# Morphology-Agnostic Visual Robotic Control

**Brian Yang**[*†] **Dinesh Jayaraman**[*†‡] **Glen Berseth**[†] **Alexei Efros**[†] **Sergey Levine**[†]
† UC Berkeley ‡ Facebook AI Research

## Abstract

Existing approaches for visuomotor robotic control typically require characterizing the robot in advance by calibrating the camera or performing system identification. We propose MAVRIC, an approach that works with minimal prior knowledge of the robot's morphology, and requires only a camera view containing the robot and its environment and an unknown control interface. MAVRIC revolves around a mutual information-based method for self-recognition, which discovers visual "control points" on the robot body within a few seconds of exploratory interaction, and these control points in turn are then used for visual servoing. MAVRIC can control robots with imprecise actuation, no proprioceptive feedback, unknown morphologies including novel tools, unknown camera poses, and even unsteady handheld cameras. We demonstrate our method on visually-guided 3D point reaching, trajectory following, and robot-to-robot imitation.

## 1 Introduction

Current robotic control methods typically require precise knowledge of the robot's configuration and kinematics. As an example, a typical rigid robot arm's degrees of freedom are fully specified by its joint angles, available through servomotor encoders. Such proprioception-driven control methods do not generalize to many important settings. What if the robot were made of deformable material, so that its degrees of freedom are not easily enumerated or measured? Even for the rigid robot arm above, introduce a previously unseen chalk piece into its gripper and the position of its tip is now unknown and therefore not possible to control. Humans easily handle such control tasks by relying on rich sensory feedback, such as from vision, rather than purely on precise proprioception.

We propose MAVRIC, a self-recognition-enabled approach to IBVS that works "out of the box" on arbitrary new or altered robots with no manual specification of any points of interest. We use simple techniques to accomplish this: a mutual information measure [1] evaluates the responsiveness of various points in the environment, tracked using Lucas-Kanade optical flow computation [2], to the robot control commands. MAVRIC is lightweight, flexible, and fast to adapt, producing responsive "control points" for a new robot within a few seconds of interaction. As we will show, MAVRIC handles settings that are challenging to today's state-of-the-art robotic control approaches: imprecise actuation, unknown robot morphologies, unknown camera poses, novel unmodeled tools, and unsteady handheld cameras.

## 2 MAVRIC: Morphology-Agnostic Visual Robotic Control

We operate in the following setting: at each time step $t$, a controller has access to raw RGBD image observations from a camera, and the ability to set a $d$-dimensional control input $A(t)$ for the robot. The images contain the robot's body as well as other portions of its environment.

To maximize generality, we make very few assumptions about factors that are commonly treated as fully known in robotic control: (i) We do not know the nature of robot's embodiment, such as the degrees of freedom, rigidity, or the number and lengths of links in a robot arm. (ii) For the control interface, aside from the standard assumptions made in uncalibrated visual servoing, we make one

NeurIPS 2019 Workshop on Robot Learning: Control and Interaction in the Real World, Vancouver, Canada


Figure 1: [Best seen in pdf] Self-recognition for MRCP search illustrated on data from a handheld, shaky camera. Frame 1 shows the camera view of the robot during exploration. Frame 2 shows the results of coarse tracking on the exploration video, where the track points overlaid on the image are colored more green if they are more responsive. Frame 3 shows how fine tracking is initialized on the same video around most responsive tracks from the coarse stage, and Frame 4 shows surviving tracks after the fine stage of MRCP search. Frame 5 shows the final result, with the top-15 control points in green and their average position in red. Video in Supp.

additional assumption that the displacements of points on the robot are a probabilistic function of the control commands. We do not assume any kind of camera calibration whatsoever.

### 2.1 Self-Recognition: Robot as a Collection of Responsive Particles

In the self-recognition phase, we aim to resolve the question: what is the body of the robot? We start by decomposing the observable environment containing the robot into "particles"— points in 3D space that may or may not be part of its body. Each particle $P_i$ has an associated position $S_i(t)$ in RGBD camera coordinates ($x$, $y$, depth) from our uncalibrated camera.

