Extended Abstract

Motivation The motivation of the project is to study the foundations required for a goal-oriented Robot that when given a task and desired goal image can come up with a plan to execute it.

Method A real-life robot was used to demonstrate and validate the fundamental concepts behind action primitives based plan whereby LLM produced sequence of action primitives can help achieve a goal state. A simulation environment was used to validate and explore the self-learning aspect of the project.

Implementation A SO100 arm was used to implement and test the action primitives needed for a cleaning task and a multitude of objects and tasks were tested. Both Behaviour Cloning and Offline RL based policies were trained. A rewards model for RL training was done with the help of a YOLOv8 based object detection. A simulation based recreation of the SO100 arm was done on mani-skill environment and an initial policy to bootstrap a real-life pick task was trained.

Results With SO100 arm, it was shown that by a combination of simple primitives like pick, place, dip and wipe; a basic cleaning robot can be demonstrated by extending it to mobile robots like Lekiwi robot. The trained model was seen to be adaptive to multiple objects, orientations and task types. Using mani-skill simulation environment, a SO100 arm was used to simulate a pick object task and train an initial policy.

Discussion Lot of prior work in task robots is to build end-to-end models trained to complete a single monolithic task. As the task complexity gets harder and environments under which is operated gets varied, it becomes harder to generate such end-to-end models. Also the Vision-Language-Action models directly generate motor actions which makes it harder to port these trained model across robots. By breaking down the task into action primitives each of which can be trained on their own, this project attempts to simplify the training complexity and yet be effective over complex long horizon tasks in a multitude of environments. The project idea leverages both real-life evaluations runs on real robots, simulation based bootstrapping of policies.

Conclusion An effective action primitive based plan to do long horizon tasks aided by LLMs is proposed and a cleaning task based on sequence of pick and place tasks was demonstrated. The policy for each primitive based task can be bootstrapped either by Behavior Cloning, Reinforcement Learning based models with real-life expert demos, exploration runs or by simulation based learning or online videos. When combined with the planning tools provided by LLMs, this becomes a foundation for goal-based long horizon task planning and opens up scope for lot of future research on Vision-Language-Primitive based training.

Goal based Long-Horizon Multi-task Robot aided by LLM or VLP with Action Primitives

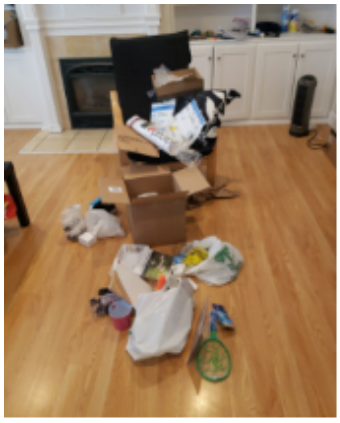
Harish Balasubramaniam
Stanford University
harishb@stanford.edu

Abstract

The motivation of the project is to study the foundations required for a goal-oriented Robot that when given a task and desired goal image can come up with a plan to execute it. The project aims to demonstrate and validate the fundamental concepts behind action primitives based robot planning whereby LLM produced sequence of action primitives can be used to achieve a desired goal state.

1 Introduction

The field of Robotics and AI are at an inflection point and poised to make a huge difference to human lives. In order to create a 'chatGPT' moment in robotics, there is a need to create versatile, robust, generalized robots that can self-learn and adapt to deployed environments like homes or factories. Some examples are Garden robots, Cleaning robots, Kitchen robots. A typical use case is that the robots should be able to take a text command from the user along with image(s) that describe the problem, then start to work on the problem with minimal human intervention as needed. Let's consider an example indoor clutter cleanup task as shown in Figure 1.



Help me clear up the clutter in my room

Desired Goal Image selected by user

Figure 1: Example user command for an Indoor Cleaning Robot

Solving this problem requires world understanding, long horizon planning with hierarchical steps, breaking the problem into low level atomic action planning, self-learning and asking for human help as needed. LLMs allow us the capability to come up with a high level plan.
Assume that the robot supports the action primitives 1) Pick 2) Place 3) Push_or_Rotate. Lets consider an action plan for the above example provided by an LLM based on these action primitives.

