A Training-Free Framework for Precise Mobile Manipulation of Small Everyday Objects

Author Names Omitted for Anonymous Review.

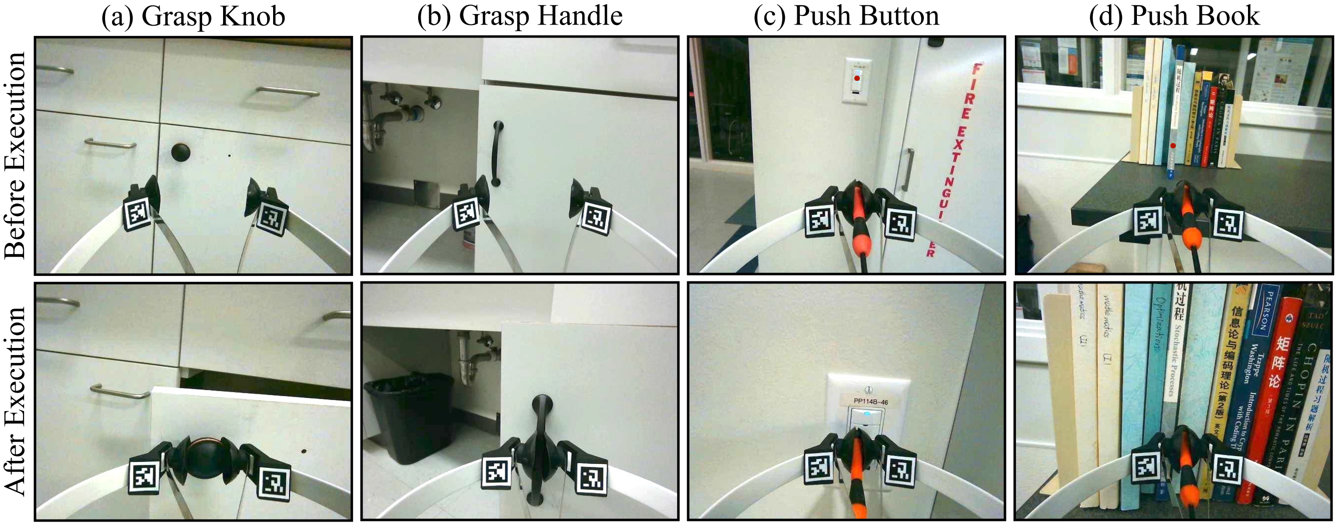

Fig. 1: Many everyday mobile manipulation tasks require reaching a precise interaction site before executing a motion primitive, *e.g.* precise reaching of a knob / handle to pull open a cupboard in (a) and (b), or precisely reaching a user-indicated button / book before pushing it in (c) and (d) (shown via the red dot). Open loop execution is unable to meet the high-precision needed for these tasks. In this paper, we develop Servoing with Vision Models (SVM), a training-free framework that closes the loop to enable a commodity mobile manipulator to tackle these tasks.

***Abstract*— Many everyday mobile manipulation tasks require precise interaction with small objects, such as grasping a knob to open a cabinet or pressing a light switch. In this paper, we develop Servoing with Vision Models (SVM), a closed-loop training-free framework that enables a mobile manipulator to tackle such precise tasks involving the manipulation of small objects. SVM employs an RGB-D wrist camera and uses visual servoing for control. Our novelty lies in the use of state-of-the-art vision models to reliably compute 3D targets from the wrist image for diverse tasks and under occlusion due to the end-effector. To mitigate occlusion artifacts, we employ vision models to out-paint the end-effector thereby significantly enhancing target localization. We demonstrate that aided by out-painting methods, open-vocabulary object detectors can serve as a drop-in module to identify semantic targets (*e.g.* knobs) and point tracking methods can reliably track interaction sites indicated by user clicks. This training-free method obtains an 85% zero-shot success rate on manipulating unseen objects in novel environments in the real world, outperforming an open-loop control method and an imitation learning baseline trained on 1000+ demonstrations by an absolute success rate of 50%.**

I. Introduction

Mobile manipulators hold the promise of performing a wide range of useful tasks in our everyday environments. However, a major obstacle to realizing this vision lies in the lack of precise mobile manipulation capabilities of current systems. Many real world mobile manipulation tasks require precise interaction with small objects, such as grasping a knob to pull open a cabinet or pressing a light switch, where even a small deviation can cause failure. A mobile manipulator's mobility makes these tasks even more challenging. For example, small errors during navigation, say on a thick carpeted surface, can easily exceed the tight tolerance required for precise tasks, and a non-holonomic base may limit precise repositioning. Furthermore, mobility means that mobile manipulators have to manipulate in varied locations under diverse lighting and unnatural viewpoints (*e.g.* looking top-down at a drawer from very close by), demanding significantly broader generalization than stationary manipulators confined to a fixed environment. As a result, developing mobile manipulators capable of performing precise tasks while generalizing to diverse environments remains an open problem.

Many precise mobile manipulation tasks involve *stylized* interactions with small objects: reaching a precise interaction site before executing a simple motion. The difficulty in these tasks lies in reaching the accurate pre-task pose around the small target object whereas the subsequent motion is easy to execute. For getting to the accurate pre-task pose, open loop execution using a sense-plan-act paradigm does not work because inaccuracies in perception and control prevent correct engagement with the small interaction site. This necessitates a closed loop approach. Imitation learning would be a natural choice, but as our experiments will show, policies trained with imitation fail to achieve sufficient precision and generalization even when trained on 1000+ real world demonstrations. So, how can we achieve precise mobile manipulation?