**Responsiveness** We define the responsiveness of a particle as the mutual information (MI) [1; 3] between its motions and the control inputs. Specifically, we execute a random sequence of exploratory control commands $A(t)$ assign a non-negative responsiveness score to each particle $P_i$:

$$R_i \triangleq I(\Delta S_i; A), \tag{1}$$

where $\Delta S_i(t) = S_i(t+1) - S_i(t)$ is the change in position of $P_i$ in response to $A(t)$, and $I(\cdot;\cdot)$ is the MI between two random variables.

**Control Points and MRCP** We define the "body" $B_\delta$ of a robot as the set of particles whose responsiveness is higher than some threshold $\delta$: $B_\delta \triangleq \{P_i : R_i > \delta\}$. We call the constituent particles of this set "control points." In Appendix A, we discuss why this aligns with our intuitive definition of a robot's body, in our setting.

The maximally responsive control point (MRCP) $P^*$ is the particle with the highest responsiveness $R^* = \max_i R_i$, with position $S^*(t)$. In practice, we average the positions of the top-$k$ most responsive particles to compute a robust MRCP.

**Handling Rigid Objects** Points on a rigid object, such as a single link of a standard robot arm, all exhibit the same motion, modulo invertible affine transformations. Mutual information is known to be invariant under such smooth, invertible mappings (see [4] for a simple proof). This has the problematic implication that points on a rigid object all have the same responsiveness score. To break such ties, we preferentially select points with larger motions by exploiting the fact that any loss of precision leads naturally to a preference for large motions. We add Gaussian noise to $\Delta S_i$ in Eq. 1 to artificially lower the precision and express this preference for large motions.

**Implementation Details** For tracking the positions $S_i(t)$ of particles over time, we use the Lucas-Kanade optical flow estimator [2]. We use the mutual information estimator proposed in [4], as implemented in [5]. In our experiments, control points on a robot arm are discovered within 20 seconds of exploration, sufficient to execute about 100 randomly sampled small actions. Finally, we find it useful to search for control points in a coarse-to-fine manner, using the most responsive points to initialize a second round of tracking. See Appendix B for more details. Fig 1 shows an example of the various stages of MRCP identification with a handheld, shaky camera.

**Prior Work** Prior methods have been proposed that learn to recognize the robot's body [6; 7; 8]. Early methods use simple temporal correspondences between actions and observed motion in the visual field to identify the body — [6] use learned characteristic response delays, and [8] rely on periodicity. Closest to MAVRIC, [7] also use a mutual information-based approach for self-recognition, but different from MAVRIC, they assume known aspects of the robot's morphology, such as the number of its links and the visual appearance of its body, and the correspondences between action dimensions and the various servos on the robot, and also rely on manually demonstrated robot poses during exploration. Despite these advantages, they report requiring four minutes of exploration, compared to about 20s for MAVRIC. They also propose a different approach to tracking, which we empirically compare against in our setting.

### 2.2 Visual Servoing in Control Point Coordinates

Once the MRCP is identified, MAVRIC performs uncalibrated image-based visual servoing [9; 10; 11; 12; 13; 14; 15; 16; 17; 18; 19; 20; 21; 22; 23; 24; 25] to transport the MRCP point $S^*$ to a specified goal point $G$. We use online regression to fit $(A, \Delta S^*)$ tuples to estimate local Jacobian matrices "on the fly," using modified Broyden updates [11; 12]. See Appendix C for more details. The Jacobian matrix initialization $J_0$ is computed as follows: we start at a random arm position, sequentially set the control inputs $A_i$ to scaled unit vectors $\epsilon e_i$ along each control dimension, and set the $i$-th column of $J_0$ to the response $\Delta S_i/\epsilon$. In our experiments, we repeat this initialization procedure whenever servoing has failed to get closer to the target in the last 20 steps.

## 3 Experiments

We perform experiments using the REPLAB standardized hardware platform [26; 27] with an imprecise low-cost manipulator (Trossen WidowX) and an RGB-D camera (Intel Realsense SR300). We evaluate how well MAVRIC's self-recognition phase works under varied conditions, and also the overall effectiveness of MAVRIC for visuomotor servoing tasks.

