

Steertuning Policies from a Few Human Videos

Billy Yan Isaac Tu* Kelly Lee* Blake Chang Kenny Ma Kanad Patel
Dhawal Kabra Neer Patel Kevin Wu Lerrel Pinto Irmak Guzey

New York University

steertuning.github.io

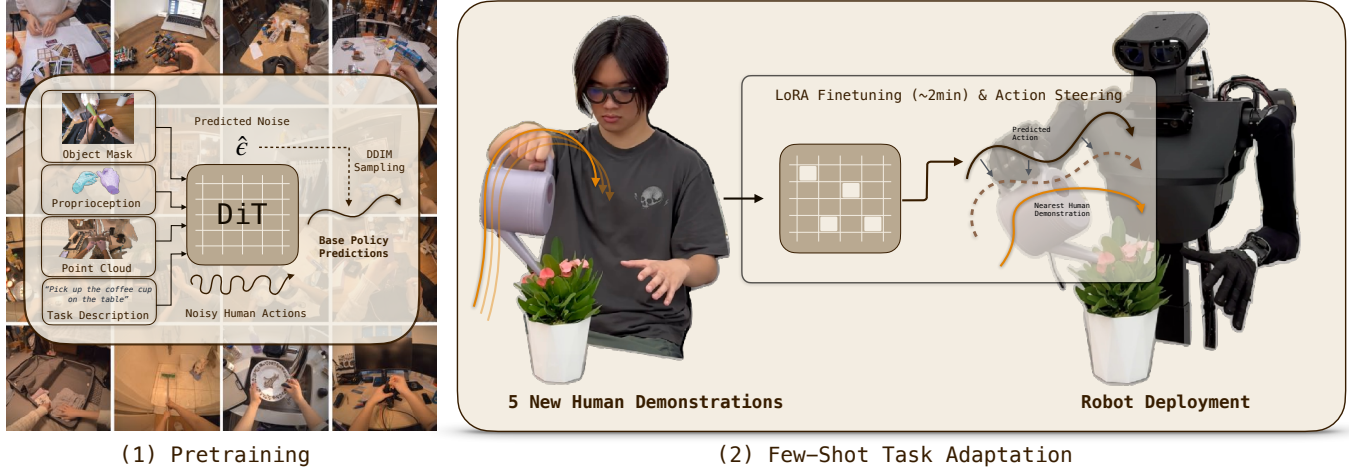


Fig. 1: Steertuning is an end-to-end system for few-shot task generalization in completely new environments. It consists of two parts: (1) pretraining a 400M-parameter diffusion policy on our human dataset consisting of 100 hours of densely annotated random interactions, and (2) 2-minute finetuning and diffusion steering, used together on 5 new task demonstrations collected with Aria Gen 2 glasses. By carefully combining the two, our mobile manipulator completes 12 new tasks in 12 new environments in under 5 minutes with a 71% average success rate.

Abstract—Current recipes for robot learning require massive amounts of robot demonstrations to obtain robust and adaptive policies. In this work, we introduce Steertuning, a system that enables robots to learn new tasks in new environments from 5 human demonstrations in under 5 minutes.

Steertuning consists of two parts. (1) We pretrain a 400M-parameter diffusion policy on our egocentric human dataset, which consists of 100 hours of densely annotated random interactions. This pretraining makes Steertuning robust to background differences and enables spatial and object generalization when deployed on new tasks. (2) Given 5 demonstrations of a new task, we finetune our model for only 2 minutes (2000 training updates) and steer the policy’s actions during reverse sampling using the demonstration whose observation is closest to that of the deployment environment. Together with a carefully designed data processing pipeline for embodiment transfer, this enables Steertuning to achieve a 71% success rate across 12 tasks in 12 new environments, ranging from lab settings to kitchens in new homes. The entire process, from data collection for a new task to robot policy deployment, takes less than 5 minutes. Our full dataset, model weights, codebase, and robot videos will be made available at <https://steertuning.github.io>.

I. INTRODUCTION

Humans learn new tasks every day: we adapt and improve through new interactions and by watching others. Achieving similar flexibility in robots requires systems capable of the same kind of adaptation. Recent work on robot learning from human videos has enabled many ways of using scalable human demonstrations for downstream robotic tasks. Some combine robot and human data to enable quick adaptation [1], [2], where models trained on large-scale human data adapt to new tasks from a few robot demonstrations. Others learn few-shot robot policies from human demonstrations alone [3]–[5]. However, learning representations that are both scalable and robust, while adapting quickly from human data only, remains unexplored. In this work, we introduce Steertuning, an end-to-end mobile dexterous manipulation system trained and finetuned only on egocentric human data, which adapts to completely new environments and homes within 5 minutes.

Initial efforts toward general robot policies focused on behavior cloning (BC), where a robot is teleoperated [6]–[8] to complete tasks under different configurations and the recorded data is used to train a policy that clones these behaviors. However, this requires hundreds, if not thousands,

Correspondence to irmakguzey@nyu.edu and yy4435@nyu.edu.
* denotes equal contribution.

of demonstrations per task. To reduce the data requirement per task, past work showed that pretraining policies on very large datasets [9]–[11] and then *finetuning* a small subset of the parameters [12] on task-specific demonstrations can boost performance. Yet, for large enough models, this process can still take days. Alternatively, with denoising diffusion probabilistic models (DDPMs), a desired-yet-incomplete distribution can be used to *steer* the predicted distribution toward it [13], i.e., inpainting with a provided mask. Similar approaches have succeeded in robotics as well [14]: given a large pretrained diffusion model and a few task demonstrations, steering enables task adaptation while keeping the deployed policy safe and robust. Steertuning carefully combines finetuning and action steering, so that adaptation takes only 2 minutes and reaches an 83% success rate across four of our tasks, compared to 23% with steering only and 0% with finetuning only (Section IV-B).

Meanwhile, given the challenges of teleoperation, human data is receiving increasing attention in the robot learning community. Initial efforts collected human demonstrations in a calibrated robot environment with multiple cameras, enabling accurate 3D annotations such as hand poses and object points [5], [14]–[16]. While these works show the importance of 3D annotations for human-to-robot retargeting, a calibrated environment is too strong an assumption for general-purpose home robots. Others instead leveraged internet-scale egocentric human videos [17]–[19], showing how scalable data can improve robots’ understanding of interactions; however, the lack of annotations such as 3D hand poses limited their deployability. More recent works [3], [4], [20], [21] show that smart glasses (e.g., Aria Gen 1/2) are a sweet spot in this scalability–annotation trade-off, enabling deployment of robot policies trained purely on human data.