Visual servoing with an eye-in-hand camera is an effective technique to close the loop to precisely pursue targets [1] without requiring large-scale task-specific training for broad generalization, unlike closed-loop imitation learning. However, vanilla visual servoing makes strong assumptions, *e.g.* requiring known 3D objects or target images, which are not available in our in-the-wild setting. Our proposed approach, Servoing with Vision Models or SVM, marries together visual servoing with modern perception systems to mitigate these limitations. This leads to an effective system which is able to operate in a closed loop manner, and at the same time is versatile enough to operate in novel environments on previously unseen objects.

SVM leverages modern perception systems in two ways. First, we use them to specify targets for the visual servoing module. This alleviates the need for known 3D objects or target images. We experiment with two ways to specify targets: a) semantic categories, and b) points of interaction. For objects that have a well-known semantic category (*e.g.* drawer knobs or cabinet handles), we use an open-world object detector (*e.g.* Detic [2]) to continuously detect the target during visual servoing. However, not all mobile manipulation interaction sites, *e.g.* the different buttons on a microwave, correspond to a semantic category. We tackle such cases by using point trackers (*e.g.* CoTracker [3]) to continuously track a user-indicated interaction site (*e.g.* a single user click in the image, specifying which button on the microwave to push) over the course of visual servoing. Thus, the use of strong perception systems takes care of the target specification problem in visual servoing.

One problem however still remains. The use of visual servoing with an eye-in-hand camera for manipulation tasks suffers due to occlusion of the environment by the manipulator. Such occlusion can be particularly detrimental if it leads to out-of-distribution input to the perception system that now starts producing erroneous predictions (see second row of Figure 2). We mitigate this issue using yet another advance in computer vision: video in-painting models [4]–[6]. We out-paint the robot end-effector to obtain a *clean* view of the scene (see last row of Figure 2). This improves the detection performance of the vision system, leading to improved overall success.

We test SVM across several real world tasks: grasping a knob to pull open drawers / cupboards, grasping a handle to pull open a cabinet, pushing onto light buttons, and pushing books into place on bookshelves. We obtain a 85% success rate on these challenging tasks, *zero-shot on novel objects in previously unseen environments*. As expected, SVM performs much better than open-loop control which only succeeds on 35% trials. SVM without end-effector out-painting only obtains a 70% success rate, suggesting that occlusion caused by the end-effector does degrade performance and that our out-painting strategy is able to successfully mitigate it. Somewhat surprisingly, large vision models (Detic [2], CoTracker [3], and ProPainter [5] in our experiments) performs quite well on out-of-distribution wrist camera images.

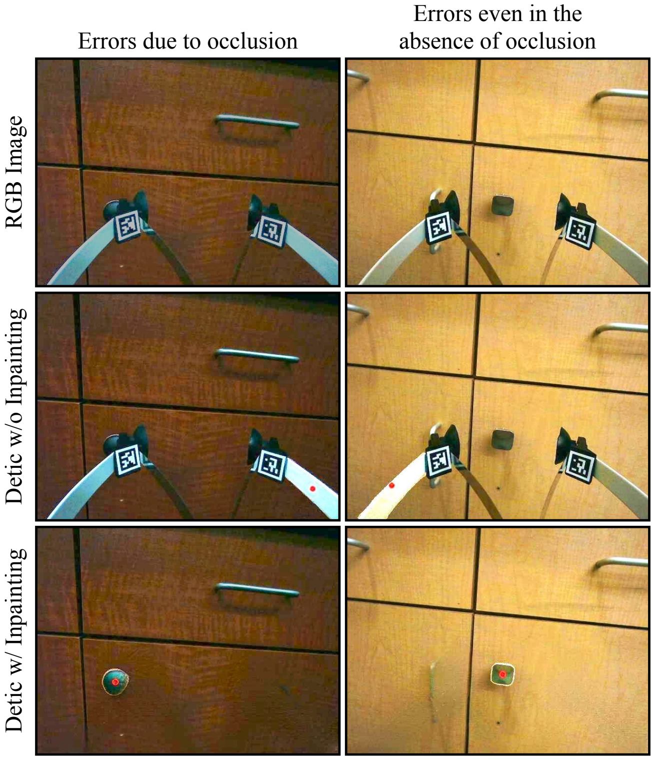

Fig. 2: When using off-the-shelf vision detectors on wrist camera data, knob detections (indicated by the red point in the second row) on the raw image are incorrect. Errors stem from occlusion due to the end-effector (**left**) and due to the presence of the end-effector (out-of-distribution object) in the image even when the target itself is unoccluded (**right**). Painting out the end-effector (**bottom row**) fixes this issue.

Perhaps most interesting is the comparison of SVM *vs.* imitation learning. As discussed before, precise, closed-loop tasks are precisely where one would expect imitation learning to excel over a modular approach such as SVM. So it is natural to ask how well does SVM perform *vs.* imitation learning. We conduct this comparison on the tasks of pulling on knobs and handles for opening articulated objects. Specifically, we compare to the recent Robot Utility Models (RUM) work from Etukuru *et al.* [7]. RUM is a closed-loop policy trained on 1200 demonstrations for opening cabinets and 525 demonstrations for opening drawers and thus serves as a very strong imitation learning comparison point. Quite surprisingly, we find that SVM outperforms RUM by an absolute success rate of 50%. We believe this was made possible by SVM's design: effectiveness and robustness of a) servoing for control and b) large-scale vision models for perception.

To summarize, we develop SVM, a training-free approach to enable precise mobile manipulation of small everyday objects. This is made possible by marrying together visual servoing with large-vision models. Our experiments reveal the effectiveness of our proposed approach, particularly over imitation learning, even when it is trained on 1000+ demonstrations. Our findings suggest that SVM can serve a practical and effective alternative to imitation learning for generalizable and precise mobile manipulation.