### 3.1 Self-Recognition: Discovering End Effectors, Tools, and Robot Morphology

In each run, a sequence of 100 random exploratory control commands are executed, which requires about 20 seconds of interaction. Fig 3 shows some example results of detected control points and MRCP points in different settings, with various tools inserted in the end effector. We quantitatively evaluate how closely MAVRIC's MRCP matches the manually annotated "true end-effector" of the robot. We introduce a simple baseline that selects the points that move the largest distance over the exploration phase ("Max-motion"). Fig 2 shows this end-effector identification error, in terms of how far the MRCP was from a single "end-effector"/"tooltip" point ("point identification error") and also whether or not the MRCP was on the tool held in the robot ("region identification rate"). Max-motion peforms very poorly in nearly all settings, while most variants of MAVRIC get close to perfect end-effector region identification success rate.

Appendix D contains more detailed explanations of the MAVRC ablations in Fig 2. Additionally, Appendix E also evaluates MAVRC self-recognition in additional settings, such as with a shaky hand-held camera. It also shows how MAVRC succeeds in the presence of moving distractors, where other identical robots are present in the scene, and how MAVRIC deteriorates gracefully under exploration sequences shorter than the 20s used in the above experiments. Appendix F compares MAVRIC's tracking system with the approach proposed in [7].

### 3.2 Visually Guided 3D Point Reaching, Trajectory Following, and Imitation

We now evaluate MAVRIC (self-recognition + servoing) on 3D-point reaching tasks. We manually set 9 goal positions in the RGBD camera view at varying elevations and azimuths centered at the end effector's initial position at a distance of about 15 cm. We compare MAVRIC to two methods that have access to additional manually specified information: "Oracle VS," which servos a manually annotated end effector using the same visual servoing approach (based on [11; 12]) as our method, and "ROS MoveIt" [28] which has knowledge of the full robot morphology and kinematics models, camera calibration matrices, and proprioception. We allow a maximum of 150 steps for Oracle VS and MAVRIC, and servoing terminates early if the MRCP has reached within a 5 pixel radius of the goal (in 640x480 views).

Tab 1 reports (i) the median 3D distance error of the manually annotated end effector point from the goal, and (ii) early termination rate (ETR). ROS MoveIt is not feasible for servoing unmodeled tools, so we report its performance only in the no-tool setting. Its error is higher than Oracle VS; this may be due to WidowX robot model inaccuracy, servo encoder position errors, and camera calibration error.

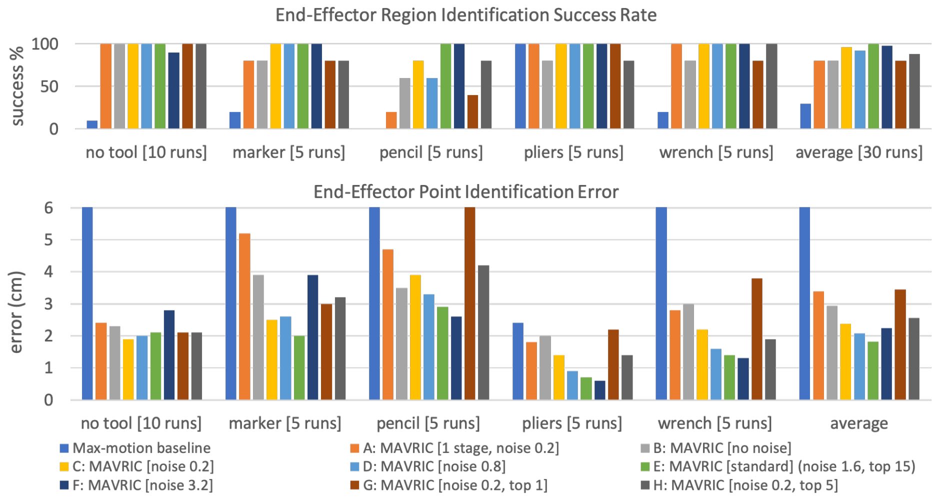



Figure 2: [Best seen in pdf] **(Top)** End-effector region identification success rate (higher is better), and **(Bottom)** End-effector identification error (in cm, lower is better) in various settings for the maximum-motion baseline, and various ablations of our method. Note that max-motion performs very poorly in most situations (average error 8.8 cm): the error axis is clipped at 6 cm here. Among ablations of MAVRIC, we study three hyperparameters: number of stages of end-effector ID (default: 2), noise variance (in squared pixel units) before responsiveness computation (default: 1.6), number of top points averaged to compute the MRCP (default: 15).

