The Role of Tactile Sensing in Learning and Deploying Grasp Refinement Algorithms

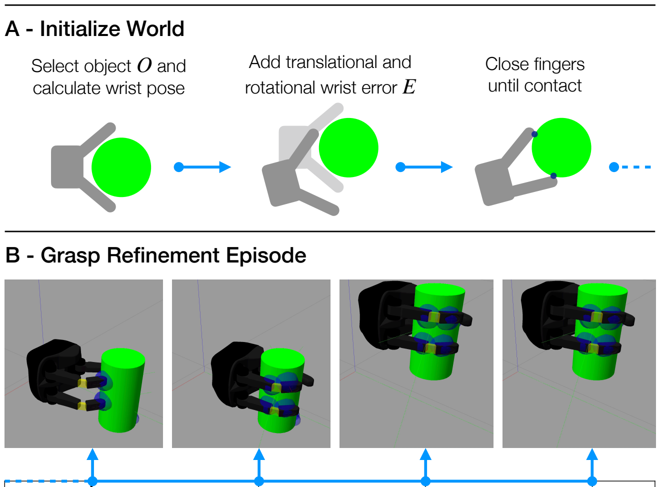
Alexander Koenig[1,2], Zixi Liu[2], Lucas Janson[3] and Robert Howe[2,4]

***Abstract*— A long-standing question in robot hand design is how accurate tactile sensing must be. This paper uses simulated tactile signals and the reinforcement learning (RL) framework to study the sensing needs in grasping systems. Our first experiment investigates the need for rich tactile sensing in the rewards of RL-based grasp refinement algorithms for multi-fingered robotic hands. We systematically integrate different levels of tactile data into the rewards using analytic grasp stability metrics. We find that combining information on contact positions, normals, and forces in the reward yields the highest average success rates of 95.4% for cuboids, 93.1% for cylinders, and 62.3% for spheres across wrist position errors between 0 and 7 centimeters and rotational errors between 0 and 14 degrees. This contact-based reward outperforms a non-tactile binary-reward baseline by 42.9%. Our follow-up experiment shows that when training with tactile-enabled rewards, the use of tactile information in the control policy's state vector is drastically reducible at only a slight performance decrease of at most 6.6% for no tactile sensing in the state. Since policies do not require access to the reward signal at test time, our work implies that models trained on tactile-enabled hands are deployable to robotic hands with a smaller sensor suite, potentially reducing cost dramatically.**

I. Introduction

Tactile sensing provides essential information about local object geometry, surface properties, contact forces, and grasp stability [1]. Hence, tactile sensors can be a valuable tool in robotic grasp refinement tasks [2] where a grasping system recovers from calibration errors. Computer vision approaches for grasp refinement often face limitations due to the occlusion of contact events. Tactile sensors are often expensive and fragile hardware components. Hence, for cost-effective robotic hand design, it is essential to understand when robot hands need precise sensing and how accurate it should be to achieve good grasping performance.

A few research papers investigated the effect of tactile sensor resolution on grasp success. Wan et al. [3] found that reduced spatial resolution of tactile sensors negatively impacts grasp success since inaccuracies in contact position and normal sensing can influence grasp stability predictions. Other works analyzed the effect of contact sensor resolution on grasp performance in the context of reinforcement learning. In simulated experiments, Merzić et al. [4] found that contact feedback in a policy's state vector improves the performance of RL-based grasping controllers, and [5], [6] presented similar results for in-hand manipulation. However, [5], [6] also concluded that models trained with binary contact signals perform equally well as models that receive accurate normal force information. Furthermore, [5], [6] found that tactile resolution (92 vs. 16 sensors) has no noticeable effect on performance and sample efficiency of reinforcement learned manipulation controllers.


