.05
Input Dropout	0.0	0.0
Optimizer	AdamW	AdamW
Betas	[0.9, 0.9]	[0.9, 0.9]
Transformer Weight Decay	1e-3	1e-3
Other weight decay	1e-6	1e-6
Batch Size per GPU	128	128
Train Epochs	15	20
Trainable Parameters	86.2 million	98.0 million
σ_{max}	80	80
σ_{min}	0.001	0.001
σ_t	0.5	0.5
Time steps	Exponential	Exponential
Sampler	DDIM	DDIM

Table 3: Summary of all the Hyperparameters for the MDT policy used in the CALVIN experiments