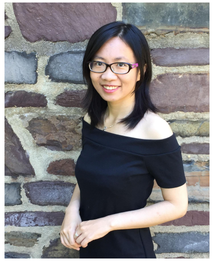
Learning Synergies between Pushing and Grasping with Self-supervised Deep Reinforcement Learning

11/10/2019

Authors

Andy Zeng

PhD Student in Princeton, Researcher in Google Brain.



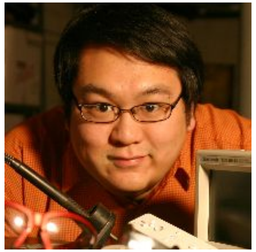
Shuran Song

Assistant Professor in Columbia University.

Stefan Welker

Staff Engineer in Google.

No Tossingbot :((



Johnny Lee

Researcher in Google Brain, entrepreneur, Kinect developer.

Alberto Rodriguez

Associate Professor in MIT.

Thomas Funkhouser

Emeritus Professor in Princeton, Researcher in Google Brain.

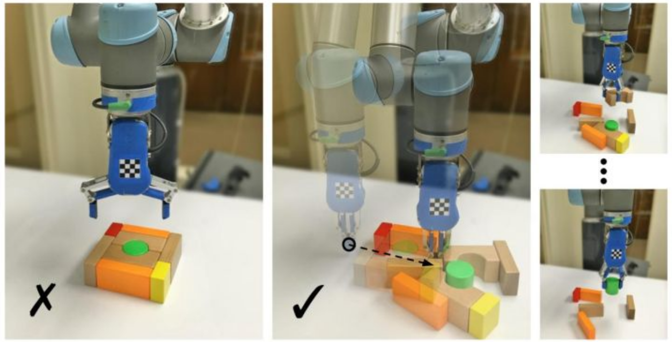
★ Best Cognitive Robotics Paper Award Finalist, IROS ★

Introduction

Motivation

Benefit from the 'synergies' between pushing and grasping actions.

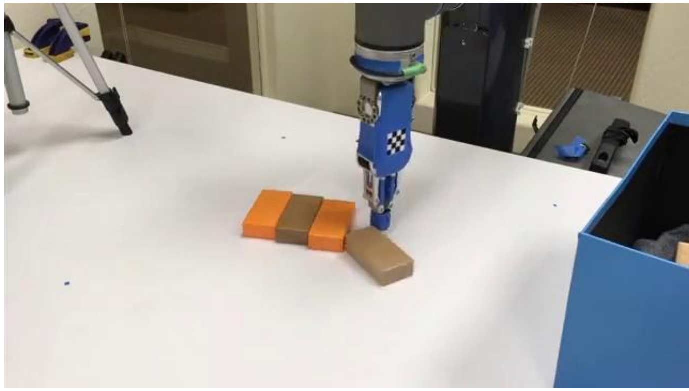
- Pushing rearranges cluttered objects to make room for grasping.
- Grasping makes further pushes more precise.
- Tightly packed blocks, no immediate grasping afforded.



Approach

Learn synergies from scratch via model-free deep reinforcement learning.

- Train 2 separate DQNs (ϕ_p, ϕ_g) that map from visual observations to actions. (sample dense pixel-wise end effector orientations and locations)
- Joint policy configuration for selecting between grasping and pushing.
- Entirely self-supervised, rewards provided by successful grasps.



Contributions

- Learn complementary pushes and graspes simultaneously.
- Learn from complex, cluttered environments quickly.
- Less training, better results than the state of the art.
- Generalizes to novel objects.

