Affordance-based Robot Manipulation with Flow Matching

Fan Zhang and Michael Gienger

***Abstract*—We present a framework for assistive robot manipulation that addresses two fundamental challenges: efficient adaptation of large-scale models for scene affordance understanding and effective learning of robot actions by grounding the visual affordance. To tackle the first challenge, we adopt a parameter-efficient prompt tuning method, prepending learnable text prompts to a frozen vision model to predict affordances, while considering spatial and semantic relationships in multi-task scenarios. For the second challenge, we propose a flow matching method, representing a robot visuomotor policy as a conditional process of flowing random waypoints to desired robot actions. We introduce a real-world dataset with 10 tasks to evaluate our approach. Experiments show our prompt tuning method achieves competitive or superior performance to other finetuning protocols across data scales, while satisfying parameter efficiency. Flow matching yields more stable training and faster inference, while maintaining comparable generalization performance to diffusion policy. Our framework seamlessly unifies parameter-efficient affordance learning and robot action generation with flow matching.**

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 [11] make the fewest assumptions on tasks and are formulated in language-image-to-action prediction models. 2) Off-the-shelf-vision-language models for robot manipulation. These works have prompted pre-trained VLMs in various contexts of robot motion learning, including reward design for reinforcement learning [10], python coding [8], joint actions [16], 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 to alleviate the sample inefficiency problem of end-to-end learning, including affordances [4], primitive skills [5], etc. In this paper, we follow the third line of work to unify a parameter-efficient affordance model and a low-level robot flow matching policy.

This work seamlessly grounds VLM-based affordance with flow matching for real-world robot manipulation. We leverage the capability of the flow matching policy to represent multi-modal action distributions for learning accurate 6D robot actions, while 2D affordance maps readily provide sufficient guidance in shaping the policy. We have further systematically evaluated robot manipulation with flow matching on several benchmarks, including various input representations, robot control types, and manipulation tasks. 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. Our code is publicly available.

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 and vision-language fusion architectures.

The authors are with Honda Research Institute EU. (e-mail: fan.zhang@honda-ri.de; michael.gienger@honda-ri.de).

II. 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 [6], we propose two frameworks: shallow and deep network architectures.

1) Shallow Architecture: The vision transformer layer takes the image patch embeddings $\boldsymbol{E}_0$ as input and passes through various layers $\boldsymbol{L}_i^v$ to achieve vision features $\boldsymbol{E}_i$, where $\boldsymbol{E}_i \in \mathbb{R}^{M\times C}$ and C is the channel dimension.

$$\boldsymbol{E}_i = \boldsymbol{L}_i^v(\boldsymbol{E}_{i-1}) \qquad i = 1, 2, \cdots, N$$

Similarly, the text transformer layer could be represented as

$$\boldsymbol{P}_i = \boldsymbol{L}_i^p(\boldsymbol{P}_{i-1}) \qquad i = 1, 2, \cdots, N$$

where $\boldsymbol{P}_0$ denotes the text tokens, text features $\boldsymbol{P}_i$ are obtained through various layers $\boldsymbol{L}_i^P$, where $\boldsymbol{p}_i \in \mathbb{R}^{K\times C}$.

As shown in Fig. 2, for the shallow structure, only one text transformer layer is used to compute text features $\boldsymbol{P}_1$, which are then treated as prompts and inserted into the first vision transformer Layer:

$$[\boldsymbol{Z}_1, \boldsymbol{E}_1] = \boldsymbol{L}_1^v([\boldsymbol{P}_1, \boldsymbol{E}_0])$$
$$[\boldsymbol{Z}_i, \boldsymbol{E}_i] = \boldsymbol{L}_i^v([\boldsymbol{Z}_{i-1}, \boldsymbol{E}_{i-1}])$$

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

$$\text{Affordance} = \text{Decoder}(\boldsymbol{Z}_N, \boldsymbol{E}_N)$$

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

$$[\_, \boldsymbol{E}_1] = \boldsymbol{L}_1^v([\boldsymbol{P}_1, \boldsymbol{E}_0])$$
$$[\_, \boldsymbol{E}_i] = \boldsymbol{L}_i^v([\boldsymbol{P}_i, \boldsymbol{E}_{i-1}])$$

