Insights towards Sim2Real Contact-Rich Manipulation

Michael Noseworthy* CSAIL, MIT
Iretiayo Akinola NVIDIA
Yashraj Narang NVIDIA
Fabio Ramos NVIDIA
Lucas Manuelli NVIDIA
Ankur Handa NVIDIA
Dieter Fox NVIDIA

Abstract

Recent work has shown promise towards training policies for contact-rich tasks in simulation with the hope that they can be transferred to the real world. However, to close the *sim2real* gap, it is important to consider the effects of partial observability that are unavoidable in the real world and particularly relevant when dealing with small parts that require precise manipulation. In this work, we perform a detailed simulation-based analysis of how *pose-estimation error*, *object geometry variability*, and *controller variability* affect deep reinforcement learning algorithms. We show that using asymmetric actor-critic architectures leads to more robust training under noise.

1 Introduction

With recent advances in simulation technologies and deep reinforcement learning (DRL), there has been increased interest in *sim2real* approaches for robotic control [2, 3, 6]. In the manipulation domain, Narang et al. [13] develop techniques to efficiently simulate contact-rich tasks such as plug insertion or nut-on-bolt fastening – an important foundation for *sim2real* transfer in industrial assembly. Further, they demonstrate that DRL algorithms succeed at nut-on-bolt-fastening within their simulator. Although an impressive result, the simulated policy relies on full state observations and does not consider relevant factors for real world deployment such as partial observability and model misspecification. This is a significantly more challenging scenario, as exploratory behavior may be necessary to perform a task (e.g., the robot may need to search for nut/bolt alignment to account for pose uncertainty).

In the real world, partial observability is unavoidable and arises from many sources. Sensors are noisy, have limited range, and suffer from occlusion – all exacerbated by small parts typical in robotic assembly. Object models may be unknown and are subject to manufacturing tolerances. Furthermore, intrinsic robot parameters (e.g., inertial parameters or joint friction) may have been estimated through an imperfect system identification (SysID) process. Indeed, a large body of evidence suggests that task performance degrades as a function of all these types of uncertainty [8, 21, 22].

To move towards real-world policy deployment, we perform an extensive simulation-based analysis of a policy's ability to adapt to sources of uncertainty that would be present on the real robot: pose uncertainty, manufacturing tolerances, and controller differences. These insights are especially relevant for *sim2real* approaches where the model of the environment (i.e., the simulator) is unlikely to exactly match the real world. We evaluate three components of the PPO algorithm [18] (and its extensions) that we hypothesize will be relevant when dealing with partial observability: the discount factor, recurrent policy and value networks, and asymmetric actor-critics architectures. We find that using a privileged critic contributes the most to robust success under moderate levels of noise.

*Corresponding author: mnosew@mit.edu. Work done while interning at NVIDIA.

NeurIPS 2022 Workshop on Robot Learning: Trustworthy Robotics, Virtual, Virtual

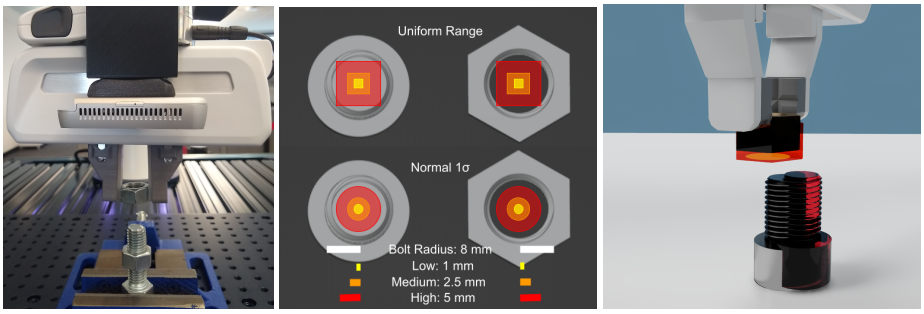


Figure 1: [Left] The real robot will be subject to pose estimation error and model inaccuracies. [Middle] Visualization of the uncertainty regions from noise added to the initial nut/bolt poses. Both uniform (top) and normal (bottom) noise distributions are considered. [Right] Initial state visualization where the red assets represent the initial pose estimates under high noise.