A - Initialize World
Select object O and calculate wrist pose
Add translational and rotational wrist error E
Close fingers until contact
B - Grasp Refinement Episode

Stage	Refine	Lift	Hold	End
Steps	15	6	6	-
Duration	5 s	2 s	2 s	-
ϵ *and* δ	$\epsilon_f + \alpha_1\epsilon_\tau + \alpha_2\delta_{task}$	$\epsilon_f + \alpha_1\epsilon_\tau + \alpha_2\delta_{cur}$	$\epsilon_f + \alpha_1\epsilon_\tau + \alpha_2\delta_{cur}$	0
δ	δ_{task}	δ_{cur}	δ_{cur}	0
ϵ	$\epsilon_f + \alpha_1\epsilon_\tau$	$\epsilon_f + \alpha_1\epsilon_\tau$	$\epsilon_f + \alpha_1\epsilon_\tau$	0
β	0	0	0	$\{0,1\}$

Fig. 1: Overview of one algorithm episode. (A) Initialization of hand and object. (B) We split the grasp refinement algorithm into four stages and compare four reward frameworks: (1) ϵ *and* δ, (2) only δ, (3) only ϵ and (4) the non-tactile binary reward baseline β. The weighting factors of $\alpha_1 = 5$ and $\alpha_2 = 0.5$ were empirically determined.

In this paper, we use accurate tactile signals from simulation and the reinforcement learning framework to explore the tactile sensing needs in robotic systems. RL algorithms aim to produce a policy $\pi(\boldsymbol{a}|\boldsymbol{s})$ that outputs actions $\boldsymbol{a}$ given state information $\boldsymbol{s}$ such that the cumulative reward signal r is maximized. The reward function is a critical part of every RL algorithm [10]. While the previous work in [4], [5], [6] only studied the tactile resolution in the policy's state, our first contribution investigates the impact of tactile information in the reward signal. Table I shows an overview of the reward functions used in other RL-based

This material is based upon work supported by the US National Science Foundation under Grant No. IIS-1924984 and by the German Academic Exchange Service.

[1] Department of Informatics, Technical University of Munich
[2] School of Engineering and Applied Sciences, Harvard University
[3] Department of Statistics, Harvard University
[4] RightHand Robotics, Inc., 237 Washington St, Somerville, MA 02143 USA. Robert Howe is corresponding author `howe@seas.harvard.edu`.

TABLE I: Reward functions of RL grasping controllers.

Paper	Reward
Chebotar 2016 [7]	Maximize predicted grasp success from learned stability predictor
Merzić 2019 [4]	Maximize (1) number of links in contact and (2) binary drop test reward Minimize (1) distance object to gripper, (2) distance fingertips to object, (3) joint torques and (4) object velocity
Wu 2019 [8]	Maximize binary pick-up reward at episode end Minimize finger reopening
Hu 2020 [9]	Maximize (1) number of contact points and (2) number of object key-points contained in convex hull of hand and finger key-points Minimize (1) distance from hand key-points to object key-points, (2) angle between hand key-point normals and vectors pointing from hand key-points to object center, (3) number of contacts on outside of fingers and (4) object linear velocity

grasping controllers that process tactile information from multi-fingered hands. These reward functions are insufficient to study the effect of different types of contact information, because they either directly encode the experiment outcome [4], [8] or consist of manually engineered cues (e.g., number of contacts [4], [9]) that do not include contact position, normal, and force information. Hence, we propose a unified framework to systematically incorporate different levels of tactile information from robotic hands into a reward signal via analytic grasp stability metrics. As shown in Fig. 1, we conduct grasp refinement experiments and define three types of rewards: ϵ calculated from contact positions and normals, a force-based reward δ, and a binary task execution reward β. By comparing the individual and combined performance of ϵ and δ, we estimate the relevance of contact position, normal, and force sensing for the reward signal.

Calculating grasp stability metrics requires costly tactile sensing capabilities on physical grippers. However, the reward signal is only required during the training of policies but not while testing, which suggests that sensing needs in both stages could be different. We hypothesize in Figure 2 that policies trained with grasp stability metrics on a robotic hand H_f with a *full* tactile sensor suite are deployable to structurally similar but more affordable hands H_r with *reduced* tactile sensing at a small performance decrease. Hence, our second experiment gradually decreases tactile resolution in the state vector to find realistic training and deployment workflows for grasping algorithms.

