

# Affordance-based Robot Manipulation with Flow Matching

Fan Zhang, Michael Gienger  
Honda Research Institute EU  
Email: firstname.lastname@honda-ri.de

**Abstract**—We present a framework for assistive robot manipulation, which focuses on two fundamental challenges: first, efficiently adapting large-scale models to downstream scene affordance understanding tasks, especially in daily living scenarios where gathering multi-task data involving humans requires strenuous effort; second, effectively learning robot action trajectories by grounding the visual affordance model. We tackle the first challenge by employing a parameter-efficient prompt tuning method that prepends learnable text prompts to the frozen vision model to predict manipulation affordances in multi-task scenarios. Then we propose to learn robot action trajectories guided by affordances in a supervised flow matching method. Flow matching represents a robot visuomotor policy as a conditional process of flowing random waypoints to desired robot action trajectories. Finally, we introduce a real-world dataset with 10 tasks across Activities of Daily Living to test our framework. Our extensive evaluation highlights that the proposed prompt tuning method for learning manipulation affordance achieves competitive performance and even outperforms some other finetuning protocols across data scales, while satisfying parameter efficiency. Learning multi-task robot action trajectories with flow matching leads to consistently favorable results in several robot manipulation benchmarks than some alternative behavior cloning methods. This includes more stable training and evaluation, and noticeably faster inference, while maintaining comparable generalization performance to diffusion policy, where flow matching performs marginally better in most cases. Our framework seamlessly unifies affordance learning and action generation with flow matching for robot manipulation. <https://hri-eu.github.io/flow-matching-policy/>

## I. INTRODUCTION

Recent advances in vision-language models (VLMs) present unprecedented opportunities to solve robot manipulation problems. Attempts in the field have focused on three primary aspects: 1) End-to-end learning manipulation from scratch. These approaches [35] make the least assumptions on tasks and are formulated in language-image-to-action prediction models. 2) Off-the-shelf-vision-language models for robot manipulation. Such line of works have explored using pre-trained VLMs with prompt engineering in various contexts of robot motion learning, including reward design for reinforcement learning [31], python coding [26], joint actions [47], etc. 3) Intermediate substrate to bridge high-level language-image instructions and low-level robot policies. These works usually introduce some form of prior derived from human knowledge as an intermediate stage, including affordances [22], pre-trained visual representations [48], primitive skills [23], etc. In this paper, we follow the third line of work to unify an affordance model and low-level robot policies, which helps to alleviate the sample inefficiency problem of end-to-end learning.

Extracting affordance knowledge has long inspired the robot community [48]. Humans heavily rely on affordances to efficiently perform day-to-day tasks across environments. The concept of affordance has been introduced in [12], referring to the ability to perform certain actions with objects in the context of a given scene. In this work, we first focus on learning affordance maps related to the locations of object areas that permit or enable specific actions. Then, the knowledge of affordance maps is used to plan and execute appropriate physical interactions with objects with our robot learning policy.

We particularly concentrate on multi-task scenarios with text prompting. As shown in Fig. 1, given the same visual scene but with different language instructions, we aim to extract different affordances for robot policy learning through our proposed model. To leverage the ability of pre-trained foundation models while avoiding expensive computational constraints, recent works have explored parameter-efficiently fine-tuning (PEFT) large vision-language models for visual recognition applications [24]. PEFT could be mainly categorized into two main groups: adapter-based methods (e.g., LoRA [20]) and prompt-based methods [29]. One representative line of such research has concentrated on prompt tuning methods, which prepend learnable prompts to the input of a large frozen pre-trained model and optimize them via gradients during finetuning. Studies on randomly-generalized trainable prompts [24] for universal use or condition-admitted prompt variables [43] for better specific task performance have been both explored. It has shown that prompt tuning could match the performance of full finetuning but with substantially less parameter storage in various domains, including visual tracking [53] and cross-domain tasks such as language-dance assessment [52]. Inspired in part by the notion of human cognitive penetrability mechanism [32] that uses linguistic knowledge to tune ongoing visual processing, we target to incorporate learnable text-conditioned prompts into any vision foundation model while keeping it frozen, preserving its visual understanding capabilities, to learn instruction-relevant manipulation affordance maps.

The subsequent challenge involves deploying the visual affordance across robot manipulation learning paradigms. From the traditional behavior cloning with convolutional networks [49] to transformer-based learning structures [42], extensive research has modeled robot action trajectories from visual scenes. A recent line of work builds on successes in diffusion models [3] with Denoising Diffusion Probabilistic Models (DDPM [19])

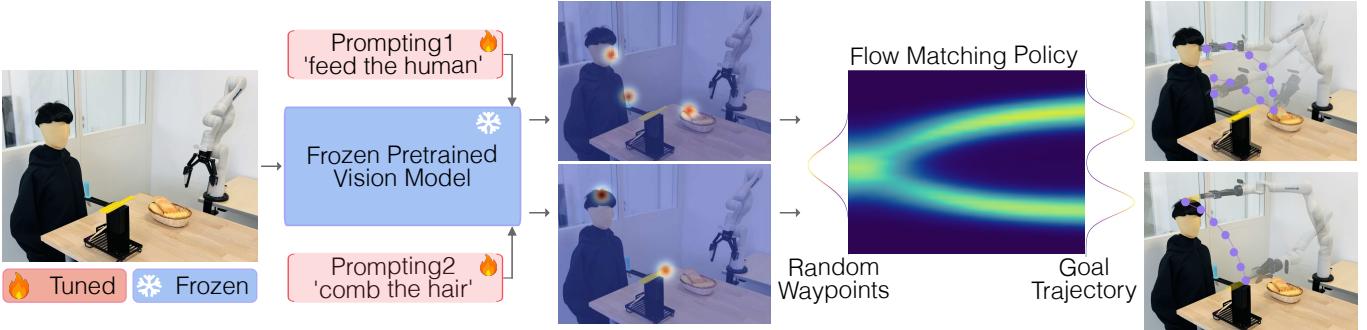


Fig. 1: The proposed framework of unifying affordance map learning and action generation for robot manipulation. Given the same visual scene with different language instructions, the model first extracts instruction-relevant manipulation affordances. This is achieved through a prompt tuning method that prepends learnable text-conditioned prompts in a frozen vision foundation model. Then a flow matching policy is proposed to transform the random waypoints to the desired action trajectories, guided by task-relevant affordance maps.

and Denoising Diffusion Implicit Models (DDIM [44]) to generate motion trajectories to capture multimodal action distributions. Flow Matching is another novel generative method. Sharing theoretical similarities with diffusion process, flow matching aims to regress onto a deterministic vector field to flow samples toward the target distribution. [27] has proven that the simplicity of flow matching objectives allows favorable performance in stable training and generation quality compared to solving complex stochastic differential equations in stochastic denoising diffusion process. We extend flow matching to the robotics domain. As shown in Fig. 1, the proposed method would flow the random waypoints to the desired action trajectories based on multi-task affordances in a single flow matching policy.

We also construct a Real-world Activities of Daily Living (ADLs) dataset with 10 tasks. The novelty of our dataset is that it contains the same scenarios but with multi-task affordance and robot trajectories. Experimental evaluation on our dataset empirically demonstrates that the prompt tuning method for learning affordances achieves performance competitive, and sometimes beyond other finetuning protocols across data scales, vision-language fusion architectures, and prompt variants. Furthermore, we showcase that across several benchmarks, flow matching attains favorable performance in training stability, generation quality, and computational efficiency amongst competing methods of behavior cloning.

This is the first attempt to ground VLM-based affordance with flow matching for real-world robot manipulation. We have also first systematically evaluated robot manipulation with flow matching on several benchmarks, including various input representations, robot control types, and manipulation tasks. Our code has been released <https://github.com/HRI-EU/flow-matching>.