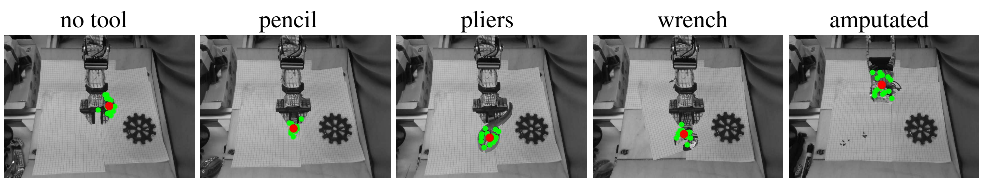



Figure 3: [Best seen in pdf] MRCP identification in various settings. In each setting, the red point is the MRCP, computed as the average of the 15 most responsive points, shown in green.

Oracle VS does well in most settings, and MAVRIC takes slightly longer (lower ETR), but its error is within 3 cm of Oracle VS, which may largely be explained by the end-effector point identification error (Fig 2).

Appendix G demonstrates embodiment mapping with MAVRIC for robot-to-robot imitation.

**Discussion:** We have presented MAVRIC, an approach that performs fast robot body recognition and uses this to accomplish morphology-agnostic visuomotor control. Reinforcement learning-based approaches operating in the same setting typically require days of robot data [19; 18; 17; 29], compared to 20s for MAVRIC. However, MAVRIC currently has several important drawbacks: (i) it inherits the common problems of visual servoing methods, namely local minima and singularites during Jacobian estimation. [30; 31; 32; 33] and (ii) it is currently not robust to occlusions, since it relies on unbroken visual tracks during both self-recognition and control. We are working to address these.

| Method | no-tool | | wrench | | pliers | | pencil | | marker | | average | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | error | ETR | error | ETR | error | ETR | error | ETR | error | ETR | error | ETR |
| ROS MoveIt | 4.4 | - | - | - | - | - | - | - | - | - | - | - |
| Oracle VS | 1.2 | 1.0 | 2.3 | 0.9 | 2.2 | 1.0 | 2.2 | 0.3 | 3.6 | 0.8 | 2.3 | 0.8 |
| mavric | 4.4 | 0.3 | 3.5 | 0.7 | 5.7 | 0.9 | 6.8 | 0.6 | 5.9 | 0.6 | 5.2 | 0.6 |

Table 1: 3D point-reaching error (in cm) between the manually annotated "true end-effector point" and the target (median over 9 goals), and early termination rate (ETR). MAVRIC, which automatically recognizes its own end-effector and uses it to servo, performs comparably with methods that have access to more information.

## References


[1] C. E. Shannon, "A mathematical theory of communication," *Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[2] B. D. Lucas, T. Kanade *et al.*, "An iterative image registration technique with an application to stereo vision," 1981.

[3] T. M. Cover and J. A. Thomas, *Elements of information theory*. John Wiley & Sons, 2012.

[4] A. Kraskov, H. Stögbauer, and P. Grassberger, "Estimating mutual information," *Physical review E*, vol. 69, no. 6, p. 066138, 2004.

[5] G. Ver Steeg, "Non-parametric entropy estimation toolbox (npeet)," 2000.

[6] P. Michel, K. Gold, and B. Scassellati, "Motion-based robotic self-recognition," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 3. IEEE, pp. 2763–2768.

[7] A. Edsinger and C. C. Kemp, "What can i control? a framework for robot self-discovery," in *6th International Conference on Epigenetic Robotics*, 2006.

[8] L. Natale, F. Orabona, G. Metta, and G. Sandini, "Sensorimotor coordination in a "baby" robot: learning about objects through grasping," *Progress in brain research*, vol. 164, pp. 403–424, 2007.

[9] K. Hosoda and M. Asada, "Versatile visual servoing without knowledge of true jacobian," in *IROS*. IEEE, 1994.

[10] B. H. Yoshimi and P. K. Allen, "Active, uncalibrated visual servoing," in *ICRA*. IEEE, 1994.

[11] M. Jagersand, O. Fuentes, and R. Nelson, "Experimental evaluation of uncalibrated visual servoing for precision manipulation," in *ICRA*, 1997.

[12] M. Bonkovi, K. Jezernik *et al.*, "Population-based uncalibrated visual servoing," *IEEE/ASME Transactions on Mechatronics*, vol. 13, no. 3, pp. 393–397, 2008.