- Step 1 - Clear loose items from floor
  - pick() each loose item from the floor (e.g. bags, red cup, packets, small boxes)
  - place() them temporarily on chair or shelf for staging
- Step 2 - Stack boxes
  - pick() each box one by one
  - place() them in a neat stack near the chair
- Step 3 - Organize small items
  - pick() staged small items
  - place() them beside the chair in grouped fashion (bag + net + small packet)
- Step 4 - Adjust orientation
  - Use push_or_rotate() to slightly rotate or nudge items for alignment (e.g. boxes perfectly stacked, bags upright)
- Step 5 - Chair cleanup
  - pick() any remaining loose packaging from chair
  - place() the pillow back on chair
  - place() cylindrical roll on chair

Any Robot that is capable of the 3 action primitives should be able to execute this plan subject to their hardware constraints. Each primitive can be trained by a behavior cloning, reinforcement learning or other imitation learning policies like flow model or diffusion with expert demos or self-learn within simulation. A Robot may even ask for human help if any of the intermediate task is beyond its capability.

2 Related Work

Policy optimization for long-horizon multi-task high DoF robots operating in highly unstructured environments is a challenge. Behavior Cloning or Imitation learning based policies require expert demonstrations that are time-consuming to collect. Though Reinforcement Learning algorithms can learn with an environment setup, online learning has risks of safe exploration or has to depend on simulation which suffer from sim-to-real gaps.
Other problems with policy optimizations for robots are a) They are hard to generalize across robots b) hard to share partial solutions across task domains.

RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control

Google's RT-2 (2) introduces large-scale Vision-Language-Action (VLA) models capable of directly mapping web-scale visual-language knowledge into real-world robotic control. The approach uses frozen foundation models such as PaLM-E and PaLI to encode multi-modal inputs, and fine-tunes action heads using robotic demonstrations. This allows zero-shot generalization to unseen instructions and enables task reasoning directly from user commands, leveraging large pretrained web datasets.

- Trains full VLA model directly
- Transformer outputs directly predict motor actions specific to the robot
- Needs relatively high compute. Not ideal for onboard inference

pi-0: Toward Real-World Deployment of Foundation Models on Physical Systems

The pi-0 (1) architecture focuses on compressing foundation models for real-world robotic deployment on resource-constrained hardware. The approach modularizes vision, language, and control by freezing pretrained vision-language encoders and training lightweight control policy heads on top of embedded features. This separation reduces computational demands while preserving multimodal task reasoning.

- Trains lightweight policy heads with frozen vision, language encoders
- Transformer outputs directly predict motor actions specific to multiple classes of robots
- Relatively less compute. Can run on robot onboard hardware

The pi-0 model informs our design that we can combine LLM or Vision-Language encoders with actions primitive based policy heads that can be executed via efficient per-primitive lightweight policies suitable for embedded robots.

SLAC: Scaling Robot Skills with Large-Scale Skill Libraries (June 4, 2025)

SLAC (3) proposes building large-scale skill libraries by training reusable primitive policies across diverse tasks and object types. Each skill is trained with goal-conditioning and behavior cloning, allowing skill-conditioned policies to generalize across many combinations of tasks.

- Highlights issues with RL - Bridging sim-to-real reality gap, achieving safe exploration and sample efficiency
- Leverages a low-fidelity simulator to pretrain a task-agnostic **latent action space** which is temporally long, disentangled and safe
- Once a latent action space is learned, SLAC uses it as the action interface for a novel off-policy RL algorithm to autonomously learn downstream tasks through real-world interactions
- Enables a high-DoF mobile manipulator to learn with RL in the real world without relying on any demonstrations or hand-crafted behavior priors

While SLAC learns the latent action space Z in low-fidelity simulation that is not replicated in visual/physical reality, our approach uses action primitives that may be reflected in physical reality. We call it the **Reduced Instruction Set Robot (RISR)** framework, where reusable primitives are combined dynamically using LLM-generated plans to efficiently compose complex long-horizon behaviors with minimal data collection overhead. Each primitive can be bootstrapped with fine-tuned low-level policy learned from Behavior Cloning, Imitation Learning or RL approaches either with expert demos or from simulation. Such an approach could have an additional benefit of a 'portable robot intelligence software' similar to how Java Virtual Machine enabled software to run across hardware.