The main contributions can be summarized as follows:

- 1) A parameter-efficient prompt tuning method for adapting pretrained vision foundation model conditioned on language instructions to learn manipulation affordances.
- 2) A novel formulation using flow matching for closed-

loop 6D robot manipulation learning from various inputs, including visual affordances.

- 3) Empirical and extensive results show that flow matching leads to consistently favorable results than some alternative behavior cloning methods in various manipulation tasks. This includes **more stable training and evaluation, and noticeably faster inference than diffusion policy, while maintaining comparable generalization performance to DDIM and slightly better performance than DDPM**.

Note that our goal is not to achieve the state-of-the-art general robot manipulation performance, but instead to broadly explore a new paradigm of efficiently adapting VLMs for affordance learning, and robot policy for multimodal action distributions.

## II. RELATED WORK

### A. Robot Learning from Demonstration

Imitation learning has been a common paradigm for robots which requires simulated or real-world demonstration data collection [50]. To improve data efficiency, extensive works have been proposed to learn robot policies on the top of visual representations [28], such as keypoints or affordance heatmaps [28], instead of end-to-end raw images [13]. This paper concentrates on using affordances to guide the low-level robot manipulation. In terms of network architectures for robot learning, prior works have successfully investigated convolutional networks [49], Transformers [45], generative adversarial networks [18], Energy-Based Models [7], etc. However, the collected data is usually expected to be non-convex and multi-modal due to the variability in human demonstrations. Recent works have addressed this problem by reformulating the robot policy as a generative process. Diffusion policy [3] has emerged as a powerful class of generative models for behavior cloning by representing a robot's visuomotor policy as a conditional denoising diffusion process. In this work, we investigate flow matching [27], a novel generative model that has demonstrated its superiority in image generation, but is much less explored in robotics domains.

## B. Parameter-Efficient Finetuning

Instruction-aware vision encoding [15] has been extensively studied for language-vision fusing tasks [39]. Given the dominance of large-scale vision-language models, many approaches have been proposed to efficiently finetune a frozen pretrained model for downstream tasks to speed up training and reduce memory. Two representative methods among them are adapters and prompting. The first line of research varies depending on the adapter that could add extra lightweight modules [10, 20]. Other work focuses on prompting [29], which originally primes a frozen pretrained language model for downstream tasks by including a hard text prompt. Recent works on prompt tuning [30] treat soft prompts as continuous vectors and compute their gradients with backpropagation during training. The extension of prompt tuning to vision tasks has gained massive success. Visual prompt tuning [24] has manipulated visual prompts to steer models in arbitrary vision tasks. [43] have explored extending the random learnable prompts to condition-based prompts that are less universal but more accurate. Inspired by its recent success, we extend the prompt tuning technologies to address the challenge of adapting large pretrained vision-language models to affordance learning for robot manipulation. The intuition is clear: if the model understands the posed text instruction and the inherent context, it should extract visual affordances that directly correspond to the relevant image aspects. Our method achieves the above goal by integrating learnable text-conditioned prompts into a large vision encoder, while keeping it frozen to preserve visual understanding capabilities.

## C. Flow Matching in Robotics

Despite its recent progress in image generation [1], the application of flow matching in robotics domains remains underexplored [21, 41]. Few prior studies have concentrated exclusively on certain robot scenarios for deploying flow matching, for pointcloud environment [4], Riemannian manifold [5], etc. We propose to use flow matching to learn multi-task robot behaviors from raw observations, including visual affordances obtained from a vision-language model, in a single supervised policy. We have first systematically evaluated robot manipulation with flow matching on several benchmarks, including various input representations, robot control types, and manipulation tasks.

## III. METHODS

### A. Prompt Tuning for Affordance Map Learning

Providing any type of pre-trained vision transformer, our objective is to learn a set of text-conditioned prompts to maximize the likelihood of correct affordance labels, as shown in Fig. 2. Only the prompt-related layers and the decoder are being updated during the training, while the vision transformer remains frozen. Inspired by Vision Prompt Tuning [24], we propose shallow and deep network architectures.

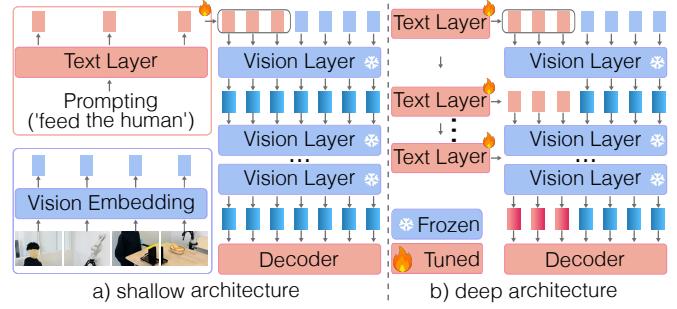


Fig. 2: Overview of prompt tuning structures used for affordance learning. (Left) For the shallow structure, text-conditioned prompts are prepended to the first vision transformer layer. (Right) For the deep structure, prompts are inserted to every vision layer. Only the prompt-related layers and the decoder are being updated during the training, while the vision transformer remains frozen.

*1) Shallow Architecture:* The vision transformer layer takes the image patch embeddings  $E_0$  as input and passes through various layers  $L_i^v$  to achieve vision features  $E_i$ , where  $E_i \in \mathbb{R}^{M \times C}$  and  $C$  is the channel dimension.

$$E_i = L_i^v(E_{i-1}) \quad i = 1, 2, \dots, N$$

Similarly, the text transformer layer could be represented as

$$P_i = L_i^p(P_{i-1}) \quad i = 1, 2, \dots, N$$

where  $P_0$  denotes the text tokens, text features  $P_i$  are obtained through various layers  $L_i^p$ , where  $p_i \in \mathbb{R}^{K \times C}$ .

As shown in Fig. 2, for the shallow structure, only one text transformer layer is used to compute text features  $P_1$ , which are then treated as prompts and inserted into the first vision transformer Layer:

$$\begin{aligned} [Z_1, E_1] &= L_1^v([P_1, E_0]) \\ [Z_i, E_i] &= L_i^v([Z_{i-1}, E_{i-1}]) \end{aligned}$$

Then a decoder is added on the global output flattened token sequence to generate visual affordance tokens.

$$\text{Affordance} = \text{Decoder}(Z_N, E_N)$$

*2) Deep Architecture:* For the deep architecture, the only difference is that text features  $P_i$  are computed through each layer and introduced at the corresponding vision transformer layer's input space:

$$\begin{aligned} [_, E_1] &= L_1^v([P_1, E_0]) \\ [_, E_i] &= L_i^v([P_i, E_{i-1}]) \end{aligned}$$

*3) Implementation Details:* Our goal is to integrate textual representations into any vision encoder while keeping it frozen, preserving its visual understanding capabilities. Thus, we have chosen the most basic vision backbone, a pretrained ViT-B-16 transformer. The text layers adopt the classic CLIP setup [39], including Multiheaded Self-Attention, Feed-Forward Networks with LayerNorm and residual connections, and output 76 tokens.