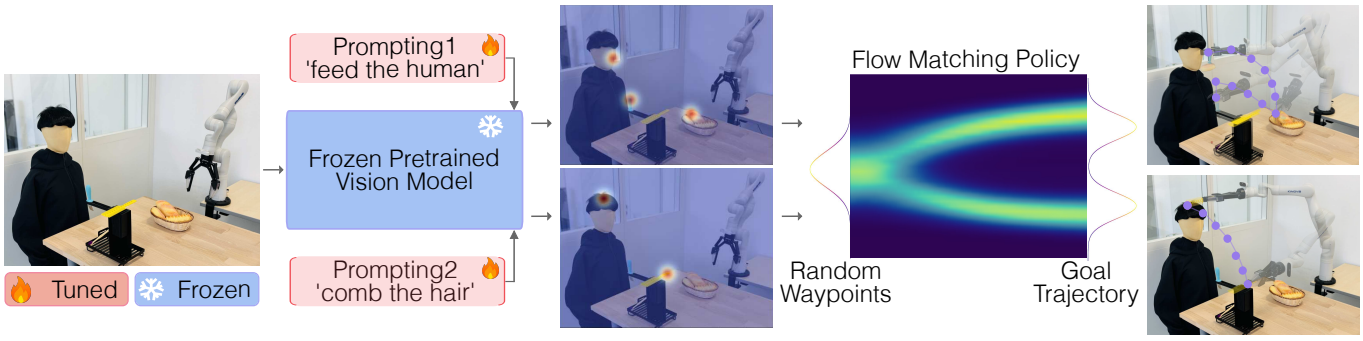

Fig. 1: The proposed framework of unifying affordance map learning and action generation for robot manipulation. Given the same visual scene with different language instructions, the model first extracts instruction-relevant manipulation affordances. This is achieved through a prompt tuning method that 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.

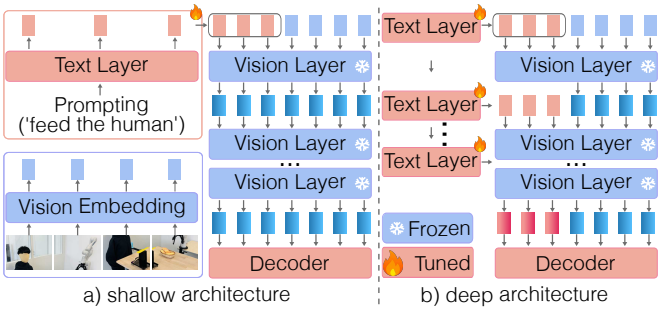

Fig. 2: Overview of prompt tuning structures used for affordance learning. (Left) For the shallow structure, text-conditioned prompts are prepended to the first vision transformer layer. (Right) For the deep structure, prompts are inserted into every vision layer. Only the prompt-related layers and the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

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), which is based on a stochastic differential equation by introducing noise.

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

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

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

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

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

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

where we define $\boldsymbol{x}_t$ as the linear interpolation between $\boldsymbol{x}_0$ and $\boldsymbol{x}_1$ with respect to time $\boldsymbol{x}_t = t\boldsymbol{x}_1 + (1-t)\boldsymbol{x}_0$, following the linear conditional flow theory [12]. $v_{\boldsymbol{\theta}}$ is a network of the flow model. Thus Equation (1) could be reformatted as

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

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

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 $\boldsymbol{o}$; ii) changing the output $\boldsymbol{x}$ to represent robot actions.

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

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

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 $\boldsymbol{o}$ as input and predicts Tp steps of actions, of which Ta steps of actions are executed on the robot without re-planning. Tp is the action prediction horizon and Ta is the action execution horizon.

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 use multiple steps $1/\Delta t$ for inference:

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

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 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, the feeding task requires affordance heatmaps centered on the fork handle, food, and the human mouth.

III. Experiments

In this section, we systematically evaluate the performance of the proposed prompt tuning and flow matching methods.