3 Method

3.1 Policy training for each RISR primitive

The Figure 2 shows how a policy is trained for each RISR primitive (e.g pick, place, push, wipe). The policy is bootstrapped either via

- Expert tele-operated demonstrations on a real compatible hardware. This has the disadvantage of being time-consuming.
- or from simulation environment where a compatible world scene is created with relevant objects. The policy can be trained either by collecting demos or through online RL based iterative policy optimization.

The policy MLP should be lean and minimal such that it can work robustly across multiple tasks, objects and also robot embodiments. For example, the camera and sensor placement in each robot can be different. Care should be taken that such that policy training is agnostic to the specific robot

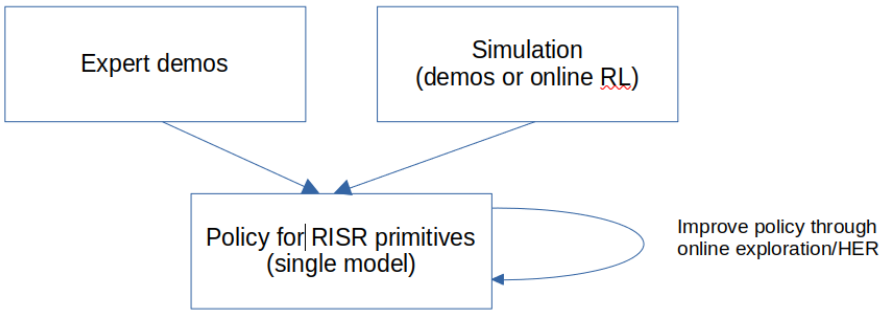


Figure 2: Policy training for each RISR primitive

design and can transfer well across tasks, object types. For example, to ensure the pick and place actions are generic across tasks, objects and robots; bounding boxes are provided as inputs to the Condition MLP policy as shown in Figure 3.

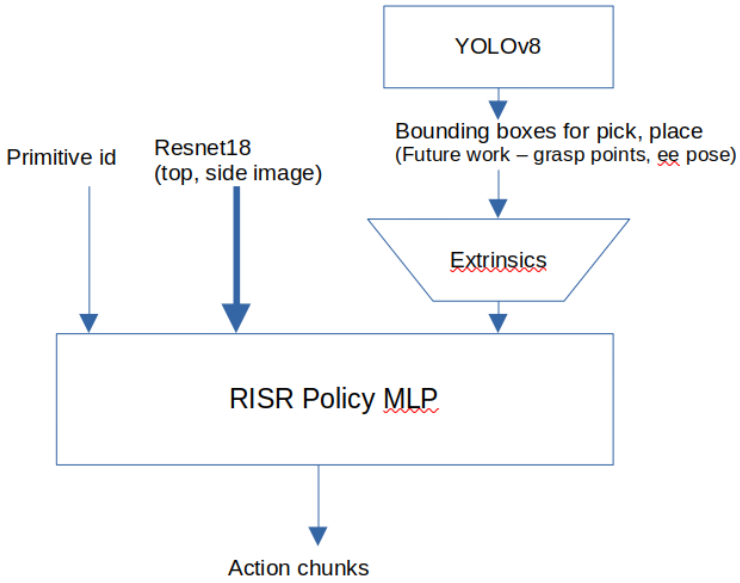


Figure 3: Conditional Policy MLP inference inputs/outputs

Details on some policies that were attempted for this project are in next sections.

3.1.1 Action Chunking Transformer based policy

ACT proposed in (6) is a behavior cloning model designed to learn long-horizon behaviors from demonstration trajectories by compressing full action sequences into compact latent chunks.

- It uses a vision encoder (e.g ResNet18) to process camera images into visual embeddings.
- A transformer-based encoder maps sequences of observations and actions into discrete latent tokens (z) representing full action chunks.
- The decoder reconstructs full action trajectories from these latent tokens conditioned on current observations.
- ACT allows efficient behavior cloning over temporally extended sequences rather than individual time steps.

- During inference, the model generates full action sequences (100 steps) per latent chunk, improving temporal consistency.