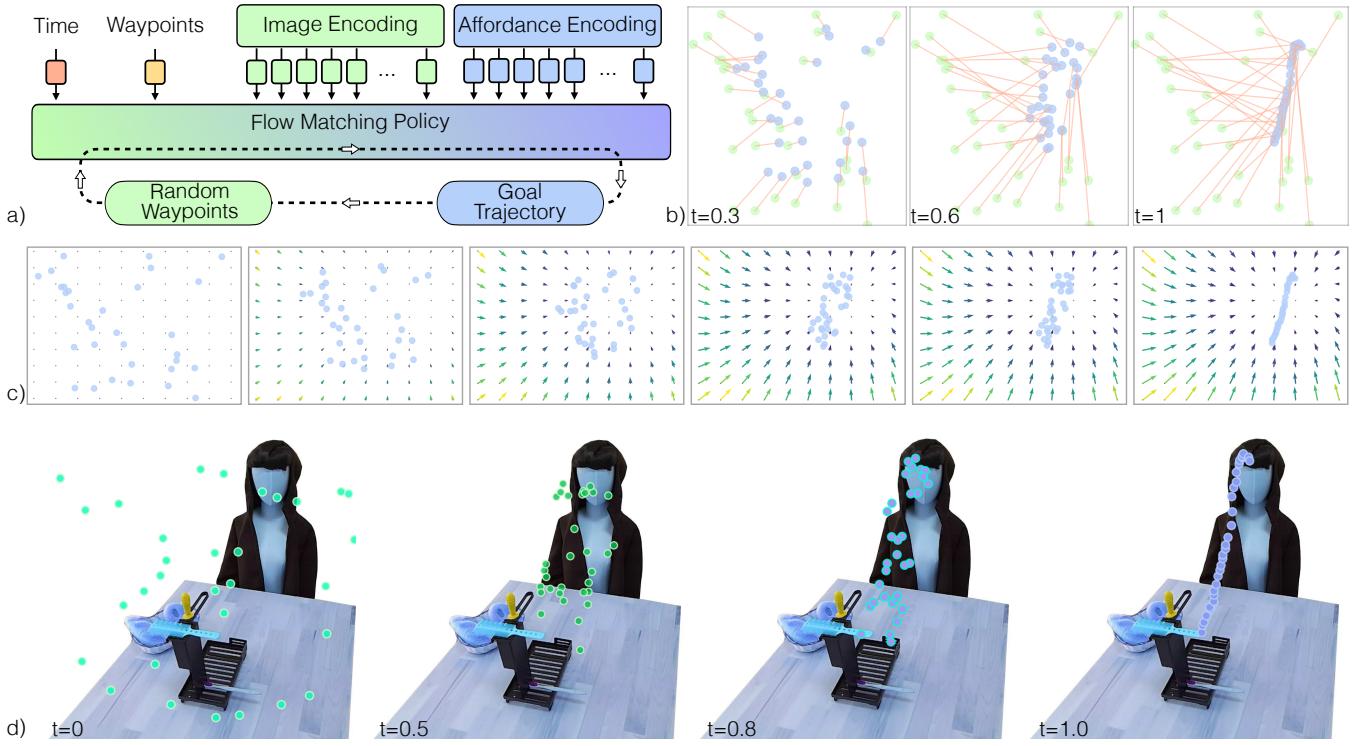


Fig. 3: Framework of flow matching policy. (a) General formulation. At each time step, flow matching takes visual observation  $\mathbf{o}$  (e.g., state-based inputs, RGB-D images, visual affordances) as input, and outputs robot actions (e.g., 6D robot end-effector trajectories, robot joint actions, gripper actions). (b-d) Visualization of the inference process of transforming random waypoints to target actions over time from 0 (green) to 1 (purple). Green lines in (b) denote the flow paths. Vector fields are shown in (c).

As suggested by MAE [17], the decoder is only used for downstream tasks and could be flexible and lightweight. Thus we use one single transformer decoder layer.

We adopt the L2 Mean Squared loss between the predicted and ground truth affordances for network training. The training parameters for the prompt tuning network include an image size of  $224 \times 224$ , AdamW with a learning rate of  $1.5e-4$  including Warmup with step-decay, and batch size of 256. We add positional embeddings to all the image and language tokens to preserve the positional information. In the subsequent experiments, we will further ablate multiple model variants, including text and vision fusion structures, prompt depth, pretrained weights for vision transformers, etc.

### B. Flow Matching Policy

We build the robot behavioral cloning policy as a generative process of flow matching, which constructs a flow vector that continuously transforms a source probability distribution toward a destination distribution. Flow matching leverages an ordinary differential equation to deterministically mold data distribution, contrasting with Denoising Diffusion Probabilistic Models (DDPM) that is based on a stochastic differential equation through introducing noise.

1) *Flow Matching Model:* Given a conditional probability density path  $p_t(\mathbf{x}|\mathbf{z})$  and a corresponding conditional vector field  $\mathbf{u}_t(\mathbf{x}|\mathbf{z})$ , the objective loss of flow matching could be

described as:

$$\mathcal{L}_{\text{FM}}(\boldsymbol{\theta}) = \mathbb{E}_{t, q(\mathbf{z}), p_t(\mathbf{x}|\mathbf{z})} \|\mathbf{v}_t(\mathbf{x}, \boldsymbol{\theta}) - \mathbf{u}_t(\mathbf{x}|\mathbf{z})\|^2 \quad (1)$$

where  $\mathbf{x} \sim p_t(\mathbf{x}|\mathbf{z})$ ,  $t \sim \mathcal{U}[0, 1]$ . Flow matching aims to regress  $\mathbf{u}_t(\mathbf{x}|\mathbf{z})$  with a time-dependent vector field of flow  $\mathbf{v}_t(\mathbf{x}, \boldsymbol{\theta})$  parameterized as a neural network with weights  $\boldsymbol{\theta}$ .  $\mathbf{u}_t(\mathbf{x}|\mathbf{z})$  can be further simplified as:

$$\mathbf{u}_t(\mathbf{x}|\mathbf{z}) = \mathbf{x}_1 - \mathbf{x}_0 \quad \mathbf{x}_0 \sim p_0, \mathbf{x}_1 \sim p_1$$

$p_0$  represents a simple base density at time  $t = 0$ ,  $p_1$  denotes the target complicated distribution at time  $t = 1$ ,  $\mathbf{x}_0$  and  $\mathbf{x}_1$  are the corresponding samplings.  $\mathbf{v}_t(\mathbf{x}, \boldsymbol{\theta})$  could be described as:

$$\mathbf{v}_t(\mathbf{x}, \boldsymbol{\theta}) = v_{\boldsymbol{\theta}}(\mathbf{x}_t, t) \quad (2)$$

where we define  $\mathbf{x}_t$  as the linear interpolation between  $\mathbf{x}_0$  and  $\mathbf{x}_1$  with respect to time  $\mathbf{x}_t = t\mathbf{x}_1 + (1-t)\mathbf{x}_0$ , following the Optimal Transport theory [37]. And  $v_{\boldsymbol{\theta}}$  is a network of the flow model. Thus Equation (1) could be reformatted as

$$\mathcal{L}_{\text{FM}}(\boldsymbol{\theta}) = \mathbb{E}_{t, \sim p_0, \sim p_1} \|v_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - (\mathbf{x}_1 - \mathbf{x}_0)\|^2 \quad (3)$$

This represents the progression of the scalar flow that transforms data from source to target between time 0 and 1.

---

**Algorithm 1** Robot Flow Matching Policy Training

---

**Input:** observation  $\mathbf{o}$ , target robot actions  $\mathbf{x}_1$ , source random waypoints  $p_0$   
**Output:** flow  $\mathbf{v}_\theta$

- 1: **while** not converged **do**
- 2:    $\mathbf{x}_0 \sim p_0$ , sample random robot waypoints
- 3:    $t \sim \mathcal{U}[0, 1]$ , sample time steps
- 4:    $\mathbf{x}_t = t\mathbf{x}_1 + (1-t)\mathbf{x}_0$ , linear interpolation
- 5:    $\mathbf{v}_t(\mathbf{x}|\mathbf{o}) = v_\theta(\mathbf{x}_t, t|\mathbf{o})$ , flow estimation
- 6:    $\nabla_\theta \|\mathbf{v}_\theta(\mathbf{x}_t, t|\mathbf{o}) - \mathbf{x}_t\|$ , gradient step
- 7: **end while**
- 8: Stopping criteria: training epochs reached

---

2) *Flow Matching for Visuomotor Policy Learning*: We extend flow matching to learn robot visuomotor policies. This requires two modifications in the formulation: i) modeling the flow estimation conditioned on input observations  $\mathbf{o}$ ; ii) changing the output  $\mathbf{x}$  to represent robot actions. Fig. 3 illustrates our model structures.