II. Related Work

A. Visual Servoing

Visual servoing (image-based, pose-based, and hybrid approaches) outputs control commands that convey the camera (and the attached manipulator) to a desired location with respect to the scene through [1], [8], [9]. Research has investigated use of different features to compute distance between current and target images: photometric distance [10], matching histograms [11], features from pre-trained neural networks [12], and has even trained neural networks to directly predict the relative geometric transformation between images [13]. Visual servoing has been applied for manipulation [14], navigation [15]–[17], 1-shot visual imitation [18] and also for seeking far away targets via intermediate view synthesis [19]. Most similar to our work, [20] leverage visual servoing to solve tasks with a similar structure. We differ in our training-free approach that leverages pre-trained vision models rather than training a model on a fixed set of objects. This allows us to interact with arbitrary user selected objects.

B. Eye-in-hand Imitation Learning

Imitation learning [21], [22] is a general tool for learning closed-loop manipulation policies and has been applied to eye-in-hand settings [7], [23]–[26]. However, this generality comes with the need for a large number of demonstrations for generalization [27]. Recent one-shot imitation learning methods [28], [29] leverage the structure of the task (getting to a bottleneck pose + motion replay) to learn from a single demonstration but are then restricted to interacting with the object they were trained on. We also leverage the same structure in tasks, but by employing vision foundation models trained on large datasets, our framework is able to operate on novel objects in novel environments.

C. Detection, Point Tracking, and In-painting

Training on Internet-scale datasets [30]–[32] with large-capacity models [33] has dramatically improved the generalization performance of vision systems. This coupled with alignment of visual representations with ones from language (*e.g.* CLIP [30]) has lead to effective open-vocabulary object detectors, *e.g.* Detic [2], OVR-CNN [34], OWL-ViT [35]. Similar advances in diffusion-based generative models [36]–[39] and large-scale training have led to effective image generation models. These models have been leveraged for image and video in-painting [4], [5]. In-painting models have also been used in robotics to mitigate domain gap between human and robot data [4], [40]. Last, point-based tracking in videos is seeing renewed interest in recent times [3], [41]–[43]. Given a set of 2D points in the first frame, these models are able to track them over a video. Use of machine learning makes these new approaches more robust than earlier versions [44]. Forecasts of point tracks into the future has been used as a intermediate representation for policy learning in robotics [45], [46].

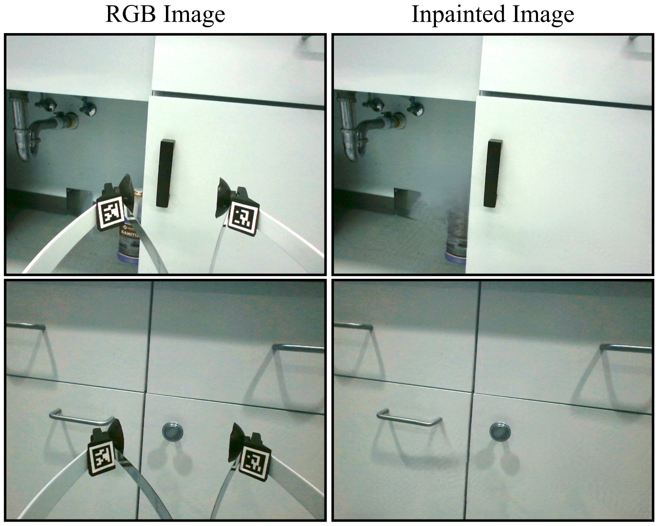

Fig. 3: Visualizations for end-effector out paintings.

III. Task

Many everyday household tasks involve precise manipulation followed by execution of a motion primitive. Examples include grasping a knob or a handle to pull open a drawers / cupboard, or pushing a button on a microwave. We consider two variants, where the interaction site can be identified via a semantic label (*e.g.* knob / handle) or via a user-specified point (*e.g.* a click on an image specifying the button to push); and assume that the motion primitive is given or easy to specify. Our goal is to enable a commodity mobile manipulator equipped with a RGB-D wrist camera to accomplish such tasks in previously unseen environments.

IV. Method

At a high-level, our method employs visual servoing on eye-in-hand camera images (Section IV-C) to control the end-effector to reach the interaction site. Our innovation lies in the use of state-of-the-art vision models to reliably detect / track the interaction site (Section IV-B) to provide the visual feedback for visual servoing. As we will see, occlusion due to the end-effector in the wrist camera view causes nuisance. We deal with this by painting out the end-effector (Section IV-A) before running the perception models on images. Let's denote images from the wrist camera with I_t, current robot state by x_t. The output actions are computed as follows:

$$a_t = \pi\left(x_t, g\left(f\left(I_t, [I_1, \ldots, I_{t-1}]\right)\right)\right), \tag{1}$$

where $f(I, \mathbf{I})$ is a video inpainting function that paints out the end-effector from image I using images in $\mathbf{I}$ as reference, $g(I)$ localizes the target in 3D in the wrist camera frame,