Building on this line of work, Steertuning combines the scalable, richly annotated data from smart glasses with the test-time adaptation capabilities of finetuning and action steering. As illustrated in Fig. 1, Steertuning consists of two parts. The first is pretraining a 400M-parameter diffusion transformer on our random-interaction dataset; 100 hours of egocentric Aria Gen 2 recordings of humans performing everyday tasks. The transformer takes environment observations and a task description as input and predicts the noise ϵ added to the human actions. The second is 2-minute finetuning and action steering, used together on 5 new egocentric demonstrations, which enables adaptation to a new task in a new environment in under 5 minutes total. Our contributions are:

- 1) Steertuning is a new recipe that combines finetuning and action steering to enable quick yet robust task adaptation for diffusion policies. We ablate the design decisions behind it and show a 71% success rate across 12 tasks in 12 new environments (Section III-D.1).
- 2) To our knowledge, Steertuning is the first end-to-end system trained purely on human data that robustly adapts to new tasks in completely new environments (Section III-A).
- 3) We integrate Steertuning with an open-knowledge nav-

igation system [22] and a VLM orchestrator [23] to deploy robot policies from a single language command, showing that Steertuning generalizes even when navigation and docking are handled by our mobile manipulator.

Our dataset, model weights, robot videos, and software stack will be made available at <https://steertuning.github.io>.

II. RELATED WORK

A. Fine-Tuning and Steering Pretrained Policies

Fine-tuning. The dominant recipe for few-shot task learning is to pretrain a large policy on open-source robot datasets [6], [7], [11], [24], [25] and post-train it on a small set of task-specific demonstrations. Since full fine-tuning is too flexible for a handful of demonstrations, parameter-efficient methods such as low-rank adapters are used instead [12], [26], [27], and others fine-tune diffusion policies with RL [28], [29]. In all cases, adaptation requires new robot data and, for large models, hours to days of training.

Steering. A complementary line of work adapts a pretrained diffusion model at inference time, without changing its weights, by modifying the reverse sampling process. This started in image generation with guidance [30] and inpainting [13], and was later adopted by diffusion planners [31]–[33] and diffusion policies [34], where sampling is steered with a value function, a dynamics model, or human corrections [35]–[37]. Closest to ours is DemoDiffusion [14], which steers a pretrained diffusion policy with a single retargeted human demonstration. Steering alone, however, can only bend the policy toward the demonstration; it cannot teach behaviors the base policy does not know. Steertuning instead combines a 2-minute LoRA fine-tuning with demonstration-based steering, which we show to be both faster and more robust than either alone (Section IV-B).

B. Policy Learning from Human Data

Learning robot policies from human data can be split by the data source used. Initial work used in-domain data, where humans demonstrate tasks in the same calibrated environment as a table-top manipulator [4], [14]–[16]. While effective, these approaches are not scalable, since robots are no longer stationary. On the other hand, internet-scale in-the-wild videos are scalable but lack annotations needed for robot deployment, such as 3D hand pose or depth. Past work that used internet videos therefore either extracted higher-level interaction information such as affordances [17], [38], [39] or object tracks [18], [19], [40], [41], or first retargeted human demonstrations to robots [42]–[46] and then trained policies on the retargeted trajectories. The first option lacks the low-level annotations needed for dexterous interactions, and the second requires extensive preprocessing for every video with a high failure rate. More recently, human videos have also been used to pretrain policies that are then fine-tuned on robot data [1], [2], [47]–[49], but this still requires robot demonstrations for every new task.

Smart glasses provide a sweet spot between these two branches. While enabling in-the-wild, scalable data collection, they provide useful 3D annotations such as head and hand pose and stereo depth. Past work showed promising results for learning robot policies from data collected with Aria glasses [3], [4], [20]. But these works either only worked for parallel grippers, relied on point tracks of the objects, or lacked the object generality needed for general robots. Steertuning introduces a new system that (a) works end-to-end in new environments and homes, showcasing extensive visual generalization, (b) can adapt to new tasks with under 5 minutes, and (c) uses data collected purely from humans wearing Aria Gen 2 glasses.

III. METHOD

A. Overview

Steertuning is an end-to-end mobile dexterous manipulation system that completes tasks from language commands and adapts to new tasks in under 5 minutes of processing time. The system consists of three parts: (a) collecting and processing egocentric human data for scalable and robust robot learning, (b) learning robot policies from this data that robustly adapt to new environments and tasks, and (c) closing the human-to-robot embodiment gap and combining the policies with open-knowledge navigation [22] so that new tasks can be completed from a single language command. We describe each part below.

B. Collecting and Processing Smart-Glass Data

1) *Data Collection*: Steertuning uses two sources of egocentric human data, both collected by multiple operators wearing Aria Gen 2 smart glasses [54] at 10 Hz. The first, $\mathcal{D}_{\text{wild}}$, contains 109.6 hours of everyday interactions such as cooking, dishwashing, and cleaning, totaling 3.69M retained frames and 48.7k auto-generated instructions. The second, $\mathcal{D}_{\text{task}}$, contains 5 task-specific demonstrations for each robot evaluation task. We use $\mathcal{D}_{\text{wild}}$ for pretraining and $\mathcal{D}_{\text{task}}$ for task-specific fine-tuning and action steering (i.e., Steertuning).

Aria Gen 2 glasses provide synchronized RGB and stereo images, SLAM head pose, and per-hand tracking with 21 hand landmarks and a 6-DoF wrist pose. We additionally recover metric depth from the stereo images using S2M2 [55]. For each training snippet, we express the point cloud, hand trajectories, and head pose in the RGB camera frame of the last observed frame, giving a consistent camera-relative representation across demonstrations.

2) *Task Segmentation*: Raw smart-glass recordings are long and contain many interactions without explicit task boundaries or language annotations. Rather than asking a VLM to interpret an entire recording at once, we first split each video into short interaction snippets using hand-scene contact, then annotate each snippet with language and the manipulated object.

a) *Contact-based segmentation*: We remove the demonstrator’s arms and hands using SAM3 [56], unproject the remaining depth into a scene point cloud, mark a hand as active when its tracked landmarks come within 5 cm of the scene, and form snippets from intervals with a consistent active-contact state.

b) *Language annotation*: For each snippet, we sample up to 10 ordered frames and prompt Qwen3-VL-8B [57] to generate a short, verb-first instruction that describes the interaction without referring to the human embodiment, e.g., “placing bread in toaster”.

c) *Object grounding*: We additionally condition the policy on a mask of the manipulated object, similar to [58]. For each snippet, we prompt Gemini 2.5 Flash to place two keypoints on the contacted object, which initialize SAM3 to propagate the object mask through the snippet. Human arm and hand regions are removed from the resulting mask before it is used for training.