**Visual observation Conditioning:** We modify Equation (2) to allow the model to predict actions conditioned on observations:

$$\mathbf{v}_t(\mathbf{x}|\mathbf{o}) = v_\theta(\mathbf{x}_t, t|\mathbf{o})$$

**Closed-loop action trajectory prediction:** We execute the action trajectory prediction obtained by our flow matching model for a fixed duration before replanning. At each step, the policy takes the observation data  $\mathbf{o}$  as input and predicts  $T_p$  steps of actions, of which  $T_a$  steps of actions are executed on the robot without re-planning.  $T_p$  is the action prediction horizon and  $T_a$  is the action execution horizon. The whole training process of flow matching is illustrated in Algorithm 1.

**Inference:** For the inference procedure, random waypoints are sampled from the source distribution and then flowed into the target trajectory by estimating the flow from  $t = 0$  to  $t = 1$  over steps. We could adopt multiple steps  $1/\Delta t$  for inference:

$$\mathbf{x}_{t+\Delta t} = \mathbf{x}_t + \Delta t f(\mathbf{x}_t, t|\mathbf{o}), \quad \text{for } t \in [0, 1] \quad (4)$$

3) *Implementation Details*: For the network structures of flow matching, we first adopt ResNet [16] for visual embeddings  $\mathbf{o}$ . The flow model  $f_\theta$  is represented with U-Net [40]. The flow model predicts vectors  $\mathbf{v}_t$  conditioned on visual observation embeddings  $\mathbf{o}$  with Feature-wise Linear Modulation (FiLM) [36] as well as the interpolated waypoints  $\mathbf{x}_t$ . In the subsequent experiments, we will further study multiple model variants, including transformer-based structure, trajectory representation, etc.

In our case of robot manipulation,  $\mathbf{x}_1$  in Equation (3) represents the demonstration robot action trajectories.  $\mathbf{x}_0$  is the random generated waypoints following a multivariate normal distribution  $\mathbf{x}_0 \sim \mathcal{N}(0, I)$ .  $\mathbf{x}$  here could denote 6D robot end-effector trajectories, robot joint actions, gripper actions, etc. The visual embeddings  $\mathbf{o}$  include various types of inputs, such as state-based inputs, RGB-D images, and visual affordances.

---

**Algorithm 2** Robot Diffusion Policy (DDPM) Training

---

**Input:** observation  $\mathbf{o}$ , target robot actions  $\mathbf{x}_1$ , source Gaussian noises  $p_0$   
**Output:** noise  $\epsilon_\theta$

- 1: **while** not converged **do**
- 2:    $\mathbf{x}_0 \sim p_0$ , sample Gaussian noises
- 3:    $t \sim \mathcal{U}[0, 1]$ , sample time steps
- 4:    $\mathbf{x}_t = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t} \mathbf{x}_0, (1 - \bar{\alpha}_t) \mathbf{I})$ , forward process
- 5:    $\epsilon_t(\mathbf{x}|\mathbf{o}) = \epsilon_\theta(\mathbf{x}_t, t|\mathbf{o})$ , noise estimation
- 6:    $\nabla_\theta \|\epsilon_\theta(\mathbf{x}_t, t|\mathbf{o}) - \epsilon_t\|$ , gradient step
- 7: **end while**
- 8: Stopping criteria: training epochs reached

---

4) *Comparisons against Diffusion Policy*: In this section, we provide some insights and intuitions about flow matching and its comparisons against diffusion policy for clarification. Algorithm 1 and Algorithm 2 respectively shows the pseudocode of training flow matching and diffusion policy with DDPM. We can see several differences between these two methods:

- By solving a Stochastic Differential Equation (SDE), DDPM generates a clean sample from Gaussian noise. Flow matching regresses onto a target flow vector field that generates a deterministic mapping from source to target data distributions by solving an ordinary differential equation (ODE).
- DDPM reverses a diffusion process that adds noise to a clean sample until it becomes Gaussian noise. Flow matching drops all the Gaussian assumptions.
- Flow matching includes a particularly interesting family of probability paths: the vector field that corresponds to the principle of Optimal Transport with linear interpolant. Flow matching paths with linear interpolation are simpler than diffusion paths, forming straight-line trajectories whereas diffusion paths result in curved paths. These properties seem to empirically translate to more stable training, faster generation, and better performance.

### C. Activities of Daily Living Dataset

We construct a real-world dataset with 10 tasks across Activities of Daily Living. Each task includes 1,000 sets of RGB-D images, demonstrated robot action trajectories, and labeled ground truth of affordance maps. Thus 10,000 demonstrations have been collected in total. The data has been manually collected by moving robot end-effectors with kinesthetic teaching. The novelty of our dataset includes: (i) It contains the same scenarios with multiple objects, multi-task affordances, and the demonstrated robot trajectories. (ii) All tasks are related to Activities of Daily Living that involve (simulated) human data.

We label the affordance heatmaps with 2D Gaussian blobs centered on the object pixels of the demonstrated action. The affordance maps model the locations of all relevant object areas that physically interact with robots given each task. For example, 'feeding task' requires affordance heatmaps centered on the fork handle, food, and human mouth.

Objects are randomly placed within the range of the table (1.5 meters  $\times$  1 meter). The human (manikin) is placed randomly around the table in the camera view. We have around 30 different objects. Our tasks include prompt primitives: ‘sweep the trash’, ‘pass the water to the human’, ‘hang the towel’, ‘put on the hat’, ‘cover the food’, ‘wipe the nose’, ‘wipe the forearm’, ‘feed the human’, ‘comb the hair’, and ‘brush the teeth’. We can also add a pretrained LLM layer (e.g., GPT) in the very front for zero-shot text classification, allowing for linking other language instructions to one of the ten prompt primitives. The camera is positioned with some variations but generally oriented towards the table and objects. Please refer to the supplementary materials for more videos of each task.

#### IV. EXPERIMENTS

We systematically evaluate the proposed prompt tuning and flow matching methods against baseline studies. We also ablate how design choices would affect their performance.

##### A. Affordance Evaluation with Prompt Tuning

*1) Baseline Studies:* We benchmark our proposed prompt tuning structures against several commonly used finetuning and instruction-aware vision encoding protocols:

- Full finetuning: fully update the text and vision transformer layers and the decoder.
- Adapter-based methods: insert MLP layers with residual connections between pretrained frozen transformer layers of vision and language, as customary in the literature [10].
- Decoder-based methods: These methods treat the pretrained backbone as a feature extractor with fixed weights during tuning, and only the decoder is tuned, as customary in the literature [16].
- Side-network methods: train a text transformer network on the side and append pretrained vision features and sidetuned text features before being fed into the decoder, as customary in the literature [9].
- Cross-attention methods: Similarly to the above side-network methods, the difference here is using cross-attention fusing instead of simple prepending. An example of cross-attention fusing vision and language could be found in the literature [25].

For a fair comparison, all the baselines here use self-supervised pretrained MAE weights on ImageNet-21k dataset for the vision transformer model. We randomly split our Real-world Activities of Daily Living (ADLs) dataset with 80%-20% percentage of training and testing. The results reported here are obtained after 1,000 epochs of training.

