Learning Motor Skills of Reactive Reaching and Grasping of Objects

Wenbin Hu, Chuanyu Yang, Kai Yuan, Zhibin Li

***Abstract*— Reactive grasping of objects is an essential capability of autonomous robot manipulation, which is yet challenging to learn such sensorimotor control to coordinate coherent hand-finger motions and be robust against disturbances and failures. This work proposed a deep reinforcement learning based scheme to train feedback control policies which can coordinate reaching and grasping actions in presence of uncertainties. We formulated geometric metrics and task-orientated quantities to design the reward, which enabled efficient exploration of grasping policies. Further, to improve the success rate, we deployed key initial states of difficult hand-finger poses to train policies to overcome potential failures due to challenging configurations. The extensive simulation validations and benchmarks demonstrated that the learned policy was robust to grasp both static and moving objects. Moreover, the policy generated successful failure recoveries within a short time in difficult configurations and was robust with synthetic noises in the state feedback which were unseen during training.**

I. Introduction

Adaptive and reactive grasping of various objects under disturbances are underpinning capabilities for autonomous robot manipulation. In particular, the abilities to react quickly to sudden changes and robust recovery from failures are essential to the robot autonomy, which can significantly increase success rate in a broad range of real-world scenarios. However, a coherent control scheme combining both dynamical reaching and grasping remains an open research.

In the conventional control, reaching and grasping are addressed separately and executed in a cascaded manner. Generally, conventional planning based methods have good results in solving either reaching [1] or grasping problem [2] individually, but the switch between controllers is designed manually. As a next-level performance with increased robustness, reaching, grasping, and even failure recovery need to be addressed *simultaneously within one unified policy*.

Learning based approaches are very attractive alternatives for producing autonomous control policies in general, as they alleviate tedious manual design and demonstrated success in robot locomotion [3], we believe that learning can be explored for solving robotic grasping problems as well. Compared with classical grasp synthesis, training models in learning is data hungry, although the performance of grasping unknown objects is improved as in [4], [5]. Also, collecting data from simulation [4] or self-supervised real robot experiments [5] is time-consuming. In case of failures, the whole pipeline has to reset, instead of an online reactive recovery strategy [4].

All authors are with the School of Informatics, University of Edinburgh. Email: wenbin.hu@ed.ac.uk

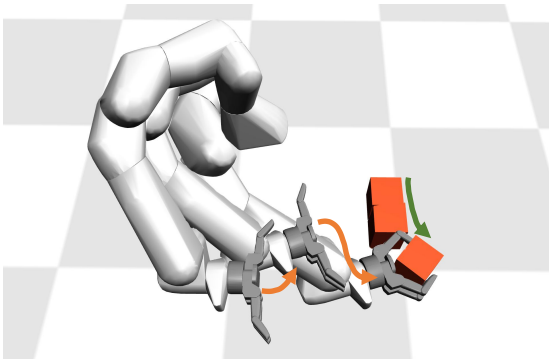


Fig. 1: Reactive and coordinated hand-finger motions for grasping a dynamically moving object.

Recent research in Deep Reinforcement Learning (DRL) has shown promising capabilities of solving continuous robot control tasks in high-dimensional state space, such as multi-finger grasping [6], in-hand manipulation [7]. In contrast to supervised and unsupervised learning, no pre-collected training data is required as the agent autonomously generates it by interacting with the environment, and infers the quality of the state-action pairs through reward signals.

In this work, we aim to propose a unified learned policy for reactive reaching and grasping of rigid objects in presence of disturbances and uncertainties. Unlike common approaches that first predict the grasping pose and then reach to the pose and grasp, we directly generate the control commands for both hand and fingers based on current state observation. As a proof of concept, we define the scope of such grasping problems as: planar grasps of objects that are placed on a level surface. We show that despite the policy is learned via grasping static objects placed on the ground, the learned skills can reach and grasp moving objects nevertheless.