3) *Visual Domain Alignment*: Although task segmentation provides structured supervision, the visual observations still differ between human demonstrations and robot deployment: human arms and hands appear during training, while robot arms and hands appear at inference. We reduce this embodiment gap by removing the embodiment geometry from the observation in both settings.

During training, SAM3 provides the human arm and hand mask; at inference, the robot’s joint configuration and kinematic model provide the corresponding robot geometry. In both cases, we remove the masked depth points before constructing the colored point cloud, so the policy observes the surrounding scene rather than either embodiment. We further restrict the point cloud to a camera-centered workspace defined by a $0.85 \times 1.5 \times 1.5$ m ellipsoidal crop, discard points beyond 1.7 m in depth, and resample the remaining cloud to 5,000 points. This yields a consistent observation representation between human training data and robot inference.

C. Observation and Action Representation

a) *Coordinate frame*: We express each training snippet in the RGB camera frame at the anchor frame t_a , defined as the last observed frame before action prediction. The point cloud, hand history, future hand trajectories, and head targets are all transformed into this common frame using the Aria SLAM pose and camera extrinsics.

b) *Action representation*: Each hand is represented by its 21 tracked landmarks plus three additional points encoding the wrist orientation, giving 24 points per hand. Four additional points represent the head pose, for a total of 52 points in $\mathbb{R}^3$. Each action chunk contains 15 future steps at stride 3, which at the 10 Hz collection frequency corresponds to 4.5 seconds of motion.

D. Pretraining and Steertuning Policies from Human Videos

Inspired by the success of diffusion transformers [59] in robot learning [51], [60], we train an ϵ -predictor with a transformer backbone. Following [61], we use $T = 100$

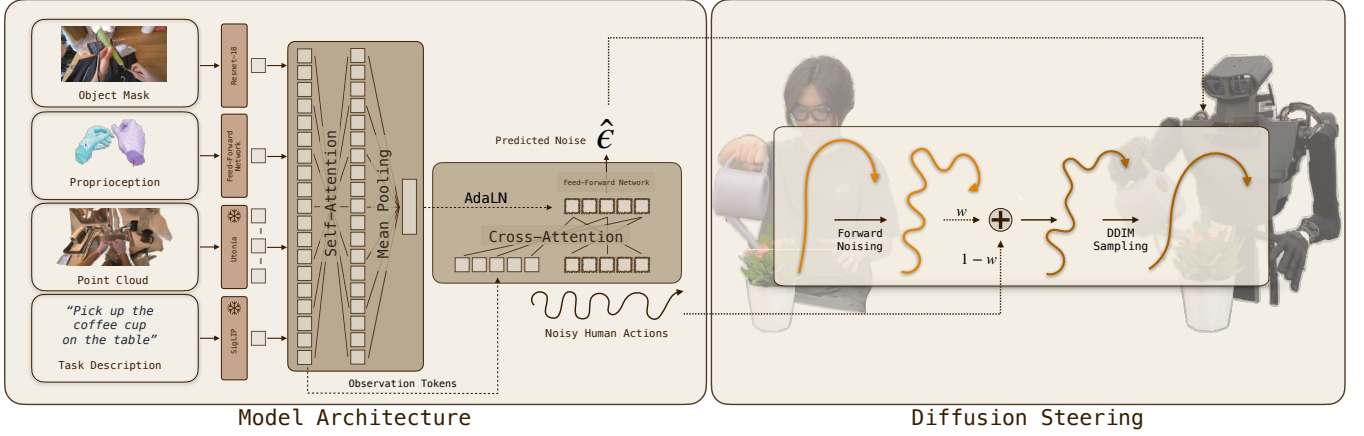


Fig. 3: **Policy.** Our architecture and action steering mechanism. The policy takes four modalities as input: a point cloud, the manipulated-object mask, the fingertip locations of both hands (10 3D points) in the anchor frame, and a text description of the task. Each modality is passed through its own encoder and the resulting tokens go through a self-attention transformer, after which we mean-pool them into a *context latent* that conditions the diffusion transformer via AdaLayerNorm [50]. The observation tokens are also fed to the diffusion transformer so that the noisy action tokens can cross-attend to them [51]. The point-cloud encoder (Utonia [52]) and text encoder (SigLIP [53]) are kept frozen; everything else is trained end-to-end with an MSE loss on the predicted ϵ . During finetuning, LoRA adapters are trained for 2k steps; at deployment, the predicted actions are steered toward the nearest human trajectory. A short finetuning combined with action steering enables robust yet quick task adaptation.

diffusion timesteps during training and deterministic DDIM sampling ($\eta = 0$) with $N = 8$ steps at inference. Our architecture and the Steertuning logic are illustrated in Fig. 3.

Similar to [51], observation embeddings condition the action prediction in two ways: (1) a context token, obtained by self-attention across all observation tokens, is added to each transformer block via AdaLN conditioning [50], [59]; and (2) the full observation-token sequence is fed to the transformer alongside the noisy action tokens, enabling cross-attention between the two. The resulting action tokens pass through a final feed-forward network (FFN) to predict the noise $\hat{\epsilon}$ at each denoising step. The loss is the mean-squared error between $\hat{\epsilon}$ and the ground-truth noise ϵ .

We use four observation modalities: a cropped and sampled point cloud $P \in \mathbb{R}^{5000 \times 3}$, a binary manipulated-object mask $O \in \mathbb{R}^{255 \times 255}$, proprioception given by the 5 fingertip points of each hand $S \in \mathbb{R}^{2 \times 5 \times 3}$, and a task description D . The point cloud is encoded with Utonia [52] into $P_e \in \mathbb{R}^{512 \times 1024}$. The object mask is passed through a ResNet-18 [62] trained from scratch to obtain a token $O_e \in \mathbb{R}^{1 \times 1024}$, proprioception through an FFN to obtain $S_e \in \mathbb{R}^{1 \times 1024}$, and the language annotation through a pretrained SigLIP [53] encoder to obtain $D_e \in \mathbb{R}^{1 \times 1024}$. These are concatenated into the observation-token sequence $Z = (P_e | O_e | S_e | D_e) \in \mathbb{R}^{515 \times 1024}$ and fed to the transformer. The Utonia and SigLIP encoders are kept frozen; everything else is trained end-to-end on $\mathcal{D}_{\text{wild}}$.

a) Implementation details: We apply per-head query/key normalization [63] for stable token mixing, and mean-pool the self-attended tokens to obtain the context latent used for AdaLN. Training hyperparameters are provided on our website.