2 Problem Description

Nut-on-Bolt Fastening Domain: We are motivated by contact-rich manipulation tasks. In particular, we focus on the nut-on-bolt assembly environment introduced by Narang et al. [13] in the industrial assembly domain. Nut-on-bolt assembly has been identified as an important benchmark task as it is found in $\sim 40\%$ of mechanical assembly operations [11]. In this environment, a Franka Panda robot initially grasps a nut, which is not in contact with the bolt. The nut and bolt assets are defined to match ISO standards and for each size (e.g., M4 or M16) can vary between *tight* and *loose* configurations representing the manufacturing bounds of the clearances between parts. The goal is to fasten the nut to the bolt by first engaging it, and then lowering it by two threads (see Figure 1 left). For a formal description, including a list of state variables, please see Appendix B.1.

Sources of Error: We consider observation uncertainty for the nut, bolt, and robot model. These assumptions are used to justify the noise models introduced in our experiments section:

Pose Estimation Error: From our experience with real-world laboratory settings, camera intrinsics for off-the-shelf cameras (e.g., Intel RealSense D415) can be reliably obtained from manufacturers. Calibration procedures for camera extrinsics obtain the transform between camera and robot. Through multiple image observations, this error can be small. We also assume the use of RGB-only sensors, which are not subject to the artifacts frequently present in depth sensors. Thus, the dominant source of error is instead from a downstream pose estimator. For a state-of-the-art estimator that leverages known object models [9], we estimate the maximum position and orientation error of the nut and bolt to be 1 cm (Euclidean distance) and $3-7$ deg. We also note that the pose of the nut in the gripper could be estimated more accurately by use of tactile sensors (e.g., GelSight).

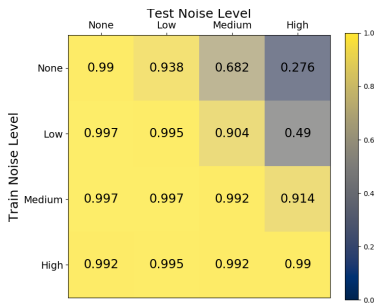
Object Variation: In industrial assembly, we often have access to CAD part models. However, since parts are subject to manufacturing tolerances, these models have limited instance-level accuracy (e.g., tolerances can cause the clearance for an M16 nut/bolt pair to differ up to $0.5mm$).

Robot Arm Model: As observed in previous work [22], there may also be inaccuracies in the robot model that can manifest as inaccurate proprioceptive state estimation or noisy control. The causes include robot arm tolerances, joint encoder calibration error, structural loading, as well as repeatability of the motor, gearboxes, and joint encoders. Of these, joint encoder calibration error (also referred to as *joint-offset error*) is known to be the most significant [22]. This error is on the order of 3-6 mm. These differences can lead to similar controllers having different behaviours across robot instances.

3 Approach: Asymmetric Recurrent RL with Domain Randomization

In this section, we describe a robust pipeline for training policies for contact-rich tasks using the *Proximal Policy Optimization* (PPO) algorithm. We combine *PPO* with *Domain Randomization* and identify important hyper-parameters that lead to robust performance under partial observability.

Preliminaries: Across experiments, we use the *Proximal Policy Optimization* algorithm [18] due to its previously demonstrated success in contact-rich domains [3, 13]. The action space for the robot



Noise Level (Full Model)	Success Rate (Time)
None ($0mm$)	0.992 ± 0.02 (3.82s)
Low ($1mm$)	0.990 ± 0.01 (4.06s)
Medium ($2.5mm$)	0.992 ± 0.00 (4.52s)
High ($5mm$)	0.979 ± 0.02 (4.61s)
Ablation (High Noise)	**Success Rate (Time)**
Full Model	0.979 ± 0.02 (4.61s)
(MLP Baseline)	0.995 ± 0.01 (5.18s)
(Low Discount Baseline)	0.974 ± 0.04 (4.75s)
(Symm Baseline)	0.943 ± 0.01 (7.28s)
(MLP, Symm Baseline)	0.451 ± 0.42 (9.41s)

Figure 2: Pose Estimation Noise Results: [Left] Success rates for the policies trained on a given noise level and evaluated on another. [Top Right] Success rates (and average time to success) for policies trained for the full model. [Bottom Right] Success rates (and average time to success) with ablated model components. For all experiments, noise is added once at the start of the episode.

is a 4-dimensional *Operational Space Controller* (OSC), where the policy outputs *x, y, z, and yaw* offsets for the end-effector while the OSC separately maintains an upright orientation with *pitch and roll* dimensions. As in previous work[13], the end-effector has no rotational limit.[2] To cover a variety of task instances, the initial state of each episode is randomized by varying the nut's pose relative to the gripper, the bolt's pose, and the hand's relative pose above the bolt (see Appendix B.2).