[13] A. Shademan, A.-M. Farahmand, and M. Jägersand, "Robust jacobian estimation for uncalibrated visual servoing," in *ICRA*, 2010.

[14] A. Dame and E. Marchand, "Mutual information-based visual servoing," *IEEE Transactions on Robotics*, vol. 27, no. 5, pp. 958–969, 2011.

[15] T. Lampe and M. Riedmiller, "Acquiring visual servoing reaching and grasping skills using neural reinforcement learning," in *IJCNN*, 2013.

[16] K. Mohta, V. Kumar, and K. Daniilidis, "Vision-based control of a quadrotor for perching on lines," in *ICRA*, 2014.

[17] A. X. Lee, S. Levine, and P. Abbeel, "Learning visual servoing with deep features and fitted q-iteration," *arXiv preprint arXiv:1703.11000*, 2017.

[18] F. Ebert, C. Finn, A. X. Lee, and S. Levine, "Self-supervised visual planning with temporal skip connections," *arXiv preprint arXiv:1710.05268*, 2017.

[19] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *IJRR*, 2018.

[20] Q. Bateux, E. Marchand, J. Leitner, F. Chaumette, and P. Corke, "Training deep neural networks for visual servoing," in *ICRA*, 2018.

[21] U. Viereck, A. t. Pas, K. Saenko, and R. Platt, "Learning a visuomotor controller for real world robotic grasping using simulated depth images," *arXiv preprint arXiv:1706.04652*, 2017.

[22] R. L. Andersson, "A robot ping-pong player," *Experiment in Real-Time Intelligent Control.*, 1988.

[23] W. Jang, K. Kim, M. Chung, and Z. Bien, "Concepts of augmented image space and transformed feature space for efficient visual servoing of an "eye-in-hand robot"," *Robotica*, 1991.

[24] C. Cai, E. Dean-León, N. Somani, and A. Knoll, "6d image-based visual servoing for robot manipulators with uncalibrated stereo cameras," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*. IEEE, 2014, pp. 736–742.

[25] A. McFadyen, M. Jabeur, and P. Corke, "Image-based visual servoing with unknown point feature correspondence." *RAL*, 2017.

[26] B. Yang, J. Zhang, D. Jayaraman, and S. Levine, "Replab: A reproducible low-cost arm benchmark platform for robotic learning," *ICRA*, 2019.

[27] B. Yang, J. Zhang, V. Pong, S. Levine, and D. Jayaraman, "Replab: A reproducible low-cost arm benchmark platform for robotic learning," *arXiv preprint arXiv:1905.07447*, 2019.

[28] "ROS MoveIt package," https://moveit.ros.org/, accessed: 2019-07-06.

[29] S. James, P. Wohlhart, M. Kalakrishnan, D. Kalashnikov, A. Irpan, J. Ibarz, S. Levine, R. Hadsell, and K. Bousmalis, "Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 627–12 637.

[30] F. Chaumette, "Potential problems of stability and convergence in image-based and position-based visual servoing," in *The confluence of vision and control*. Springer, 1998, pp. 66–78.

[31] S. Hutchinson, G. D. Hager, and P. I. Corke, "A tutorial on visual servo control," *IEEE transactions on robotics and automation*, vol. 12, no. 5, pp. 651–670, 1996.

[32] C. Cai, "6d visual servoing for industrial manipulators applied to human-robot interaction scenarios," 2017.

[33] S. Cho and D. H. Shim, "Sampling-based visual path planning framework for a multirotor uav," *International Journal of Aeronautical and Space Sciences*, pp. 1–29, 2019.

[34] J. Shi and C. Tomasi, "Good features to track," Cornell University, Tech. Rep., 1993.

# A On the Definition of Control Points and Robot Body

In Sec 2.1, we defined the "body" $B_\delta$ of a robot as the set of particles whose responsiveness is higher than some threshold $\delta$: $B_\delta \triangleq \{P_i : R_i > \delta\}$. We call the constituent particles of this set "control points."