Our paper reviews and defines grasp quality metrics in section II. We use these metrics as reward signals in section III and thereby study the impact of contact position, normal, and force sensing in the reward. In section IV, we investigate the importance of tactile data in the state vector.

II. Grasp Stability Metrics

A. Largest-minimum resisted forces and torques

Ferrari and Canny [11] define grasp quality as the largest-minimum perturbing wrench that the grasp can resist given the grasp's force constraints. Ferrari's metric [11] suffers from the non-comparability of forces (in N) and torques (in Nm). Hence, Mirtich and Canny [12] refine this popular

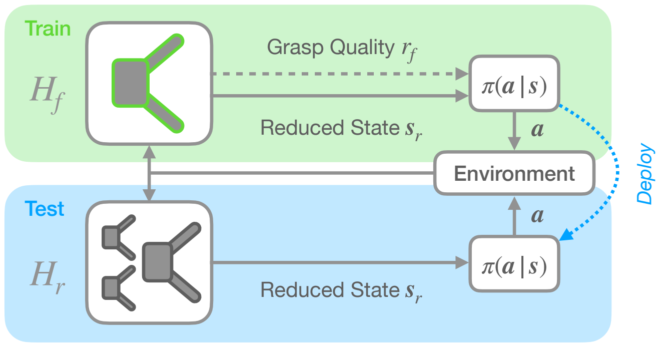


Fig. 2: The hypothesized workflow for training and deploying RL-controlled grasping systems. First, train a policy $\pi(\boldsymbol{a}|\boldsymbol{s})$ on a hand H_f with a *full* tactile sensor suite (e.g., contact position, normal and force sensors) where the grasp quality metrics are available as a reward r_f to learn a task, but only provide a subset of the available contact data in the state vector $\boldsymbol{s}_r$. Afterwards, deploy the policy to many structurally similar hands H_r with a *reduced* sensor set to save cost.

metric by decoupling the wrench space into a force and torque component, and thereby evaluate how well a grasp resists pure forces and torques.

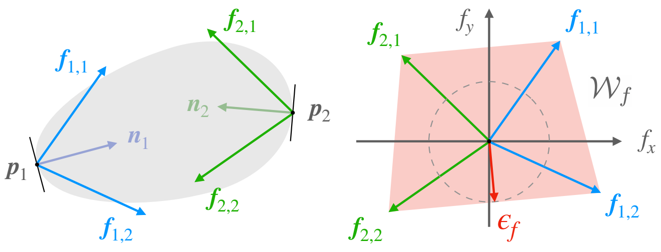


Fig. 3: Left: a grasp on a grey object with two contact points $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$, contact normals $\boldsymbol{n}_i$ and friction cones. Right: the quality metric ϵ_f is the radius of the largest ball contained in the convex hull $\mathcal{W}_f$ over the set of resisted forces.

Let us examine how to measure resistance to disturbing forces. The contact force $\boldsymbol{f}_i$ at each contact i is constrained via the friction cone $f_{i,t} \leq \mu f_{i,n}$, where μ is the coefficient of friction and $f_{i,t}$ and $f_{i,n}$ are the tangential and normal components of $\boldsymbol{f}_i$, respectively. The friction cone is discretized using m edges $\boldsymbol{f}_{i,j}$. The set of forces $\mathcal{W}_f$ that the contacts can apply to the object is $\mathcal{W}_f = \text{ConvexHull}\left(\bigcup_{i=1}^{n}\left\{\boldsymbol{f}_{i,1}, \ldots, \boldsymbol{f}_{i,m}\right\}\right)$, where n is the number of contacts. Finally, the quality metric ϵ_f in equation (1) is the shortest distance from the origin to the nearest hyperplane of $\mathcal{W}_f$. Hence, the metric defines a lower bound on the resisted force in all directions. As shown in Fig. 3, ϵ_f can be geometrically interpreted as the radius of the largest ball centered at the origin and contained inside $\mathcal{W}_f$.

$$\epsilon_f = \min_{\boldsymbol{f} \in \partial \mathcal{W}_f} \|\boldsymbol{f}\| \tag{1}$$

