MoFlo: Language-Conditioned Flow Matching for Policy Mobilization

Anonymous Author(s)
Affiliation
Address
email

Abstract: Open-vocabulary mobile manipulation (OVMM) requires a robot to carry out a natural-language instruction on a mobile platform, and the success of learned manipulation policies hinges on placing the base within the policy's training distribution. We focus on policy mobilization, the problem of predicting an $SE(2)$ base pose from which a fixed pre-trained manipulation policy will succeed, given a single ego-centric RGB-D frame and a language instruction. Prior work estimates a per-task, instruction-agnostic success likelihood over base poses, which limits generalization across tasks and language-based disambiguation. To address these limitations, we propose MoFlo (Mobilization Flow), a multi-task model that learns a language- and observation-conditioned transport from the robot's current base pose to a policy-compatible endpoint, trained by conditional flow matching with x_1 prediction. On five RoboCasa kitchen tasks, MoFlo achieves an 80% mean success rate, outperforming prior policy-mobilization baselines, and further extends to language-based disambiguation among multiple fixtures of the same category. In real-world experiments on a mobile manipulator, MoFlo places the base for successful manipulation from a single ego-centric observation, including cases that require selecting a language-specified target in an ambiguous scene, outperforming a representative baseline.

Keywords: Open-Vocabulary Mobile Manipulation, Flow Matching

1 Introduction

Service robots that work alongside humans are becoming increasingly important in homes, hospitals, and warehouses, especially with rising labor shortages and an aging population. Based on a free-form natural-language instruction, such robots must navigate to a target region in the scene and manipulate the object the instruction names. This problem is studied as open-vocabulary mobile manipulation (OVMM) [1]. Recent OVMM systems combine open-vocabulary semantic navigation with modular manipulation [2, 3, 4, 5], but their manipulation step remains heuristic, relying on analytic grasp detection [6] and scripted motion primitives. Learned manipulation policies, from visuomotor policies [7, 8, 9] to vision-language-action models [10, 11, 12] and world-action models [13, 14, 15], have advanced general-purpose manipulation, but they have not yet been integrated into OVMM pipelines at scale.

In this study, we focus on policy mobilization, the problem of placing the base of a mobile robot at a pose from which a fixed pre-trained manipulation policy will succeed. Given a language instruction and a single ego-centric RGB-D frame at the robot's current pose, the model must predict an $SE(2)$ target pose that lies within the policy's training distribution, and the robot then navigates to this pose before the manipulation policy executes.

The primary challenge is that the correct base pose depends jointly on the task, on which target object the instruction names among visually-similar candidates, and on the manipulation policy's own

Submitted to the 10th Conference on Robot Learning (CoRL 2026). Do not distribute.

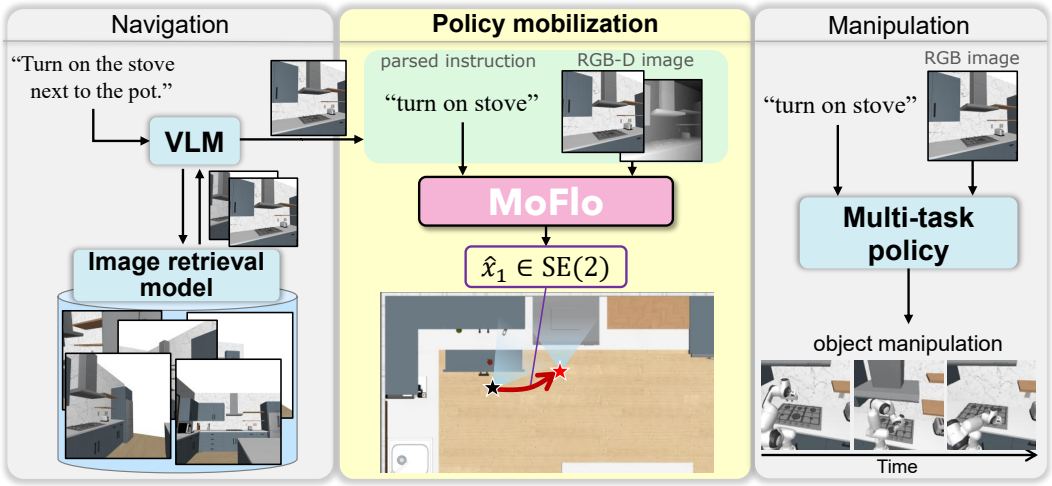
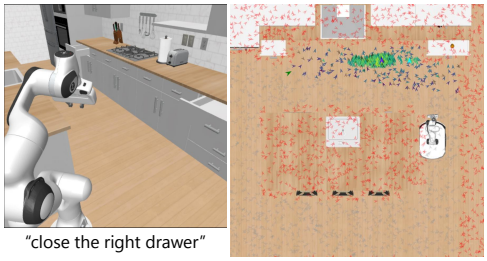


Figure 1: **MoFlo in an OVMM pipeline.** An upstream retrieval module parses a free-form natural-language instruction (e.g. "turn on the stove next to the pot") and grounds it to a target view in the scene. MoFlo takes the parsed task instruction together with a single ego-centric RGB-D image and predicts a target $SE(2)$ base pose $\hat{x}_1$ in a single forward pass, and the robot navigates to $\hat{x}_1$ while a fixed multi-task visuomotor policy executes the named manipulation.

viewpoint distribution. Existing systems navigate to the object centroid or pick a nearby pose from task-agnostic reachability heuristics [2, 16], neither of which can express these joint dependencies, and this gap is increasingly identified as a system-level bottleneck of OVMM pipelines [16].

Two recent methods study policy mobilization directly [17, 18]. These methods are policy-aware, in the sense that the predicted pose is trained against a downstream manipulation policy's success signal, but their awareness is limited to a single task. Each method trains one pose distribution per task, and the prediction conditions only on the visual scene rather than on the instruction. As a result, the same prediction is returned when two instructions name two same-type fixtures in one scene (Figure 2b), and a separate model must be trained for every task in deployment. These limitations suggest that policy mobilization should be reformulated as a direct language-conditioned prediction that handles every task in a single multi-task model.



(a) (b)

Figure 2: **Motivation.** (a) Task setup on CLOSEDRAWERDISAMBIG, where the robot is given the instruction "close the right drawer" in a scene with two open drawers. (b) A representative instruction-agnostic baseline [17] assigns pose mass in front of both drawers and cannot resolve which one the prompt refers to.

In this paper, we propose MoFlo (Mobilization Flow), a multi-task policy-mobilization model that learns a language- and observation-conditioned transport from the robot's current base pose to a policy-compatible $SE(2)$ endpoint in a single forward pass (Figure 1). MoFlo is trained by conditional flow matching [19, 20] with x_1 prediction [21], where the network predicts the endpoint $\hat{x}_1$ and supervises the induced rectified-flow velocity against the conditional flow-matching target. To evaluate the language-conditioned placement that this enables, we further introduce language disambiguation (Figure 2a), where several same-type fixtures are open and the instruction names one, instantiated by the CLOSEDRAWERDISAMBIG and CLOSEDRAWERDISAMBIGTHREE tasks on which MoFlo reaches the named fixture where instruction-agnostic methods cannot.