1) Post-Training: We explore different ways of enabling rapid task adaptation from only five new human demonstra-

tions, and find that the best recipe is to combine fine-tuning with low-rank adapters [12] and steering the noise distribution during reverse sampling with the provided demonstrations, similar to [14]. In both cases, the new demonstrations are processed as described in Section III-B.2 and used as described below. Fig. 7 shows the impact of both on downstream robot tasks.

a) Low-Rank Adaptation (LoRA): Full fine-tuning is too flexible when only five demonstrations are available per task [26]. We therefore freeze the pretrained weights and insert LoRA adapters [12] of rank 8 (scaling 16) into the attention projections and selected linear layers. The action normalizer is loaded from pretraining so that post-training does not change the coordinate scale predicted by the denoiser. We fine-tune for only 2000 steps, totaling 2 minutes. To cover a broader distribution than the limited in-scene data provides, we augment the anchor frame, wrist pose, RGB, and text. We apply rigid transformations to the anchor frame t_a and the proprioception points, with translations sampled uniformly within ± 5 cm and ± 3 cm and rotations within $\pm 5^\circ$ and $\pm 5^\circ$, respectively, for generalization to varying robot head and hand poses. We apply brightness, contrast, hue, saturation, and small noise jitter to the point-cloud RGB values. For text, we sample paraphrases that preserve the action while matching the short verb-object-preposition style of the pretraining interactions.

b) Action Steering: While necessary, 2 minutes of fine-tuning is not sufficient to complete tasks robustly. Inspired by [13], we therefore steer the action predictions during reverse sampling using the human actions from the demonstration whose initial frame is closest to the current observation. Specifically, at diffusion timestep t and given a steering action a^* , we first noise it to the same noise level

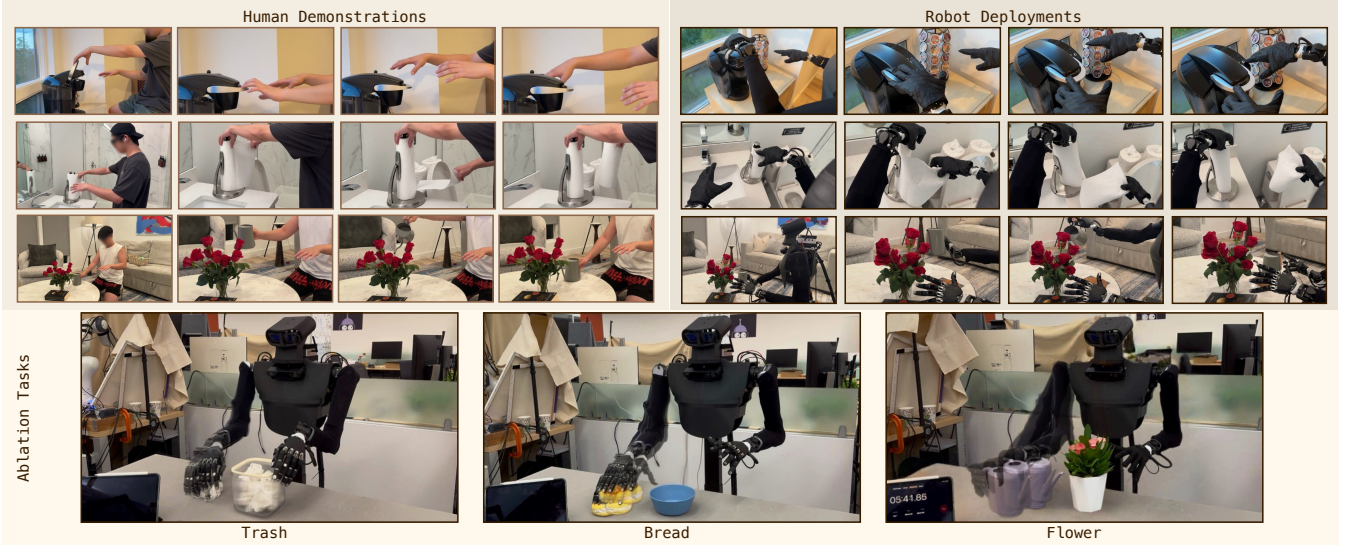


Fig. 4: In-the-wild and ablation tasks. We deploy Steertuning on 9 new tasks in new in-the-wild environments. For each task, we collect 5 new human demonstrations (top left) and deploy the policy on our robot YOR (top right). All ablation experiments are run on three tasks with spatial generalization (bottom). Steertuning achieves an overall success rate of 71% (7.1/10) across these 12 tasks; per-task success rates are given on our website.

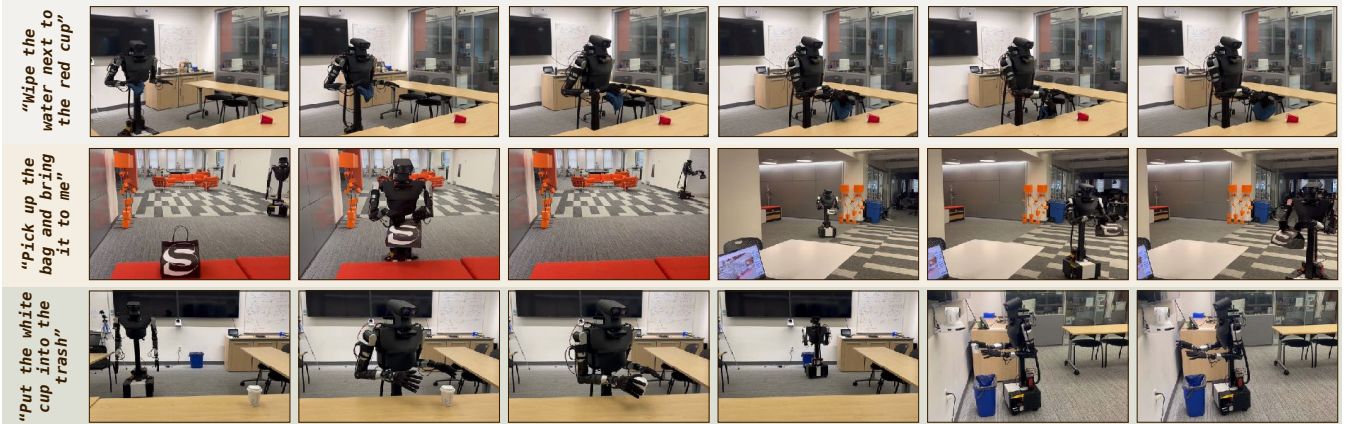


Fig. 5: Full-stack deployments of different policies from a single language command. The left-most column shows the language annotations used by the system. For example, *Put the white cup into the trash* is split into navigate white cup, pick up white cup, navigate trash, and drop white cup. Demonstrations for picking up and dropping the cup are collected and deployed in the same way as in Steertuning.

using the forward diffusion process,

$$a_t^* = \sqrt{\alpha_t} a_0^* + \sqrt{1 - \alpha_t} \epsilon, \quad (1)$$

and then, given a steering weight w , shift the noisy actions as

$$x_t \leftarrow w a_t^* + (1 - w) x_t, \quad (2)$$

before continuing with standard DDIM sampling.