This concept is easily extended to the torque domain. The reaction torque $\boldsymbol{\tau}_{i,j}$ resulting from a friction cone edge $\boldsymbol{f}_{i,j}$ is calculated by $\boldsymbol{\tau}_{i,j} = \boldsymbol{r_i} \times \boldsymbol{f_{i,j}}$, where $\boldsymbol{r_i}$ is a vector pointing from the object's center of mass to the contact point $\boldsymbol{p_i}$. The set of torques $\mathcal{W}_\tau$ that the grasp can resist is defined by $\mathcal{W}_\tau = \text{ConvexHull}\left(\bigcup_{i=1}^{n}\{\boldsymbol{\tau}_{i,1},\ldots,\boldsymbol{\tau}_{i,m}\}\right)$. The metric ϵ_τ in equation (2) evaluates the grasp's quality by identifying the magnitude of the largest-minimum resisted torque.

$$\epsilon_\tau = \min_{\boldsymbol{\tau}\in\partial\mathcal{W}_\tau} \|\boldsymbol{\tau}\| \tag{2}$$

B. Minimum distance to the friction cone

The quality metrics ϵ_f and ϵ_τ analyze the forces that each contact can theoretically exert on the object. However, these metrics do not consider the actual contact forces that the contacts apply to the object. To this end, we define two force-based quality metrics δ_{cur} and δ_{task}. While δ_{cur} is a general-purpose grasp quality metric, δ_{task} is applicable when a task definition exists.

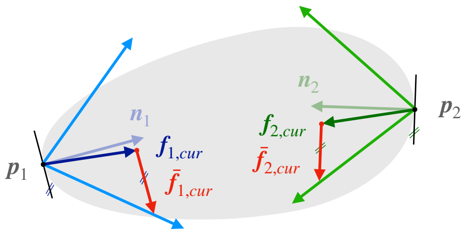


Fig. 4: Grasp with current contact forces $\boldsymbol{f}_{i,cur}$ and tangential force margins $\bar{\boldsymbol{f}}_{i,cur}$ to the friction cones.

Similar to Buss et al. [13], we measure grasp stability in terms of how far the contact forces are from the friction limits. Fig. 4 shows a grasp with the current contact forces $\boldsymbol{f}_{i,cur}$ and the tangential force margins $\bar{\boldsymbol{f}}_{i,cur}$. The vectors $\bar{\boldsymbol{f}}_{i,cur}$ are forces in the tangential direction that point from $\boldsymbol{f}_{i,cur}$ to the closest point on the friction cone, thereby identifying the direction in which the contact can take the least tangential force before slipping. A grasp with large tangential force margins $\bar{\boldsymbol{f}}_{i,cur}$ is desirable since the contacts are less prone to sliding when an object wrench is applied. Hence, the metric δ_{cur} in equation (3) measures the average magnitude of the safety margins $\|\bar{\boldsymbol{f}}_{i,cur}\|$. Contacts with larger forces contribute more to grasp stability because they can have larger tangential force margins $\bar{\boldsymbol{f}}_{i,cur}$ and can thereby compensate for more disturbing object wrenches. Therefore, we weigh the safety margins $\|\bar{\boldsymbol{f}}_{i,cur}\|$ by their respective contact force magnitudes $\|\boldsymbol{f}_{i,cur}\|$.

$$\delta_{cur} = \frac{\sum_{i=1}^{n_c}\|\boldsymbol{f}_{i,cur}\|\|\bar{\boldsymbol{f}}_{i,cur}\|}{\sum_{i=1}^{n_c}\|\boldsymbol{f}_{i,cur}\|} \tag{3}$$

In many grasping tasks, a clear task definition exists. Let $T = \{\boldsymbol{w_1}, \boldsymbol{w_2}, \ldots, \boldsymbol{w_m}\}$ be the set of task wrenches that the grasp must resist during task execution (e.g., object weight or wrenches from expected collisions). Several task-oriented quality metrics measure how well a convex set of T is contained within the convex set of all wrenches that the grasp can resist [14], [15], [16]. However, since these approaches reason about the theoretically applicable contact forces, which are commonly bounded to unity [17], [11], it is not possible to evaluate whether the *current* contact forces of a grasp are suitable to balance the anticipated task wrenches.