Despite reaching and grasping can be realized by the existing planning or data-driven methods separately, our work focused on the study of learning a *unified control policy* to generate coherent and coordinating motor skills for reaching and grasping. In particular, the proposed policy is robust to the *sudden changes and movements of objects* and is capable of *recovering quickly from failures*. Reactive failure recoveries in real-time can significantly increase the grasping efficiency, compared to the conventional methods which simply restart a cascaded planning-and-control pipeline and reattempt the tasking using the same procedure.

The contributions of this paper are: (1) A deep reinforcement learning based framework for acquiring feedback control policies for reactive reaching and grasping; (2) Design of a task-orientated reward function, i.e. multiple geometric

metrics for object grasping, for learning a robust policy (Section III); (3) Design of state initialization to generate learning data to effectively learn reactive policies that can recover from grasping failures (Fig. 4).

In the remainder of this paper, we discuss the related work in Section II, and present the learning algorithm and elaborate on the simulation design for policy learning in Section III. The detailed results of learned policies are validated and analyzed in Section IV with success rate, and we conclude the paper and future work in Section V.

II. Related Work

Reaching and grasping are proven to be coupled in humans [8]. The developed controller in [9] used a coupled dynamical system to replan reaching and grasping according to sudden changes of objects. Coupled modeling of dynamical systems is shown to be feasible in learning highly coordinated hand-arm motions, e.g. catching flying objects [10]; however, human demonstrations are required to learn such coupled motion-models and the graspable space of the objects.

Other research solved the reaching and grasping through a combination of trajectory planning with policy learning [11]. The classic visual servoing was combined with DRL in [12], which included a long-range controller for reaching. and a short-range controller for more precise motion of reaching and grasping – they were activated by hand-crafted rules. The typical framework for grasping moving objects with robotic grippers contains different modules: offline generation of grasps, online selection of grasp trajectories, and a motion planner. Metrics, such as task space errors [13] or the reachability, can be used to select the optimal grasp from a candidate set [14]. The timing of closing the grippers is manually programmed by pre-defined criteria, instead of being learned automatically. Compared to the hand-crafted controllers, our approach learns both reaching and grasping in a holistic manner as one policy.

One major deficiency of learning-based grasp detection via visual input is the expensive computation and long wait time caused by large CNNs, individually sampled and ranked grasp candidates [5], [15]. This problem was overcome by a lightweight network structure that enabled reactive close-loop control [16], which dynamically tracked and grasped novel objects in clutter with a high success rate. However, this approach has not yet considered recovery strategies in case of failures, i.e. the object slips out of the robot hand. In our approach, we proposed an effective method of using state initialization to start policy training from difficult configurations, which induce failure grasps more often and prevent data samples being skewed, and therefore train recovery policies more effectively.

III. Methodology

A. Proximal Policy Optimization

We used an on-policy deep reinforcement learning algorithm named Proximal Policy Optimization (PPO) [17] for policy optimization. PPO is in an actor-critic fashion, with the actor consisting of a policy π_θ and a critic consisting of estimated value function V_ϕ. Both of the policy π_θ and the value function V_ϕ are parameterized with a fully-connected neural network with two hidden layers, as shown in Fig. 2.

B. Proposed Framework

The system's schematics is shown in Fig. 2. One training iteration consists of two procedures: data acquisition and update of network weights. Guided by the latest policy, the agent actively interacts with the environment and gathers the state-action-reward data tuples, and then the weights of actor and critic networks are updated separately.

The control framework consists of the neural network policy and the low-level joint controller. Before fed into the policy network as state input, the robot proprioception is low-pass filtered, and the object surface point cloud is down-sampled. To acquire smooth robot motion, the output commands from the policy are also processed by the lowpass filter. The filtered actions are executed by the low-level proportional-derivative (PD) controller at 500Hz.

C. Simulation and Training Setup

For realistic contacts and dynamics, we use the physics simulator MuJoCo [18], where the Barrett Hand is the end-effector that attached on the Franka Emika robot arm. The training objects consist of a cube and three household objects, as shown in Fig. 2. All objects are rigid with a uniform density. Every training episode starts with the randomized object pose on the ground (friction $\mu = 0.8$) within the workspace of the arm. Then the robot interacts with the environment for 8 seconds. Any self-collision will trigger the early termination. For simulating moving objects, we set an initial horizontal velocity at the object's center of mass, and the object slides on the ground due to its inertia.