To choose the closest human trajectory, we first re-express each demonstration scene in the manipulated-surface coordinate frame with the gravity axis pointing upward. We then compare the object layout in the current observation to those in the demonstrations and select the closest one. The object layout is extracted by segmenting each object on the table and computing its 3D centroid with respect to the camera. At each frame, we track the robot’s progress

through the selected demonstration by identifying the closest corresponding wrist and hand motion. This selection has a significant impact on success rate; we ablate it in Section IV-E.

E. Embodiment Transfer and Robot Deployment

1) *Embodiment Transfer*: Our main robot embodiment is a YOR robot [64] with Wuji hands [65] as end effectors. We attach the Aria glasses to the robot’s head with a 3D-printed mount. During deployment, as with the human data, we remove the robot arm and hand from the observation point cloud using the robot’s kinematic chain. We move the robot’s wrists to the predicted wrist poses with Cartesian inverse kinematics and then apply kinematic retargeting [66] so the Wuji hands follow the predicted landmarks relative to the

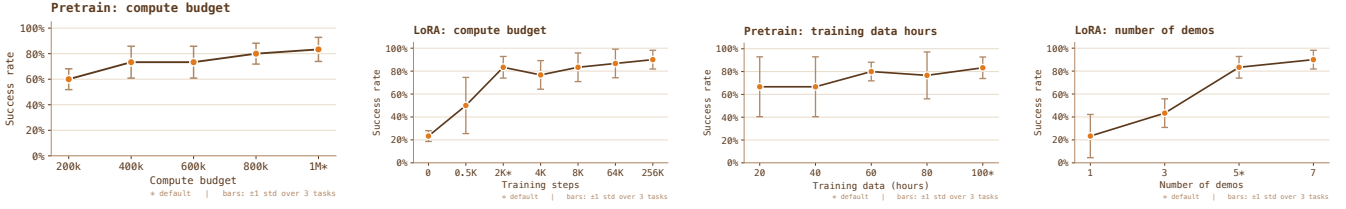


Fig. 6: Scaling ablations for (a) pretraining compute budget, (b) LoRA compute budget, (c) pretraining data hours, and (d) number of LoRA demonstrations. Points are the mean success rate over the three tasks; bars show ± 1 standard deviation. Steertuning settings are marked with *.

wrists. To minimize jitter, we apply real-time chunking [67] to temporally smooth the predictions.

2) *End-to-End Deployment*: To showcase the generalization of Steertuning, we build an end-to-end platform that lets the robot complete tasks in new environments from a language command. We use Gemini ER 1.6 [23] to decompose a long-horizon command into `navigate <object>` and `manipulate <task>` sub-tasks. For `navigate`, we follow [22]: we extract a voxelized SLAM map of the environment in which each voxel is associated with a SAM3 [56] latent, find the nearest neighbor of the target object’s SAM3 latent in the map, and navigate the robot to that location. Once there, the `manipulate` sub-task runs the multi-task manipulation policy described above. Fig. 5 shows a few end-to-end tasks with this setup; robot rollouts are on our website.

IV. EXPERIMENTS

In this section, we run a series of robot experiments to answer the following questions:

- 1) What is the impact of fine-tuning and action steering on quick task adaptation?
- 2) What is the impact of the training budget (i.e., training steps and data) on downstream robot policies?
- 3) How well does Steertuning perform when ablated against different architectural components?
- 4) How much does the trajectory-selection strategy used during action steering impact downstream task performance?
- 5) How well does Steertuning generalize to new objects?

A. Task Descriptions

We deploy Steertuning on 12 tasks in total, illustrated in Fig. 4: 9 in an Airbnb rented for this purpose and 3 in our lab. For each task, we collect 5 entirely new human demonstrations and deploy the policy immediately afterwards. All ablation experiments are run on the 3 lab tasks. In *Trash* and *Bread*, the robot is asked to pick up the corresponding object and drop it into a bowl or a can, respectively. In *Flower*, the robot is asked to pick up the watering can and water the flower. During evaluation, we move the trash, bread, and watering can within a 20×20 cm region. Steertuning achieves an overall success rate of 71% (7.1/10) across these 12 tasks. All robot videos are available on our website.

B. What is the impact of fine-tuning and action steering on quick task adaptation?

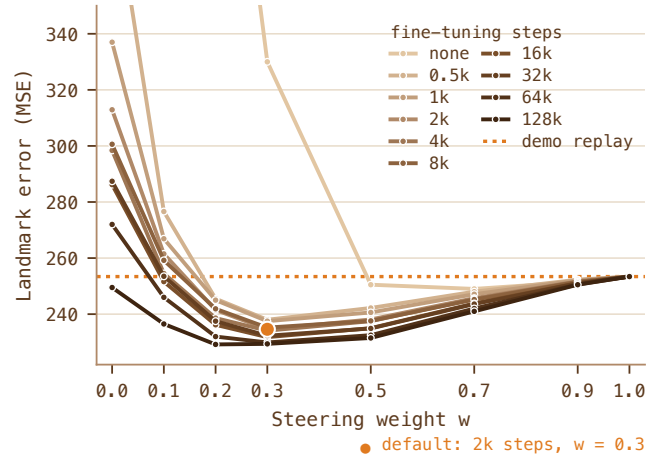


Fig. 7: Mean landmark error as a function of the steering weight w for different fine-tuning budgets, averaged over our 12 tasks. Error is the MSE between the predicted and ground-truth action landmarks on a held-out demonstration; the dotted line is the error of replaying the demonstration directly. Values above 350 (no fine-tuning at $w \leq 0.2$, 0.5k steps at $w = 0$) are cut off for readability.