*2) Main Results:* Table I presents the results of prompt tuning on our ADLs testing dataset for affordance learning, comparing against baselines. We use two metrics to evaluate our results: (i) L2 error of affordance heatmap estimation, and (ii) L2 distance between the predicted and ground truth of heatmap centers. We fit Gaussian Mixture Models on predicted heatmaps to determine the inferred heatmap centers. The heatmap error is averaged on each map, and the center error is averaged on per center point. Three observations could be made:

	Methods	Learnable Params (M) ↓	Affordance Heatmaps ( $\times 10^{-3}$ ) ↓	Heatmap Centers (pixel) ↓
Baselines	Full	153.8	<b>0.76</b>	<b>1.15</b>
	Decoder	3.9	1.51	13.48
	Adapter	19.2	1.17	6.22
	Cross-attention side-network	43.5	1.26	8.89
Ours	PT-shallow	8.0	1.42	12.04
	PT-deep (self-supervised weights)	42.1	<b>0.80</b>	<b>2.93</b>
Ablations	PT-deep (supervised weights)	42.1	1.48	10.13
	PT-deep (image output)	42.1	1.56	13.27

TABLE I: Results of prompt tuning baseline and ablation studies. We report the number of learnable parameters, the heatmap estimation error (the fourth column) and the heatmap center error (the fifth column). Our method outshines other baselines except for the full finetuning.

• **Prompt tuning against full finetuning** The deep structure of prompt tuning outperforms other baselines except for full finetuning. Full finetuning slightly outperforms deep prompt tuning in terms of heatmap estimation error and heatmap center error. However, the distinction of heatmap center errors (1.78 pixels) remains subtle, given the full image size of 224  $\times$  224. This outcome is favorable as it indicates that most heatmap errors are caused by the tails of the Gaussian distribution, instead of the center area where the robot actions actually applied on. We will further ablate the impact of dataset size on these two methods.

• **Generalizability:** We also observe that the trained model could be generalized to new objects. For example, the training dataset only includes a manikin. We found out that it generates well on our testing data with real humans. Affordances on objects with similar shapes (e.g., forks and spoons) could also be transferred. Note that as the proposed tuning method is parameter-efficient, it is envisaged that the method could be readily transferred to different tasks with a small amount of task-specific data.

• **What do prompts learn?** We show a t-SNE [46] visualization of the embeddings after the last vision transformer layer (before the decoder) in Fig. 4. We can see that the points of the same color (e.g., tasks with the same language prompts) are embedded together, implying that the representations recover the underlying manifold structure of discriminative task information.

3) *Ablations:* We further ablate model design choices:

**Pretrained Weights:** We evaluate using MAE self-supervised pretrained weights and supervised pretrained weights trained on ImageNet-21k dataset for the vision transformer model. The results in Table I show self-supervised pretrained weights perform better. We are aware of other more complicated variants of vision transformers, for example, CLIP vision encoder and its pretrained weights. As our goal is to integrate textual representations into any vision encoder while keeping it frozen, we have chosen the most basic ViT-B-16 transformer backbone and commonly used pretrained weights and achieved competitive results. We have also tried GPT directly to infer affordance, but no constant performance was achieved without carefully designed prompts or finetuning.

**Decoder Input:** We apply the decoder on the global output

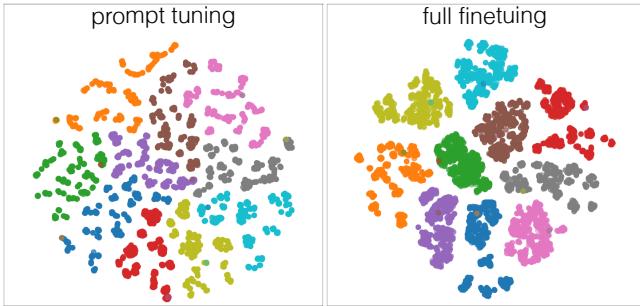


Fig. 4: t-SNE visualizations of the embeddings before the decoder. The points of the same color denote the tasks with same language prompts, which are embedded together. The prompt tuning method could produce instruction-relevant features without updating vision backbone parameters.

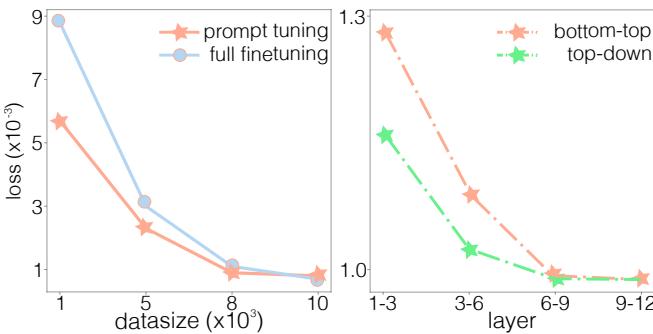


Fig. 5: Ablation studies of prompt tuning. We investigate the effect of various design choices on affordance learning performance, including pretrained weights, decoder input, dataset size and prompt location.

and image-corresponding output after the vision transformer respectively and report results in Table I.

**Dataset Size:** We use various amounts of data for training. Fig. 5-left shows that prompt tuning has better adaptability than full finetuning when downstream data is scarce.

**Prompt Location:** We have seen different conclusions from prior works about whether the vision-language fusion should be integrated at early or late transformer layers. We conduct experiments to insert prompts at various layers. From Fig. 5-right, we can see that inserting prompts to early layers (for example, layer 1-3 from bottom to top) achieves higher loss than inserting to late layers (for example, layer 1-3 from top to bottom). Thus in our case, prompts have greater significance at the late transformer layers. These results are also supported by the nature of vision transformer hierarchy: lower layers mainly capture low-level fundamental visual details, while higher layers focus on high-level concepts that might be vital for downstream tasks.

In conclusion, we observe no single method that outperforms all the rest. For scenarios where a small number of parameters or datasize is available, we reckon that prompt tuning remains the preferred approach.

Methods	2D Trajectory	3D Trajectory	Inference
	Prediction (pixel) ↓	Prediction (cm) ↓	Times (ms) ↓
Ours	Flow Matching (Transformer)	2.151	4.677
	Flow Matching (CNN, 16-step)	<b>0.840</b>	<b>1.009</b>
Ablations	Flow Matching (CNN, 1-step)	0.888	1.031
	Flow Matching (CNN, 4-step)	0.846	1.014
	Flow Matching (CNN, 8-step)	0.842	1.013
	Flow Matching (without affordance)	1.107	13.450
Baselines	DDPM (1-step)	2.851	13.791
	DDPM (4-step)	0.890	2.411
	DDPM (8-step)	0.884	2.403
	DDPM (16-step)	0.882	2.398
	DDIM (1-step)	4.328	10.78
	DDIM (4-step)	0.876	2.299
	DDIM (8-step)	0.875	2.272
	DDIM (16-step)	0.874	2.266
Transformer-based BC		2.797	4.911
			7.59

TABLE II: Results of flow matching policy against baselines and ablations. We report the average error of 2D and 3D trajectory estimation and the inference time. Our flow matching method achieves the best trajectory estimation accuracy. We also investigate the effect of various design choices on flow matching performance, including network structures, training, and inference steps.

### B. Flow Matching Policy Evaluation

**1) Baseline Studies:** We compare our flow matching policy against: (i) Diffusion Policy [3] with DDPM and DDIM, and (ii) Transformer-based behavior cloning, as customary in RVT [13], RT-X [35].

Some previous research [34, 11] has mathematically proven that DDIM trajectories are equivalent to flow matching trajectories with a different scaling of time. This scale difference may be beneficial for flow matching. In this experiment, we aim to practically evaluate such speculation. Table V shows the hyperparameters used in flow matching and diffusion policy.

Note that we are aware of other competitive robot behavior cloning methods, including energy-based IBC [7], GAIL [18], etc. Since extensive studies have been conducted and showed better performance of the diffusion policy against these methods, we choose the representative transformer and diffusion baselines for evaluation.