PPO with Partial Observability: We will investigate how the following hyperparameter and architecture choices affect the performance of PPO. The *Full Model* uses each of these components:

Asymmetric Actor-Critic: In simulation, we have access to privileged state. Following the work of Pinto et al. [15], we propose for the actor to use noisy observations while the critic uses the noiseless state. The *Symm Baseline* refers to using a symmetric actor-critic architecture.

Discount Factor: The discount factor affects how far into the future the agent can reason. We hypothesize that having a larger discount factor may be important for developing exploratory behaviours. The *Full Model* uses a discount factor of 0.999 while the *Low Discount Baseline* uses 0.99.

Recurrent Policies: We postulate that using a history of observations trains the agent to adapt to environment parameters relevant to the task. Recurrent neural networks (RNNs) lend themselves naturally to sequence processing, so we adopt LSTM layers in the actor and critic networks. The *MLP Baseline* uses MLP layers instead.

Domain Randomization: Domain randomization has been shown to be an effective way to improve robustness and transferrability of learning agents [3, 20]; however, not all types of randomization are valuable and useful [12]. We investigate what types of domain randomization are valuable for contact-rich manipulation motivated by the discussion of real-world challenges in Section 2.

Part-Pose Randomization: This is the most evident randomization parameter based on experience with real-robot policy deployment. To mimic a noisy pose-estimation pipeline, we consider a scenario where the pose estimator is only run once at the beginning of each episode (see Appendix B.3). Noise is added to the initial pose of both the nut and the bolt, and then the robot must use these noisy estimates for the entire episode. We introduce three levels of observation noise (for both uniform and normal distributions) during training to capture typical levels of noise expected in the real world (see Section 2). Figure 1 shows a visualization of the noise levels in our experiments.

Part-Geometry Randomization: Parts typically have manufacturing tolerances that prescribe a level of variability beyond mere scaling. For example, nuts and bolts have variabilities in their thread spacing (i.e., pitch), which are not captured by the typical XYZ scale randomization. To randomize over this parameter, we use multiple assets that represent the ends of standard manufacturing tolerances.

Controller-Parameter Randomization: We also investigate robot-related partial observability. For the same action space, policy behavior might vary depending on robot controller parameters and robot kinematic/dynamic parameters (link lengths, masses, etc.), among others. While robot dynamic parameters are difficult to *efficiently* randomize during parallelized training, we start to address this

[2]Although this makes the task easier, it is an important limitation to remove for sim2real transfer.

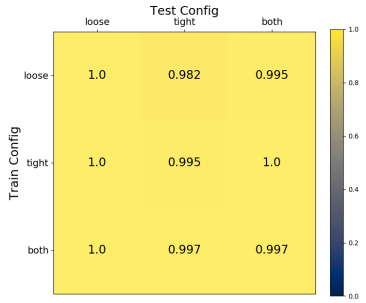


Figure 3: Success rates for policies trained using assets with different thread clearances within the manufacturing tolerance range.

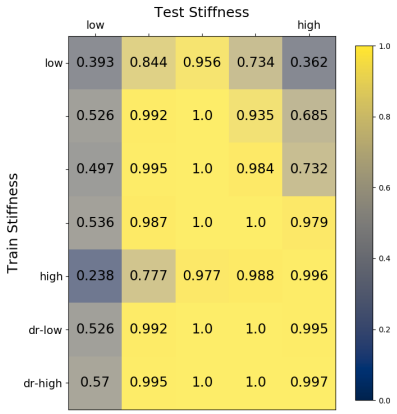


Figure 4: Success rates for policies trained with different control stiffnesses. Controllers 1–5 use fixed gains, while DR-low & DR-high had variable gains.

question by analyzing how well policies generalize to controllers of different stiffnesses. We hope this will give initial insight into how policies can generalize to controllers with different behaviours and leave a full treatment of unobserved robot models to future work.