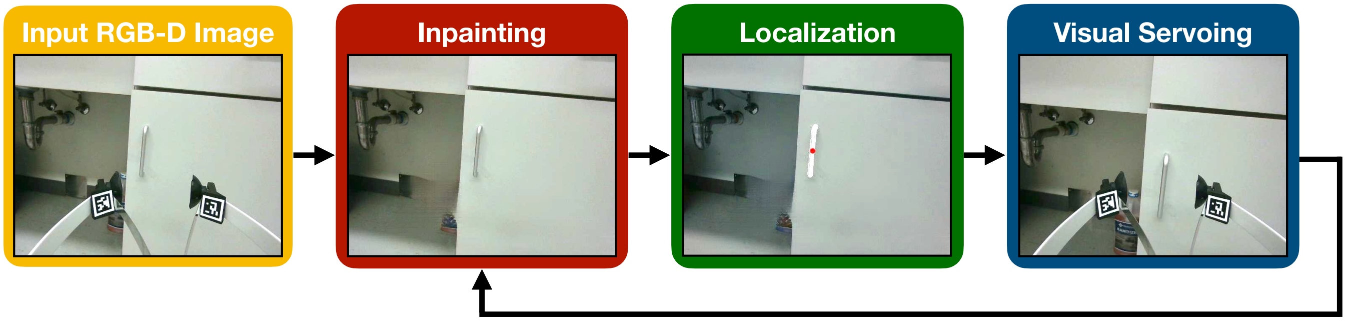

Fig. 4: Servoing with Vision Models (SVM) is a framework for precise reaching for mobile manipulators. Starting from an input RGB-D wrist camera image with a target specified either via a semantic label (*e.g.* handle) or a user-clicked point on the image, SVM outputs whole-body control commands to convey the end-effector to the target location by closing the loop with visual feedback. SVM first paints out the end-effector using a video outpainting model (Section IV-A), uses vision models to continuously detect the target object (or track the desired target point) to compute 3D servoing targets (Section IV-B), which are passed to a servo to obtain whole-body control commands (Section IV-C).

and π computes the desired joint velocities using the current robot state x_t and the current target location output by g. Figure 4 shows an overview of our proposed method and we describe each component below.

A. Inpainting

Given RGB images from the wrist camera, the inpainting function f uses past frames from the wrist camera to inpaint the current frame I_t. We utilize a video inpainting method (as opposed to an image inpainting method) for better performance: a video inpainting model has access to previous frames (where the object may not be occluded), which can lead to improved inpainting. We adopt the ProPainter model [5] to realize f. It is a transformer-based video inpainting model trained on the YouTube-VOS [47] dataset.

If the object of interest is occluded by the end-effector in the first frame, even a video inpainting method may not be able to accurately reconstruct the object. To combat this, we design a "look-around" primitive that moves the end-effector around (vertically and laterally) to obtain contextual information about the scene. To limit the inference time in each iteration, we limit the inpainting model to only look at the ten past images. The "look-around" provides an initial set of ten images.

Out painting the end-effector also requires a mask of the end-effector. We use a manually constructed mask that coarsely covers the end-effector. We find this to work better for out painting than a fine mask of the end-effector obtained using the segmentation model SAM [32].

B. Interaction Site Localization

Given an image with the end-effector painted out, our next goal is to localize the object of interest to obtain 3D location for the target. We handle the two specifications for the interaction site, via a semantic label or a user click, separately as described below.

1) Detection: For semantically specified targets (*e.g.* knobs / handles), we use Detic [2], an open-vocabulary detector trained on large-scale datasets. We prompt Detic with the object class 'handle' for handles and 'knob' for knobs. If Detic detects multiple handles in the image, we select the handle closest to the center of the image. Detic also outputs a mask for the object. We compute the center of the mask and use this as 2D position of the object of interest.

2) Tracking: For tasks specified via a user click, *e.g.* the point on the book or the button in Figure 1, we use of CoTracker [3], a point tracking method for videos. Given a point in the first frame, CoTracker is able to track it over subsequent frames seen during execution. CoTracker notes that tracking performance is better when tracking many points together. We therefore sample 40 points randomly around the user click and found that to drastically improve tracking performance.

Either of these methods provides a 2D location in the image. We lift this 2D target location to 3D using the depth image. In the case that the 2D position of the target point is within the mask of the end-effector (i.e. occluded by the end-effector), we utilize the depth from the nearest previous frame for this 3D lifting.

C. Closed-loop Control

Given the interaction sites' 3D location, we employ visual servoing to realize π to compute velocity control commands. Visual servoing computes whole-body velocity control commands that minimize the distance between the end-effector and the 3D target point [1].

V. Experiments

Our experiments are designed to test a) the extent to which open loop execution is an issue for precise mobile manipulation tasks, b) how effective are blind proprioceptive correction techniques, c) do object detectors and point trackers perform reliably enough in wrist camera images for reliable control, d) is occlusion by the end-effector an issue and how effectively can it be mitigated through the use of video in-painting models, and e) how does our proposed SVM methodology compare to large-scale imitation learning?