A. Affordance Evaluation with Prompt Tuning

1) Baseline Studies: We benchmark our proposed prompt tuning structures against several commonly used parameter-efficient finetuning protocols and SoA VLM-based affordance learning methods:

- Full fine-tuning: 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 [9, 14].
- Side-network methods: train a language-based network on the side, append pretrained vision features and sidetuned text features before being fed into the decoder, as customary in the literature [1]. This also shares similarities with parallel adaptors. Our deep prompt tuning method differs in the sense that it inserts text layers into every vision layer, while parallel network methods introduce additional trainable modules that run in parallel with the main transformer layers instead of modifying existing layers.
- Decoder-based methods: adopt the pretrained backbone as a feature extractor with fixed weights during tuning, and only train the decoder, as customary in the literature [3].
- Cross-attention methods: use cross-attention fusing text and vision instead of simple prepending. An example of cross-attention fusing vision and language can be found in the literature [7].
- Mixed-design methods: mixes the favorable adaptor designs. We compare against the state-of-the-art MAM adaptor [2], which is a Mix-And-Match adapter designed based on the practical findings on NLP.
- VLM-based affordance learning: We also compare against two state-of-the-art affordance learning methods based on fine-tuning large VLMs: AffordanceLLM [13] and Robopoint [17].

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.

	Methods	Learnable Params ↓	Affordance Heatmaps ($\times 10^{-3}$) ↓	Heatmap Centers (pixel) ↓
Baselines	Full	153.8M	**0.76**	**1.15**
	Decoder	3.9M	1.51	13.48
	Adapter	19.2M	1.17	6.22
	Cross-attention	43.5M	1.26	8.89
	side-network	42.7M	1.35	9.20
	mixed-design	108M	0.85	2.96
	AffordanceLLM	7B	**0.72**	**1.01**
	Robopoint	12B	-	3.18
Ours	PT-shallow	8.0M	1.42	12.04
	PT-deep (self-supervised weights)	42.1M	**0.80**	**2.93**
Ablations	PT-deep (supervised weights)	42.1M	1.48	10.13
	PT-deep (image output)	42.1M	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.

2) Main Results: 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 each center point. Three observations could be made:

- **General analysis**: Table I presents the results of prompt tuning on our ADLs testing dataset for affordance learning, comparing against baselines. The deep structure of prompt tuning outperforms other parameter-efficient baselines. We can also see that deep prompt tuning achieves better performance than RoboPoint but falls short of the performance achieved by AffordanceLLM. We would like to mention that AffordanceLLM and Robopoint are not parameter-efficient models (the focus of our research), as they involve finetuning large models, including LLaVA and Vicuna, on our dataset. Full finetuning such a large model may not function optimally if only a small dataset is available. Note that, since Robopoint predicts action keypoints directly, it does not generate intermediate affordance heatmaps. Therefore it is not included in our table.
- **Prompt tuning against 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 224×224. 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 are actually applied. 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. Note that the testing experiments involved only the authors as participants and were therefore exempt

from the requirement for ethics approval.

- **What do prompts learn?** We show a t-SNE [15] visualization of the embeddings after the last vision transformer layer (before the decoder) in Fig. 3. 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.
- **Prompt tuning or adapters?** As pointed by the research of Visual Prompt Tuning [6], in contrast to comparable studies in NLP, prompt tuning outperforms full fine-tuning and adapter-based methods in the visual domain. The MAM adaptor [2] mixes the favorable adaptor designs based on the practical findings on NLP and achieves state-of-the-art results, but does not function optimally in the image-text domain.

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 that 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.

Decoder Input: We apply the decoder on the global output 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. 4-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. 4-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 the 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.

B. Real-World Robot Evaluation

We carry out 50 replications of trials for each baseline. The experimental protocol comprised 10 distinct tasks, each evaluated over five independent trials. Adhering to an open-loop testing framework on the Activities of Daily Living (ADL) dataset, any failure on the initial attempt was recorded as a categorical failure for that trial. We define task success based on three criteria: (1) the robot correctly grasps the target object, (2) the object/tool reaches the designated affordance region associated with the task, and (3) the robot performs the expected manipulation motion or achieves the intended object state change. A trial is considered successful when all three conditions are satisfied.