4 Experiments

Details on our training and evaluation pipeline can be found in Appendix B.5.

Pose Observation Noise: Figure 2 shows the results of training PPO (*Full Model*) under various levels of observation noise. First, on the left, we see that policies trained with low noise levels fail to generalize to higher noise settings (as expected). Our second observation (seen on the top right) is that episode success length is longer when more noise is present. This is likely due to the extra exploration needed to succeed. Finally, we present the results of our ablation study that show that the asymmetric actor-critic architecture was the most important factor in achieving robust performance under high noise (bottom right). More detailed ablation results can be found in Appendix C.

Object Model Uncertainty: Figure 3 shows the result of training a policy with assets of a specific configuration (i.e., tight or loose) and evaluating on the others. No other types of uncertainty are added and the policy does not receive the asset configuration as input. For these experiments, we use the *Full Model* and present an ablation of the RNN in Appendix D. The object configurations appear to be small enough to not make a difference on generalization. In Appendix D, we show that the MLP policy performs worse on the tight configuration, which is remedied by domain randomization.

Controller Uncertainty: Finally, we report the performance of five policies evaluated using controllers of different stiffnesses than those they were trained on (ranging from low to high stiffness). No other forms of uncertainty are considered. In addition, we evaluate policies that were trained with randomized controller gains (DR-low and DR-high, see Appendix B.4). Results in Figure 4 show that most policies do well when using the controller they were trained on but degrade as the evaluation stiffness changes. However, domain randomization improves policy robustness. This initial result shows promise for using DR to adapt to controller inaccuracies, and we plan to next evaluate if this result also holds for misspecified robot kinematic and dynamics parameters.

5 Conclusion

To conclude, we posit that domain transferability and partial observability are tightly connected. Our results show that success in contact-rich tasks is sensitive to parameters which are often uncertain. Careful consideration of control, estimation, and object models is important to adapt to variations expected in the real-world. One promising approach to better address this uncertainty includes integrating rich sensing modalities (e.g., tactile or force feedback) with state-of-the-art RL algorithms.

Acknowledgments

We would like to thank the reviewers for their helpful comments and feedback on this work. We would also like to thank Bingjie Tang and Michael Lin for many insightful discussions.

References

[1] A. Agarwal, A. Kumar, J. Malik, and D. Pathak. Legged Locomotion in Challenging Terrains using Egocentric Vision. In *Proceedings of the Conference on Robot Learning (CoRL)*, 2022.

[2] A. Allshire, M. Mittal, V. Lodaya, V. Makoviychuk, D. Makoviichuk, F. Widmaier, M. Wüthrich, S. Bauer, A. Handa, and A. Garg. Transferring dexterous manipulation from gpu simulation to a remote real-world trifinger. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[3] O. M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research (IJRR)*, 2020.

[4] A. Handa, A. Allshire, V. Makoviychuk, A. Petrenko, R. Singh, J. Liu, D. Makoviichuk, K. Van Wyk, A. Zhurkevich, B. Sundaralingam, Y. Narang, J.-F. Lafleche, D. Fox, and G. State. DeXtreme: Transfer of Agile In-hand Manipulation from Simulation to Reality. *arXiv preprint arXiv:2210.13702*, 2022.

[5] R. Holladay, T. Lozano-Pérez, and A. Rodriguez. Force-and-Motion Constrained Planning for Tool Use. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.

[6] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter. Learning agile and dynamic motor skills for legged robots. *Science Robotics*, 2019.

[7] Z. Jia, A. Bhatia, R. M. Aronson, D. Bourne, and M. T. Mason. A Survey of Automated Threaded Fastening. *IEEE Transactions on Automation Science and Engineering*, 2019.

[8] K. N. Kaipa, A. S. Kankanhalli-Nagendra, N. B. Kumbla, S. Shriyam, S. S. Thevendria-Karthic, J. A. Marvel, and S. K. Gupta. Addressing perception uncertainty induced failure modes in robotic bin-picking. *Robotics and Computer-Integrated Manufacturing*, 2016.