The ACT implementation from (4) is modified to also take primitive identifier (as a one-hot command), bounding boxes as training inputs.

3.1.2 IQL with Action Chunking Transformer based policy

Implicit Q learning (IQL) (7) based policy extends ACT to do offline reinforcement learning

- Rather than purely cloning expert demonstrations, IQL learns value-weighted policies using offline datasets.
- The ACT encoder extracts latent representations from trajectories.
- The IQL critic, value, and actor networks operate directly in the ACT latent space.
- IQL updates uses expectile regression to stabilize value estimation, combined with advantage-weighted behavior cloning for policy updates.
- This allows leveraging large offline datasets with imperfect demonstrations while benefiting from ACT's temporally consistent latent representations.

The IQL policy implementation also takes primitive identifier (as a one-hot command), bounding boxes as training inputs. In addition, preprocessing is done to also provide next observation state, rewards and done values during training.

3.2 LLM or VLP aided RISR primitive based action plan

Once the Robot has trained a sufficient set of RISR primitives needed to accomplish long horizon tasks, the Robot is ready to service a user command. When the user provides a user text command and a goal image, the Robot agent provides the command, goal image, supported primitives to the LLM and requests an action plan based on the primitives. The Robot agent onboard the computer then acts as a local planner executing the action plan and then periodically assessing the status of the execution. This is shown in Figure 4
Similar to pi-0, a Vision-Language-Primitive model can also be pre-trained and stored onboard the Robot to reduce the dependency on LLMs that are expensive in terms of cost and resources.

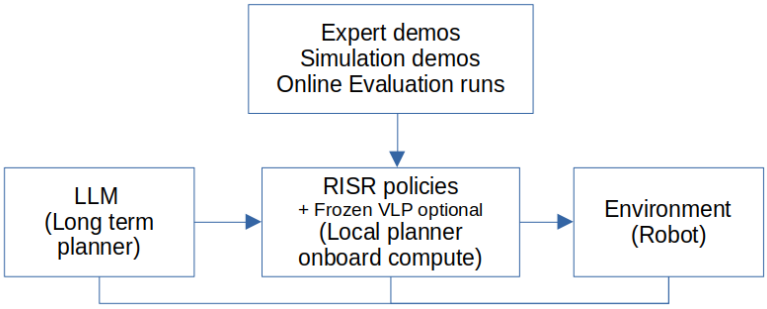


Figure 4: LLM or VLP aided RISR primitive based framework

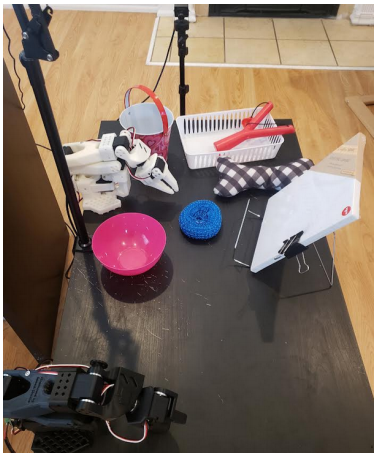
4 Experimental Setup

4.1 SO100 arm

SO100 (4) is a low-cost arm designed by Robotstudio which is a good platform for testing imitation learning and reinforcement learning algorithms on a real hardware prototype. It has a dual leader and follower arm which allows teleoperated demonstrations to be collected as a dataset. These demos can then be used for training. Figure 5 shows the setup where multiple objects were used to collect demos of pick, place, wiping based action primitives.

Two sets of tele-operated demos were collected on so100 arm

((a)) SO100 Leader Follower



((b)) SO100 setup

Figure 5: SO100 setup and objects used

- For stage 1 tests, collected 61 teleop demos of picking a scrub and placing in a bowl.
- For stage 2 tests, collected 230 teleop demos with multiple pick (scrub, crinkle, catapult, bucket) objects, place (bowl, tray) objects, dip (bowl with water) and a canvas to perform wiping action. This dataset can be found in (9) for public use.

4.2 YOLOv8 object detection

A dataset (8) of 494 images were collected and bounding boxes were recorded for different object types. The object detection was used for bounding box estimation and also for rewards modeling and task completion checks to aid Reinforcement Learning implementation.