In summary, our contributions are:

- We revisit the instruction-agnostic distribution paradigm of prior policy-mobilization methods [17, 18] and reformulate base placement as a language- and observation-conditioned transport from the robot's current base pose to a policy-compatible endpoint, predicted in a single forward pass.
- We propose MoFlo, a multi-task language-conditioned policy-mobilization model trained by conditional flow matching with x_1 prediction. By conditioning on language, MoFlo maps the same scene to different policy-compatible base poses for different instructions, a capability prior policy-mobilization methods cannot express by construction.
- We conduct comprehensive experiments in simulation and on a real mobile manipulator. On five RoboCasa kitchen tasks, MoFlo achieves an 80% mean success rate and outperforms prior policy-mobilization baselines, while real-world experiments demonstrate successful base placement for language-specified manipulation and outperform a representative baseline.

2 Related Work

Base placement and policy mobilization. Base-pose selection for downstream manipulation follows two lines [17, 18, 22, 23, 24, 25, 26, 27, 28, 29]. Policy mobilization selects poses against a downstream policy's viewpoint or success distribution, while policy-blind feasibility methods use kinematic [22, 23, 24] or VLM-grounded affordance criteria [25, 26, 27, 30, 28, 29]. The closest works, Mobi-π [17] and N2M [18], are both policy mobilization. Mobi-π reconstructs the scene as a 3D Gaussian Splatting [31] field and scores candidate poses by DINO [32] similarity to the policy's training views, while N2M predicts a Gaussian mixture over poses from an ego-centric point cloud. Both are per-task and instruction-agnostic, the limitations MoFlo removes.

Mobile manipulation systems. Open-vocabulary mobile manipulation was formalized by [1], with pipelines combining semantic navigation, language grounding, and modular manipulation [2, 3, 4, 5] that leave the manipulation base pose implicit or heuristic [16]. End-to-end policies learn base-arm coordination jointly [33, 34, 11, 35, 36, 37] but do not expose base placement as a reusable module, and MoManipVLA [38] couples base waypoints to a finetuned VLA. In contrast, MoFlo is a standalone $SE(2)$ base-placement model that pairs with any downstream policy and composes with language-conditioned navigation [39, 40, 41, 3] at the navigation endpoint.

Visuomotor policy robustness and flow matching. Policy mobilization is needed because visuomotor policies degrade when the camera or base pose drifts from training [42, 43, 44, 45]. Rather than modifying the policy, MoFlo moves the robot to where the policy was trained to operate. We cast base-pose prediction as a conditional flow [19, 20], following work that applies flow matching to manipulation [46, 47, 48, 49, 10, 50, 51, 52]. Unlike these, our source at $t{=}0$ is the robot's current base pose rather than Gaussian noise, an instance of transport between arbitrary distributions [20, 53] also used in concurrent action-prediction work [54].

3 Problem Statement

We consider the policy-mobilization stage that bridges navigation and manipulation in an open-vocabulary mobile-manipulation pipeline (Figure 1). Upstream of MoFlo, an image-based navigation module (e.g. [3, 2, 39]) retrieves and paraphrases the natural-language instruction ℓ against a memory of egocentric views and brings the robot to a query pose x_{query} at which the target fixture is visible. At this point the robot is already close to the target, but not necessarily within the success basin of the fixed pre-trained manipulation policy π, which acts on the robot's observation under the instruction ℓ, and the residual placement error is what determines whether π will succeed. Given a single ego-centric RGB-D observation o taken at x_{query} and the instruction ℓ, MoFlo predicts an $SE(2)$ target pose $x^* \in \mathbb{R}^4$ encoded as $(\hat{x}, \hat{y}, \cos\theta, \sin\theta)$ with $\hat{x}, \hat{y} \in [-1, 1]$ from which π will succeed under ℓ, after which the robot navigates from x_{query} to x^* and hands off to π. The ideal target

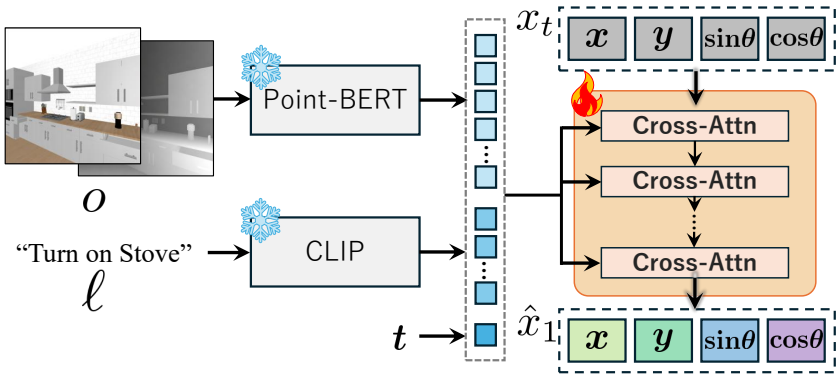


Figure 3: **Overview of MoFlo.** A single ego-centric RGB-D frame and a language instruction are encoded by a Point-BERT visual encoder and a CLIP text encoder. The resulting embeddings, together with the robot's current base pose $x_0 = (x, y, \sin\theta, \cos\theta)$, feed a transformer head that predicts the target base pose $\hat{x}_1 = (x, y, \sin\theta, \cos\theta) \in SE(2)$. The robot then navigates to $\hat{x}_1$ and the fixed manipulation policy executes.

is the pose that maximizes the probability that π succeeds under ℓ from it given the observation o. In practice we approximate this with the demonstration-induced distribution over policy-compatible base poses from the corpus on which π was trained. MoFlo predicts x^* with no per-scene reconstruction and no pre-exploration phase, in contrast to reconstruction-based methods such as Mobi-π [17] that first build a 3D Gaussian Splatting [31] field of the scene.

4 Method

Prior policy-mobilization methods fit a per-task, instruction-agnostic pose distribution and return its most likely pose. Mobi-π scores candidate poses inside a per-scene 3D reconstruction with Bayesian optimization, and N2M predicts a Gaussian mixture over poses directly from an ego-centric observation. Both formulations share a structural limitation, namely that the prediction is conditioned only on the visual observation, so two prompts referring to two fixtures of the same type in the same scene yield identical predictions (Figure 2b).

Rather than predicting a pose from scratch, MoFlo learns a language- and observation-conditioned transport that moves the robot's current base pose to a policy-compatible endpoint, trained by conditional flow matching (Figure 3). Because the transport is conditioned on both the ego-centric observation and the instruction, the same visual scene can be mapped to different endpoints depending on the prompt. A single multi-task model is shared across all tasks in the benchmark, and the instruction is consumed as an input token rather than used to select a per-task model, so fixture disambiguation within a scene (Section 5.4) is an intrinsic capability of the model.

Architecture. The model takes three inputs, an ego-centric RGB-D frame o, a natural-language instruction ℓ, and the robot's current base pose $x_0 \in \mathbb{R}^4$ encoded as $(x, y, \cos\theta, \sin\theta)$. It outputs a target base pose $\hat{x}_1 \in \mathbb{R}^4$ in the same encoding.

