Beyond Pick-and-Place: Tackling Robotic Stacking of Diverse Shapes

Alex Lee GK, Coline Devin, Yuxiang Zhou, Thomas Lampe, Jost Tobias Springenberg, Abbas Abdolmaleki, Konstantinos Bousmalis

With help and advice from: Arun Byravan, Nimrod Gileadi, David Khosid, Claudio Fantacci, Jose Chen, Akhil Raju, Rae Jeong, Stefano Sacileti, Federico Casarini, Martin Riedmiller, Raia Hadsell, and Francesco Nori
Machine learning can be extremely effective

Simple outputs

Standardized inputs

Large Datasets

| Dataset                | Quantity (tokens) | Weight in training mix | Epochs elapsed when training for 300B tokens |
|------------------------|-------------------|-------------------------|---------------------------------------------|
| Common Crawl (filtered) | 410 billion       | 60%                     | 0.44                                        |
| WebText2               | 19 billion        | 22%                     | 2.9                                         |
| Books1                 | 12 billion        | 8%                      | 1.9                                         |
| Books2                 | 55 billion        | 8%                      | 0.43                                        |
| Wikipedia              | 3 billion         | 3%                      | 3.4                                         |
ML in Robotics

What inputs should we give model?

How do we determine the “correct” action for a particular input?

How do we get enough data to train a model?
ML in Robotics

Large scale data through parallelism

(Robotics at Google, 2016)

Large scale data through simulation and domain randomization

(OpenAI, 2019)
Stacking

An arm

Some objects

Stack!
Some Terminology

Agent outputs actions according to a parametric policy to maximize future rewards.

Agent receives observations or states and rewards from an environment.
Stacking: Why use learning-based methods?

1. Move open gripper to top object centroid
2. Grasp top object
3. Move closed gripper to safe height
4. Move closed gripper to bottom object hover position
5. Move closed gripper to bottom object stack position
6. Open gripper at bottom object stack position
7. Move open gripper to top object hover position

Steps:
- Step 0: Init
- Step 1: Move open gripper to top object centroid
- Step 2: Grasp top object
- Step 3: Move closed gripper to safe height
- Step 4: Move closed gripper to bottom object hover position
- Step 5: Move closed gripper to bottom object stack position
- Step 6: Open gripper at bottom object stack position
- Step 7: Move open gripper to top object hover position
- Step 8: End
Scripted Policy

It’s easy to think of object manipulation as “draw a square around the object” and “pick up object”
Scripted Policy: Grasping is hard

For many objects, the robot needs to reason about the geometry of the object to grasp it successfully.
Scripted Policy: Placing is hard

To make a stable stack, the robot also must consider the shape and orientation of the base object.
Some prior DM work on robotic stacking

2016: Lego blocks
Popov et al 2017, arxiv

2017: Foam blocks
Zhu et al RSS 2018

2018–2019: Rigid color-coded blocks
Jeong et al ICRA 2019, Wulfmeier et al RSS 2020

2019: Color-coded squishy blocks
Cabi et al RSS 2020
Stacking is not just pick-and-place

Grasping requires *precise positioning* and/or orientation.

Objects afford different grasping/stacking behaviors, which change when on the slanted side of the basket.

After grasping, attention should be switched to the *relative positions between the two objects*.

The gripper can get jammed due to the distractor.
RGB-Stacking: From pick-and-place to diverse objects

Task (clear metric): success percentage of stacking red on blue, in 20s, ignoring green
A systematically generated set of objects that vary in physically meaningful ways
The different axes of deformation affect the relative affordances of the objects for stacking.

| Axes of Deformation | Seed | Polygon | Trapezoid | Parallelogram | Rectangle |
|---------------------|------|---------|-----------|---------------|-----------|
| Seed                | ![Seed](image) | ![Polygon](image) | ![Trapezoid](image) | ![Parallelogram](image) | ![Rectangle](image) |
| Polygon             | ![Polygon](image) | ![Polygon](image) | ![Trapezoid](image) | ![Parallelogram](image) | ![Rectangle](image) |
| Trapezoid           | ![Trapezoid](image) | ![Trapezoid](image) | ![Trapezoid](image) | ![Parallelogram](image) | ![Rectangle](image) |
| Parallelogram       | ![Parallelogram](image) | ![Parallelogram](image) | ![Parallelogram](image) | ![Parallelogram](image) | ![Rectangle](image) |
| Rectangle           | ![Rectangle](image) | ![Rectangle](image) | ![Rectangle](image) | ![Rectangle](image) | ![Rectangle](image) |
One Benchmark, Two Tasks

Skill Mastery
- **Train** and **Test** objects are the same

Skill Generalization
- **Train**: random RGB objects from non-heldout axes
- **Test**: 5 eval triplets from held-out axes

Train and Test

| Triplet 1 | Triplet 2 | Triplet 3 | Triplet 4 | Triplet 5 |
|-----------|-----------|-----------|-----------|-----------|
| ![Image](link) | ![Image](link) | ![Image](link) | ![Image](link) | ![Image](link) |

Test

![Image](link)

Training axes
Benchmark Challenges

Grasping at wrong angle

Stacking impossible on sloped surface

Center of mass alignment

Align flat surfaces before stacking

Red object can easily roll off

Triplet 1

Triplet 2

Triplet 3

Triplet 4

Triplet 5
Reinforcement Learning

Since we can’t just write out the steps to tell a robot how to stack, we instead use reinforcement learning so the robot can learn through trial and error.
Reinforcement Learning: What is the reward?

In the real world, all we have is a sparse (binary) success label: the center of mass of the red object is above the center of mass of the blue object, and the gripper is open.
Reinforcement learning in the real world?

5 robots running in parallel

We would probably need on the order of 1 million stack attempts to learn from images with a sparse reward in the real world.