[9] Y. Labbe, L. Manuelli, A. Mousavian, S. Tyree, S. Birchfield, J. Tremblay, J. Carpentier, M. Aubry, D. Fox, and J. Sivic. MegaPose: 6D pose estimation of novel objects via render & compare. In *Proceedings of the Conference on Robot Learning (CoRL)*, 2022.

[10] G. B. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal. Rapid Locomotion via Reinforcement Learning. In *Proceedings of Robotics: Science and Systems (RSS)*, 2022.

[11] L. A. Martin-Vega, H. K. Brown, W. H. Shaw, and T. J. Sanders. Industrial perspective on research needs and opportunities in manufacturing assembly. *Journal of manufacturing systems*, 1995.

[12] B. Mehta, M. Diaz, F. Golemo, C. J. Pal, and L. Paull. Active domain randomization. In *Proceedings of the Conference on Robot Learning (CoRL)*, 2020.

[13] Y. Narang, K. Storey, I. Akinola, M. Macklin, P. Reist, L. Wawrzyniak, Y. Guo, A. Moravanszky, G. State, M. Lu, A. Handa, and D. Fox. Factory: Fast contact for robotic assembly. In *Proceedings of Robotics: Science and Systems (RSS)*, 2022.

[14] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel. Sim-to-Real Transfer of Robotic Control with Dynamics Randomization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[15] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel. Asymmetric actor critic for image-based robot learning. In *Proceedings of Robotics: Science and Systems (RSS)*, 2018.

[16] N. Rudin, D. Hoeller, M. Hutter, and P. Reist. Learning to Walk in Minutes Using Massively Parallel Deep Reinforcement Learning. In *Proceedings of the Conference on Robot Learning (CoRL)*, 2021.

[17] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel. High-dimensional continuous control using generalized advantage estimation. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.

[18] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[19] D. Son, H. Yang, and D. Lee. Sim-to-Real Transfer of Bolting Tasks with Tight Tolerance. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[20] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[21] K. Van Wyk, M. Culleton, J. Falco, and K. Kelly. Comparative peg-in-hole testing of a force-based manipulation controlled robotic hand. *IEEE Transactions on Robotics*, 2018.

[22] K. Van Wyk, J. Falco, and G. Cheok. Efficiently improving and quantifying robot accuracy in situ. *arXiv preprint arXiv:1908.07273*, 2019.

Initial State Variable	Min	Max
Bolt Position (x, y, z)	$(-0.01m, -0.01m, -0.01m)$	$(0.01m, 0.01m, 0.01m)$
Nut Position (x, z)	$(-0.006m, -0.003m)$	$(0.006m, 0.003m)$
EE Position (x, y, z)	$(-0.01m, -0.01m, -0.005m)$	$(0.01m, 0.01m, 0.005m)$
EE Orientation (yaw)	$-3.14rad$	$3.14rad$

Table 1: Randomization ranges used to sample offsets from a default initial state for each episode.

Appendix

A Related Work

Nut-on-Bolt Fastening: Nut-and-bolt fastening is an ubiquitous task in robotic assembly [7]. Many works consider tool-use (e.g., nut runners) for this task which aid in applying the appropriate torque and reducing uncertainty [7]. Holladay et al. [5] consider the task of twisting a nut that is already on a bolt. They focus on multi-stage planning where it may be necessary to fixture the bolt while applying a torque to the nut. In our work, we instead focus on the initial nut engagement but assume the bolt is fixed to the workspace. Narang et al. [13] consider the entire *Screw* task (e.g., picking, engaging, and lowering), but do not address uncertainity that would be present in the real world. Son et al. [19] successfully show sim2real transfer for an M48 nut/bolt pair while considering sensing noise. We build upon the insights of this work by considering much smaller parts (M16 nut/bolt pair) and additional forms of uncertainty.

Sim2Real Transfer: With advancements in reinforcement learning and parallelizable simulation, there has been much interest in *sim2real* transfer for complicated control problems. Of note include legged locomotion [6, 1, 10, 16] and in-hand manipulation [2, 3, 4]. These works use techniques such as domain randomization and policy distillation to bridge the sim2real gap. We are interested in using similar techniques in the industrial assembly domain [14, 19].