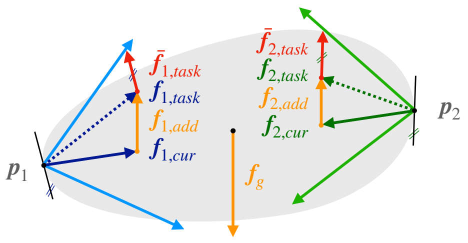


Fig. 5: Grasp with predicted task contact forces $\boldsymbol{f}_{i,task}$ after mapping the task force $-\boldsymbol{f}_g$ onto the contacts.

We define an alternative task-oriented metric δ_{task}. We calculate the additional contact force $\boldsymbol{f}_{i,add}$ that each contact i must react with to compensate the task wrench $\boldsymbol{w} \in T$ with $\mathbf{G}^+\boldsymbol{w} = \begin{pmatrix}\boldsymbol{f}_{1,add}^T & \boldsymbol{f}_{2,add}^T & \ldots & \boldsymbol{f}_{n_c,add}^T\end{pmatrix}^T$, where $\mathbf{G}^+$ is the pseudoinverse of the grasp matrix as defined in [18]. Fig. 5 shows that the task contact force $\boldsymbol{f}_{i,task} = \boldsymbol{f}_{i,cur} + \boldsymbol{f}_{i,add}$ is the sum of the current contact force $\boldsymbol{f}_{i,cur}$ and $\boldsymbol{f}_{i,add}$ which results from a task wrench (here the object weight $-\boldsymbol{f}_g$). We use a hard contact model and assume that the internal grasp forces stay the same after applying $\boldsymbol{f}_{i,add}$. The metric δ_{task} in equation (4) measures the expected grasp stability during task execution by computing the average magnitude of the tangential force margins $\|\bar{\boldsymbol{f}}_{i,task}\|$ of the task contact forces $\boldsymbol{f}_{i,task}$. The metric δ_{task} is a lower bound over all task wrenches $\boldsymbol{w} \in T$ and we thereby identify the worst-case task wrench.

$$\delta_{task} = \min_{\boldsymbol{w}\in T} \frac{\sum_{i=1}^{n_c}\|\boldsymbol{f}_{i,task}\|\|\bar{\boldsymbol{f}}_{i,task}\|}{\sum_{i=1}^{n_c}\|\boldsymbol{f}_{i,task}\|} \tag{4}$$

III. Tactile Sensing and the Reward Function

A. Simulation Environment

We simulate the grasp refinement episodes of the three-fingered ReFlex TakkTile hand (RightHand Robotics, Somerville, MA USA) using a custom robotics simulator based on the Gazebo [19] simulation environment, the DART [20] physics engine, and the ROS [21] communication framework. We model the under-actuated distal flexure as a rigid link with two revolute joints (one between the proximal and one between the distal finger link). Further, we approximate the finger geometries as cuboids to reduce computational load. We activate simulated gravity in our experiments (unlike in [4]), and the object can freely interact with the hand and the world. Our source code is publicly available under `github.com/axkoenig/grasp_refinement`.

B. Train and Test Dataset

Each training sample consists of a tuple (O, E), where O is the object, and E is the wrist pose error sampled uniformly before every episode. There are three object types (cuboid, cylinder, and sphere) with a mass $\in [0.1, 0.4]$ kg and randomly sampled sizes. Fig. 6 visualizes the minimum and maximum object dimensions. The wrist pose error E consists of a translational and a rotational error. We uniformly sample the translational error (e_x, e_y, e_z) from $[-5, 5]$ cm and the rotational error (e_ξ, e_η, e_ζ) from $[-10, 10]$ deg for each variable, respectively.