For the object observation, two synthetic depth cameras are shooting from different angles in the simulation, so that even when the hand is grasping the object we can obtain enough surface points. For simplicity, the object point cloud is segmented using the color and the object meshes are imported from YCB database [19]. We are not matching the observed object point clouds between adjacent frames, so they can be different. However, the learned policy is able to extract the encoded object pose and surface information from partial point cloud, and generate stable actuator commands.

D. State and Action Space

The state vector is $\mathcal{S} = \{\boldsymbol{P}, \boldsymbol{q}, \theta, \boldsymbol{\tau}, \boldsymbol{d}, \boldsymbol{V}\}$, where $\boldsymbol{P}$ refers to the object point cloud positions relative to the position of the robot palm link, representing the object's shape and pose; $\boldsymbol{q}, \theta$ refer to the finger joint positions and hand rotation angle around z axis; $\boldsymbol{\tau}$ refers to the target finger joint torques; $\boldsymbol{d}, \boldsymbol{V}$ refer to the distances and unit vectors between the in-hand key points and their nearest object surface points. $\boldsymbol{d}, \boldsymbol{V}$ are also related to the reward computation, accelerating the policy convergence. In the computation of $\boldsymbol{d}, \boldsymbol{V}$, we use the processed dense object point cloud for better precision; but in $\boldsymbol{P}$, the number of points is downsampled to 32 using the

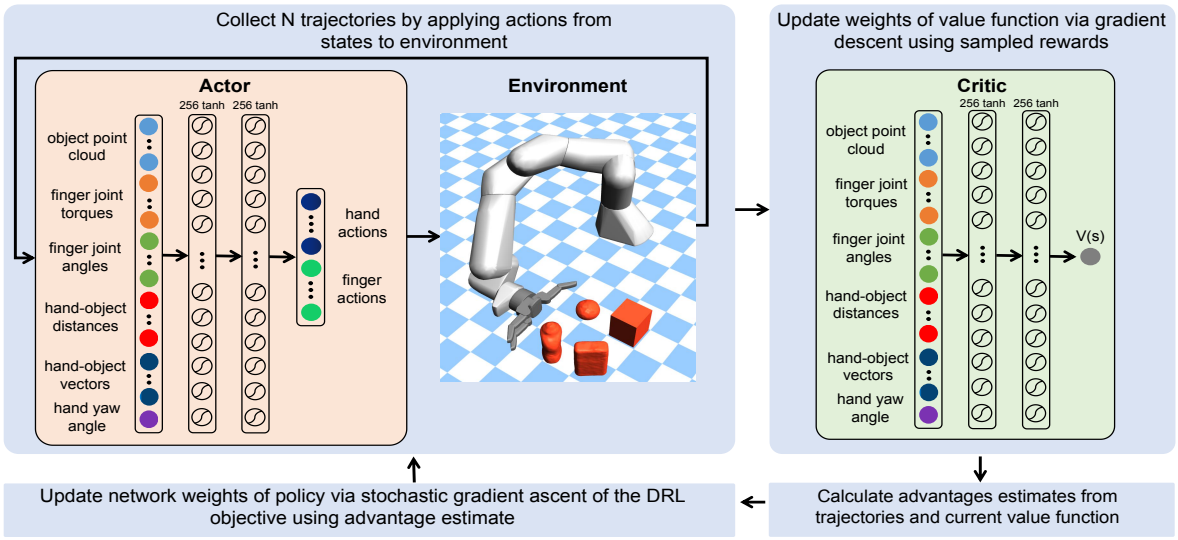


Fig. 2: Control framework of learning reactive reaching and grasping, detailing state observation and action space.