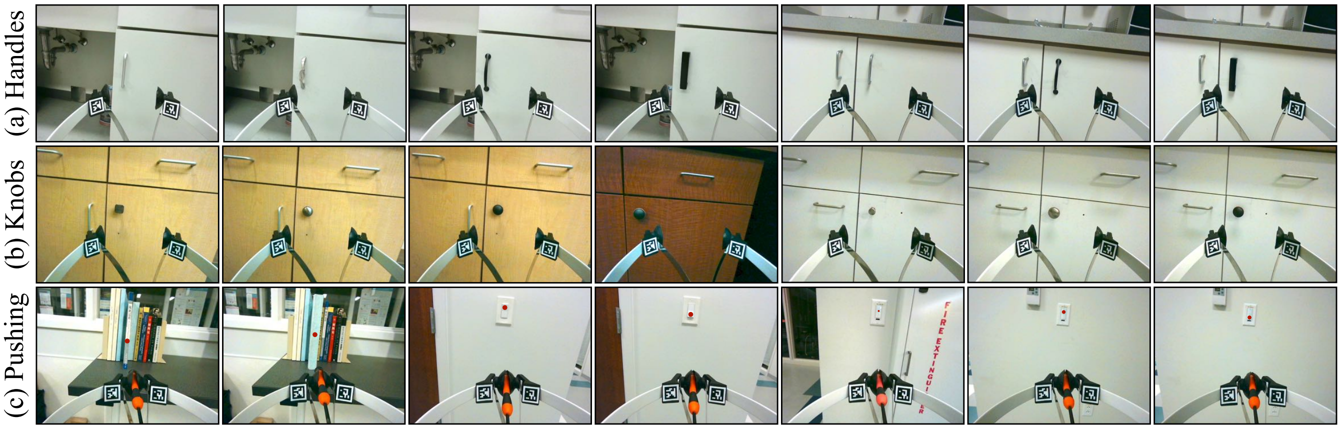

Fig. 5: Visualization of objects and environments for the three different tasks: a) and b) pulling on a variety of handles / knobs to open articulated objects and c) pushing on user defined objects (books in a bookshelf and buttons). Note that we exclusively test on novel objects in novel environments not used for development in any manner.

A. *Tasks and Experimental Setup*

We work with the Stretch RE2 robot. Stretch RE2 is a commodity mobile manipulator with a 5DOF arm mounted on top of a non-holomonic base. We upgrade the robot to use the Dex Wrist 3, which has an eye-in-hand RGB-D camera (Intel D405). We consider 3 task families for a total of 6 different tasks: a) holding a knob to pull open a cabinet or drawer, b) holding a handle to pull open a cabinet, and c) pushing on objects (light buttons, books in a book shelf, and light switches). Our focus is on generalization. *Therefore, we exclusively test on previously unseen instances, not used during development in any way.* Figure 5 shows the instances that we test on.

All tasks involve some precise manipulation, followed by execution of a motion primitive. **For the pushing tasks**, the precise motion is to get the end-effector exactly at the indicated point and the motion primitive is to push in the direction perpendicular to the surface and retract the end-effector upon contact. The robot is positioned such that the target position is within the field of view of the wrist camera. A user selects the point of pushing via a mouse click on the wrist camera image. The goal is to push at the indicated location. Success is determined by whether the push results in the desired outcome (light turns on / off or book gets pushed in). The original rubber gripper bends upon contact, we use a rigid known tool that sticks out a bit. We take the geometry of the tool into account while servoing.

For the opening articulated object tasks, the precise manipulation is grasping the knob / handle, while the motion primitive is the whole-body motion that opens the cupboard. Computing and executing this full body motion is difficult. We adopt the modular approach to opening articulated objects (MOSART) from Gupta *et al.* [48] and invoke it after the gripper has been placed around the knob / handle. The whole tasks starts out with the robot about 1.5m way from the target object, with the target object in view from robot's head mounted camera. We use MOSART to compute articulation parameters and convey the robot to a pre-grasp location with the target handle in view of the wrist camera. At this point, SVM (or baseline) is used to center the gripper around the knob / handle, before resuming MOSART: extending the gripper till contact, close the gripper, and play rest of the predicted motion plan. Success is determined by whether the cabinet opens by more than 60° or the drawer is pulled out by more than 24*cm*, similar to the criteria used in [48].

For the precise manipulation part, all baselines consume the current and previous RGB-D images from the wrist camera and output full body motor commands.

B. *Baselines*

We compare against three other methods for the precise manipulation part of these tasks.

1) Open Loop (Eye-in-Hand): To assess the precision requirements of the tasks and to set it in context with the manipulation capabilities of the robot platform, this baseline uses open loop execution starting from estimates for the 3D target position from the first wrist camera image.

2) MOSART [48]: The recent modular system for opening cabinets and drawers [48] reports impressive performance with open-loop control (using the head camera from 1.5m away), combined with proprioception-based feedback to compensate for errors in perception and control when interacting with handles. We test if such correction is also sufficient for interacting with knobs. Note that such correction is not possible for the smaller buttons and pliable books.

3) SVM (no inpainting): To understand how much of an issue occlusion due to the end-effector is during manipulation, we ablate the use of inpainting.

4) Robot Utility Models (RUM) [7]: For the opening articulated object tasks, we also compare to Robot Utility Models (RUM), a closed-loop imitation learning method recently proposed by Etukuru et al. [7]. RUM is trained on a substantial dataset comprising expert demonstrations, including 1,200 instances of cabinet opening and 525 of drawer opening, gathered from roughly 40 different environments. This dataset stands as the most extensive imitation

TABLE I: Execution Success Rates. We compare Servoing with Vision Models (SVM) to a previous system (MOSART [48]), open loop execution using target computed in the wrist camera, and a version of SVM without inpainting. Tasks require precise control and open loop execution fails. MOSART's contact correction works for handles but struggles with knobs and it can't handle user-clicked targets. In-painting matters for semantic targets.