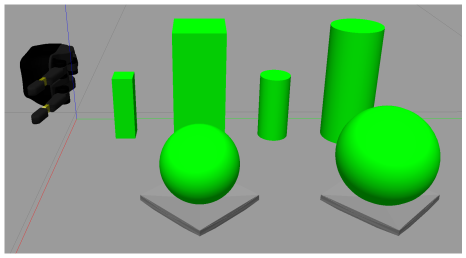


Fig. 6: Minimum and maximum object sizes. Sizes in mm: cuboid height $\in [130, 230]$, length and width $\in [40, 100]$, cylinder height $\in [130, 230]$, radius $\in [30, 50]$, sphere radius $\in [65, 80]$. We place the spheres on a concave mount to prevent rolling.

We define 8 different wrist error cases for the test dataset. Let $d(a, b, c) = \sqrt{a^2 + b^2 + c^2}$ be the L2 norm of the variables (a, b, c). Table II shows the wrist error cases, where case A corresponds to no error and case H means maximum wrist error. Fig. 7 visualizes two wrist error cases. The test dataset consists of 30 random objects O (10 cuboids, 10 cylinders, and 10 spheres). Per object O, we randomly generate the eight wrist error cases $\{A, B, \ldots, H\}$ from Table II. Hence, we run $30 \times 8 = 240$ experiments to test one model.

TABLE II: Wrist error cases

Wrist Error Case	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**
$d(e_x, e_y, e_z)$ in cm	0	1	2	3	4	5	6	7
$d(e_\xi, e_\eta, e_\varsigma)$ in deg	0	2	4	6	8	10	12	14

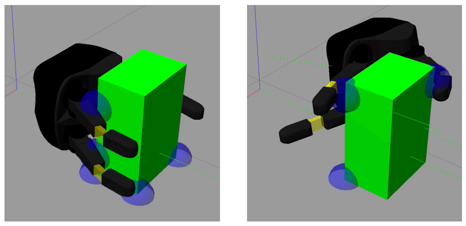


Fig. 7: Left: wrist error case **A** (no wrist error), Right: wrist error case **H** (maximum wrist error) after closing the fingers. Contact points in blue.

C. State and Action Space

The state vector $\boldsymbol{s}$ consists of 7 joint positions (1 finger separation, 3 proximal bending, 3 distal bending degrees of freedom), and 7 contact cues (3 on proximal links, 3 on distal links, and 1 on palm) that include contact position, contact normal and contact force, which have 3 (x, y, z) components each. The dimension of the state vector is $\boldsymbol{s} \in \mathbb{R}^{7+7\times(3\times3)=70}$. Note that we do not assume any information about the object (e.g., object pose, geometry, or mass) in the state vector. The action vector $\boldsymbol{a}$ consists of 3 finger position increments, 3 wrist position increments and 3 wrist rotation increments. The action vector's dimension is $\boldsymbol{a} \in \mathbb{R}^{3+3+3=9}$. The policy $\pi_{\boldsymbol{\theta}}$ is parametrized by a neural network with weights $\boldsymbol{\theta}$. The network is a multi-layer perceptron (MLP) with four layers [70, 256, 256, 9]. We use the `stable-baselines3` [22] implementation of the soft actor-critic (SAC) [23] framework to train the stochastic policy $\pi_{\boldsymbol{\theta}}$. We evaluate the policy deterministically when testing.

D. Algorithm Overview

Fig. 1 shows an overview of one training episode. Before starting the control algorithm, we reset the world. Thereby, we randomly generate a new object, wrist error tuple (O, E) (or we select one from the test dataset). We assume a computer vision system and a grasp planner that produces a side-ways facing grasp at a fixed 5 cm offset from the object's center of mass. We add the wrist pose error E to this grasp pose to simulate calibration errors and close the fingers of the robotic hand in the erroneous wrist pose until the fingers make contact with the object. Consequently, the grasp refinement episode starts. We divide each episode into three stages, as displayed in Fig. 1. Firstly, the policy $\pi_{\boldsymbol{\theta}}$ *refines* the grasp in five seconds and 15 algorithm steps. Afterward, the agent *lifts* the object by 15 cm via hard-coded increments to the wrist's z-position in two seconds and six algorithm steps. Finally, the policy *holds* the object in place for two seconds and six algorithm steps to test the grasp's stability. The policy $\pi_{\boldsymbol{\theta}}$ can update the wrist and finger positions while lifting and holding. The control frequency of the policy in all stages is 3 Hz, while the update frequency of the low-level proportional–derivative (PD) controllers in the wrist and the fingers is 100 Hz.