Does this definition align with our intuitive notion of a robot's body? Should all points on a robot's body be "responsive", i.e., do their displacements $\Delta S$ have high MI with the control inputs $A$? For this to hold, $\Delta S$ must be a *probabilistic function* of $A$, i.e., a fixed control input must induce a fixed distribution over $\Delta S$. This is true for velocity control commands as long as the states $S$ explored during the self-recognition phase lie within a small neighborhood. For example, consider a single motor controlling a rigid rod, as in Fig 4 (left). A small angular shift $\Delta\theta$ in the servomotor corresponds to a displacement $r\Delta\theta$ for a particle at a distance $r$ along the rod, in a direction perpendicular to the *current orientation* of the rod. With a significantly different orientation of the rod, the same angular shift would produce a very different displacement.

To account for this, our experiments employ velocity control and a small number of small exploration actions, so that all exploration happens within a small state neighborhood. As we will show empirically, this yields good performance.[1]

[1] An alternative definition of responsiveness may be much more general, but much less efficient to compute in practice: $R_i = I(S_i(t), S_i(t+1); A(t))$ — state changes $\Delta S_i$ are replaced throughout by state transition pairs $(S_i(t), S_i(t+1))$, and the rest of the approach remains the same.

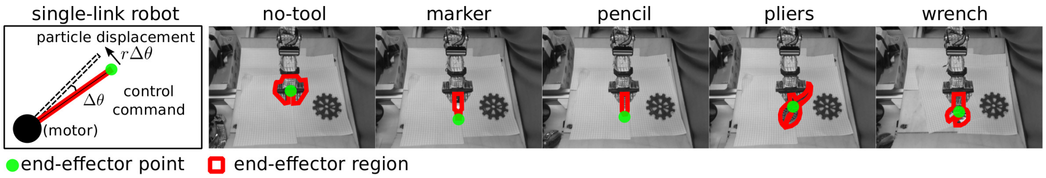



Figure 4: Manual annotations to illustrate the end-effector point (green point) and region (red outline) in different settings: a schematic single link robot (leftmost), followed by our experimental setups with various tools held in a robot arm.

## B Coarse to Fine Search for MRCP

In Sec 2.1, we alluded to a coarse-to-fine approach during self-recognition. This strategy targets the MRCP, since we are primarily interested in it for control. It works as follows: In the coarse stage, we initialize point tracking with Shi-Tomasi corner points [34], compute responsiveness scores and select the top-$k$ candidate particles. In the fine stage, we reinitialize tracking with a grid of 15x15 points around each of the selected candidates, and recompute responsiveness scores. Our experiments in Sec 3.1 evaluate the impact of various implementation details including coarse-to-fine search, noise addition, and the value of $k$.

## C Visual Servoing Details

In Sec 2.2, we briefly described how MAVRIC uses visual servoing for control. We provide a more elaborate explanation here.

Once the MRCP is identified, MAVRIC performs visual servoing for control to transport the MRCP point $S^*$ to a specified goal point $G$. This is appropriate for tasks like reaching and pushing, which are normally performed by hand-specified end-effectors in standard control settings.

We use online regression to fit $(A, \Delta S^*)$ tuples to estimate local Jacobian matrices "on the fly," using Broyden updates [11]: $\hat{J}_t = \hat{J}_{t-1} + (\Delta S_t - J_{t-1} A_t) A_t^T / \|A_t\|^2$.

The new control input $A_t$ for the current step is then computed using the pseudoinverse of the Jacobian, as $A_t = \eta \hat{J}_t^\dagger (G - S^*(t))$, where $\eta$ is a rate hyperparameter. Once $A_t$ is executed and the new position $S^*$ is measured, the Jacobian matrix is updated as above, and the process repeats until $S^* \approx G$. The Broyden update above is susceptible to noise, since it only uses a single $\Delta S_t$ measurement, hence we apply a batched update comprising the last $T$ tuples of $(A_\tau, \Delta S_\tau)$ as proposed in [12]. In our experiments, we set $T = 10$.

The Jacobian matrix initialization $J_0$ is computed as follows: we start at a random arm position, sequentially set the control inputs $A_i$ to scaled unit vectors $\epsilon e_i$ along each control dimension, and set the $i$-th column of $J_0$ to the response $\Delta S_i / \epsilon$. In our experiments, we repeat this initialization procedure whenever servoing has failed to get closer to the target in the last 20 steps.

**Handling tracking failures.** The above discussion of visual servoing depends on reliably continuing to track the MRCP throughout the servoing process. In practice, tracking is imperfect, and the control points are often dropped midway through the task due to occlusions, lighting changes etc. For robustness to such errors, we take the MRCP to be the average of the $k = 15$ most responsive control points. If any one point is dropped by the tracker during servoing, the MRCP is set to the average of the remaining points.