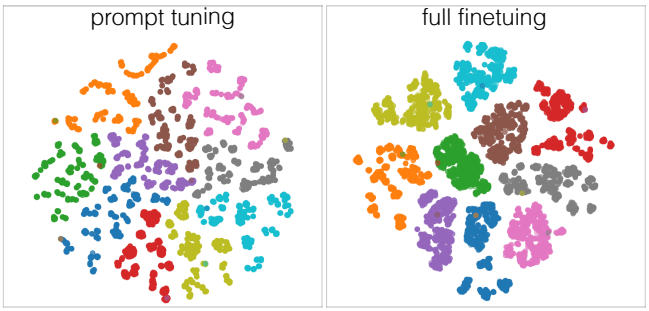

Fig. 3: t-SNE visualizations of the embeddings before the decoder. The points of the same color denote the tasks with the same language prompts, which are embedded together. The prompt tuning method could produce instruction-relevant features without updating vision backbone parameters.

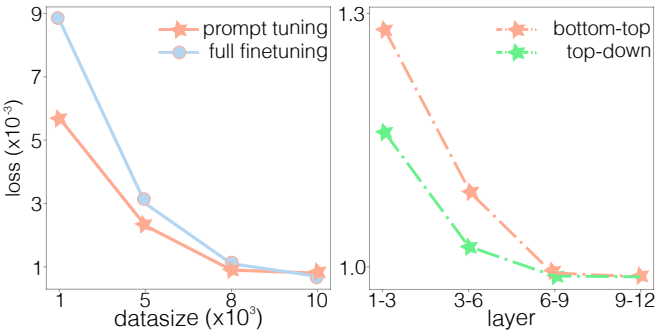

Fig. 4: Ablation studies of prompt tuning. We investigate the effect of various design choices on affordance learning performance, including pretrained weights, decoder input, dataset size and prompt location.

Methods (Inference Step)	Flow Matching (16-step) ↑	Diffusion Policy (16-step) ↑	Transformer BC ↑	Flow Matching end-to-end ↑
Activities of Daily Living	**0.82**	0.76	0.44	0.74

TABLE II: Real-world robot experimental results.

We have respectively compared the proposed flow matching policy against SoA baselines, including DDPM and Transformer policies. We also investigate whether our framework can seamlessly unify affordance learning and action generation by comparing against the end-to-end policy and off-the-shelf VLM-based trajectory generation. From Table II, we can see that flow matching outperforms DDPM and Transformer baselines.

IV. Conclusion

We have formulated a prompt tuning method for affordance map learning and flow matching policy for robot manipulation. 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] 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.

[2] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*, 2021.

[3] 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.

[4] 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.

[5] Nils Ingelhag, 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.

[6] 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.

[7] 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.

[8] 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.

[9] Zuxin Liu, Jesse Zhang, Kavosh Asadi, Yao Liu, Ding Zhao, Shoham Sabach, and Rasool Fakoor. Tail: Task-specific adapters for imitation learning with large pretrained models. *arXiv preprint arXiv:2310.05905*, 2023.

[10] 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.

[11] 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.

[12] 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.

[13] Shengyi Qian, Weifeng Chen, Min Bai, Xiong Zhou, Zhuowen Tu, and Li Erran Li. Affordancellm: Grounding affordance from vision language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7587–7597, 2024.

[14] Mohit Sharma, Claudio Fantacci, Yuxiang Zhou, Skanda Koppula, Nicolas Heess, Jon Scholz, and Yusuf Aytar. Lossless adaptation of pretrained vision models for robotic manipulation. *arXiv preprint arXiv:2304.06600*, 2023.

[15] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[16] 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.

[17] Wentao Yuan, Jiafei Duan, Valts Blukis, Wilbert Pumacay, Ranjay Krishna, Adithyavairavan Murali, Arsalan Mousavian, and Dieter Fox. Robopoint: A vision-language model for spatial affordance prediction for robotics. *arXiv preprint arXiv:2406.10721*, 2024.