4.3 Mani-skill simulation

ManiSkill (11) is a framework for robot simulation and training, with a strong focus on manipulation skills. It includes support for multiple robot embodiments including SO100. Custom implementation was added to streamline the dataset collection from simulation to match the training requirements for the real robot setup

- Camera views similar to the SO100 setup in 5
- Bounding boxes for the object
- Rewards needed for reinforcement learning

Figure 6 shows the setup to train a policy for the pick task. The cube object was selected as it is not present in the tele-operated dataset collected with the physical so100 arm.

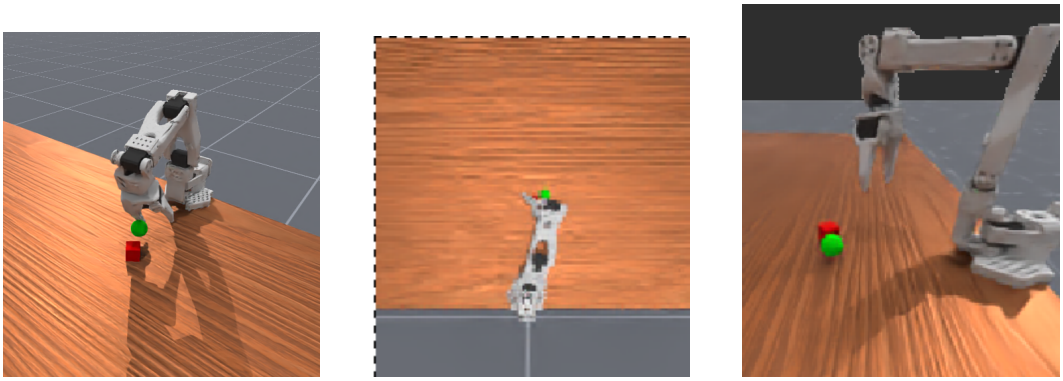
4.4 LeKiwi Mobile manipulator

In order to test cross-robot adaptation and goal-oriented long horizon tasks, a LeKiwi (5) robot was setup as shown in Figure 7. The challenges in reusing policy trained on SO100 robot is that the camera placement is different in the LeKiwi robot. The goal of the policy MLP in Figure 3 is to make the cross-robot adaptation of the trained model agnostic to hardware setup. More extensive testing on LeKiwi robot setup is considered as part of **future work**

5 Results

5.1 SO100 leader follower arm

Stage 1: Single monolithic task on SO100



((a)) Maniskill pick cube ((b)) Top view ((c)) Side view

Figure 6: Maniskill simulation setup for SO100 pick cube

((a)) Lekiwi robot ((b)) Lekiwi setup

Figure 7: Lekiwi cross-robot test setup

RISR primitive supported

- Pick_and_Place(pick_object, place_object)

The policies tested were

- ACT policy (8)
- IQL with ACT policy implemented with expectile=0.9. Also implemented rewards, completion check with YOLOv8 object detection. (8)

Both policies ended up successfully completing the task. However, the success rate of ACT policy was better and had smoother action. The challenge for the IQL policy is it needs additional fine-tuning of critic, value and actor networks; and hand-crafted rewards shaping (see (10)).

Stage 2: Compound task with RISR primitives on SO100

RISR primitives supported

- Pick(pick_object, pick bounding box)
- Place(place_object, place bounding box)
- Dip(place_object, place bounding box)
- Wipe(target_object, target bounding box)

The policies tested were

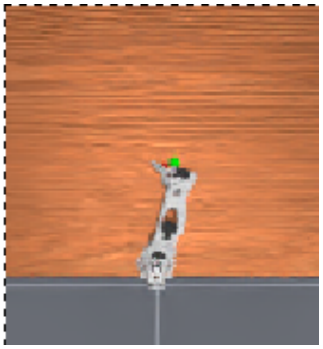
- Conditional ACT policy for various tasks where primitive id is coded as one-hot-vector input along with bounding boxes.

The compound tasks are

- Pick and Place (tested with multiple object types) - See videos in (10)
- Pick and Dip and Wipe (****future work****)

5.2 Mani-skill sim-to-real

With dataset of 50 episodes collected from ManiSkill so100 based pick-cube episodes, the policy model was retrained. Evaluation on a real physical robot to pick a small cubical ball was attempted. It was able to pick the ball 50% of the time with failures possibly due to sim-to-reality gap, and also the small size and slippery nature of the ball (see Figure 8).