Steertuning uses fine-tuning and diffusion steering together in a novel way. Fig. 7 shows how the mean action error changes with longer fine-tuning and a higher steering weight w on a held-out demonstration. The dotted line is the mean error of the trajectory selected to steer the policy on that held-out configuration. As we steer more strongly toward that demonstration, the error converges to the dotted line (at $w = 1$, DDIM sampling reduces to replaying the demonstration). Darker lines correspond to policies fine-tuned for more steps, from no fine-tuning to 128k steps. While several configurations bring the error below the dotted line, fine-tuning for only 2k steps (about 2 minutes) and steering with $w = 0.3$ strikes the best balance: it reaches a minimum error very close to that of the policy fine-tuned for 128k steps (roughly a day of training) at a small fraction of the cost.

Beyond landmark errors, we run robot experiments on the 3 lab tasks and compare Steertuning against three baselines: fine-tuning only, for 2k and for 150k steps, and steering only, without any fine-tuning (Table I). Steertuning outperforms

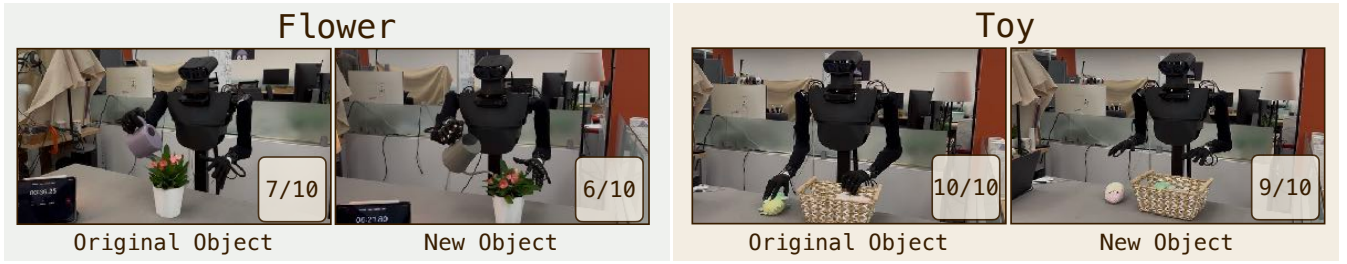


Fig. 8: Object generalization. For each task, the left image shows the original object and the right image the new object used at deployment, with the corresponding success rate.

TABLE I: Success rates of the full Steertuning pipeline compared to fine-tuning only and steering only. Ours in bold.

Pipeline	Trash	Flower	Bread	Avg.
Steertuning	9/10	7/10	9/10	8.3/10
Steering only	2/10	2/10	3/10	2.3/10
Short fine-tune (2k)	0/10	0/10	0/10	0/10
Full fine-tune (150k)	7/10	8/10	8/10	7.7/10

all baselines. The closest, fine-tuning for 150k steps, reaches comparable success rates but requires a full day of training, whereas Steertuning adapts in minutes. The same 2k-step fine-tuning that Steertuning uses fails completely on its own, and steering alone succeeds in only 23% of trials. Together, these results show that combining the two is what enables few-shot task adaptation that is both robust and fast.

C. What is the impact of the training budget (i.e., training steps and data) on downstream robot policies?

Steertuning brings together many data sources and methodologies for learning robot policies. To quantify the impact of data and compute, we vary the compute budget for both pretraining and LoRA fine-tuning, as well as the amount of data used in each stage, and report success rates with standard-deviation bars in Fig. 6. Success rate increases with compute budget. For pretraining, we use the maximum budget to reach the highest performance; for fine-tuning, we instead choose the cheapest checkpoint that still achieves a high success rate, keeping task adaptation as fast as possible.

D. How well does Steertuning perform when ablated against different architectural components?

Steertuning uses Utonia [52] as the point-cloud encoder and a Diffusion Transformer (DiT) [59] as the action decoder. We ablate both by replacing the encoder with PointNeXt [68] or PTv3 [69], and the decoder with a UNet [70]. Pretraining, post-training, and evaluation are identical for all variants. As shown in Table II, Utonia and DiT achieve the highest average success rates; although other variants outperform Steertuning on individual tasks, the averages support our design choices.

E. How much does the trajectory-selection strategy used during action steering impact downstream task performance?

Steertuning selects the nearest neighbor among the newly collected human demonstrations by comparing the object

TABLE II: Success rates for encoder and decoder architecture ablations. Ours in bold.

		Trash	Flower	Bread	Avg.
Encoder	PointNeXt	5/10	6/10	7/10	6/10
	PTv3	10/10	4/10	9/10	7.7/10
	Utonia (Ours)	9/10	7/10	9/10	8.3/10
Decoder	UNet	8/10	8/10	6/10	7.3/10
	DiT (Ours)	9/10	7/10	9/10	8.3/10

layout at deployment time to that of each demonstration, as described in Sec. III-D.1. We compare this against two alternatives for choosing the steering trajectory (Table III):

- **VINN** [71] uses the mean-pooled context latent of the initial observation to find the nearest-neighbor demonstrations, then takes a weighted average of their actions, with weights given by the normalized L2 distances between the context latents of the deployment and demonstration observations.
- **Wrist pose** uses the initial wrist pose to find the nearest-neighbor demonstration.

TABLE III: Success rates for trajectory-selection strategies at inference. Ours in bold.

Trajectory selection	Trash	Flower	Bread	Avg.
VINN [71]	7/10	2/10	7/10	5.3/10
Wrist pose	6/10	7/10	4/10	5.7/10
Object-based (Ours)	9/10	7/10	9/10	8.3/10

Once a trajectory is selected, steering is performed identically for all methods. As shown in Table III, the object-based selection of Steertuning outperforms both alternatives.

F. How well does Steertuning generalize to new objects?

Finally, we evaluate Steertuning when the object at deployment differs from the one in the demonstrations. Results and the objects used are shown in Fig. 8. The *Toy* task, where the goal is to pick up a toy and drop it into a box, was added specifically for this evaluation. Steertuning remains robust to visual changes in the object as long as the required action does not differ significantly from the demonstrated one. We attribute this to large-scale pretraining and to the augmentations applied during fine-tuning.

V. DISCUSSION AND LIMITATIONS

We presented Steertuning, a system for few-shot task adaptation from in-the-wild human videos in under 5 minutes. Pretraining provides robustness to visual perturbations, while the combination of fine-tuning and action steering enables rapid learning of new tasks in new environments. We note two major limitations. (1) Human video data contains no contact information; all actions and hand poses are annotated purely visually. As a result, the learned low-level actions remain relatively primitive, which limits the dexterity of the system. Incorporating contact information or a form of RL-based adaptation should extend the range of tasks Steertuning can learn. (2) Steertuning relies on kinematic retargeting being sufficient to complete the tasks; even when the human-policy predictions are accurate, the robot can fail due to retargeting errors.