We first train and test flow matching and baselines on our collected Real-world Activities of Daily Living (ADLs) dataset in a supervised manner with Mean Square Error Loss. In the following sections, we will also evaluate on other benchmarks (e.g., Push-T, Franka Kitchen, and Robomimic).

**• Real-world Activities of Daily Living (ADLs) dataset:** For 2D data, the training uses a RGB image with visual affordances as input, and the output is a trajectory in 2D pixel space. For the counterparts in 3D, the training takes the concatenation of RGB-D images with visual affordances as input, and outputs a trajectory in 3D Cartesian space. We randomly split the dataset with 80%-20% percentage of training and testing. The results reported are obtained after 1,000 epochs of training. For each baseline, all 10 tasks in our dataset are trained in a single policy in a supervised manner. To ensure fair comparisons, the corresponding hyperparameters across the flow matching policy and baseline methods are selected to remain consistent. Table VI shows the task summary.

**2) Main Results:** Table II presents the results of flow matching policy on our ADLs testing dataset for robot trajectory

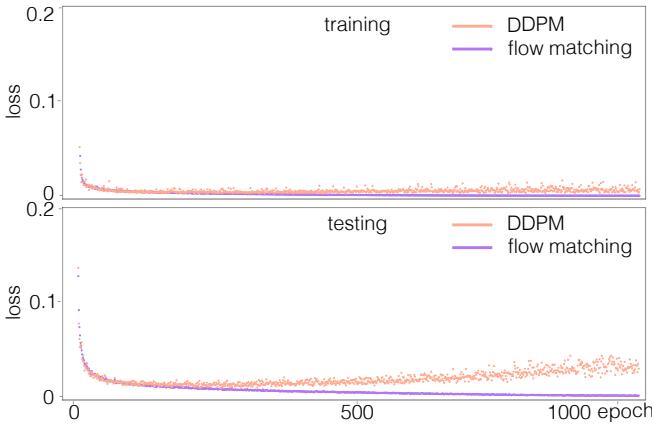


Fig. 6: Training and testing loss throughout the training process. Flow matching exhibits greater stability on both training and evaluation than diffusion policy.

learning, comparing against baselines. We use two metrics for evaluation: (i) the error of 2D and 3D trajectory estimation, and (ii) the inference time, performed with RTX 4090 GPU. The trajectory error is averaged on each point of the trajectory.

Four observations could be made from this result:

- **Generation Quality:** Flow matching (CNN-based, 16 steps) outperforms diffusion policy and Transformer baselines in terms of 2D and 3D trajectory prediction accuracy. The Transformer behavior cloning achieves marginal precision. This is expected as it is hindered by the nature of multi-modal action distribution, causing the averaging out across non-convex spaces.
- **Stability:** Fig. 6 shows an example of training and testing loss of flow matching and diffusion policy with DDPM throughout the training process. We can see **flow matching exhibits greater stability on both training and evaluation than diffusion policy**.

• **Inference Time:** We have two observations here: i) Table II showcases that flow matching with 16 steps achieves faster inference time compared to diffusion policy with 16 steps. We hypothesize that flow matching with linear pointwise flows generates straighter flows than DDPM and DDIM, and thus causes faster inference. ii) More importantly, Table II also showcases that 1-step flow matching (error: 1.031cm, time: 13.228ms) has achieved comparable performance as 16-step DDIM (error: 2.266cm, time: 98.680ms), but **noticeably lowered inference time roughly by 85%**. We hypothesize that this is because diffusion models solve a stochastic differential equation with a series of discrete steps to progressively refine the generated sample. Contrarily, flow matching trains continuously normalize flow models, leading to no significant improvements when increasing inference steps. Thus 2-step flow matching has achieved comparable performance as 16-step diffusion policy, which considerably reduces the inference time for closed-loop robot manipulation. This is in line with some results obtained in the image generation domain. As pointed out by Stable Diffusion 3 [6], flow matching performs better with fewer inference steps than diffusion policy, especially

when faster inference is required. The extensive evaluation on multiple robot benchmarks in the next section will further reinforce our argument.

• **Training Resources:** DDPM training and benchmarking demand significant resources for various training and inference steps. DDIM decouples the number of denoising iterations in training and inference, thereby allowing the algorithm to be trained one time with a large training iteration and use fewer iterations for inference to speed up the process. However, flow matching still achieves faster inference than DDIM.

3) *Real-World Robot Experiments:* We deploy flow matching, DDPM and Transformer policies on real robot manipulation for evaluation. We carry out 50 replications of trials for each baseline. We use a KINOVA Gen3 arm and an Azure Kinect camera for real-world robot experiments. Details can be found in the supplementary video.

From Table III, we can see that flow matching outperforms other baselines. We also observe that most failures occurred due to some out-of-distribution factors, including background and significant camera view changes. Larger data size and more diversity should alleviate this problem.

**RGB images with/without affordances:** We also investigate how the affordance would guide the flow matching policy. We trained flow matching taking the raw RGB images and language tokens as input. This is similar to our proposed method but without the intermediate stage of affordance learning. It also involves some resemblance while not being entirely identical to the method in [2]. Interestingly, comprehensive examinations reinforce the argument that flow matching could handle multimodal action distribution. Fig. 7 shows one example. From the left figure, we can see that when training a policy without affordances, the predicted trajectory (yellow) of moving the towel toward the trash for sweeping could be detached from the ground truth (red), but still a reasonable solution that allows for a successful robot execution. With affordance guidance, the prediction is closely aligned with the truth (Fig. 7-right). We also observe that for applications that demand higher precision in manipulation, like grasping the toothbrush handle, affordances offer greater guidance for shaping the manipulation policy.

4) *Ablations:* We further ablate policy design choices.

**Network Structure:** As shown in Table II, CNN-based flow matching achieves better results than transformer-based architecture. We hypothesize that transformer might need additional hyperparameter tuning.

**Trajectory Representation:** We empirically test trajectory representation with 8, 16, 32 and 64 waypoints. More waypoints are not necessary, while fewer waypoints are unable to entirely encapsulate the complete long-horizon trajectories. We have found that, in general, the trajectory representation does not wield a significant influence on flow matching performance. The performance reported in the above main results section is achieved by using 32 waypoints.

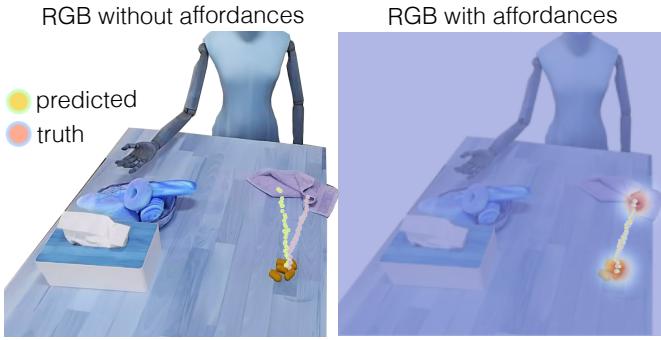
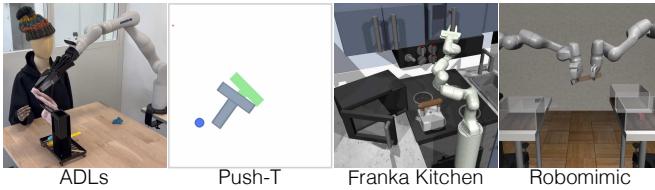


Fig. 7: Ablations of using RGB images with/without affordances for policy training. Visual affordance guides the flow matching policy to generate a trajectory closely aligned with the truth. The policy without affordances might generate a trajectory detached from the ground truth, but potentially still a reasonable solution. This reinforces the argument that flow matching policy could handle multimodal robot action distributions.