The RGB-D frame is back-projected to a point cloud and encoded by a frozen Point-BERT encoder [55] into N_V visual embeddings $\phi_V(o) \in \mathbb{R}^{N_V \times d_V}$. The instruction is encoded by a frozen CLIP ViT-B/16 text encoder [56] into N_L language embeddings $\phi_L(\ell) \in \mathbb{R}^{N_L \times d_L}$. A transformer head g_ψ then predicts the target pose from the current pose x_0 conditioned on the visual and language tokens,

$$\hat{x}_1 = g_\psi\big(x_0,\, \phi_V(o),\, \phi_L(\ell)\big). \tag{1}$$

Flow-matching pose head. We train the transformer head as a conditional flow. Given the current pose x_0 and an oracle target pose x_1, conditional flow matching [19, 20] fits a velocity field v_ψ that

transports x_0 to x_1 along the straight-line interpolant

$$x_t = (1-t)\,x_0 + t\,x_1, \quad t \in [0,1], \tag{2}$$

by regressing $v_\psi(x_t, t)$ against the rectified-flow target $v^\star = x_1 - x_0$. Following recent robotics work that replaces the Gaussian source of flow matching with a task-relevant prior [54, 53], we take the source at $t{=}0$ to be a Dirac at the robot's current base pose rather than a Gaussian, so that the flow learns a goal-directed transport from the current pose to the target rather than a generic noise-to-data map.

We parameterize the model to predict the endpoint [21]

$$\hat{x}_1 = g_\psi\big(x_t,\, t,\, \phi_V(o),\, \phi_L(\ell)\big) \tag{3}$$

directly, and define the induced velocity as $\hat{v}_\psi = (\hat{x}_1 - x_t)/(1-t)$. The endpoint prediction is supervised through the velocity loss

$$\mathcal{L}(\psi) = \mathbb{E}_{(o,\ell,x_0,x_1),\,t}\left[w \left\| \frac{\hat{x}_1 - x_t}{1-t} - (x_1 - x_0) \right\|^2 \right], \tag{4}$$

with per-sample success weight w. This objective encourages the endpoint prediction to induce a rectified-flow velocity that matches the conditional flow-matching target. The flow time t is sampled from a Beta distribution rather than uniformly on $[0,1]$, concentrating supervision near $t{=}0$.

Training. We construct supervision from the same fixed manipulation policy that MoFlo hands off to at deployment, so that the target poses reflect where the deployed policy is expected to succeed. For each (task, layout, instruction paraphrase) triple, we identify an oracle base pose x_1 from the demonstrations used to train the downstream policy. We then sample a query pose x_0 near x_1, render the ego-centric RGB-D observation o at x_0, and pair it with the instruction ℓ and target endpoint x_1. Each (o, ℓ, x_0, x_1) tuple becomes one training example, with per-sample success weight w in Eq. (4). Instruction paraphrases are drawn from a per-task pool, and we hold out a disjoint set of paraphrases, never used in training, to evaluate instruction generalization in Section 5.3.

Inference. At deployment, the robot captures a single ego-centric RGB-D frame at its current base pose x_0 and encodes the observation together with the instruction. We evaluate the model once at $t{=}0$, where $x_t = x_0$, to obtain the target pose $\hat{x}_1$. The robot then navigates to $\hat{x}_1$ and executes the fixed manipulation policy. Inference therefore requires one model evaluation and does not require per-scene reconstruction, candidate-pose sampling, or iterative pose search.

5 Experiments

5.1 Simulation Setup

We evaluated MoFlo on five RoboCasa [57] kitchen tasks, following the policy-mobilization setup of [17, 18]: TURNONMICROWAVE, CLOSEDRAWER, CLOSESINGLEDOOR, TURNONSINKFAUCET, and TURNONSTOVE. Each task has 5 training layouts and 5 held-out layouts. The downstream manipulation policy is a single multi-task bc-transformer [9] trained on MimicGen [58] demonstrations from the 5 training layouts of all five tasks. We keep this manipulation policy fixed for all base-placement methods and evaluate each method by handing its predicted pose to the policy under the corresponding task instruction. Task success rate (SR) is measured by the downstream policy's success from the predicted pose, on matched episodes with the same object placements and start poses.

Baselines. MoFlo is a single multi-task base-placement model trained jointly on all five tasks and conditioned on a language instruction. In contrast, the baselines follow the single-task, instruction-agnostic protocol of prior policy-mobilization work: a separate base-placement model or pose-selection procedure is used for each task, while the same fixed multi-task manipulation policy

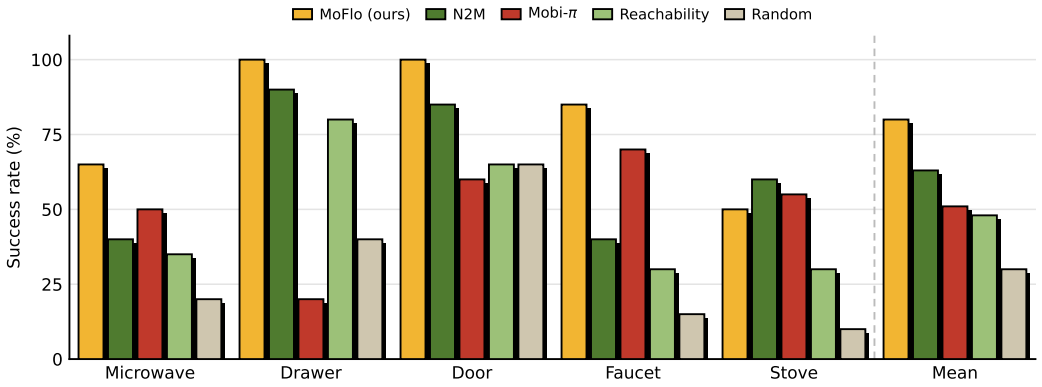


Figure 4: **Quantitative results on RoboCasa kitchen tasks.** Task success rate of the fixed downstream manipulation policy after navigation to the base pose predicted by each method. MoFlo achieves the highest mean success rate across the five tasks and outperforms prior policy-mobilization baselines while using a single language-conditioned multi-task model. All methods are evaluated on the same layouts and object-placement distribution.

is used for execution. We compare against four base-placement baselines: (1) **Random**, a pose drawn uniformly around the oracle pose; (2) **Reachability**, an analytic inverse-reachability stand-in [24]; (3) **Mobi-π** [17], per-scene 3DGS [31] reconstruction with DINO [32] novel-view scoring and Bayesian optimization; (4) **N2M** [18], a Gaussian-mixture-over-poses head trained by NLL on rollout poses. Implementation details and the matched-corpus protocol for the learned baselines are in Appendix B.

5.2 RoboCasa Kitchen Tasks

Figure 4 summarizes the quantitative results. MoFlo achieves an 80% mean success rate across the five tasks, outperforming all prior policy-mobilization baselines. Compared with N2M, the strongest learned baseline with a 63% mean success rate, MoFlo improves the mean by 17 percentage points and achieves the highest success rate on four of the five tasks, while the two methods are comparable on TURNONSTOVE. The improvement over instruction-agnostic baselines reflects the combination of language-conditioned pose prediction, which allows the output to vary with the task instruction, and the x_1-prediction flow head, which improves endpoint precision on tight-tolerance tasks.

5.3 Zero-Shot Generalization

We test MoFlo's zero-shot generalization along two axes while holding the trained model fixed, namely unseen instruction phrasings on the training layouts and novel scene appearance under data scaling.

Replacing every training instruction with a held-out paraphrase never seen during training (e.g. "slide the drawer in" for "close the drawer"), MoFlo retains a 70% mean success rate against 80% under trained phrasings (Figure 5), so the model acts on instruction content rather than memorizing trained phrasings.

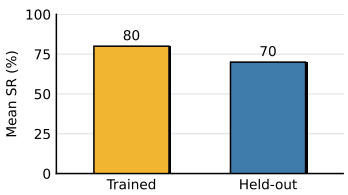


Figure 5: **Instruction-phrasing generalization.** The 5-task mean success rate of the same MoFlo model under trained instructions and held-out paraphrases.