ACKNOWLEDGEMENTS

We thank Jeff Cui, Zifan Zhao, Dandan Shan, Bowen Tan, Andre Wang, Tianxing Zhou, Liangguang Zhou, Zicheng Teng, Lucas Liu, Sissi Hu, and Charles Ji for valuable discussions and feedback, and Mais Dwal, Kikki Liu, Jingyu Li, Melody Zuo, Kevin Tang, Ayaan Mustafa, Neytiri Cheng, Wilfendjy Mathurin, Maggie Wang, Cindy Ma, John Mwangi, Chloe Wang, Tasnim Raisa, Amelia Tang, and Bruce Zhang for their help with data collection.

This work was supported by grants from LG, Qualcomm, Honda, Microsoft, NSF award 2339096, ONR award N00014-22-1-2773 and Institute of Information & Communications Technology Planning & Evaluation (IITP) with a grant funded by the Ministry of Science and ICT (MSIT) of the Republic of Korea in connection with the Global AI Frontier Lab International Collaborative Research Agreement. (No. RS-2024-00469482 & RS-2024-00509279). Lerrel Pinto is supported by the Sloan, Packard, and CIFAR Fellowships.

Artificial intelligence (AI) tools were used to improve the grammar and clarity of this paper. AI was not used to generate any of the text or references.

REFERENCES

- [1] T. Tao *et al.*, “Dexwild: Dexterous human interactions for in-the-wild robot policies,” *arXiv preprint arXiv:2505.07813*, 2025.
- [2] X. Cai *et al.*, “In-n-on: Scaling egocentric manipulation with in-the-wild and on-task data,” *arXiv preprint arXiv:2511.15704*, 2025.
- [3] V. Liu *et al.*, “Egozero: Robot learning from smart glasses,” *arXiv preprint arXiv:2505.20290*, 2025.
- [4] I. Guzey *et al.*, “Dexterity from smart lenses: Multi-fingered robot manipulation with in-the-wild human demonstrations,” *arXiv preprint arXiv:2511.16661*, 2025.
- [5] S. Haldar and L. Pinto, “Point policy: Unifying observations and actions with key points for robot manipulation,” *arXiv preprint arXiv:2502.20391*, 2025.
- [6] A. Iyer *et al.*, “Open teach: A versatile teleoperation system for robotic manipulation,” *arXiv preprint arXiv:2403.07870*, 2024.
- [7] S. P. Arunachalam *et al.*, “Holo-dex: Teaching dexterity with immersive mixed reality,” *arXiv preprint arXiv:2210.06463*, 2022.
- [8] R. Ding *et al.*, “Bunny-visionpro: Real-time bimanual dexterous teleoperation for imitation learning,” in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2025, pp. 12 248–12 255.
- [9] K. Black *et al.*, “pi0: A vision-language-action flow model for general robot control,” *arXiv preprint arXiv:2410.24164*, 2024.
- [10] O. M. Team *et al.*, “Octo: An open-source generalist robot policy,” *arXiv preprint arXiv:2405.12213*, 2024.
- [11] A. O’Neill *et al.*, “Open x-embodiment: Robotic learning datasets and rx models: Open x-embodiment collaboration 0,” in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 6892–6903.
- [12] E. J. Hu *et al.*, “Lora: Low-rank adaptation of large language models,” *arXiv preprint arXiv:2106.09685*, 2021.
- [13] A. Lugmayr *et al.*, “Repaint: Inpainting using denoising diffusion probabilistic models,” in *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*. IEEE, 2022, pp. 11 451–11 461.
- [14] S. Park, H. Bharadhwaj, and S. Tulsiani, “Demodiffusion: One-shot human imitation using pre-trained diffusion policy,” *arXiv preprint arXiv:2506.20668*, 2025.
- [15] I. Guzey *et al.*, “Bridging the human to robot dexterity gap through object-oriented rewards,” in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 3344–3351.
- [16] C. Wang *et al.*, “Dexcap: Scalable and portable mocap data collection system for dexterous manipulation,” *arXiv preprint arXiv:2403.07788*, 2024.
- [17] S. Bahl *et al.*, “Affordances from human videos as a versatile representation for robotics,” in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2023, pp. 01–13.
- [18] H. Bharadhwaj *et al.*, “Track2act: Predicting point tracks from internet videos enables generalizable robot manipulation,” in *European Conference on Computer Vision*. Springer, 2024, pp. 306–324.
- [19] J. Shi *et al.*, “Zeromimic: Distilling robotic manipulation skills from web videos,” in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 16 939–16 947.
- [20] S. Kareer *et al.*, “Egomimic: Scaling imitation learning via egocentric video,” in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 13 226–13 233.
- [21] K. Y. Wu *et al.*, “Human universal grasping,” *arXiv preprint arXiv:2606.17054*, 2026.
- [22] P. Liu *et al.*, “Ok-robot: What really matters in integrating open-knowledge models for robotics,” *arXiv preprint arXiv:2401.12202*, 2024.
- [23] G. R. Team *et al.*, “Gemini robotics: Bringing ai into the physical world,” *arXiv preprint arXiv:2503.20020*, 2025.
- [24] M. J. Kim *et al.*, “Openvla: An open-source vision-language-action model,” *arXiv preprint arXiv:2406.09246*, 2024.
- [25] S. Liu *et al.*, “Rdt-1b: a diffusion foundation model for bimanual manipulation,” in *International Conference on Learning Representations*, vol. 2025, 2025, pp. 29 982–30 009.
- [26] Z. Liu *et al.*, “Tail: Task-specific adapters for imitation learning with large pretrained models,” in *International Conference on Learning Representations*, vol. 2024, 2024, pp. 16 330–16 353.
- [27] M. J. Kim, C. Finn, and P. Liang, “Fine-tuning vision-language-action models: Optimizing speed and success,” *arXiv preprint arXiv:2502.19645*, 2025.
- [28] A. Ren *et al.*, “Diffusion policy optimization,” in *International Conference on Learning Representations*, vol. 2025, 2025, pp. 77 288–77 329.
- [29] A. Wagenmaker *et al.*, “Steering your diffusion policy with latent space reinforcement learning,” *arXiv preprint arXiv:2506.15799*, 2025.
- [30] J. Ho and T. Salimans, “Classifier-free diffusion guidance,” *arXiv preprint arXiv:2207.12598*, 2022.
- [31] M. Janner *et al.*, “Planning with diffusion for flexible behavior synthesis,” *arXiv preprint arXiv:2205.09991*, 2022.
- [32] W. Xiao *et al.*, “Safediffuser: Safe planning with diffusion probabilistic models,” in *International Conference on Learning Representations*, vol. 2025, 2025, pp. 100 391–100 412.
- [33] J. Carvalho *et al.*, “Motion planning diffusion: Learning and planning of robot motions with diffusion models,” in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 1916–1923.
- [34] C. Chi *et al.*, “Diffusion policy: Visuomotor policy learning via action diffusion,” *The International Journal of Robotics Research*, vol. 44, pp. 1684–1704, 2025.
- [35] M. Nakamoto *et al.*, “Steering your generalists: Improving robotic foundation models via value guidance,” *arXiv preprint arXiv:2410.13816*, 2024.
- [36] M. Du and S. Song, “Dynaguide: Steering diffusion policies with active dynamic guidance,” *Advances in Neural Information Processing Systems*, vol. 38, pp. 44 192–44 221, 2026.
- [37] Y. Wang *et al.*, “Inference-time policy steering through human interactions,” in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 15 626–15 633.
- [38] M. Goyal *et al.*, “Human hands as probes for interactive object understanding,” in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2022, pp. 3283–3293.
- [39] R. Mendonca, S. Bahl, and D. Pathak, “Structured world models from human videos,” *arXiv preprint arXiv:2308.10901*, 2023.
- [40] C. Wen *et al.*, “Any-point trajectory modeling for policy learning,” *arXiv preprint arXiv:2401.00025*, 2023.
- [41] M. Vecerik *et al.*, “Robotap: Tracking arbitrary points for few-shot visual imitation,” in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 5397–5403.
- [42] Y. Qin *et al.*, “Dexmv: Imitation learning for dexterous manipulation from human videos,” in *European Conference on Computer Vision*. Springer, 2022, pp. 570–587.
- [43] K. Shaw, S. Bahl, and D. Pathak, “Videodex: Learning dexterity from internet videos,” in *Conference on Robot Learning*. PMLR, 2023, pp. 654–665.
- [44] C. Wang *et al.*, “Mimicplay: Long-horizon imitation learning by watching human play,” *arXiv preprint arXiv:2302.12422*, 2023.
- [45] M. Lepert, J. Fang, and J. Bohg, “Phantom: Training robots without robots using only human videos,” *arXiv preprint arXiv:2503.00779*, 2025.