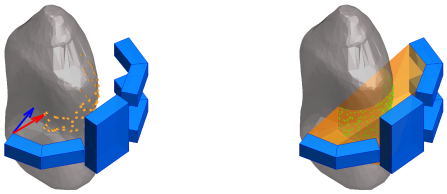


Fig. 3: (a) Divergence of vectors; (b) Object's surface points enclosed by the polyhedron formed by the hand.

voxel downsampling method, to reduce the dimension of the input vector for the neural networks.

As a proof of concept for the method, we focus on a planar case with the end-effector moving at the horizontal plane and grasping objects placed on the ground. Similar to a mobile robot, the robot hand has translational motions in the x, y coordinates at a fixed height of 0.06 m, and a rotational motion around z axis. Hence, the action space $\mathbf{A}$ includes: the translational velocities and the rotational velocity around z axis of the hand/wrist, and positions of all finger joints.

E. Reward Design

We formulated task-related quantities and proposed the following reward function, which is the weighted linear combination of positive rewards and negative penalties:

$$\mathbf{R} = \sum_{i=1}^{5} \omega_i r_i(\mathcal{P}^{\boldsymbol{o}}, \mathcal{P}^{\boldsymbol{h}}), \tag{1}$$

where $[\omega_1, \omega_2, \omega_3, \omega_4, \omega_5] = [2, 1, 1, 2, 0.3]$. $\mathcal{P}^{\boldsymbol{o}}, \mathcal{P}^{\boldsymbol{h}}$ represent the object point cloud and the robot in-hand key-points.

The term r_1 minimizes the distances between in-hand surface and the object, encouraging the hand to contact the object as much as possible, which is formulated as

$$r_1 = \exp(-\sum_{i=1}^{N_\text{h}} \|p_i^{\boldsymbol{o}} - p_i^{\boldsymbol{h}}\|), \tag{2}$$

where $p_i^{\boldsymbol{h}}, p_i^{\boldsymbol{o}} \in \mathbb{R}^3$ are the positions of in-hand key points and their nearest object points. $p_i^{\boldsymbol{o}}$ are illustrated in orange in Fig. 3. N_h is the number of hand key points. Exponential function regulates the value within $(0, 1]$.

For a stable grasp, the contact surfaces between hand and object should comply with each other; and at each contact point, the normal vectors of both surfaces should align with each other. Considering computing the surface normal vectors from the point cloud is expensive, as an approximation, we use the vector pointing from in-hand point to its nearest object point as a substitute. Therefore, the divergence between hand surface normal and such a shortest distance vector can mathematically reflect how well this alignment is. This divergence term orients finger normal to the object surface and is formulated as:

$$r_2 = \frac{1}{N_\text{h}} \sum_{i=1}^{N_\text{h}} \vec{n}_i \cdot \vec{v}_i, \tag{3}$$

where $\vec{n}_i, \vec{v}_i$ denote the unit hand surface normal vectors and unit shortest distance vectors, illustrated as blue and red vectors in Fig. 3.

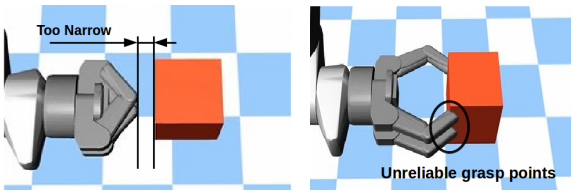
To evaluate how well the hand encloses the object, we formulated a metric to reflect the geometric closure using a convex polyhedron formed by all in-hand key points, and the term r_3 is the percentage of object surface points which are enclosed inside the polyhedron:

$$r_3 = \frac{N_\text{in}}{N_\text{obj}}. \tag{4}$$

As shown in Fig. 3, the hand polyhedron is the orange volume, and the enclosed object points are green.

A contact term is added to encourage power (enveloping) grasp, which is more stable than precision (fingertip) grasp:

$$r_4 = \text{mean}\left(\frac{\tau}{\tau_{max}} \cdot e^{-(\frac{\dot{q}}{\dot{q}_{max}})^2}\right), \tag{5}$$



(a) Potential collision. (b) Unreliable grasp pose.