We further measure how training-data coverage shapes robustness to scene appearance by retraining MoFlo on $N \in \{2, 3, 4, 5\}$ training layouts and evaluating each model on the same novel-style scene pairs (Figure 6). Novel-style success improves overall as the number of training layouts grows, though not strictly monotonically on every task, indicating that appearance robustness is bound by data coverage rather than by

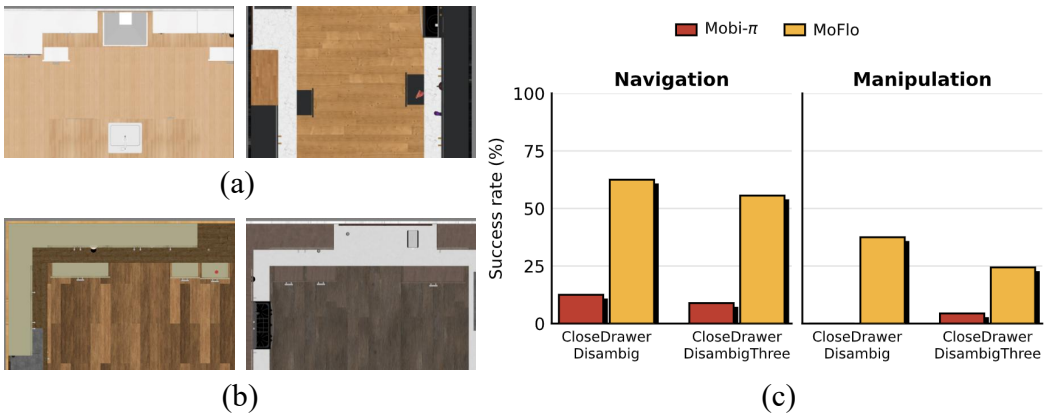


Figure 7: **Language disambiguation.** (a) Top-down views of the CLOSEDRAWERDISAMBIG scenes (two open drawers) and (b) the CLOSEDRAWERDISAMBIGTHREE scenes (three open drawers) across held-out layouts. (c) Navigation success rate and manipulation success rate for MoFlo and Mobi-π on the two tasks.

the architecture and that adding training kitchens directly buys robustness to appearance shift.

5.4 Language Disambiguation

Real OVMM scenes often contain multiple fixtures of the same type, requiring the instruction to identify the target fixture. We evaluate this ability on CLOSEDRAWERDISAMBIG (Figure 7a), where the instruction specifies either the left or right open drawer, and CLOSEDRAWERDISAMBIGTHREE (Figure 7b), which extends the choice to three drawers. MoFlo is trained only on the left/right contrast and evaluated on held-out layouts using navigation success rate, which measures whether the named drawer is visible at the reached pose, and manipulation success rate, which measures whether the downstream policy closes it from that pose.

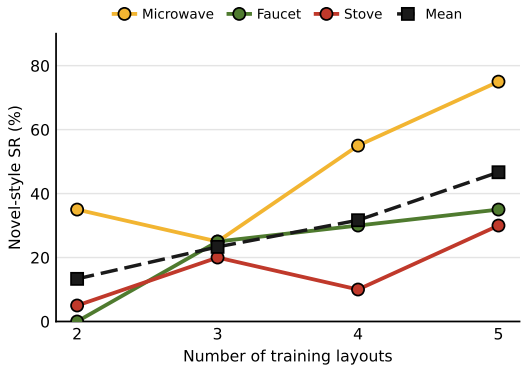


Figure 6: **Decor robustness scales with the number of training layouts.** MoFlo novel-style SR per task and 3-task mean on a common set of (seen layout $\times$ novel style) scene pairs across models trained on $N \in \{2, 3, 4, 5\}$ layouts.

Figure 7c shows the quantitative results. On CLOSEDRAWERDISAMBIG, MoFlo achieves 62.5% navigation success rate and 37.5% manipulation success rate, exceeding Mobi-π by 50.0 and 37.5 points, respectively. On CLOSEDRAWERDISAMBIGTHREE, MoFlo is applied zero-shot and achieves 55.6% navigation success rate and 24.4% manipulation success rate, remaining 46.7 and 20.0 points above Mobi-π. These results indicate that language conditioning enables MoFlo to select the correct fixture among same-category candidates, while the lower manipulation success rate mainly reflects the placement precision required by the downstream policy.

5.5 Ablation Studies

Figure 8 compares the x_1-prediction head against four alternatives that share the encoders, backbone, and language conditioning, namely N2M's GMM-NLL head,

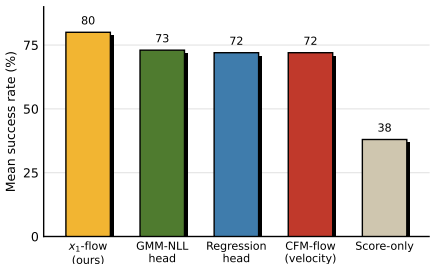


Figure 9: **Real-world deployment on a Toyota HSR.** Representative successful MoFlo rollouts for (a) PICK, (b) CLOSEDRAWER, and (c) PUSHSHELF, each shown as a left-to-right filmstrip from the predicted base pose through manipulation. (d) Navigation and manipulation success rate for each task and method (Human, Mobi-π, and MoFlo; out of 10 trials per task).

a regression head, a velocity-parameterized flow head, and a score-only variant. The x_1-prediction head gives the best 5-task mean, the GMM-NLL, regression, and velocity-parameterized heads fall within 8 points, and the score-only variant trails well behind. Predicting the endpoint rather than the velocity is the decisive choice, sharpening the placement precision that the tight-tolerance tasks require. We further ablate the visual and text encoders in Appendix F.

5.6 Real-World Experiments

We deploy MoFlo on a Toyota Human Support Robot (HSR) with a head-mounted RGB-D camera and evaluate three mobile-manipulation tasks, PICK, CLOSEDRAWER, and PUSHSHELF, of which PICK and CLOSEDRAWER require disambiguation among same-type candidates from the instruction. As in simulation, an upstream image-based retrieval module brings the robot to a query pose before MoFlo predicts the base pose and a fixed manipulation policy executes. We report navigation success rate (the robot reaches a pose from which the target is manipulable) and manipulation success rate (the policy then completes the task) against a **Human** upper bound and **Mobi-π**. Task and protocol details are in Appendix G.

Figure 9d presents the real-world success rates. MoFlo reaches the correct base pose and completes the manipulation on all three tasks, outperforming Mobi-π on every task in both navigation and manipulation (on PICK, 80 versus 60 and 50 versus 30), while the human upper bound confirms the manipulation policy is reliable once the base is well placed.

6 Limitations and Future Work

MoFlo is an in-distribution method. Like the policy it places, it is trained and evaluated on kitchens from a single distribution, and broad cross-layout transfer, likely through a stronger visual representation or fixture-relative pose reasoning rather than the flow head, remains open. It also assumes upstream grounding of a parsed instruction, and integrating a vision-language model directly would ground free-form language and resolve referents within one model while supplying visual priors that the cross-layout setting needs. End-to-end success is further bounded by the fixed downstream policy, since a pose within MoFlo's prediction error still fails when it falls outside the policy's success basin (the TURNONSTOVE case in Section 5.2), which co-training or placement augmentation

could relax. Finally, disambiguation precision is capped by a small training set, and we address only single-step placement, leaving multi-step OVMM and system-level evaluation of the full pipeline to future work.