Model Architecture

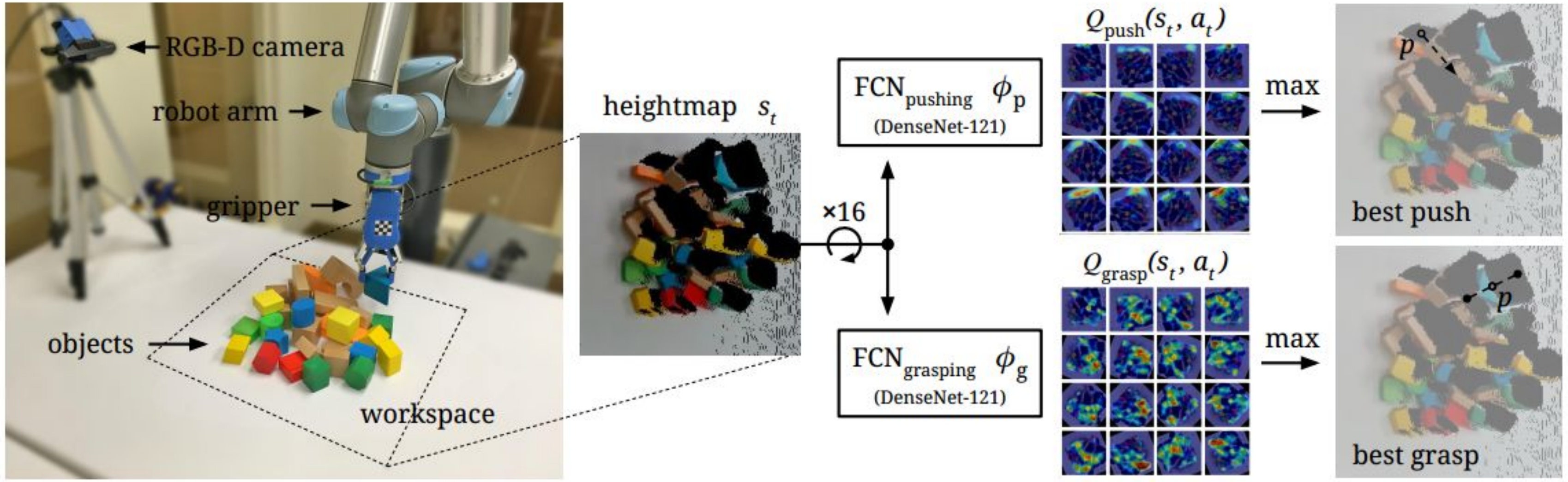

RGB-D camera
robot arm
gripper
objects
workspace
heightmap s_t
×16
FCN$_{\text{pushing}}$ ϕ_{p}
(DenseNet-121)
FCN$_{\text{grasping}}$ ϕ_{g}
(DenseNet-121)
$Q_{\text{push}}(s_t, a_t)$
$Q_{\text{grasp}}(s_t, a_t)$
max
max
p
best push
p
best grasp

Fig. 2. **Overview** of our system and Q-learning formulation. Our robot arm operates over a workspace observed by a statically mounted RGB-D camera. Visual 3D data is re-projected onto an orthographic RGB-D heightmap, which serves as a representation of the current state s_t. The heightmaps are then fed into two FCNs - one ϕ_{p} inferring pixel-wise Q values (visualized with heat maps) for pushing to the right of the heightmap and another ϕ_{g} for horizontal grasping over the heightmap. Each pixel represents a different location on which to execute the primitive. This is repeated for 16 different rotations of the heightmap to account for various pushing and grasping angles. These FCNs jointly define our deep Q function and are trained simultaneously.

Problem Formulation

- Formulated as a MDP.
- Off-policy Q-learning that maximizes the Q-function.
- DQNs are used → Q-function approximation.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\left[r_t + \lambda \max_{a} Q(s_{t+1}, a) - Q(s_t, a_t)\right]$$

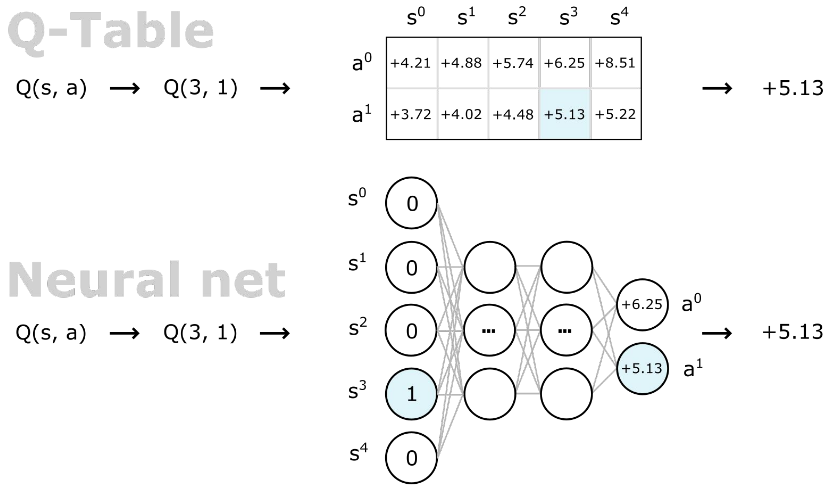
🟦 ~~Current Q-table value we are updating~~

🟪 ~~Learning rate~~

🟩 Reward

🟨 Discount

🟥 Estimated reward from our next action



	s^0	s^1	s^2	s^3	s^4
a^0	+4.21	+4.88	+5.74	+6.25	+8.51
a^1	+3.72	+4.02	+4.48	+5.13	+5.22

- Minimize $\delta_t = |Q(s_t, a_t) - (r_t + max_a Q(s_{t+1}, a)|$

Method - Visual Pushing for Grasping (VPG)

State : Each state is modeled as an RBG-D heightmap image, resolution of 224 x 224.