Fig. 4: Proposed state initialization to train recovery policies from critical and difficult poses that lead to failure grasps.

where τ_{max} and $\dot{q}_{max}$ denote the upper limit of joint torque and velocity. This term gives more reward when the policy generates larger joint torque and $\dot{q}$ is smaller during the physical contact and grasp of the object.

A penalty on the translational velocity of the object is introduced to penalize large impact to the object:

$$r_5 = e^{-\|\boldsymbol{V}_{\text{obj}}\|} - 1.0, \tag{6}$$

preventing undesirable movements once an object is grasped.

F. Training Failure Recovery Skills

To obtain the collision-free re-grasp skills, apart from the normal training episodes, we particularly proposed two special *initial states* which are crucial prevent skewed data collection: setting the configurations of object poses and finger positions as shown in Fig. 4. They are essential in learning the failure recovery skills, because these challenging configurations are hard to be encountered via a naive random exploration. These initial states trigger difficult situations that the agent can encounter during the grasp: potential collisions between hand and object, and unreliable (shallow) grasps.

IV. Validation and Analysis

As a baseline for benchmark, we designed a control-based grasping policy: the hand keeps reaching the object with fingers open until the hand-object distance is below a threshold; once within proximity, joint torques are applied to fingers and generate the grasping motion.

To evaluate the policy's robustness and generalization ability, we used 14 unseen objects with very different sizes, shapes, and masses, as shown in Fig. 5. Considering the size of robot hand, we did not include objects that are too small or too large. We also introduced synthetic noises into object point cloud and evaluated the robustness under imperfect and noisy visual feedback. For every task, we repeated the experiment with each testing object for 10 times (140 trials in total). The statistics of the success rate is shown in Table I.

A. Evaluation Metrics

We evaluated the performances of learned policy by two metrics: the success rate in lift-and-shake tests, and the reaction time specially for failure recoveries. In the lifting test, the robot lifts up the hand with fixed orientation, and the target finger joint positions will maintain the values of the last time step before lifting; and in the shaking test, the robot first lifts the hand and then an external disturbance force of $12N$ is applied to the object's center of mass, changing the direction every 0.3 seconds. If the grasp lasts for 10 seconds, then this trial is regarded as a success. We also record the time for the policy to recover from the failure and achieve a successful re-grasp.

The results in Table I indicate that the achieved grasps are robust for lifting and can resist external disturbances. The learned control policy can react to changes rapidly and perform a new grasp in case of failures, even under challenging configurations as shown in Fig. 4.

B. Grasp Static Objects

Fig. 5 shows 14 representative grasps of unseen objects and Fig. 6 shows typical control actions of such a task. When the hand is relatively far from the object, the target finger joint positions for next time step are negative, which means the fingers are opening, enlarging the grasping area. The velocity of hand reaches the maximum at the beginning and converges to zero in the end, indicating that the agent learns to slow down when the hand approaches the object.

C. Grasp Continuously Moving Objects

The learned policy has good tracking performance and is able to follow and grasp a constantly moving object without prediction of the object trajectory, as long as the object's velocity and position is within the limit of the robot arm. As shown in Fig. 1, the hand adapts the moving direction based on the object's motion.

D. Reactive Grasp and Failure Recovery

Fig. 7a and Fig. 7b show the learned recovery motions from failure in two aforementioned challenging configurations. In Fig. 7c, the object is suddenly pushed to right side at the 2nd snapshot. The fingers re-open and the hand chases the object until executing another grasping attempt. The learned policy can recover from failures within 2 seconds, which is more efficient than repeating a cascaded planning-and-control pipeline, and it can also achieve secured grasps that pass the lift-and-shake tests with high success rates.

E. Ablation Study

The learning curves shown in Fig. 8 show the contribution of each reward term in the training. In order to show the effectiveness of each term, we formulated different combinations of the reward function, and compared the resulted learned policies. Table II shows the capabilities of different policies trained with different reward functions.

The two special initial states in Fig. 4 are critical for learning the collision-free failure recovery. Without them, the hand cannot learn the retrieving motion and the exteriors of fingers would collide with the object while re-opening.