## D Details of Comparisons and Ablations in Self-Recognition Experiments

In Sec 3.1 and specifically the experiments in Fig 2, we evaluate several design decisions in MAVRIC: 1-stage vs 2-stage (coarse-to-fine) MRCP search, values of $K$ for top-K control point selection, and values of noise variance added to point tracks before responsiveness computation. We explain these now.

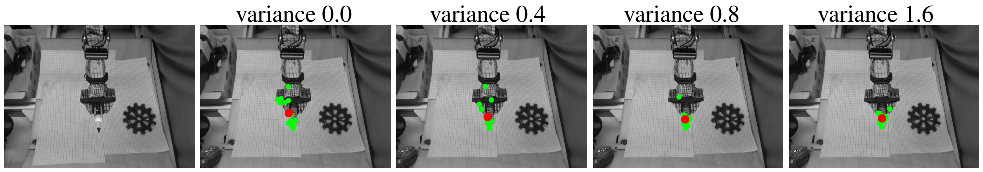
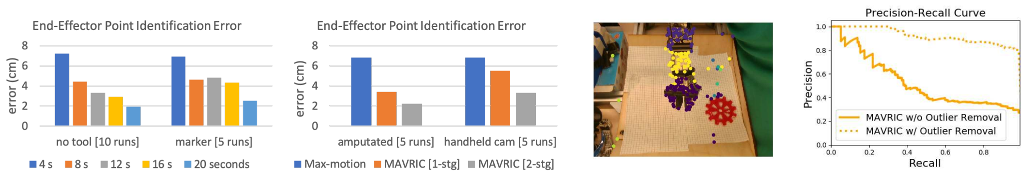



Figure 5: [Best seen in pdf] (Left to right) Original image of the arm with a marker tool, followed by MRCP identification with various values of noise variance. As noise increases, the MRCP points move closer towards the marker tip.




(a) (b) (c) (d)

Figure 6: [Best seen in pdf] **(a)** Self-recognition performance with shorter exploration phases, **(b)** Self-recognition with an amputated arm and a handheld camera, **(c)** Discovery of robot arm links from self-recognition phase data: tracks assigned to different clusters are colored differently. **(d)** Precision-Recall plot for control points.

While all MAVRIC variants work nearly perfectly on the end-effector region identification plots (Fig 2, top), the end-effector point identification error plot (Fig 2, bottom) provides a clear comparison of the MAVRIC ablations, labeled A through H in the legend. Comparing A and C (1-stage vs. 2-stage search), it is clear that coarse-to-fine MRCP search has a big impact on self-recognition success. Comparing B, C, D, E, and F (increasing noise variance), it is clear that a small amount of noise improves outcomes, but performance deteriorates when the noise is too high. Finally, comparing G, H, and C (top 1 vs top 5 vs top 15 control points), top 15 performs best in most cases. For all remaining experiments, we use variant E (2-stage, noise variance 1.6, and top 15 control point selection). Fig 3 shows examples of the detected MRCP from various runs under various settings. Fig 5 shows examples of the effect of noise on MRCP detection with the marker tool, clearly illustrating how higher noise variance biases towards selecting points closer to the tip of the marker.

## E Additional Evaluation Settings for Self-Recognition

We provide additional evidence of MAVRIC self-recognition working well, aside from the results in Sec 3.1.

**Amputated arm and shaky camera.** We also quantitatively evaluate self-recognition in two additional settings: an amputated version of the robot arm, with the last two links removed, and a shaky handheld camera. Fig 6 shows the errors. Once again, 2-stage MAVRIC works best. Fig 1 shows various steps during self-recognition with the handheld camera. Fig 3 includes an example in the amputated arm setting.

**Self-recognition phase duration.** While the above results are based on a 100-time step self-recognition phase (approx. 20 s), how much faster could this phase be? We evaluate end-effector identification with even fewer exploration steps in Fig 6, which shows that performance deteriorates gracefully under shorter exploration sequences.