Primitive Actions : Each action is parameterized as a motion primitive behavior, 3D location q projected from pixel p.

$$a = (\psi, q) | \psi \in \{push, grasp\}, q \to p \in s_t$$

Pushing : q → starting position of a 10 cm push in one of k=16 directions.

Grasping : q → middle position of top-down parallel grasp in one of k=16 orientations.

Learning FCNs: 2 independent networks ϕ_p, ϕ_g

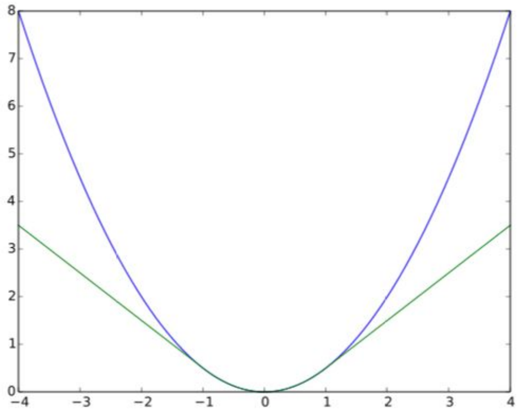
- **Input** : heightmap image representation. **Output**: Q-values for all the pixels for k=16.
- 224x224x32 → Q-values for all possible actions.
- Two parallel 121-layer DenseNet, pre-trained on ImageNet.

Rewards : Successful grasp : 1, Pushing that makes detectable changes : 0.5

Method

Training Details:

- Huber loss function. (quadratic loss for small values for the residual, linear loss for larger values)
- At each iteration backpropagate only through the single pixel p and its corresponding network.
- Prioritized experience replay.
- n objects. (10-30)
- At every training iteration;
    - Capture data.
    - Compute forward passes.
    - Executing an action.
    - Backpropagation, running a single iteration of experience replay.
    - Another forward pass and backprop. on a sample from the replay buffer.



Testing Details:

- A small learning rate to eliminate the issue of selecting always the same action when state is the same.
- Network weights are reset before each new test run.

Termination Conditions:

- All objects are successfully grasped.
- No change for 10 successive actions detected.

Experiments

Goals:

- Pushing contribution to grasping.
- Check the feasibility of training pushing policies only with grasping supervision.
- Self-supervised, end-to-end learning of pushing-for-grasping policies.

Reactive Grasping Only: Same state-action representation, single FCN supervised with binary classification for grasping, maximize immediate grasping affordance.

Reactive Pushing and Grasping Policy: two FCNs, one for pushing, one for grasping, binary classification as well.

Evaluation Metrics:

For each test, n (10,30) runs are executed.

- The average completion rate.
- The average grasp success rate per completion.
- Action efficiency → #objects in test / #actions before completion. (same as average grasp success for grasping-only)

- Simulation + Real World Experiments

Simulation

- n = 30 runs, 9 different 3D toy blocks chosen randomly during the experiment.

TABLE I
SIMULATION RESULTS ON RANDOM ARRANGEMENTS (MEAN %)

Method	Completion	Grasp Success	Action Efficiency
Grasping-only [8]	90.9	55.8	55.8
P+G Reactive	54.5	59.4	47.7
VPG	**100.0**	**67.7**	**60.9**

TABLE II
SIMULATION RESULTS ON CHALLENGING ARRANGEMENTS (MEAN %)

Method	Completion	Grasp Success	Action Efficiency
Grasping-only [8]	40.6	51.7	51.7
P+G Reactive	48.2	59.0	46.4
VPG	**82.7**	**77.2**	**60.1**

TABLE III
COMPARISON WITH MYOPIC POLICIES (MEAN %)

Method	Completion	Grasp Success	Action Efficiency
VPG-myopic	79.1	74.3	53.7
VPG	**82.7**	**77.2**	**60.1**