7 Conclusion

We introduced MoFlo, a language-conditioned policy-mobilization model that bridges navigation and manipulation in OVMM pipelines by replacing the instruction-agnostic pose-distribution paradigm of prior methods with a single conditional flow mapping one ego-centric RGB-D frame and a natural-language instruction directly to an $SE(2)$ base pose, with no per-scene 3D reconstruction. The flow head uses x_1 prediction with a velocity loss, recovering the tight-tolerance tasks on which the velocity parameterization underfits. On five RoboCasa kitchen tasks (n=20/cell), MoFlo reaches 80% mean success rate, 29 points above the reconstruction-based Mobi-π and 17 points above N2M, while running in a single forward pass with no per-scene reconstruction. A single multi-task model handles every task in the benchmark, and given multi-fixture training data the same architecture learns to disambiguate fixtures of the same type in a scene from a language instruction, generalizing to held-out layouts, a capability the per-task scoring heads of prior policy-mobilization methods cannot express by construction. Cross-layout visual generalization remains the principal limitation, and direct VLM integration is a natural route toward addressing it.

References

[1] S. Yenamandra, A. Ramachandran, K. Yadav, A. Wang, M. Khanna, T. Gervet, T.-Y. Yang, V. Jain, A. W. Clegg, J. Turner, et al. HomeRobot: Open-vocabulary mobile manipulation. In *CoRL*, 2023.

[2] P. Liu, Y. Orru, J. Vakil, C. Paxton, N. M. M. Shafiullah, and L. Pinto. OK-Robot: What really matters in integrating open-knowledge models for robotics. In *RSS*, 2024.

[3] D. Yashima, R. Korekata, and K. Sugiura. Open-vocabulary mobile manipulation based on double relaxed contrastive learning with dense labeling. *IEEE RA-L*, 2025.

[4] P. Zhi, Z. Zhang, Y. Zhao, M. Han, Z. Zhang, Z. Li, Z. Jiao, B. Jia, and S. Huang. Closed-loop open-vocabulary mobile manipulation with GPT-4V. *arXiv preprint arXiv:2404.10220*, 2024.

[5] S. Tan, D. Zhou, J. Tang, Y. Ji, H. Liu, and Y. Cui. Language-conditioned open-vocabulary mobile manipulation with pretrained models. In *IJCAI*, 2025.

[6] H.-S. Fang, C. Wang, H. Fang, M. Gou, J. Liu, H. Yan, W. Liu, Y. Xie, and C. Lu. AnyGrasp: Robust and efficient grasp perception in spatial and temporal domains. *IEEE T-RO*, 2023.

[7] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *RSS*, 2023.

[8] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware. 2023.

[9] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *CoRL*, 2021.

[10] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, L. X. Shi, J. Tanner, Q. Vuong, A. Walling, H. Wang, and U. Zhilinsky. π_0: A vision-language-action flow model for general robot control. In *RSS*, 2025.

[11] Physical Intelligence, K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, M. Y. Galliker, D. Ghosh, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, D. LeBlanc, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, A. Z. Ren, L. X. Shi, L. Smith, J. T. Springenberg, K. Stachowicz, J. Tanner, Q. Vuong, H. Walke, A. Walling, H. Wang, L. Yu, and U. Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In *CoRL*, 2025.

[12] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, Q. Vuong, T. Kollar, B. Burchfiel, R. Tedrake, D. Sadigh, S. Levine, P. Liang, and C. Finn. OpenVLA: An open-source vision-language-action model. In *CoRL*, 2024.

[13] S. Yang, Y. Du, K. Ghasemipour, J. Tompson, D. Schuurmans, and P. Abbeel. Learning interactive real-world simulators. In *ICLR*, 2024.

[14] F. Zhu, H. Wu, S. Guo, Y. Liu, C. Cheang, and T. Kong. IRASim: Learning interactive real-robot action simulators. *arXiv preprint arXiv:2406.14540*, 2024.

[15] Y. Hu, Y. Guo, P. Wang, X. Chen, Y.-J. Wang, J. Zhang, K. Sreenath, C. Lu, and J. Chen. Video prediction policy: A generalist robot policy with predictive visual representations. In *ICML*, 2025.

[16] K. Gubernatorov, A. Voronov, R. Voronov, S. Pasynkov, S. Perminov, Z. Guo, and D. Tsetserukou. AnywhereVLA: Language-conditioned exploration and mobile manipulation. *arXiv preprint arXiv:2509.21006*, 2025.

[17] J. Yang, I. Huang, B. Vu, M. Bajracharya, R. Antonova, and J. Bohg. Mobi-π: Mobilizing your robot learning policy. In *CoRL*, 2025.

[18] K. Chai, H. Lee, and J. J. Lim. N2M: Bridging navigation and manipulation by learning pose preference from rollout. *arXiv preprint arXiv:2509.18671*, 2025.

[19] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le. Flow matching for generative modeling. In *ICLR*, 2023.

[20] X. Liu, C. Gong, and Q. Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *ICLR*, 2023.

[21] T. Li and K. He. Back to basics: Let denoising generative models denoise. *arXiv preprint arXiv:2511.13720*, 2025.

[22] B. Shao, N. Cao, Y. Ding, X. Wang, F. Gu, and C. Chen. MoMa-Pos: An efficient object-kinematic-aware base placement optimization framework for mobile manipulation. *arXiv preprint arXiv:2403.19940*, 2024.

[23] C. Li, M. Xu, A. Bahety, H. Yin, Y. Jiang, H. Huang, J. Wong, S. Garlanka, C. Gokmen, R. Zhang, W. Liu, J. Wu, R. Martín-Martín, and L. Fei-Fei. MoMaGen: Generating demonstrations under soft and hard constraints for multi-step bimanual mobile manipulation. In *ICLR*, 2026.

[24] M. Rudorfer. RM4D: A combined reachability and inverse reachability map for common 6-/7-axis robot arms by dimensionality reduction to 4D. *arXiv preprint arXiv:2410.06968*, 2024.

[25] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei. VoxPoser: Composable 3D value maps for robotic manipulation with language models. In *CoRL*, 2023.

[26] X. Li, M. Zhang, Y. Geng, H. Geng, Y. Long, Y. Shen, R. Zhang, J. Liu, and H. Dong. ManipLLM: Embodied multimodal large language model for object-centric robotic manipulation. In *CVPR*, 2024.

[27] W. Yuan, J. Duan, V. Blukis, W. Pumacay, R. Krishna, A. Murali, A. Mousavian, and D. Fox. RoboPoint: A vision-language model for spatial affordance prediction for robotics. In *CoRL*, 2024.

[28] T.-J. Lin, J.-F. Yeh, H.-T. Su, C.-Y. Lin, Y.-T. Chen, and W. H. Hsu. Affordance-guided coarse-to-fine exploration for base placement in open-vocabulary mobile manipulation. In *AAAI*, 2026.

[29] E. Tong, A. Opipari, S. Lewis, Z. Zeng, and O. C. Jenkins. OVAL-Prompt: Open-vocabulary affordance localization for robot manipulation through LLM affordance-grounding. In *ICRA Workshop (VLMNM)*, 2024.