**Moving distractors.** Next, we evaluate self-recognition with moving distractors by evaluating it on videos with two robots, where one of the robots is controlled by our method, while the other moves autonomously, thereby creating a moving distractor. We create such videos by spatially concatenating two separate exploration videos. Fig 7 shows an example result. MAVRIC correctly selects the end-effector of the *correct* arm, based on which arm's control commands it receives as input. See Supp for example videos. Max-motion does not have any control inputs, so it produces the same prediction in both cases.

video 1, left arm control inputs video 1, right arm control inputs

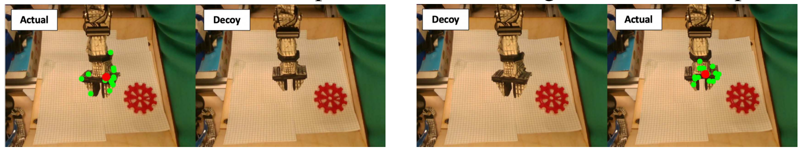



Figure 7: The moving distractors test: given the same video of two arms operating side by side (produced by concatenating individual frames side-by-side from two exploration videos), MAVRIC correctly ignores the decoy arm and selects the arm that is being controlled based on which control input sequence it receives as input. **(left)** MAVRIC is fed the left arm's controls, and it selects the MRCP (red point) on the left arm's end effector. **(right)** With right arm's control inputs, MAVRIC selects the right arm's end effector.

**Evaluating control points.** While the above results evaluate end-effector identification alone, MAVRIC finds control points all along the robot body. We now annotate the full robot body to evaluate whether these discovered control points are indeed on the robot body. Treating points on the robot body as ground truth positives, and those outside as negatives, Fig 6 (d) shows the precision-recall plot as the threshold $\delta$ on the responsiveness scores are varied ("MAVRIC w/o outlier removal"): while the precision is very high at low recall, it drops off quickly. This is intuitive: the lower the true responsiveness, the more noisy the measurements are. We expect that less responsive control points would benefit from a longer self-recognition phase. However, even with 20 seconds, it is possible to filter the points to improve the precision-recall performance. We perform simple outlier removal as follows: we measure the 2D position variance of each candidate track over the length of self-recognition, and set a heuristic threshold on this value, below which points are discarded. This simple outlier removal scheme proves sufficient to significantly improve precision-recall, as shown in Fig 6 (d). These control points may then be clustered based on spatial coherence to discover various links of a rigid robot, and their associated responsiveness scores. Fig 6 (c) shows an example. We use $K$-means clustering ($K = 10$) on position history features.

## F Validating Tracking Approach

We now compare MAVRIC's tracking approach against an alternative tracking scheme. [7] track moving objects for self-recognition by finding image patches that match the expected appearance of the robot and clustering them based on appearance. We implement their tracking scheme for self-recognition, so that the output is an image patch tracked through the video, representing the end-effector. On the same "no-tool" videos where MAVRIC correctly identifies the end-effector 10 out of 10 times, this method produces an output image patch that is centered on the end-effector only 2 out of 10 times. Further, since this clustering scheme relies on appearance similarities, it completely breaks down in the moving distractors setting above, where multiple identical-looking robots are present — the same appearance cluster teleports across the different robots, making responsiveness computation extremely noisy.

## G Robot-to-Robot Imitation

Next, we perform robot-to-robot imitation through MAVRIC: the target robot servos to move its MRCP along the source robot's MRCP trajectory. Fig 8 (left) shows the source robot drawing the letter "C" with a chalk piece (cyan represents the next target, white points are future target points, and black points are previously achieved target points). Fig 8 (right) shows the target robot, a second WidowX in different pose and illumination, in the process of following the trajectory of the source robot, including its path before the "C". See Supp for videos. MAVRIC thus enables visual embodiment mapping between two robots with unknown morphologies.

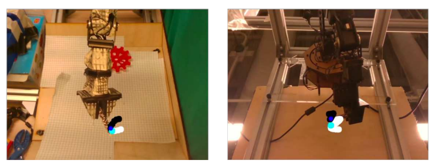


Figure 8: [Best seen in pdf] Robot-to-robot imitation with MAVRIC. In both video frames, cyan represents the next target, white points are future target points, and black points are previously reached target points. **(left)** A video frame of a source robot draws the letter C — in this case, we used MAVRIC to perform this task with visual servoing for trajectory-following. **(right)** A video frame of the target robot imitating the motions of the source robot. Full video in Supp.