The r_1 and r_2 terms guide the agent to approach the object. Without r_1, the hand cannot learn to reach the object. r_2 guides the contact surface conformation before and during the grasp, and without it, the agent fails to learn the grasping motion. The r_3 term rises as the fingers wrap around the object, and also compensates the penalty on object velocity caused by the random actions from policy search. After

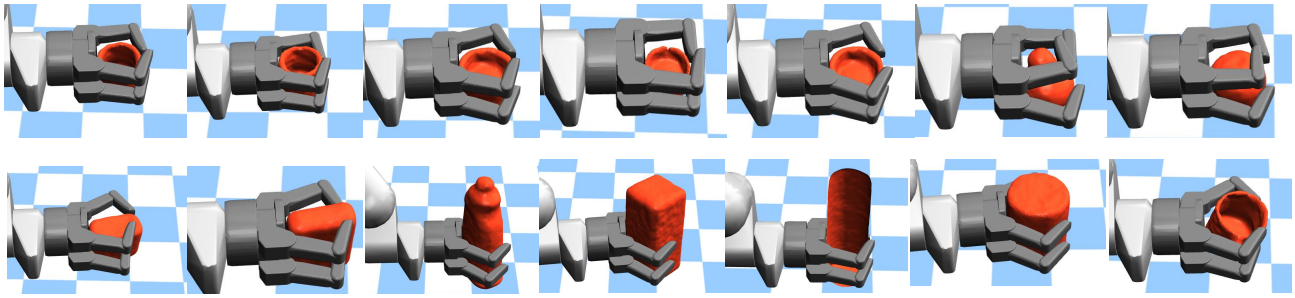


Fig. 5: Successful grasps of 14 testing objects that were unseen during training.

TABLE I: Success rate of the policy statistically evaluated in different tasks with and without sensor noises.

	Complete, Noise-free Object Point Cloud			Noisy Object Point Cloud		
	Lift [%]	Shake [%]	Recovery [s]	Lift [%]	Shake [%]	Recovery [s]
Baseline on Static Object	82	64	∅	84	61	∅
Static Object	94	85	∅	85	78	∅
Moving Object	88	79	∅	77	69	∅
Reactive Grasp	83	76	1.59	80	74	1.44
Close Fingers Recovery	92	85	0.87	84	73	0.98
Shallow Grasp Recovery	97	88	0.72	78	66	0.62

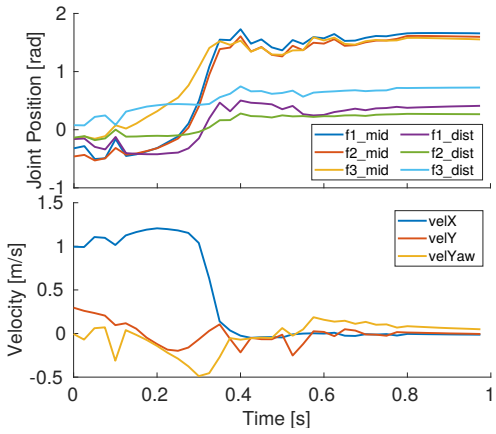


Fig. 6: Fingers' positions and torques, and velocities of the hand in a representative case – grasping a static object.

Reward combination	Reach	Grasp	Failure Recovery	Lift
No key initial states	✓	✓	✗	✓
No r_1	✗	✗	✗	✗
No r_2	✓	✗	✗	✗
No r_3	✓	✗	✗	✗
No r_4	✓	✗	✗	✗
No r_5	✓	✓	✓	✓
Complete reward set	✓	✓	✓	✓

TABLE II: Capabilities over various reward combinations.

removing r_3, the hand only learns to stay at a distance from the object with the fingers open. r_4 encourages the joints to generate enough torques and leads to a firm grasp. After removing r_4, instead of contacting and grasping the object, the hand only stays at a closer distance from the object and the fingers fail to have physical contacts. The penalty on object velocity r_5 encourages gentle grasp motions and prevents the hand from moving the object. The agent can still learn the reaching and grasping without it, but the hand will have undesirable movements after a successful grasp.

