Vision-Based Manipulation via Sim-to-Real RL

SO-101 Robotic Arm with Isaac Lab

Matthew Evans Kiran Hegde

December 2025

Project Goal & Approach

Goal: Train vision-based manipulation policies in simulation, deploy to real SO-101 arm

Two Tasks Implemented:

1. Proprioception-Only

- Maximize EE height
- 12-D obs (joint states)
- 4,096 parallel envs
- ✓ Sim-to-real success

2. Vision-Based Cube Interaction

- Point/grasp/lift cube
- 1030-D obs (vision + joints)
- 16 parallel envs
- Domain randomization
- ○ Partial sim-to-real

Stack: Isaac Sim/Lab + PPO + ResNet18 vision encoder + LeRobot deployment

Vision Task: Architecture & Observations

Policy Architecture:

- **Vision:** ResNet18 (frozen) $\rightarrow$ Spatial Softmax $\rightarrow$ 1024-D features
- **Actor:** [1024-D visual + 6-D joints] $\rightarrow$ MLP $\rightarrow$ 6-D actions
- **Critic:** 14-D privileged state (joint states, contact forces, cube pose)

Key Design Choices:

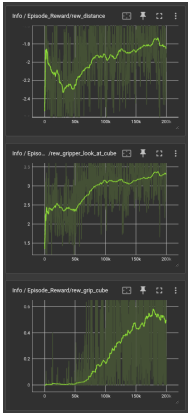
- Asymmetric actor-critic: actor sees only camera + joints (deployable)
- Spatial Softmax preserves positional information for manipulation
- Actions: normalized $[-1, 1]$ mapped to joint position targets

Reward Shaping for Vision Task

Multi-Component Reward Signal:

- **Distance:** Encourages approach ($-10.0 \times d$)
- **Grip force:** Rewards contact ($+20.0 \times f$)
- **Lift height:** Rewards elevation ($+100.0 \times h$)
- **Look-at:** Keeps cube in view ($+5.0 \times \cos\theta$)
- **Action penalty:** Discourages thrashing (-0.005)
- **Terminal:** Success bonus ($+2000$)

Key Insight: Gated rewards guide policy through sequential phases (approach $\rightarrow$ contact $\rightarrow$ lift)



Policy learns staged behavior through reward composition

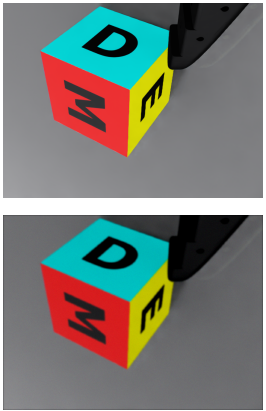
Domain Randomization: Visual Variation

Why Visual Randomization?
Real-world cameras never match simulation perfectly. We randomize:

Camera Augmentation:

- Gaussian noise (1-2%)
- Brightness variation ($\pm 15\%$)
- Motion blur

Impact: Forces vision encoder to learn robust features invariant to lighting and sensor noise



Top: Original image. Bottom: Gaussian blur, down/up sample, brightness, Gaussian noise, contrast, motion blur, JPEG compression

Domain Randomization: Physical & Geometric

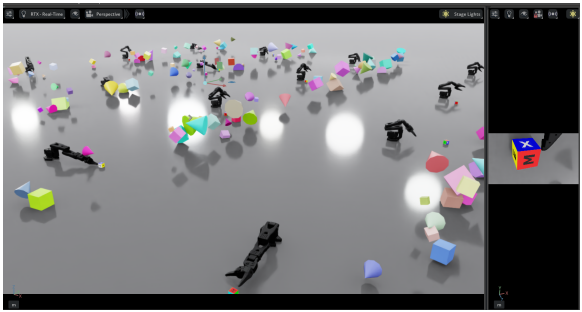
Why Physical Randomization?
Sim-to-real gap extends beyond vision—we randomize physical properties:

Geometric Variation:

- Camera pose jitter ($\pm$1mm, $\pm 0.5^\circ$)
- Cube position & size
- 10 distractor objects (80% active)

Lighting & Scene:

- Intensity: [500, 1500] range
- Enables generalization to varied workspaces

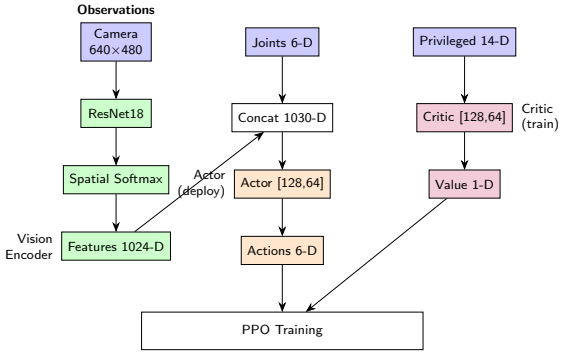


Training scenes with varying object poses, lighting, and distractors

Key Challenges Encountered

1. **RL is Hard**
   - Reward tuning critical: used live curriculum (adjust scalars mid-training)
   - Monitor TensorBoard, continue from checkpoints with updated rewards
2. **Sim-to-Real Gap**
   - Solution: Aggressive domain randomization during training
   - Careful camera calibration and observation preprocessing matching
3. **Hardware Limitations**
   - Low-quality webcam (no published specs, fuzzy image)
   - Servo imprecision: some joints $> 4\%$ error
   - GPU memory: RTX 5090 32GB needed for vision task

Vision Model Training in Isaac Sim



Architecture: Asymmetric actor-critic with frozen ResNet18 vision encoder

Results & Achievements

What We Accomplished:

- ✓ **Proprioception task:** Successful sim-to-real transfer (raise EE)
- ✓ **Vision task:** Trained policy can point at/touch cube in simulation
- ○ **Vision sim-to-real:** Partial success (inconsistent due to webcam quality)
- ✓ **Infrastructure:** Full training pipeline with reproducible artifacts
- ✓ **Safety:** Hardware protection mechanisms prevent damage

Demo Available:

1. Real robot: Proprioception-only EE raising
2. Simulation: Vision-based cube interaction (trained policy)

Key Lessons Learned

Technical Insights:

- Reward shaping is an art: live curriculum critical for avoiding local optima
- Domain randomization works: sim models can transfer to real hardware
- Hardware matters: webcam quality and servo precision limit sim-to-real success
- Spatial Softmax essential for vision-based manipulation (preserves position info)

Process Insights:

- Start simple: proprioception-only task validated our pipeline before vision
- Infrastructure first: reproducible training saves debugging time
- Iterate quickly: simpler workflows beat perfect ones when learning

Conclusion

What We Built:

- End-to-end sim-to-real RL pipeline using Isaac Lab + PPO
- Vision-based manipulation policy (ResNet18 + Spatial Softmax)
- Successful proprioception-only sim-to-real deployment
- Reproducible training infrastructure with CI/CD

Impact:

- Deep practical understanding of modern sim-to-real robotics
- Validated Isaac Lab for manipulation tasks on consumer hardware
- Foundation for future work in learned manipulation

Questions?

Next: Live Demo