Each episode can last at most $15 + 6 + 6 = 27$ algorithm steps. We end the episode earlier if the hand shifts the object by more than 10 cm during the refinement stage to discourage excessive movement of the object. Furthermore, we terminate refinement if one of the fingers exceeds a joint limit of 3 radians. We do not enter the holding stage if the object dropped after the lifting stage. The algorithm trains for 25000 steps, which corresponds to approximately 1000 training episodes depending on the episode lengths.

As shown in the table of Fig. 1, we use the analytic grasp stability metrics from section II as reward functions. We compare the following reward configurations: (1) both ϵ *and* δ, (2) only ϵ, (3) only δ and (4) the baseline β. Fig. 1 shows that δ refers to δ_{task} in the *refine* stage to measure expected grasp stability before lifting and δ_{cur} in the *lift* and *hold* stages to measure current stability. Further, ϵ is a weighted combination of ϵ_f and ϵ_τ. While ϵ *and* δ, δ, and ϵ provide stability feedback after every algorithm step, the baseline β gives a sparse reward after the holding stage, indicating if the object is still in the hand (1) or not (0). Since the SAC algorithm is sensitive to reward scaling [23], we normalize the rewards, which are based on grasp quality metrics.

E. Results

Fig. 8 shows the training results of the four reward frameworks. For all experiments in this paper, we average over 40 models trained with different seeds for each framework and smooth the training curves with a moving average filter of kernel size 30. The error bars in all plots represent ± 2

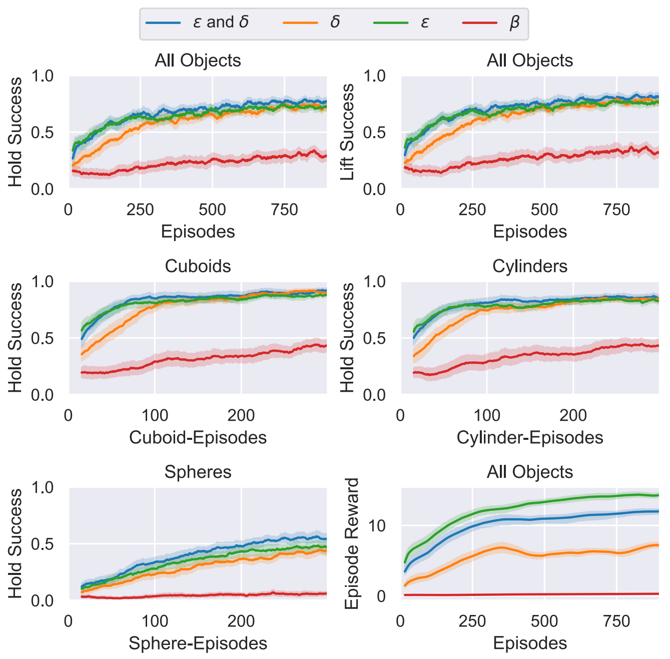


Fig. 8: Training results for reward frameworks.