F. Realistic Visual Observation with Synthetic Noises

The proprioceptive data such as joint positions are usually of good quality due to high resolution encoders, and the largest uncertainty comes from the visual feedback. Though the majority of object point cloud can be obtained from multi-view cameras, the visual data have inevitable occlusion (bottom of the objects) and noises in depth images.

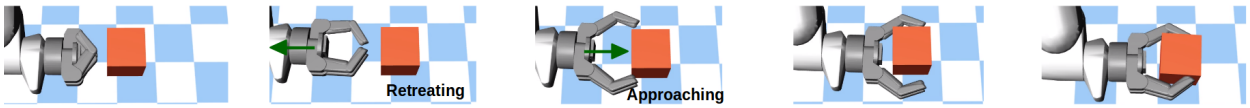
Therefore, to evaluate the robustness of the learned policy, we added synthetic sensory noises in the visual feedback during the testing of unseen objects. The noisy observation of the object point cloud is produced as

$$\mathcal{P}^{o}_{\text{noisy}} = \mathcal{P}^{o} + \|\epsilon\|\,\mathbf{u}, \tag{7}$$

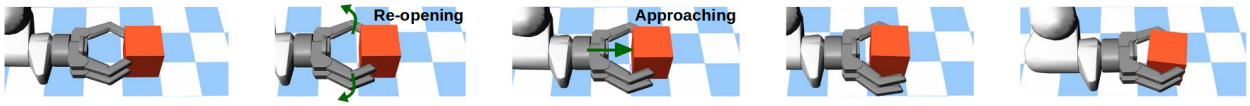
where $\mathbf{u}$ is a randomized unit vector and ϵ is sampled from a zero-mean Gaussian distribution with standard deviation $\sigma = 1.0cm$. Such variation ($\sigma = 1.0cm$) is beyond and covers the uncertainties of depth images in most common cameras, e.g., the accuracy is 0.5-1.0 cm from RealSense D435 camera at 1 m distance (more accurate at a closer distance). Fig. 9 shows the examples of downsampled point clouds of three different objects with such synthetic noises based on Eq. (7). Although the policy is trained with noise-free state feedback, we show that the learned policy is robust to noisy visual inputs which were not seen before during training.

V. Conclusion

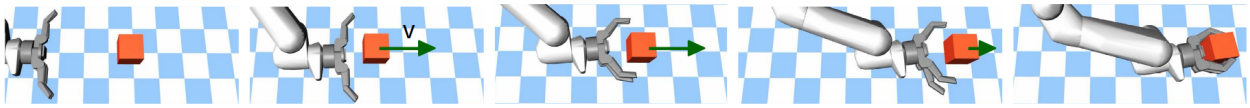
We proposed a model-free reinforcement learning scheme, which successfully learned reactive control policies, i.e. motor skills, for reaching and grasping. We formulated task-related physical quantities mathematically, designed initial state randomization of the object configurations, and incorporated two challenging initial states particularly to induce



(a) Coordination and recovery from potential collisions.



(b) Coordinated collision-free re-grasping from an unreliable grasp.



(c) Grasping of a dynamically moving object.

Fig. 7: Grasping motions of the learned policy: (a) Recovering from the initial configuration where the object is too close to the fingers; (b) Re-grasping from the unreliable grasp; (c) Chasing and grasping of a continuously moving object.

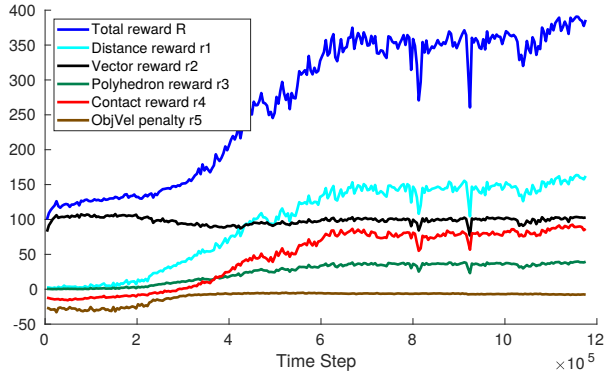


Fig. 8: Learning curves of the total reward and each term.