Methods (Inference Step)	Flow Matching (16-step) ↑	Diffusion Policy (16-step) ↑	Transformer BC ↑	Flow Matching RGB ↑
Activities of Daily Living	<b>0.82</b>	0.76	0.44	0.74

TABLE III: Real-world robot experimental results.



a sampling range:  $[(50, 450), (50, 450), (200, 300), (200, 300), (-\pi, \pi)]$

b sampling range:  $[(50, 450), (50, 450), (100, 400), (100, 400), (-\pi, \pi)]$

TABLE IV: We present the robot evaluation performance in the format of (max performance) / (average of last checkpoint with 10 trials of replications), with each averaged across 500 different environment initial conditions. The metric used here is success rate, except for the Push-T task which uses target area coverage. We have used various sampling ranges (end-effector position, T-block position and orientation) for Push-T environment initialization. For Robomimic benchmark, we specifically report results on the Transport task.

### C. Comparisons between flow matching and diffusion policy

To further investigate the performance of flow matching compared to diffusion policy, we benchmark the proposed methods on three more datasets which include closed-loop 6D robot actions and gripper actions: (i) Push-T [7], (ii) Franka Kitchen [14], and (iii) Robomimic [33].

- Push-T requires pushing a T-shaped block to a fixed target with a circular end-effector. Push-T takes RGB images with proprioception of end-effector location as inputs, and outputs

end-effector actions in a closed-loop manner. The dataset includes 200 demonstrations.

- *Franka Kitchen* contains 7 objects for interaction and comes with a human demonstration dataset of 566 demonstrations, each completing 4 tasks in arbitrary order. The goal is to execute as many demonstrated tasks as possible, regardless of order. The training takes state-based inputs, and outputs closed-loop robot joint actions and gripper actions.

- *Robomimic* consists of 5 tasks with a proficient human teleoperated demonstration dataset. We specifically focus on the transport task which includes 200 demonstrations. The policy takes state-based inputs, and outputs closed-loop robot joint actions and gripper actions.

For each benchmark, the evaluation has been carried out across 500 different environment initial conditions, using the last checkpoint of each policy with 10 trials of replications. Thus, 5,000 trials have been carried out in total per policy and benchmark. Variation is added on random initial conditions for the robot and object states. We respectively report the best and average performance in the 10 trials of replications of the last checkpoint. All state-based tasks are trained for 4,500 epochs, and image-based tasks for 3,000 epochs. Table V shows the hyperparameters we have used in flow matching and diffusion policy. Table VI shows the task summary.

Similar conclusions could be achieved from Table IV and Fig. 8, as in the Main Results section:

- **Generation Quality:** Flow matching outperforms DDPM in all three benchmarks. The performances of flow matching and DDIM are comparable, where flow matching performs marginally better in most cases.

- **Inference Time:** Fig. 8 shows how the number of inference steps affects the performance of flow matching and diffusion policy. We can observe that diffusion policy showcases better performances when applying more inference iterations with a trade-off of longer inference time, as it requires a series of discrete steps to progressively refine the generated sample. Contrarily, flow matching has not shown significant improvements when increasing inference steps. Therefore, flow matching considerably reduces the inference time for closed-loop robot manipulation. In the Push-T benchmark in Fig. 8, 2-step flow matching (coverage: 0.8803, time: 13.098ms) has achieved comparable performance as 16-step diffusion policy with DDIM (coverage: 0.8801, time: 98.268ms), **but noticeably lower inference time roughly by 86%**.

## V. LIMITATIONS

The generation quality of flow matching and diffusion policy for robot manipulation are generally comparable. Although we see only a marginal improvement in flow matching in most cases, we would like to highlight that the focus of this work is not to outperform state-of-the-art general robot manipulation research. Instead, we have systematically studied the flow matching framework, which provides an alternative to diffusion policies for robot manipulation. We can not overlook the additional advantages of flow matching, including

H-Param	Ta	Tp	ObsRes	F-Net	F-Par	V-Enc	V-Par	Lr	WDe	Iters Train (FM)	Iters Train (DDPM)	Iters Train (DDIM)	Iters Eval
Activities of Daily Living	32	32	1x224x224	ConditionalUnet1D	72	ResNet-18	11	1e-4	1e-6	N/A	1/4/8/16	16	1/4/8/16
Push-T	8	16	1x96x96	ConditionalUnet1D	80	ResNet-18	11	1e-4	1e-6	N/A	1/4/8/16	16	1/4/8/16
Franka Kitchen	8	16	1x60	ConditionalUnet1D	66	N/A	N/A	1e-4	1e-6	N/A	1/4/8/16	16	1/4/8/16
Robomimic	8	16	1x50	ConditionalUnet1D	66	N/A	N/A	1e-4	1e-6	N/A	1/4/8/16	16	1/4/8/16

TABLE V: Hyperparameters for flow matching and diffusion policy. Ta: action horizon. Tp: action prediction horizon. ObsRes: environment observation resolution. D-Net: diffusion/flow matching network. D-Par: diffusion/flow matching network number of parameters in millions. V-Enc: vision encoder. V-Par: vision encoder number of parameters in millions. Lr: learning rate. WDe: weight decay. Iters Train: number of training diffusion iterations. Iters Eval: number of inference iterations.

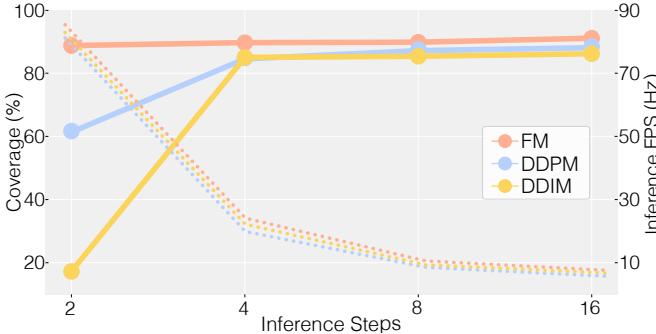


Fig. 8: Generation quality and inference time comparison of flow matching and diffusion policy for varying values of inference steps.

Tasks	Rob	Obj	ActD	PH	Steps	Img	Closed-loop
Activities of Daily Living	1	$\approx 30$	3	8,000	N/A	✓	✗
Push-T	1	1	2	200	300	✓	✓
Franka Kitchen	1	7	9	566	280	✗	✓
Robomimic	2	3	20	200	700	✗	✓

TABLE VI: Tasks Summary. Rob: number of robots. Obj: number of objects. ActD: action dimension. PH: proficient-human demonstration. Steps: max number of rollout steps. Franka Kitchen and Robomimic involve 6D robot and gripper actions in the joint space. ADLs and Push-T focus on robot end-effector trajectories. For clarity, we further explain that for our ADLs tasks, as no closed-loop motion is considered here, we assume that the gripper closes when the first waypoint has arrived, and the low-level actions are executed between waypoints using a standard proportional-derivative (PD) controller.

stable training, easy implementation, and most importantly, significantly better performance and faster inference with fewer inference steps than diffusion policy, suggesting forsaking the stochastic construction of diffusion policy in favor of learning the probability path more directly as in flow matching.

We have evaluated how varying inference steps affect the performance of flow matching and diffusion policy. The original diffusion policy research [3] adopts a large 100 training and inference steps with DDPM in their experiments. Based on our evaluation in Fig. 8 and results from other research [4], we can observe that beyond 8 steps, further increasing steps have only a marginal impact on the performance of diffusion policy, but with a trade-off of significantly longer inference time. We are aware of recent competitive research on one-step diffusion

policy with distillation [38] and shortcut models [8]. In this paper, we primarily focus on conducting a comparative analysis of the fundamental architectures underlying flow matching and diffusion policy with DDPM and DDIM.