	Semantic Targets			**User-clicked Targets**			**Total**
	Knobs (Cabinets)	**Knobs (Drawers)**	**Handles (Cabinets)**	**Light Button**	**Book**	**Light Switch**	
MOSART [48]	1/7	0/7	**6/7**	-	-	-	7/28
Open Loop (Eye-in-Hand)	2/7	0/7	**6/7**	0/3	**2/2**	0/2	10/28
Ours w/o inpainting	**6/7**	3/7	4/7	**3/3**	**2/2**	**2/2**	20/28
Ours	**6/7**	**5/7**	**6/7**	**3/3**	**2/2**	**2/2**	**24/28**

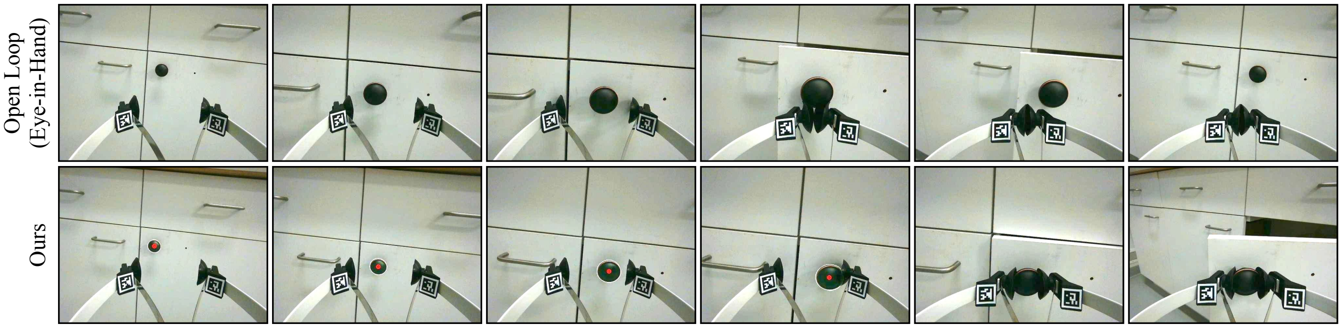

Fig. 6: Comparison of SVM with the open loop (eye-in-hand) baseline for opening a cabinet with a knob. Slight errors in getting to the target cause the end-effector to slip off, leading to failure for the baseline, where as our method is able to successfully complete the task.

	Knobs		**Handle**	**Total**
	Cabinets	**Drawer**	**Cabinets**	
RUM [7]	0/3	1/4	1/3	2/10
SVM (Ours)	2/3	2/4	3/3	7/10

TABLE II: Comparison of SVM *vs.* RUM [7], a recent large-scale end-to-end imitation learning method trained on 1200 demos for opening cabinets and 525 demos for opening drawers across 40 different environments. Our evaluation spans objects from three environments across two buildings.

learning dataset for articulated object manipulation to date, establishing RUM as a strong baseline for our evaluation.

Similar to our method, we use MOSART to compute articulation parameters and convey the robot to a pre-grasp location with the target handle in view of the wrist camera. One of the assumptions of RUM is a good view of the handle. To benefit RUM, we try out three different heights of the wrist camera, and *report the best result for RUM.*

C. Results

Table I presents results from our experiments. Our training-free approach SVM successfully solves over 85% of task instances that we test on. As noted, all these tests were conducted on unseen object instances in unseen environments that were not used for development in any way. We discuss our key experimental findings below.

1) Closing the loop is necessary for these precise tasks: While the proprioception-based strategies proposed in MOSART [48] work out for handles, they are inadequate for targets like knobs and just don't work for tasks like pushing buttons. Using estimates from the wrist camera is better, but open loop execution still fails for knobs and pushing buttons.

2) Vision models work reasonably well even on wrist camera images: Inpainting works well on wrist camera images (see Figure 2 and Figure 3). Closing the loop using feedback from vision detectors and point trackers on wrist camera images also work well, particularly when we use in-painted images. See some examples detections and point tracks in Figure 6 and Figure 7. Detic [2] was able to reliably detect the knobs and handles and CoTracker [3] was able to successfully track the point of interaction letting us solve 24/28 task instances.

3) Erroneous detections without inpainting hamper performance on handles and our end-effector out-painting strategy effectively mitigates it: As shown in Figure 8, presence of the end-effector caused the object detector to miss fire leading to failed execution. Our out painting approach mitigates this issue leading to a higher success rate than the approach without out-painting. Interestingly, CoTracker [3] is quite robust to occlusion (possibly because it tracks multiple points) and doesn't benefit from in-painting.

4) Closed-loop imitation learning struggles on novel objects: As presented in Table II, SVM significantly outperforms RUM in a paired evaluation on unseen objects across three novel environments. A common failure mode

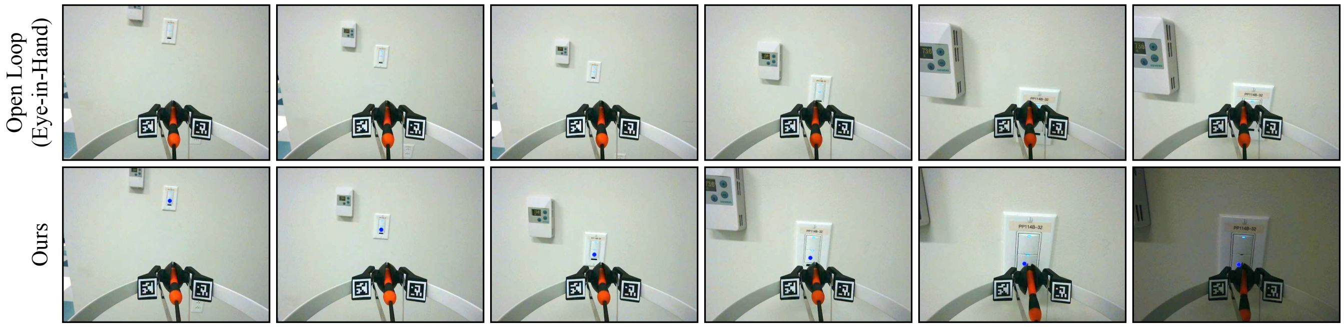

Fig. 7: SVM *vs.* open loop (eye-in-hand) baseline for pushing on user-clicked points. Slight errors in getting to the target cause failure, where as SVM successfully turns the lights off. Note the quality of CoTracker's track (blue dot).