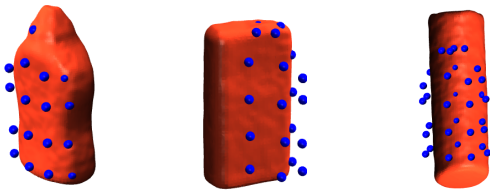


Fig. 9: Downsampled sparse point-clouds with synthetic noises, simulating offset and noises present in real data.

failure grasps and train recovery skills. Being guided by our proposed reward, the robot can explore and optimize the policy effectively.

The validations using unseen objects showed that the robot can reach and grasp both static and moving objects, generate collision-free recovery motions after a failure, and reactively re-grasp *within 2 seconds* – showing advantages of the online closed-loop control with synchronized hand-finger motions. The ablation studies and benchmarking revealed the influence of each reward term, and showed our proposed initial states indeed improved the robustness of the learned policy.

VI. Acknowledgement

This work has been supported by EU H2020 project Harmony (101017008) and EPSRC's Future AI and Robotics for Space (EP/R026092/1).

References

[1] Y. Suzuki, *et al.*, "Grasping strategy for moving object using net-structure proximity sensor and vision sensor," in *2015 IEEE International Conference on Robotics and Automation*, 2015.

[2] K. Hang, *et al.*, "A framework for optimal grasp contact planning," *IEEE Robotics and Automation Letters*, 2017.

[3] C. Yang, *et al.*, "Multi-expert learning of adaptive legged locomotion," *Science Robotics*, vol. 5, no. 49, 2020.

[4] D. Kappler, *et al.*, "Leveraging big data for grasp planning," in *2015 IEEE International Conference on Robotics and Automation*, 2015.

[5] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours," *CoRR*, 2015.

[6] H. Merzic, *et al.*, "Leveraging contact forces for learning to grasp," in *2019 IEEE International Conference on Robotics and Automation*, 2019.

[7] OpenAI, *et al.*, "Learning dexterous in-hand manipulation," *CoRR*, 2018.

[8] U. Castiello, *et al.*, "Reach to grasp: the response to a simultaneous perturbation of object position and size," *Experimental Brain Research 1998 120:1*, vol. 120, no. 1, pp. 31–40, 1998.

[9] A. Shukla and A. Billard, "Coupled dynamical system based arm–hand grasping model for learning fast adaptation strategies," *Robotics and Autonomous Systems*, vol. 60, no. 3, pp. 424–440, mar 2012.

[10] S. Kim, *et al.*, "Catching objects in flight," *IEEE Transactions on Robotics*, vol. 30, no. 5, pp. 1049–1065, oct 2014.

[11] O. Kroemer, *et al.*, "Combining active learning and reactive control for robot grasping," *Robotics and Autonomous Systems*, 2010.

[12] T. Lampe and M. Riedmiller, "Acquiring visual servoing reaching and grasping skills using neural reinforcement learning," in *The 2013 International Joint Conference on Neural Networks (IJCNN)*, 2013.

[13] N. Marturi, *et al.*, "Dynamic grasp and trajectory planning for moving objects," *Autonomous Robots*, 2019.

[14] I. Akinola, *et al.*, "Dynamic Grasping with Reachability and Motion Awareness," mar 2021.

[15] J. Mahler, *et al.*, "Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics," *CoRR*, 2017.

[16] D. Morrison, *et al.*, "Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach," *CoRR*, 2018.

[17] J. Schulman, *et al.*, "Proximal policy optimization algorithms," *CoRR*, 2017.

[18] E. Todorov, *et al.*, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012.

[19] B. Çalli, *et al.*, "Benchmarking in manipulation research: The YCB object and model set and benchmarking protocols," *CoRR*, vol. abs/1502.03143, 2015.