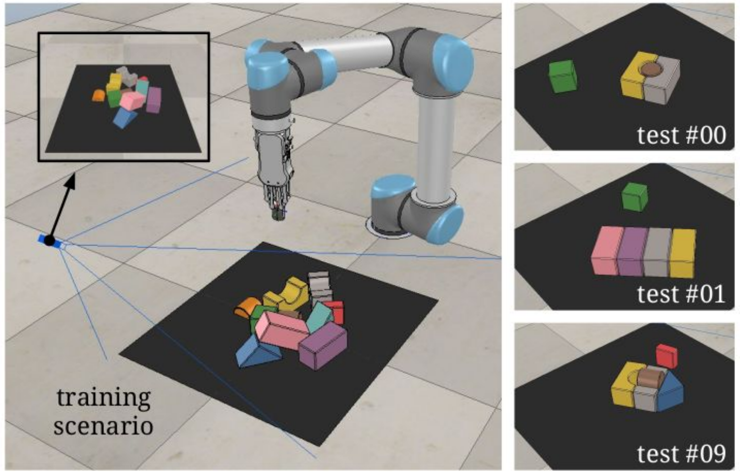


Fig. 3. **Simulation environment.** Policies are trained in scenarios with random arrangements of 10 objects (left), then evaluated in scenarios with varying degrees of clutter (10 objects, 30 objects, or challenging object arrangements). In the most challenging scenarios, adversarial clutter was manually engineered to reflect challenging real-world picking scenarios (*e.g.* tightly packed boxes, as shown on the right).

Simulation

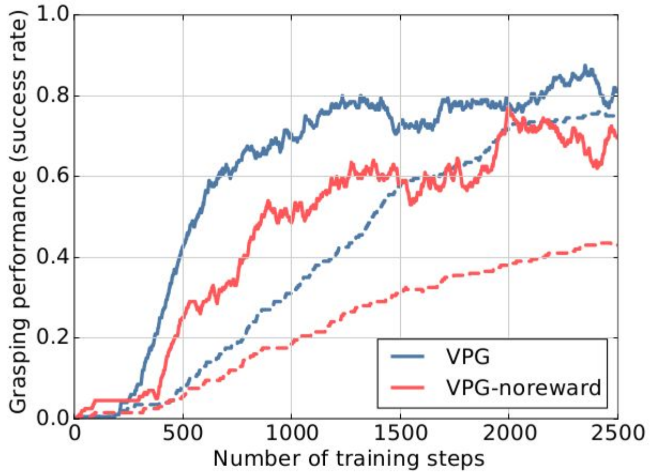


Fig. 4. Comparing performance of VPG policies trained with and without rewards for pushing. Solid lines indicate % grasp success rates (primary metric of performance) and dotted lines indicate % push-then-grasp success rates (secondary metric to measure quality of pushes) over training steps.

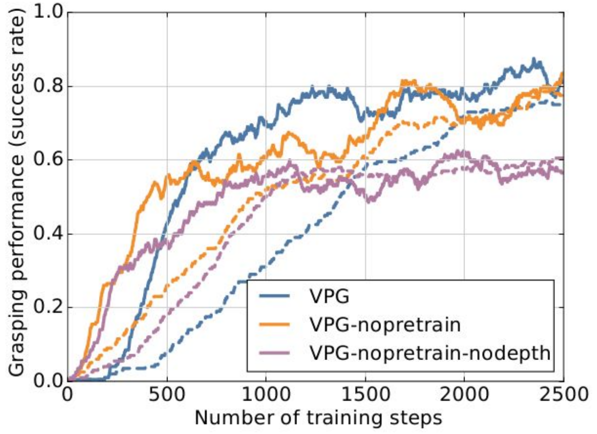


Fig. 5. Comparing performance of VPG policies initialized without weights pre-trained on ImageNet and without the depth channels of the RGB-D heightmap (*i.e.* no height-from-bottom, only color information). Solid lines indicate % grasp success rates (primary metric of performance) and dotted lines indicate % push-then-grasp success rates (secondary metric to measure quality of pushes) over training steps.