B Experiment Details

B.1 Problem Formulation

We formalize the nut-on-bolt assembly problem as a *Partially-Observable Markov Decision Process* (POMDP). We will first describe the fully-observable MDP and later evaluate the effects of making different components of the state partially observable. The corresponding MDP is defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma)$. Concretely, the state space, $\mathcal{S}$, is represented by the robot's end-effector pose and velocity, p_{ee} and v_{ee}, the nut's pose, p_{nut}, the bolt's pose, p_{bolt}, and the nut/bolt's thread configuration, $c \in \{\textit{loose, tight}\}$. The action space, $\mathcal{A}$, are setpoints for a 4-dimensional operational space controller. The transition matrix, $\mathcal{T}$, is implicitly defined using the *Factory* simulator from Narang et al. [13]. Finally, the reward function, $\mathcal{R}$, is defined using a keypoint reward that encourages the policy to align keypoints for the nut with keypoints located two threads below the top of the bolt.

B.2 Initial State Randomization

In all experiments, we randomize the initial state of the robot gripper, nut, and bolt around a default initial state. The default initial state has the robot holding the nut $1cm$ above the bolt. The nut pose is randomized to stay within the gripper and be parallel to the table (and hence no y-randomization is needed). The same randomization range is used during both training and evaluation. Initial state randomization ranges can be found in Table 1 and correspond to the range of a uniform distribution.

B.3 Pose Estimation

In this work, we add noise to the nut and bolt poses by mimicking a pose estimator running only once at the beginning of each episode. Specifically, at the start of each episode, we add noise to the bolt's pose as well as to the relative transform between the nut and the gripper. These are fixed for an entire episode, and at each timestep the global pose of the nut is computed by applying the noisy relative transform to the end-effector's pose.

Noise Level	Normal	Uniform
None ($0mm$)	0.992 ± 0.00 (3.82s)	1.000 ± 0.00 (3.66s)
Low ($1mm$)	0.990 ± 0.01 (4.06s)	1.000 ± 0.00 (3.99s)
Medium ($2.5mm$)	0.992 ± 0.01 (4.52s)	0.997 ± 0.01 (4.20s)
High ($5mm$)	0.979 ± 0.01 (4.61s)	0.992 ± 0.01 (4.32s)

Table 2: Success rates (and average time to success) for policies trained with varying levels of pose-observation noise added to both the nut and the bolt.

Discount Factor	MLP	RNN
0.99	0.992 ± 0.00	0.974 ± 0.04
0.996	0.979 ± 0.01	0.971 ± 0.02
0.998	0.979 ± 0.01	0.982 ± 0.01
0.999	0.992 ± 0.01	0.992 ± 0.01

Table 3: Success rates for policies trained under high noise with different discount factors.

Critic Architecture	MLP	RNN
Asymm	0.932 ± 0.13	0.987 ± 0.01
Symm-Separate	0.599 ± 0.30	0.951 ± 0.01
Symm-Shared	0.451 ± 0.42	0.943 ± 0.01

Table 4: Success rates for policies trained under high noise using different critic architectures.

By only mimicking an initial pose estimate, we do not rely on properly modeling the correlation between consecutive pose estimates in the real world. These correlations could be affected by factors such as occlusion, lighting conditions, and camera model properties that would be difficult to model in simulation. Overfitting to a wrong noise model could lead to policies that do not transfer to the real world. Furthermore, it is easier to implement initial pose estimation than a full tracking system that handles occlusion in the real world.

B.4 Controller Randomization

We consider a controller randomization setup where at the beginning of each episode, we sample new controller PD gains for the operational space controller. The P gains are sampled uniformly and independently in each dimension from the randomization range while the D gains are set to be critically damped as $K_d = 2\sqrt{K_p}$. The randomization range is determined by the randomization level, L, which sets the range to $(K'/L, K'L)$ where K' is a default gain (also specific to each dimension). Although typically, the policy will know the gains of the underlying controller (and is even able to control them), we design this experiment to gain insight into how well a policy can adapt to controllers with different behaviour which would more typically be caused by uncertainty in robot kinematic and dynamic parameters. This is because *IsaacGym* currently does not support randomization over robot model parameters at the beginning of each episode. We expect randomizing over the actual robot parameters will be important for *sim2real* transfer as these differences may induce different behaviour changes than those from only varying controller stiffness.