[30] Q. Gu, A. Kuwajerwala, S. Morin, K. M. Jatavallabhula, B. Sen, A. Agarwal, C. Rivera, W. Paul, K. Ellis, R. Chellappa, C. Gan, C. M. de Melo, J. B. Tenenbaum, A. Torralba, F. Shkurti, and L. Paull. ConceptGraphs: Open-vocabulary 3D scene graphs for perception and planning. In *ICRA*, 2024.

[31] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM TOG (SIGGRAPH)*, 42(4), 2023.

[32] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, pages 9650–9660, 2021.

[33] S. Chen, J. Liu, S. Qian, H. Jiang, L. Li, R. Zhang, Z. Liu, C. Gu, C. Hou, P. Wang, Z. Wang, and S. Zhang. AC-DiT: Adaptive coordination diffusion transformer for mobile manipulation. *arXiv preprint arXiv:2507.01961*, 2025.

[34] J. Dong, L. Zhang, L. Zhang, Y. Ling, Y. Fu, K. Bai, Z.-C. Márton, Z. Bing, Z. Chen, A. C. Knoll, and J. Zhang. M4Diffuser: Multi-view diffusion policy with manipulability-aware control for robust mobile manipulation. *arXiv preprint arXiv:2509.14980*, 2025.

[35] P. Sundaresan, R. Malhotra, P. Miao, J. Yang, J. Wu, H. Hu, R. Antonova, F. Engelmann, D. Sadigh, and J. Bohg. HoMeR: Learning in-the-wild mobile manipulation via hybrid imitation and whole-body control. *arXiv preprint arXiv:2506.01185*, 2025.

[36] Z. Fu, T. Z. Zhao, and C. Finn. Mobile ALOHA: Learning bimanual mobile manipulation with low-cost whole-body teleoperation. *arXiv preprint arXiv:2401.02117*, 2024.

[37] J. Wu, W. Chong, R. Holmberg, A. Prasad, Y. Gao, O. Khatib, S. Song, S. Rusinkiewicz, and J. Bohg. TidyBot++: An open-source holonomic mobile manipulator for robot learning. In *CoRL*, 2024.

[38] Z. Wu, Y. Zhou, X. Xu, Z. Wang, and H. Yan. MoManipVLA: Transferring vision-language-action models for general mobile manipulation. In *CVPR*, 2025.

[39] C. Huang, O. Mees, A. Zeng, and W. Burgard. Visual language maps for robot navigation. In *ICRA*, 2023.

[40] N. Yokoyama, S. Ha, D. Batra, J. Wang, and B. Bucher. VLFM: Vision-language frontier maps for zero-shot semantic navigation. In *ICRA*, 2024.

[41] A.-C. Cheng, Y. Ji, Z. Yang, Z. Gongye, X. Zou, J. Kautz, E. Bıyık, H. Yin, S. Liu, and X. Wang. NaVILA: Legged robot vision-language-action model for navigation. *arXiv preprint arXiv:2412.04453*, 2024.

[42] S. Tian, B. Wulfe, K. Sargent, K. Liu, S. Zakharov, V. Guizilini, and J. Wu. View-invariant policy learning via zero-shot novel view synthesis. In *CoRL*, 2024.

[43] T. Jiang, J. Ji, X. Tan, J. Fang, A. Bhattad, V. Guizilini, and M. R. Walter. Do you know where your camera is? view-invariant policy learning with camera conditioning. *arXiv preprint arXiv:2510.02268*, 2025.

[44] S. Vasudevan, S. Sagar, and R. Senanayake. Viewpoint-agnostic manipulation policies with strategic vantage selection. *arXiv preprint arXiv:2506.12261*, 2025.

[45] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu. 3D diffusion policy: Generalizable visuomotor policy learning via simple 3D representations. In *RSS*, 2024.

[46] X. Hu, B. Liu, X. Liu, and Q. Liu. AdaFlow: Imitation learning with variance-adaptive flow-based policies. In *NeurIPS*, 2024.

[47] E. Chisari, N. Heppert, M. Argus, T. Welschehold, T. Brox, and A. Valada. Learning robotic manipulation policies from point clouds with conditional flow matching. In *CoRL*, 2024.

[48] Q. Zhang, Z. Liu, H. Fan, G. Liu, B. Zeng, and S. Liu. FlowPolicy: Enabling fast and robust 3D flow-based policy via consistency flow matching for robot manipulation. In *AAAI*, 2025.

[49] S. Jiang, X. Fang, N. Roy, T. Lozano-Pérez, L. P. Kaelbling, and S. Ancha. Streaming flow policy: Simplifying diffusion/flow-matching policies by treating action trajectories as flow trajectories. In *CoRL*, 2025.

[50] B. Lim, J. Kim, J. Kim, Y. Lee, and F. C. Park. EquiGraspFlow: SE(3)-equivariant 6-dof grasp pose generative flows. In *CoRL*, 2024.

[51] D. Yashima, K. Seno, S. Kurita, Y. Oda, and K. Sugiura. HiFlow: Tokenization-free scale-wise autoregressive policy learning via flow matching. *arXiv preprint arXiv:2603.27281*, 2026.

[52] F. Zhang and M. Gienger. Affordance-based robot manipulation with flow matching. *arXiv preprint arXiv:2409.01083*, 2024.

[53] M. S. Albergo, N. M. Boffi, and E. Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv preprint arXiv:2303.08797*, 2023.

[54] J. Jia, G. Li, X. Chen, T. An, Y. Hu, J. Li, X. Guo, and J. Yang. Action-to-action flow matching. *arXiv preprint arXiv:2602.07322*, 2026.

[55] X. Yu, L. Tang, Y. Rao, T. Huang, J. Zhou, and J. Lu. Point-BERT: Pre-training 3d point cloud transformers with masked point modeling. In *CVPR*, pages 19313–19322, 2022.

[56] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

[57] S. Nasiriany, A. Maddukuri, L. Zhang, A. Parikh, A. Lo, A. Joshi, A. Mandlekar, and Y. Zhu. RoboCasa: Large-scale simulation of everyday tasks for generalist robots. In *RSS*, 2024.

[58] A. Mandlekar, S. Nasiriany, B. Wen, I. Akinola, Y. Narang, L. Fan, Y. Zhu, and D. Fox. MimicGen: A data generation system for scalable robot learning using human demonstrations. In *CoRL*, 2023.

[59] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

A Disambiguation Data Collection

Environment. CLOSEDRAWERDISAMBIG modifies the RoboCasa CLOSEDRAWER task: at every reset both the leftmost and rightmost top drawer of the counter are opened (lateral separation $> 1\,\text{m}$) and both are within the field of view from the start pose. The instruction is "close the left drawer" or "close the right drawer"; the success criterion checks only the named drawer, so a placement that closes the wrong drawer is a failure. An instruction-agnostic placement method commits to one pose regardless of the prompt and so rarely brings the prompted drawer into view.

Disambiguation training data. We collect data on the four training layouts of CLOSEDRAWERDISAMBIG. For each layout we obtain both drawers' oracle base poses and form training tuples $(o,\ \ell,\ x_{\text{cur}},\ x_{\text{tgt}})$ paired with each side's prompt and the corresponding drawer oracle. Both prompts are emitted for every scene with different targets. This contrast within a scene is the signal that forces the language-conditioned model to act on the left/right token. Current poses x_{cur} are sampled with a mixed scheme: 65% are uniform perturbations ($\pm 20\,\text{cm}$ box, $\pm 15^\circ$) around the two-drawer midpoint, where both drawers are equidistant, so only language resolves the target, and 35% are uniform perturbations around the target drawer's oracle, supplying within-drawer transport precision. Ego RGB-D is rendered at 224×224 and encoded with the frozen Point-BERT encoder.