((a)) Pick cube in simulation

((b)) Pick cube (ball) in real with so100

Figure 8: Maniskill sim to real

Hindsight Experience Replay (HER) updates Evaluation runs can be utilized by detecting the final state (using visual detection of final bounding boxes reached by gripper) and then use them to improve the policy. This can also be used to self-learn as an exploration policy. Adding these evaluation runs with HER updates, helped supplement the Sim-to-real bootstrapped policy model and improve its success rate.

5.3 LeKiwi Mobile manipulator

(****Future work****)

Goal is to perform cross-robot long-horizon tests on LeKiwi. The compound tasks are

- Navigate, Pick, Navigate and Place
- Cleaning a room task $\{Navigate, Pick, Place, Dip, Wipe\}^*$

6 Discussion

Some learning from the experiments performed to implement a multi-task long-horizon robot are

- For low-cost physical robots, onboard compute constraints how large the inference model can be in terms of memory and latency. If the sensors (like camera) have a frame rate of 30, it implies inference has to complete in $\sim$ 30 ms. Action chunking (e.g with ACT of 100 steps) thus produces smoother performance but may have limitations for highly dexterous tasks.
- Our approach of using single conditional model for all primitives reduces latency and burden on memory resources.

- The benefit of training simplified RISR primitives is that it reduces dataset preparation efforts (e.g A pick and place of 5 different pick objects into 5 different place objects usually requires 5x5=25 set of demos. With simplified RISR primitives, we need 5 demos for pick and 5 demos for place i.e total of 10. This also allows us to focus on objects that need more dexterous handling.)
- Rewards shaping and Task completion check using visual tools e.g YOLOv8 requires a lot of hand-crafted effort.
- Our approach of a lean trained policy MLP model is required towards making the model generalize across tasks, objects and robots.

For future work, more focus is needed on following

- On-demand sim-to-real training of new tasks and object types
- Generate a rewards model without explicit hand-crafting
- Work on a compact 'Vision-Language-Primitive' model that can run on real robots.
- Make the robots self-learn through experience and exploration

7 Conclusion

The project was able to validate and demonstrate the foundations needed for a goal-oriented long-horizon multi-task robot trained with both expert demonstrations and sim-to-real training. Recommendations were suggested to make the policy training agnostic to variations in tasks, objects and robot embodiments. The project was able to identify the challenges for designing a self-learning and planning multi-task robot.

8 Team Contributions

- **Harish Balasubramaniam:** Project idea, research, software design and implementation. In addition this project also required considerable time to setup and calibrate the physical robots.

Changes from Proposal Initial project proposal had greater emphasis on sim-to-real approach. The project evolved to use expert demos on a physical robot supplemented by mani-skill based simulation runs.

References

[1] π-0: A VLA Flow Model for General Robot Control `https://www.physicalintelligence.company/download/pi0.pdf`

[2] RT2: Vision Language Action Models `https://robotics-transformer2.github.io/`

[3] SLAC: Simulation-Pretrained Latent Action Space for Whole-Body Real-World RL `https://arxiv.org/abs/2506.04147v1`

[4] SO100 arm - LeRobot `https://github.com/huggingface/lerobot/blob/main/examples/10_use_so100.md`

[5] LeKiwi - LeRobot `https://github.com/huggingface/lerobot/blob/main/examples/11_use_lekiwi.md`

[6] Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware `https://arxiv.org/abs/2304.13705`

[7] Offline Reinforcement Learning with Implicit Q-Learning `https://arxiv.org/abs/2110.06169`

[8] Code changes for SO100 arm and YOLOv8 object detection dataset https://github.com/harishbalasub/lerobot

[9] Hugging Face dataset with teleop demos for multi-task SO100 model https://huggingface.co/datasets/harishbalasub/so100_diverse_tasks

[10] Videos of SO100 pick and place https://photos.app.goo.gl/mK4ZB4vRsdZUX5Kg9

[11] ManiSkill simulation framework https://maniskill.readthedocs.io/en/latest/index.html