B.5 Evaluation Methodology

For each experiment we first train a policy on the *nut-on-bolt engagement* task. During training, we save the policy that achieves the best return. At evaluation time, we evaluate task success averaged over 128 environments. In total, each reported statistic is averaged over 3 seeds with 128 evaluation episodes per seed. For both training and evaluation we use a maximum episode length of 1200 steps (20 seconds). Generalized Advantage Estimation is used for all models [17].

C Additional Pose Uncertainty Results

C.1 Uniform vs. Random Noise

We consider two different noise distributions for adding noise to the initial pose estimates: Uniform and Gaussian. As seen in Table 2, the *Full Model* (RNN, high discount factor, and asymmetric architecture) performs well in both scenarios.

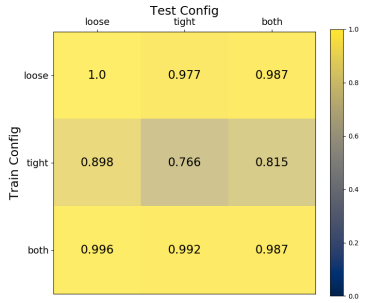


Figure 5: Success rates for MLP policies trained using assets with different thread clearances within the manufacturing tolerance range.

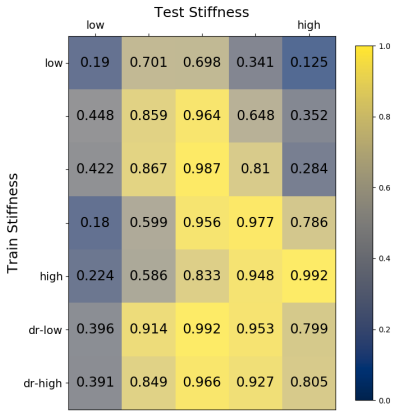


Figure 6: Success rates for policies trained with different control stiffnesses and high Gaussian pose noise. Controllers 1–5 were trained with fixed parameters, while DR-low & DR-high had variable gains.

C.2 Discount Factor Ablation

To determine if the discount factor has an impact on policy performance under noise, we consider 4 different values: $\gamma = 0.99, 0.996, 0.998, 0.999$. For this ablation we use an RNN policy and asymmetric critic. We find that there is not a significant difference between different values of γ, suggesting that the training pipeline is fairly robust to this parameter.

C.3 Asymmetric Critic Ablation

To measure how important it is to use a critic that has access to ground-truth state, we ablate the critic in the full model (Asymm). Specifically, we consider baselines where the critic uses the noisy observation with a separate value network (Symm-Separate) as well as a shared base network with the actor (Symm-Shared). Note the Asymm model uses a separate value network.

In Table 4, we report success rates for policies trained under a high-level of Gaussian observation noise. we see that for both MLP and RNN policies, the asymmetric critic outperforms both baselines (the gap is wider in the MLP case). Using an asymmetric critic significantly helps in training policies that are robust to noise.

D Additional Object Model Uncertainty Results

Although in Section 4 we showed the *Full Model* was insensitive to variation in object models, Figure 5 shows that the model using an MLP (instead of RNN) struggles for the tight configuration. This appears to be due more to the difficulty of training on only the tight configuration rather than to partial observability, as training on only the loose configuration generalizes well to evaluation on the tight configuration.

E Additional Controller Uncertainty Results

E.1 Success Times

Controllers with different stiffnesses lead to different task completion times. Table 5 shows the average time-to-success for policies trained and evaluated using controllers of varying stiffness. As expected, stiff controllers lead to faster task execution; however, success time increases when more noise is added.

Stiffness	1 (Low)	2	3	4	5 (High)
No Noise ($0mm$)	$14.6s$	$11.8s$	$7.7s$	$4.2s$	$1.9s$
High Noise ($5mm$)	$14.4s$	$12.2s$	$8.9s$	$4.6s$	$2.5s$

Table 5: Success times for different controller stiffnesses and different levels of Gaussian noise.

E.2 Pose and Control Uncertainty

Figure 6 repeats the experiment from Figure 4 with high Gaussian pose estimation noise. Comparing these two figures shows that changing the controller stiffness at test time leads to more pronounced performance degradation when there is also observation noise present. Domain randomization of control stiffnesses during training (see last two rows) leads to increased robustness to test-time variability.