Disambiguation training. The same architecture (Section 4) is trained on the disambiguation corpus, warm-started from the main MoFlo model, with no architectural change. Training uses AdamW (lr $10^{-4} \to 10^{-5}$ cosine), batch 128, 200 epochs, bf16.

Evaluation. We evaluate on held-out layouts never seen during disambiguation training, reporting the four that admit a solvable target on both sides. Navigation success rate measures whether the prompted drawer is within the robot's head-camera view at the reached pose. Manipulation success rate runs the fixed bc-transformer policy from the produced pose and checks the CLOSEDRAWERDISAMBIG success criterion. Layout 9 is excluded for both methods because its right-drawer oracle is itself unsolvable by the policy, so it cannot form a balanced left/right pair, leaving four layouts (eight cells, 80 rollouts at ten episodes per cell).

Mobi-π baseline. Mobi-π cannot condition on the instruction and predicts a single base pose per scene regardless of the prompt, so we measure its manipulation success rate without handing it the two drawer oracles. For each held-out layout we take the pose that Mobi-π's own pipeline selects on CLOSEDRAWER, the candidate that maximizes its DINO-similarity score, and navigate to that pose in CLOSEDRAWERDISAMBIG under both prompts. Because this single pose targets at most one drawer and is imprecise on these held-out layouts, the prompted drawer is in the robot's camera view only 12.5% of the time, and the policy never closes it (0% manipulation success rate).

B Implementation Details

Transformer architecture. The pose scorer and the conditional flow head are 8-block cross-attention transformers with $d_{\text{model}} = 384$, 8 attention heads, FFN multiplier 4, dropout 0.1. The pose $x_t = (\hat{x}, \hat{y}, \cos\theta, \sin\theta)$ is mapped to a single query token by a two-layer MLP. The query token cross-attends, in each block, to the concatenation of the visual token sequence and the language token sequence. The flow head additionally injects time conditioning at each block via AdaLN scale/shift parameters derived from a sinusoidal embedding of $t \in [0, 1]$. The scorer outputs a scalar logit; the flow head predicts a 4-D data point $\hat{x}_1$ (x_1 prediction), from which the velocity $(\hat{x}_1 - x_t)/(1 - t)$ is derived.

Encoders. The visual tower is a frozen Point-BERT encoder, applied to the point cloud back-projected from a single ego-centric RGB-D frame rendered at 224×224 from the RoboCasa robot's wrist-mounted camera; it emits 65 tokens of dimension 256, projected to 384 d. The text tower is the frozen CLIP ViT-B/16 text encoder [56] (hidden dimension 512); we use the per-token last hidden state rather than the pooled CLS embedding, projected to 384 d. Point-BERT and CLIP are not jointly contrastively pre-trained; the cross-attention pose head learns the alignment between the two projected token streams. Both encoders are frozen; only the scorer and flow head are trained.

Scorer training. The scorer is trained with BCE-with-logits on $\sim$31 k training tuples (one per rollout, labeled by binary task success), a $90/10$ train/val split, 100 epochs, batch 256, AdamW (lr $10^{-4} \to 10^{-5}$ cosine), weight decay 0.01, gradient clip 1.0, bf16. Paraphrases are sampled round-robin per task; positive and negative poses are both present in the corpus, so pairs are not constructed explicitly.

Flow training. The flow head is trained by conditional flow matching with x_1 prediction (Section 4). For each training query the source is the current (uniformly perturbed) pose x_0 and the target x_1 is the oracle pose; the straight-line path is $x_t = (1 - t)x_0 + t\,x_1$ with $t \sim \text{Beta}(1, 4)$ (density $4(1 - t)^3$, mean 0.2), which puts most of the supervision mass on the small-t region where the deployment-time ODE integrator starts. The head predicts the endpoint $\hat{x}_1 = f_\psi(x_t, t, \phi_V(o), \phi_L(\ell))$ from which the velocity $\hat{v} = (\hat{x}_1 - x_t)/(1 - t)$ is recovered analytically; the loss is the per-sample weighted velocity error $w\|\hat{v} - (x_1 - x_0)\|^2$, with $w = 1$ for tuples drawn from successful demonstrations and $w = 0.3$ otherwise. To avoid the $1/(1 - t)$ singularity in Eq. 4 at $t \to 1$, we clip $1 - t$ to $\geq 10^{-3}$ during training. Training uses AdamW (lr $10^{-4} \to 10^{-6}$ cosine), batch 128, 150 epochs, gradient clip 1.0, bf16, seed 42. Checkpoints are selected by the integrated end-pose error on the validation split.

Layout-aware candidate sampler. Following N2M, the scorer ranks candidates sampled conditioned on per-layout fixture and floor polygons (provided by RoboCasa) and the robot footprint. Heading is sampled within $\pm 60^\circ$ of the oracle heading; candidates outside the reachable polygon are rejected and resampled.

Inference. At runtime the flow source x_0 is the robot's current base pose; the derived velocity is Euler-integrated from $t{=}0$ to $t{=}1$ to obtain the target pose. Because the x_1-prediction flow is near-straight, a single Euler step matches 10 steps to within 0.02 cm validation error (Section 5.5); MoFlo therefore deploys as a single forward pass of the encoder plus one of the flow head.

Table 1: Success rate (%) on five RoboCasa kitchen tasks (n=20 per cell). The block above the rule is MoFlo's pose-head ablation (encoder, backbone, and language conditioning held fixed); the block below is baselines. MoFlo is a single multi-task model conditioned on the instruction; all baselines are per-task and do not use the language instruction. All methods, including Mobi-π [17], are re-evaluated by us on the same five training layouts with the same object-placement distribution under a single protocol.

Method	Microwave	Drawer	Door	Faucet	Stove	**Mean**
MoFlo (ours)	65	100	100	85	50	**80**
MoFlo GMM-NLL head	60	95	85	70	55	73
MoFlo regression head	55	100	95	65	45	72
MoFlo CFM-flow (velocity param)	65	100	85	70	40	72
MoFlo score-only	25	65	50	30	20	38
N2M [18]	40	90	85	40	60	63
Mobi-π [17]	50	20	60	70	55	51
Reachability	35	80	65	30	30	48
Random	20	40	65	15	10	30

Table 2: Decomposition of the gap from N2M to MoFlo (5-task mean SR, %, n=20 per cell). Each row adds one design change. Language conditioning is the largest single contribution.

Configuration	Mean SR	Δ
N2M baseline	63	—
+ multi-task training	66	+3
+ language conditioning	73	+7
+ x_1-prediction flow head	**80**	+7

Baselines. **Random** draws a pose uniformly from a $\pm$20 cm box around the oracle pose with heading in $\pm 60^\circ$ of the oracle heading. **Reachability** [24] places the robot in front of the manipulation site at 0.30–0.85 m, heading constrained to $\pm 60^\circ$, and is instruction-agnostic. **Mobi-π** [17] runs a per-scene 3DGS [31] reconstruction, scores candidate poses by DINO [32] feature similarity to the policy's training views, and searches with Bayesian optimization. **N2M** [18] is a per-task Gaussian-mixture head trained by NLL on rollout poses, with no instruction conditioning; to isolate the modeling choice from corpus and capacity, N2M is trained on the same Point-BERT encoder and the same training tuples as MoFlo. For the head ablations (regression, velocity-CFM, GMM-NLL, score-only) the encoders, backbone, and language conditioning are held fixed and only the head differs.