We have not explored applying flow matching for robot action generation in the reinforcement learning setting. Recent research [51] has modeled a policy as a return-conditional flow matching model with classifier-free guidance to eliminate many of the complexities that come with traditional offline reinforcement learning and outperformed standard imitation learning. These methods are anticipated to be seamlessly adaptable to robotic manipulation scenarios; however, comprehensive validation through practical simulations and real-world experimentation remains imperative.

## VI. CONCLUSION

We have formulated a prompt tuning method for affordance map learning and flow matching policy for robot manipulation. The core idea of prompt tuning is to maximally exploit the pretrained foundation model, and rapidly excavate the relevance of foundation and downstream affordance learning tasks. We have proposed a flow matching policy constructing paths that allow faster inference, and improved generation amongst robot behavior cloning methods. We qualitatively and quantitatively experiment on multiple robot manipulation benchmarks to prove that flow matching produces better trade-offs between computational cost and sample quality compared to prior competing diffusion-based methods.

## REFERENCES

- [1] Michael S Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. *arXiv preprint arXiv:2209.15571*, 2022.
- [2] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolò Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.
- [3] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.
- [4] Eugenio Chisari, Nick Heppert, Max Argus, Tim Welscheshold, Thomas Brox, and Abhinav Valada. Learning robotic manipulation policies from point clouds

- with conditional flow matching. *arXiv preprint arXiv:2409.07343*, 2024.
- [5] Haoran Ding, Noémie Jaquier, Jan Peters, and Leonel Rozo. Fast and robust visuomotor riemannian flow matching policy. *arXiv preprint arXiv:2412.10855*, 2024.
- [6] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first International Conference on Machine Learning*, 2024.
- [7] Pete Florence, Corey Lynch, Andy Zeng, Oscar A Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. In *Conference on Robot Learning*, pages 158–168. PMLR, 2022.
- [8] Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. *arXiv preprint arXiv:2410.12557*, 2024.
- [9] Roy Ganz, Yair Kittenplon, Aviad Aberdam, Elad Ben Avraham, Oren Nuriel, Shai Mazor, and Ron Litman. Question aware vision transformer for multimodal reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13861–13871, 2024.
- [10] Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Qiao. Clip-adapter: Better vision-language models with feature adapters. *International Journal of Computer Vision*, 132(2):581–595, 2024.
- [11] Ruiqi Gao, Emiel Hoogeboom, Jonathan Heek, Valentin De Bortoli, Kevin P. Murphy, and Tim Salimans. Diffusion meets flow matching: Two sides of the same coin. 2024. URL <https://diffusionflow.github.io/>.
- [12] James J Gibson. *The ecological approach to visual perception: classic edition*. Psychology press, 2014.
- [13] Ankit Goyal, Jie Xu, Yijie Guo, Valts Blukis, Yu-Wei Chao, and Dieter Fox. Rvt: Robotic view transformer for 3d object manipulation. In *Conference on Robot Learning*, pages 694–710. PMLR, 2023.
- [14] Abhishek Gupta, Vikash Kumar, Corey Lynch, Sergey Levine, and Karol Hausman. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning. *arXiv preprint arXiv:1910.11956*, 2019.
- [15] Tanmay Gupta, Amita Kamath, Aniruddha Kembhavi, and Derek Hoiem. Towards general purpose vision systems: An end-to-end task-agnostic vision-language architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16399–16409, 2022.
- [16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [17] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.
- [18] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016.
- [19] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.
- [20] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.
- [21] Xixi Hu, Bo Liu, Xingchao Liu, and Qiang Liu. Adaflow: Imitation learning with variance-adaptive flow-based policies. *arXiv preprint arXiv:2402.04292*, 2024.
- [22] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023.
- [23] Nils Ingelhart, Jesper Munkeby, Jonne van Haastregt, Anastasia Varava, Michael C Welle, and Danica Kragic. A robotic skill learning system built upon diffusion policies and foundation models. *arXiv preprint arXiv:2403.16730*, 2024.
- [24] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European Conference on Computer Vision*, pages 709–727. Springer, 2022.
- [25] Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. Vima: General robot manipulation with multimodal prompts. *arXiv preprint arXiv:2210.03094*, 2(3):6, 2022.
- [26] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500. IEEE, 2023.
- [27] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.
- [28] Fangchen Liu, Kuan Fang, Pieter Abbeel, and Sergey Levine. Moka: Open-vocabulary robotic manipulation through mark-based visual prompting. *arXiv preprint arXiv:2403.03174*, 2024.
- [29] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35, 2023.
- [30] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Lam Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint*

- arXiv:2110.07602*, 2021.
- [31] Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. *arXiv preprint arXiv:2310.12931*, 2023.
  - [32] Martin Maier and Rasha Abdel Rahman. No matter how: Top-down effects of verbal and semantic category knowledge on early visual perception. *Cognitive, Affective, & Behavioral Neuroscience*, 19:859–876, 2019.
  - [33] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *arXiv preprint arXiv:2108.03298*, 2021.
  - [34] Preetum Nakkiran, Arwen Bradley, Hattie Zhou, and Madhu Advani. Step-by-step diffusion: An elementary tutorial. *arXiv preprint arXiv:2406.08929*, 2024.
  - [35] Abhishek Padalkar, Acorn Pooley, Ajinkya Jain, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anikait Singh, Anthony Brohan, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.
  - [36] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.
  - [37] Gabriel Peyré, Marco Cuturi, et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.
  - [38] Aaditya Prasad, Kevin Lin, Jimmy Wu, Linqi Zhou, and Jeannette Bohg. Consistency policy: Accelerated visuomotor policies via consistency distillation. *arXiv preprint arXiv:2405.07503*, 2024.
  - [39] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.
  - [40] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5–9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.
  - [41] Quentin Rouxel, Andrea Ferrari, Serena Ivaldi, and Jean-Baptiste Mouret. Flow matching imitation learning for multi-support manipulation. In *2024 IEEE-RAS 23rd International Conference on Humanoid Robots (Humanoids)*, pages 528–535. IEEE, 2024.
  - [42] Mohit Sridhar, Lucas Manuelli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning*, pages 785–799. PMLR, 2023.
  - [43] Kihyuk Sohn, Huiwen Chang, José Lezama, Luisa Polania, Han Zhang, Yuan Hao, Irfan Essa, and Lu Jiang. Visual prompt tuning for generative transfer learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19840–19851, 2023.
  - [44] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.
  - [45] Kaustubh Sridhar, Souradeep Dutta, Dinesh Jayaraman, James Weimer, and Insup Lee. Memory-consistent neural networks for imitation learning. *arXiv preprint arXiv:2310.06171*, 2023.
  - [46] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.
  - [47] Yen-Jen Wang, Bike Zhang, Jianyu Chen, and Koushil Sreenath. Prompt a robot to walk with large language models. *arXiv preprint arXiv:2309.09969*, 2023.
  - [48] Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked visual pre-training for motor control. *arXiv preprint arXiv:2203.06173*, 2022.
  - [49] Fan Zhang and Yiannis Demiris. Learning garment manipulation policies toward robot-assisted dressing. *Science robotics*, 7(65):eabm6010, 2022.
  - [50] Fan Zhang and Yiannis Demiris. Visual-tactile learning of garment unfolding for robot-assisted dressing. *IEEE Robotics and Automation Letters*, 2023.
  - [51] Qingqing Zheng, Matt Le, Neta Shaul, Yaron Lipman, Aditya Grover, and Ricky TQ Chen. Guided flows for generative modeling and decision making. *arXiv preprint arXiv:2311.13443*, 2023.
  - [52] Yun Zhong and Yiannis Demiris. Dancemvp: Self-supervised learning for multi-task primitive-based dance performance assessment via transformer text prompting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 10270–10278, 2024.
  - [53] Jiawen Zhu, Simiao Lai, Xin Chen, Dong Wang, and Huchuan Lu. Visual prompt multi-modal tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9516–9526, 2023.