- [46] J. Ren *et al.*, “Motion tracks: A unified representation for human-robot transfer in few-shot imitation learning,” in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 8802–8810.
- [47] S. Ye *et al.*, “Latent action pretraining from videos,” in *International Conference on Learning Representations*, vol. 2025, 2025, pp. 28 213–28 239.
- [48] R. Yang *et al.*, “Egovla: Learning vision-language-action models from egocentric human videos,” *arXiv preprint arXiv:2507.12440*, 2025.
- [49] R.-Z. Qiu *et al.*, “Humanoid policy~ human policy,” *arXiv preprint arXiv:2503.13441*, 2025.
- [50] X. Huang and S. Belongie, “Arbitrary style transfer in real-time with adaptive instance normalization,” in *2017 IEEE international conference on computer vision (ICCV)*. IEEE, 2017, pp. 1510–1519.
- [51] G. Yan *et al.*, “Maniflow: A general robot manipulation policy via consistency flow training,” *arXiv preprint arXiv:2509.01819*, 2025.
- [52] Y. Zhang *et al.*, “Utonia: Toward one encoder for all point clouds,” *arXiv preprint arXiv:2603.03283*, 2026.
- [53] X. Zhai *et al.*, “Sigmoid loss for language image pre-training,” in *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, 2023, pp. 11 941–11 952.
- [54] The Project Aria Team at Meta, “Aria Gen 2: An advanced research device for egocentric AI research,” <https://www.projectaria.com/ariagen2devicepaper/>, Meta Reality Labs Research, Whitepaper, 2025, accessed: 2026-09-13.
- [55] J. Min *et al.*, “S2m2: Scalable stereo matching model for reliable depth estimation,” *arXiv preprint arXiv:2507.13229*, 2025.
- [56] N. Carion *et al.*, “Sam 3: Segment anything with concepts,” in *International conference on learning representations*, vol. 2026, 2026, pp. 138 846–138 923.
- [57] J. Bai *et al.*, “Qwen technical report,” *arXiv preprint arXiv:2309.16609*, 2023.
- [58] Z. J. Cui *et al.*, “Contact-anchored policies: Contact conditioning creates strong robot utility models,” *arXiv preprint arXiv:2602.09017*, 2026.
- [59] W. Peebles and S. Xie, “Scalable diffusion models with transformers,” in *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, 2023, pp. 4172–4182.
- [60] S. Dasari *et al.*, “The ingredients for robotic diffusion transformers,” in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 15 617–15 625.
- [61] J. Song, C. Meng, and S. Ermon, “Denoising diffusion implicit models,” *arXiv preprint arXiv:2010.02502*, 2020.
- [62] K. He *et al.*, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [63] A. Henry *et al.*, “Query-key normalization for transformers,” in *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020, pp. 4246–4253.
- [64] M. H. Anjaria *et al.*, “Yor: Your own mobile manipulator for generalizable robotics,” *arXiv preprint arXiv:2602.11150*, 2026.
- [65] Wuji Technology, “WUJI Hand: A 20-dof dexterous robotic hand,” <https://wuji.tech/en/>, 2026, accessed: 2026-09-13.
- [66] Y. Qin *et al.*, “Anyteleop: A general vision-based dexterous robot arm-hand teleoperation system,” *arXiv preprint arXiv:2307.04577*, 2023.
- [67] K. Black, M. Galliker, and S. Levine, “Real-time execution of action chunking flow policies,” *Advances in Neural Information Processing Systems*, vol. 38, pp. 33 383–33 407, 2026.
- [68] G. Qian *et al.*, “Pointnext: Revisiting pointnet++ with improved training and scaling strategies,” *Advances in neural information processing systems*, vol. 35, pp. 23 192–23 204, 2022.
- [69] X. Wu *et al.*, “Point transformer v3: Simpler, faster, stronger,” in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2024, pp. 4840–4851.
- [70] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.
- [71] J. Pari *et al.*, “The surprising effectiveness of representation learning for visual imitation,” *arXiv preprint arXiv:2112.01511*, 2021.