C Full Main-Result Numbers

Table 1 lists the exact per-task success rates summarized in Figure 4 and Figure 8.

D Decomposition of the Gap to N2M

To attribute the gap from N2M to MoFlo, we add N2M's design changes one at a time (Table 2). Language conditioning and the x_1-prediction flow head are the two largest contributions, +7 points each, while multi-task training adds a smaller +3. Language conditioning resolves the task identity that the multi-task instruction-agnostic setting alone leaves ambiguous, and the x_1-prediction head supplies the endpoint precision that the tight-tolerance tasks require.

E Timestep-Schedule Sweep

We sweep the Beta concentration parameter $b \in \{1, 2, 3, 4, 5, 8\}$ at fixed seed (Table 3). The mean prefers small-t concentration over uniform sampling, because uniform sampling lets the late-t resid-

Table 3: **Timestep-schedule ablation.** Sweeping the Beta concentration parameter b on the x_1-prediction + velocity-loss recipe (Eq. 4) at fixed seed 42, identical data and architecture. Under x_1 prediction the velocity-loss residual is $(\hat{x}_1 - x_1)/(1-t)$, so the implicit per-sample pose-space loss weight at $t \sim \text{Beta}(1, b)$ is $b(1-t)^{b-3}$: b=1 (uniform) gives the broken $1/(1-t)^2$ that upweights deployment-irrelevant late-t samples and corrupts training; b=4 gives the deployment-aligned $4(1-t)$ that emphasizes small-t exactly where the ODE integrator starts; larger b undertrains the integrator endpoint. Success rate % on the five Seen-layout tasks at fixed seed, n=20 per cell.

b	Effective weight	MW	Drawer	Door	Faucet	Stove	**Mean**
1 (uniform)	$1/(1-t)^2$	50	100	100	65	50	73
2	$2/(1-t)$	50	100	90	75	45	72
3	3	55	100	90	75	55	75
4 **(Ours)**	$4(1-t)$	**65**	**100**	**100**	**85**	50	**80**
5	$5(1-t)^2$	55	100	95	70	45	73
8	$8(1-t)^5$	65	100	100	75	60	80

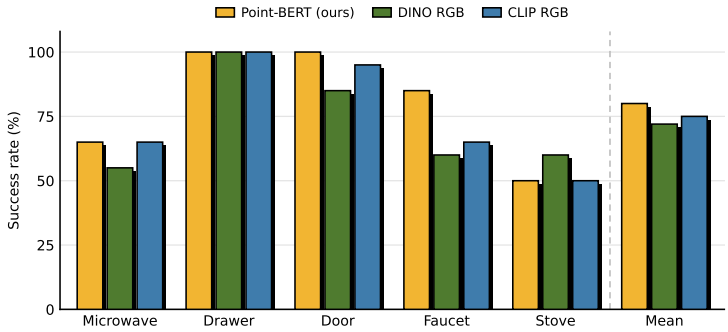


Figure 10: **Visual-encoder ablation.** MoFlo trained with a Point-BERT RGB-D point cloud, DINOv2 RGB, and CLIP RGB encoder.

ual dominate and corrupts small-t accuracy where the deployment ODE integrator starts. Within the small-t-concentrated range the 5-task mean is stable, and we adopt b=4 as our default. An ODE step-count sweep further confirms that the learned flow is well rectified at b=4, with 1-step and 10-step integration agreeing to 0.02 cm.

F Encoder Ablations

Visual encoder. We replace the Point-BERT RGB-D encoder with two 2D RGB encoders, DINOv2-base [59] and CLIP ViT-B/16, and retrain the pose head on the same augmented rollouts with identical hyperparameters (Figure 10). Removing the point cloud costs a modest, single-digit drop on the 5-task mean, concentrated on the tight-tolerance tasks where depth geometry matters most. MoFlo is therefore not point-cloud-dependent, and a 2D RGB backbone is a viable substitute.

Text encoder. We replace the default CLIP ViT-B/16 text encoder with three alternatives and retrain the pose head on a corpus whose language tokens are re-encoded by the new encoder, leaving the visual side unchanged (Figure 11, Table 4). The default CLIP ViT-B/16 gives the best 5-task mean and the larger CLIP ViT-L/14 stays comparable to it, while T5-base and the much larger Qwen3-Embedding both trail. Contrastive alignment to the visual stream is the dominant factor, and scaling text-side capacity alone does not transfer to base-pose prediction.

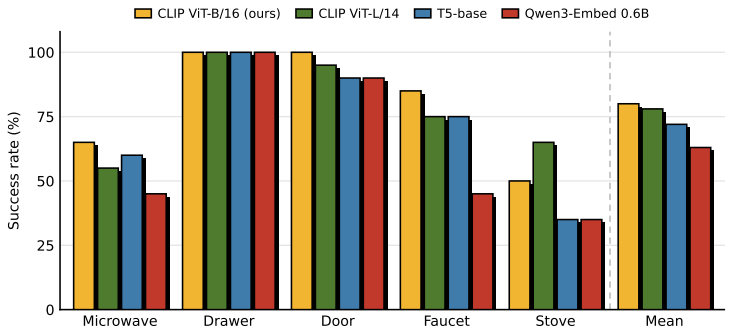


Figure 11: **Text-encoder ablation.** MoFlo trained on a corpus whose language tokens are re-encoded by each text encoder, with the visual side held fixed.

Table 4: **Text-encoder ablation.** MoFlo trained on a corpus whose language tokens are re-encoded by the listed text encoder; visual side (Point-BERT RGB-D), backbone, and all training hyperparameters are held fixed. Cells are success rate (%), n=20 per cell. The default CLIP ViT-B/16 (contrastively aligned to the Point-BERT visual features) gives the best mean; the larger CLIP ViT-L/14 trades -2 pp, while T5-base (-8 pp) and Qwen3-Embedding (-17 pp) trail further. Contrastive alignment to the visual stream is the dominant factor, not encoder scale.

Text encoder	dim	Microwave	Drawer	Door	Faucet	Stove	**Mean**
CLIP ViT-B/16 (default)	512	**65**	100	**100**	**85**	50	**80**
CLIP ViT-L/14	768	55	100	95	75	**65**	78
T5-base (encoder, language-only)	768	60	100	90	75	35	72
Qwen3-Embedding 0.6B	1024	45	100	90	45	35	63

G Real-World Experiment Details

Tasks. PICK asks the robot to pick a named object from a table; MoFlo and the multi-task visuomotor policy are both given "pick (object)", and MoFlo must select the correct object and reach a pose in front of it before the policy executes. CLOSEDRAWER asks the robot to close a specified drawer under instructions such as "close the left drawer" or "close the orange drawer", requiring navigation to the correct drawer and to a pose from which closing is likely to succeed. PUSHSHELF asks the robot to push a caster-mounted shelf toward a seated person under the instruction "push the shelf to the person", requiring it to infer the person's position and approach from the side consistent with the manipulation policy, which is trained to push from the front.

Protocol. A trial is a navigation success when the robot reaches a pose from which the target is manipulable, that is, visible in the robot's camera, and a manipulation success when the downstream policy then completes the task. We compare three base-placement methods. **Human** drives the base to where a person judges manipulation will succeed, **Mobi-π** reconstructs the scene and searches for a pose, and MoFlo predicts the pose from a single ego-centric frame.