Real-World Experiments

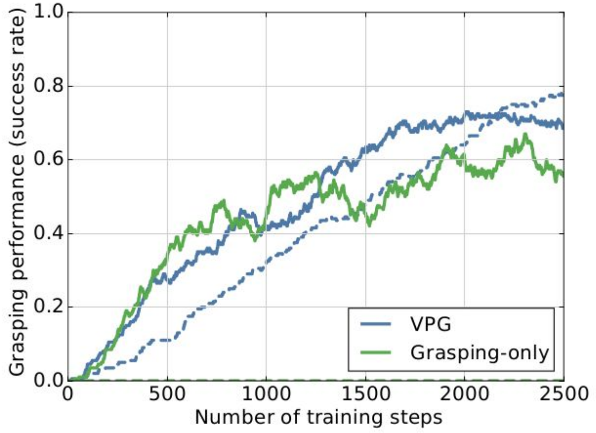


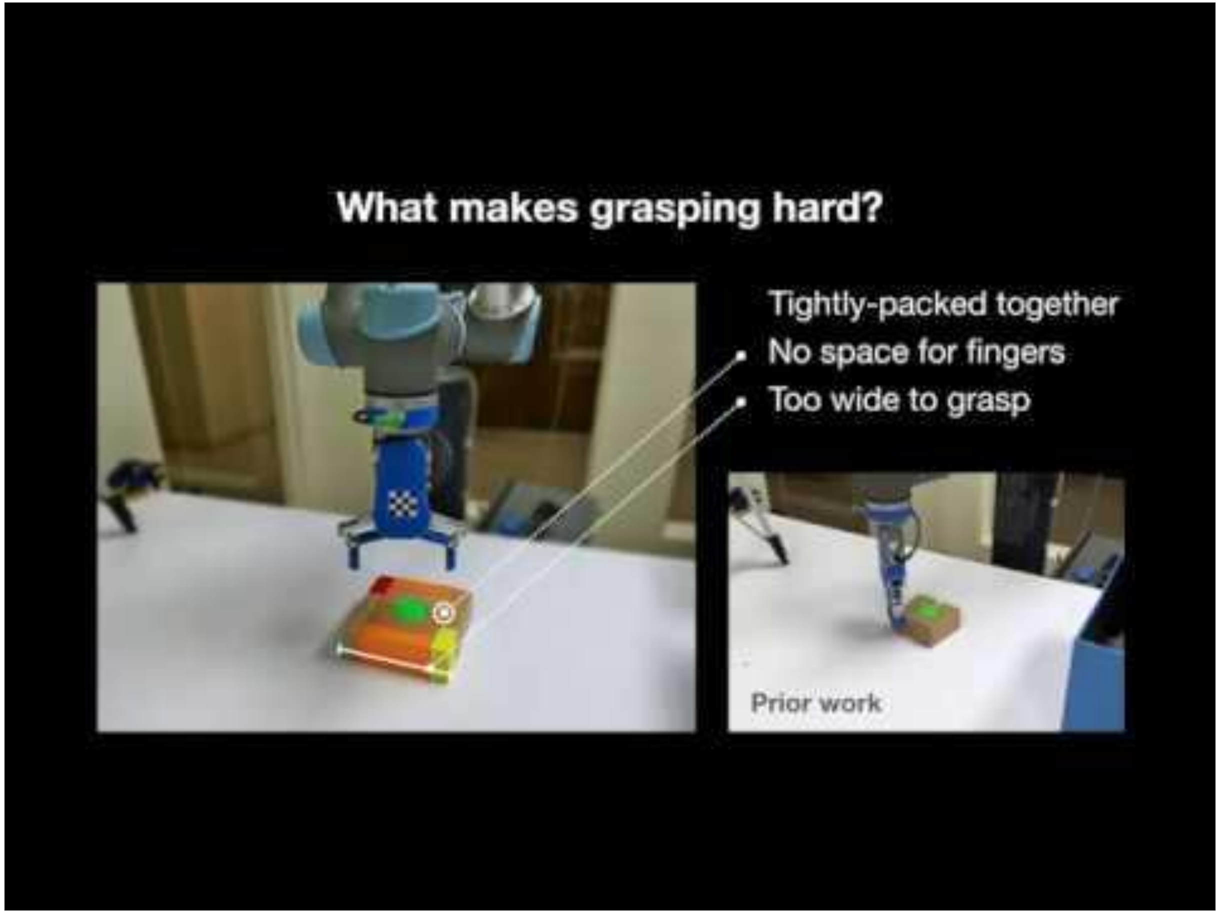
Fig. 6. Evaluating VPG in real-world tests with random 30+ object arrangements. Solid lines indicate % grasp success rates (primary metric of performance) and dotted lines indicate % push-then-grasp success rates (secondary metric to measure quality of pushes) over training steps.

TABLE IV

REAL-WORLD RESULTS ON CHALLENGING ARRANGEMENTS (MEAN %)

Method	Completion	Grasp Success	Action Efficiency
Grasping-only [8]	42.9	43.5	43.5
VPG	**71.4**	**83.3**	**69.0**

Video



What makes grasping hard?

- Tightly-packed together
- No space for fingers
- Too wide to grasp