Each can do 1000 stack attempts per day

200 days of continuously running for 1 experiment.

RL has many hyperparameters to tune, requiring many experiments to get a good, reproducible, setup.
Reinforcement learning: simulation
In simulation, we can use the object poses directly to compute a “dense” reward.
Approach: Sim2Real with interactive Distillation + offline RL

We approach the problem using a learning pipeline split into three decoupled stages:

- Policy training from state
- Interactive distillation from images with randomization
- One-step policy improvement (Offline RL)
Reinforcement learning from state in simulation

State
Object poses, object parameters, proprioception, simulation state

Dense reward
Shaped stacking reward

Simulation environment

Action

State-based agent
Interactive imitation learning in domain-randomised simulation

72% success in simulation

68% success when evaluated in the real word
One-step policy improvement from real data
Collect data on robots using sim-to-real zero-shot vision-based policy
One-step policy improvement from real data

Collect data on robots using sim-to-real zero-shot vision-based policy

Observations
Image pair and proprioception

Sparse reward
Stacking indicator

Real robot environment

Sim-to-real zero-shot vision-based agent

Dataset

Offline reinforcement learning from dataset of real data

Observations, sparse reward, action

Improved real-world vision-based agent
## Results on Skill Mastery and Generalization

|                  | Skill Mastery | Skill Generalization |
|------------------|---------------|----------------------|
|                  | Sim2Real zero-shot | Improved Vision Policy | Sim2Real zero-shot | Improved Vision Policy |
| Human            | 47%           | 68%                  | 52%               | 33%                 |
| Scripted         | 51%           | 68%                  | 54%               |                     |
| IIL-Sim2Real     | 68%           | 82%                  |                    |                     |
| Data             |               |                      |                    |                     |
| CRR-IMP          |               |                      |                    |                     |

* Data from earlier policy

Full per triplet results + baselines in the paper
Best Skill Mastery Agent (CRR-Improvement)

Best **Skill Mastery** agent: One-step policy improvement on Sim2Real with CRR achieves **82%**
Generalization Policy Examples

Triplet 4

Triplet 5

Real Robot Stacking Success

- Skill Mastery
- Skill Generalization

Triplet 1
Triplet 2
Triplet 3
Triplet 4
Triplet 5
Average
Generalization Policy Examples

Triplet 4

Triplet 5

Triplet 2

Triplet 1

Real Robot Stacking Success

- Skill Mastery
- Skill Generalization

| Triplet 1 | Triplet 2 | Triplet 3 | Triplet 4 | Triplet 5 | Average |
|-----------|-----------|-----------|-----------|-----------|---------|
| 100       | 75        | 75        | 75        | 75        | 75      |
| 75        | 50        | 50        | 75        | 75        | 75      |
| 75        | 50        | 50        | 75        | 75        | 75      |
| 75        | 50        | 50        | 75        | 75        | 75      |
| 75        | 50        | 50        | 75        | 75        | 75      |
Takeaways

- We introduced the **RGB-Stacking** challenges of stacking diverse objects in two settings: **Skill Mastery** and **Skill Generalization**
- **Simulation to real world transfer** with interactive improvement achieves: 82% (Mastery) and 54% (Generalization)

→ We are good when the objects are then same for training (in simulation) and testing (in the real world), but do not generalize well to new objects.
→ Can we **quickly adapt** the generalist to new objects in the real world?
What if we are given new objects only in the real world?

Problem Setting:

- We have some stacking teacher policy trained on some set of training objects in simulation.
- We are given new test objects in the real world.
- We want to produce the best stacking policy for the test objects in a fixed amount of time.

“Data Budget”

- To improve on the test objects we need to collect real world interactions using those objects.
- Real world data is expensive!!
- We have several options of how to collect this data
  - Run the teacher policy on these new objects and do CRR-IMP or other offline algorithm on the resulting data.
  - Run an online algorithm directly, using the teacher as a prior.
  - Some combinations of the two
We investigate this problem through the lens of specializing to individual triplets

- Each object requires different behaviors
- Let’s train policies one only one triplet, using a generalist teacher to accelerate learning.
In this work:

Goal:
- be good at a specific target task

We have:
- A suboptimal, queryable teacher
- Access to the target task environment
  - for a limited number of episodes (“Data Budget”)
  - sparse reward
One way to do this is CRR-IMP, as before

Offline training from dataset

1. Collect dataset by the teacher in the environment
2. Offline RL from dataset (teacher and environment not used anymore)
One version is interactive distillation, but it cannot improve upon the teacher.

- All data is sampled collected by the student
- All supervision is from the teacher
Another version is CRR-IMP as before

- All data is sampled collected by the teacher
- All supervision is from reward
  - Can improve!
We can combine all of these ideas:

- Collect some data by running the teacher
- Collect some data from the student
- Supervise the student using both the reward and the teacher.
After improving upon the generalist for 40k episodes on Triplet 1

Successes (81.5%)
- Rotates to the ideal gripper orientation before grasping
- Grasps from the riskier orientation

Failures (18.5%)
- Out-of-distribution corner case
- Early termination
And similarly for Triplet 2

Successes (54.5%)
- Flips blue object with the grasped red object
- Grasps from the riskier orientation

Failures (45.5%)
- Attempting to stack on a non-horizontal surface
- Same
Where does this leave us?

- If you have suboptimal data lying around, Offline RL (like CRR-IMP) is a great way to get a step of improvement without any additional data collection.
- Collecting some data interactively can lead to more improvement if the right hyperparamers.

But:
- Real world experiments are always difficult to reproduce: differences in the hardware, lab, etc all affect the results.
- Simulation results often don’t match real world results.
Thank you!

Stacking random & unseen objects

Successes

Failures

2x