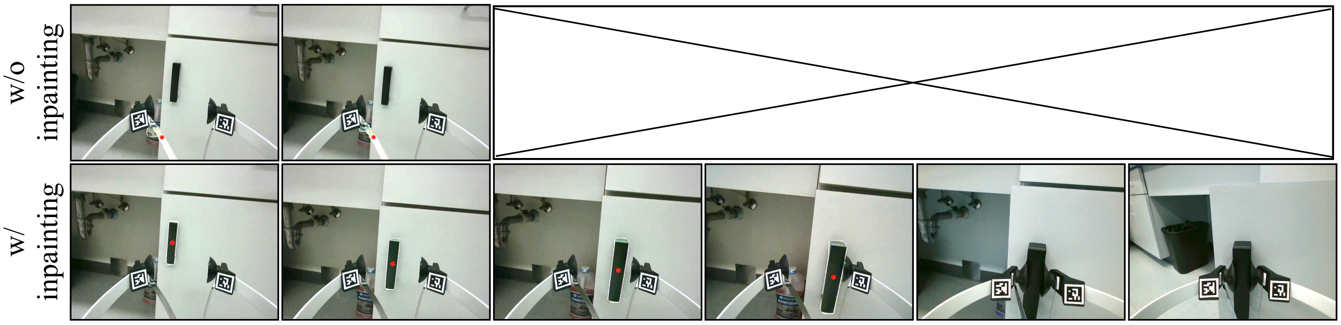

Fig. 8: Comparison of SVM with and without inpainting. Erroneous detection without inpainting causes execution to fail, where as with inpainting the target is correctly detected leading to a successful grasp and a successful execution.

of RUM is its inability to grasp the object's handle, even when it approaches it closely. Another failure mode we observe is RUM misidentifying keyholes or cabinet edges as handles, also resulting in failed grasp attempts. These result demonstrate that a modular approach that leverages the broad generalization capabilities of vision foundation models is able to generalize much better than an end-to-end imitation learning approach trained on 1000+ demonstrations, which must learn all aspects of the task from scratch.

VI. Discussion

In this paper, we describe SVM, a training-free framework for precise manipulation tasks that involve reaching a precise interaction site followed by execution of a primitive motion. Strong vision models help us mitigate issues caused by occlusion by the end-effector thereby enabling the use of off-the-shelf open-vocabulary object detectors and point trackers to estimate servoing targets during execution. Use of strong off-the-shelf models also provides broad generalization for perception while the use of servoing provides robust control. This enables SVM to solve tasks in novel environments on novel objects, obtaining a 85% zero-shot success rate across 4 precise mobile manipulation tasks. Most surprisingly, even though SVM is modular, it outperforms the strong end-to-end imitation learning system RUM [7] that was trained on 1000+ demonstrations. This is particularly striking as imitation learning is the tool of choice for precise tasks requiring closed-loop execution.

VII. Limitations

Even though SVM performs quite well across many tasks on novel objects in novel environments, it suffers from some shortcomings. Running these large vision models is computationally expensive and we have to off load computation to a A40 GPU sitting in a server. Even with this GPU, we are only able to run the vision pipeline at a 0.1 Hz leading to slow executions. Building vision models specialized to wrist camera images may work better and faster. A second limitation is the reliance on depth from the wrist camera which may be poor in some situations *e.g.* shiny or dark objects. Use of learned disparity estimators [49] with stereo images could mitigate this. As our focus is on the precise reaching of interaction sites, we work with a hand-crafted task decomposition into the interaction site and primitive motion. In the future, we could obtain such a decomposition using LLMs, or from demonstrations, thereby expanding the set of tasks we can tackle.

References

[1] F. Chaumette, S. Hutchinson, and P. Corke, "Visual servoing," *Springer handbook of robotics*, pp. 841–866, 2016.

[2] X. Zhou, R. Girdhar, A. Joulin, P. Krähenbühl, and I. Misra, "Detecting twenty-thousand classes using image-level supervision," in *ECCV*, 2022.

[3] N. Karaev, I. Rocco, B. Graham, N. Neverova, A. Vedaldi, and C. Rupprecht, "CoTracker: It is better to track together," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024.

[4] M. Chang, A. Prakash, and S. Gupta, "Look ma, no hands! agent-environment factorization of egocentric videos," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[5] S. Zhou, C. Li, K. C. Chan, and C. C. Loy, "ProPainter: Improving propagation and transformer for video inpainting," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2023.

[6] S. Lee, S. W. Oh, D. Won, and S. J. Kim, "Copy-and-paste networks for deep video inpainting," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019.

[7] H. Etukuru, N. Naka, Z. Hu, S. Lee, J. Mehu, A. Edsinger, C. Paxton, S. Chintala, L. Pinto, and N. M. M. Shafiullah, "Robot utility models: General policies for zero-shot deployment in new environments," *arXiv preprint arXiv:2409.05865*, 2024.

[8] P. I. Corke, "Visual control of robot manipulators–a review," *Visual Servoing: Real-Time Control of Robot Manipulators Based on Visual Sensory Feedback*, pp. 1–31, 1993.

[9] F. Chaumette and S. Hutchinson, "Visual servo control, Part I: Basic approaches," *IEEE Robotics & Automation Magazine (RAM)*, 2006.

[10] C. Collewet and E. Marchand, "Photometric visual servoing," *IEEE Transactions on Robotics (T-RO)*, 2011.

[11] Q. Bateux and E. Marchand, "Histograms-based visual servoing," *IEEE Robotics and Automation Letters (RA-L)*, 2016.

[12] A. X. Lee, S. Levine, and P. Abbeel, "Learning visual servoing with deep features and fitted q-iteration," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[13] Q. Bateux, E. Marchand, J. Leitner, F. Chaumette, and P. Corke, "Training deep neural networks for visual servoing," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[14] F. Sadeghi, A. Toshev, E. Jang, and S. Levine, "Sim2real viewpoint invariant visual servoing by recurrent control," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[15] F. Sadeghi, "Divis: Domain invariant visual servoing for collision-free goal reaching," *arXiv preprint arXiv:1902.05947*, 2019.

[16] C. Li, B. Li, R. Wang, and X. Zhang, "A survey on visual servoing for wheeled mobile robots," *International Journal of Intelligent Robotics and Applications*, vol. 5, no. 2, pp. 203–218, 2021.

[17] M. N. Qureshi, P. Katara, A. Gupta, H. Pandya, Y. Harish, A. Sanchawala, G. Kumar, B. Bhowmick, and K. M. Krishna, "Rtvs: A lightweight differentiable mpc framework for real-time visual servoing," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 3798–3805.

[18] M. Argus, L. Hermann, J. Long, and T. Brox, "Flowcontrol: Optical flow based visual servoing," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 7534–7541.

[19] N. Crombez, J. Buisson, Z. Yan, and Y. Ruichek, "Subsequent keyframe generation for visual servoing," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[20] J. A. Collins, C. Houff, Y. L. Tan, and C. C. Kemp, "Forcesight: Text-guided mobile manipulation with visual-force goals," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[21] D. A. Pomerleau, "Efficient training of artificial neural networks for autonomous navigation," *Neural computation*, vol. 3, no. 1, pp. 88–97, 1991.

[22] S. Schaal, "Learning from demonstration," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 9, 1996.

[23] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song, "Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots," in *Robotics: Science and Systems (RSS)*, 2024.

[24] S. Young, D. Gandhi, S. Tulsiani, A. Gupta, P. Abbeel, and L. Pinto, "Visual imitation made easy," in *Proceedings of the Conference on Robot Learning (CoRL)*. PMLR, 2021, pp. 1992–2005.

[25] X. Zhang, M. Chang, P. Kumar, and S. Gupta, "Diffusion meets dagger: Supercharging eye-in-hand imitation learning," in *Robotics: Science and Systems (RSS)*, 2024.

[26] N. M. M. Shafiullah, A. Rai, H. Etukuru, Y. Liu, I. Misra, S. Chintala, and L. Pinto, "On bringing robots home," *arXiv preprint arXiv:2311.16098*, 2023.

[27] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis *et al.*, "Droid: A large-scale in-the-wild robot manipulation dataset," *arXiv preprint arXiv:2403.12945*, 2024.

[28] E. Valassakis, G. Papagiannis, N. Di Palo, and E. Johns, "Demonstrate once, imitate immediately (dome): Learning visual servoing for one-shot imitation learning," in *International Conference on Intelligent Robots and Systems*, 2022.

[29] B. Wen, W. Lian, K. Bekris, and S. Schaal, "You only demonstrate once: Category-level manipulation from single visual demonstration," in *Robotics: Science and Systems (RSS)*, 2022.

[30] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2021.

[31] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat *et al.*, "GPT-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[32] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment anything," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2023.

[33] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[34] A. Zareian, K. D. Rosa, D. H. Hu, and S.-F. Chang, "Open-vocabulary object detection using captions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[35] M. Minderer, A. Gritsenko, A. Stone, M. Neumann, D. Weissenborn, A. Dosovitskiy, A. Mahendran, A. Arnab, M. Dehghani, Z. Shen, X. Wang, X. Zhai, T. Kipf, and N. Houlsby, "Simple open-vocabulary object detection with vision transformers," 2022. [Online]. Available: https://arxiv.org/abs/2205.06230

[36] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2015.

[37] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

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

[39] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," *arXiv preprint arXiv:2011.13456*, 2020.

[40] S. Bahl, A. Gupta, and D. Pathak, "Human-to-robot imitation in the wild," in *Robotics: Science and Systems (RSS)*, 2022.

[41] A. W. Harley, Z. Fang, and K. Fragkiadaki, "Particle video revisited: Tracking through occlusions using point trajectories," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.

[42] C. Doersch, Y. Yang, M. Vecerik, D. Gokay, A. Gupta, Y. Aytar, J. Carreira, and A. Zisserman, "Tapir: Tracking any point with per-frame initialization and temporal refinement," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2023.

[43] Y. Zheng, A. W. Harley, B. Shen, G. Wetzstein, and L. J. Guibas, "Pointodyssey: A large-scale synthetic dataset for long-term point tracking," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2023.

[44] J. Shi and C. Tomasi, "Good features to track," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1994.

[45] C. Wen, X. Lin, J. So, K. Chen, Q. Dou, Y. Gao, and P. Abbeel, "Any-point trajectory modeling for policy learning," in *Robotics: Science and Systems (RSS)*, 2024.

[46] H. Bharadhwaj, R. Mottaghi, A. Gupta, and S. Tulsiani, "Track2act: Predicting point tracks from internet videos enables diverse zero-shot robot manipulation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024.

[47] N. Xu, L. Yang, Y. Fan, J. Yang, D. Yue, Y. Liang, B. L. Price, S. Cohen, and T. S. Huang, "Youtube-vos: Sequence-to-sequence video object segmentation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[48] A. Gupta, M. Zhang, R. Sathua, and S. Gupta, "Opening articulated objects in the real world," *arXiv*, vol. 2402.17767, 2024.

[49] G. Xu, X. Wang, X. Ding, and X. Yang, "Iterative geometry encoding volume for stereo matching," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.