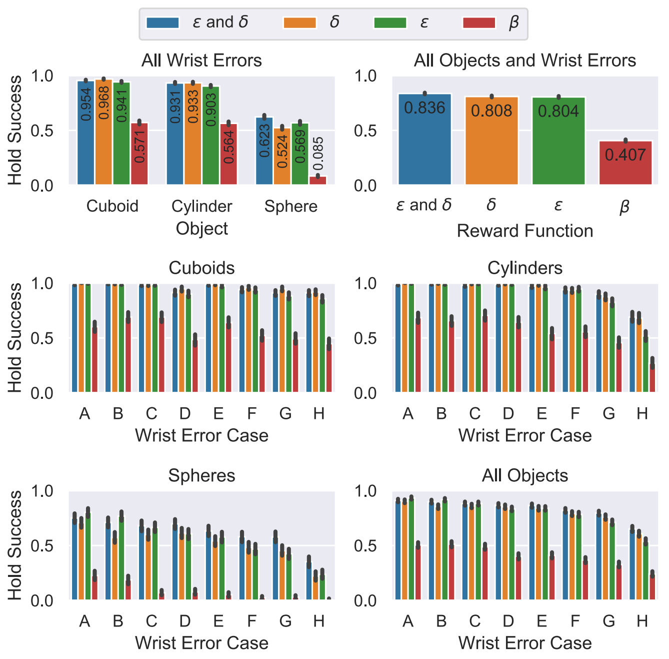


Fig. 9: Test results for reward frameworks.

standard errors. It takes approximately 20 hours to train one model on a machine with 4 CPUs. We realize from Fig. 8 that the algorithms trained with grasp stability metrics are more sample efficient and reach higher success rates than β within the defined training steps. We also notice that the combination between ϵ *and* δ is particularly helpful for spheres. The algorithms trained with β especially struggle to grasp spheres. Furthermore, the reward framework ϵ initially trains faster than the reward frameworks that include the force agnostic metric δ. Lastly, we recognize that the *Hold Success* and *Lift Success* graphs in Fig. 8 are very similar.

Fig. 9 summarizes the test results. All test results in this paper stem from 38400 grasps (40 models with different seeds $\times$ 4 frameworks $\times$ 240 test cases). Our main observation is that combining the geometric grasp stability metric ϵ with the force-agnostic metric δ yields the highest average success rates of 83.6% across all objects (95.4% for cuboids, 93.1% for cylinders, and 62.3% for spheres) over all wrist errors. The ϵ *and* δ framework outperforms the binary reward framework β by 42.9%. As expected, performance decreases for larger wrist errors. We show results of a one-sided, paired t-test in Table III (mean of framework x is μ_x and '≈ 0.0' means that value was numerically zero).

TABLE III: Results of t-test for reward comparison.

Result	$\mu_{\epsilon \text{ and } \delta} > \mu_\delta$	$\mu_{\epsilon \text{ and } \delta} > \mu_\epsilon$	$\mu_{\epsilon \text{ and } \delta} > \mu_\beta$
p-value	$3.1681\ 10^{-10}$	$2.0510\ 10^{-12}$	≈ 0.0

F. Discussion

This study investigates the tactile sensing needs in the reward of RL grasping controllers by incorporating highly accurate contact information via analytic grasp stability metrics. From the results of the t-test in Table III, we conclude that the claim 'the combination of ϵ *and* δ outperforms all other tested rewards frameworks' is statistically significant ($p < 0.01$ for all comparisons). The results demonstrate that information about contact positions and normals encoded in ϵ combines well with the force-based information in the δ reward. This result motivates building physical robotic hands capable of sensing these types of information. The low success rates for the spheres may be because they can roll and are therefore harder to grasp (cuboids and cylinders move comparatively less when touched by fingers or the palm). The observation that success rates after the *lift* and the *hold* stage are almost identical means that once the hand successfully lifts the object, the grasp is usually also stable enough to keep the object in hand until the very end of the grasp refinement episode.

The β framework performs worst after the defined number of training steps, which is unsurprising because shaped rewards are known to be more sample efficient than sparse rewards [24]. The β framework may not constitute the best-performing alternative that is not based on analytic techniques from grasp analysis. However, it should be considered as a non-tactile reward baseline often used in related works [4], [8]. Furthermore, the performance of the β framework in Fig. 8 continues to rise slowly, and it would be interesting to evaluate at which success rates it plateaus.

IV. Tactile Sensing and the State Vector

A. Experimental Setup

In a second experiment, we investigate the effect of contact sensing resolution in the state vector on grasp refinement. We compare four contact sensing frameworks. The *full* contact sensing framework receives the same state vector $\boldsymbol{s} \in \mathbb{R}^{70}$ as in section III-C. In the *normal* framework, we only provide the algorithm with the contact normal forces and omit the tangential forces ($\boldsymbol{s} \in \mathbb{R}^{56}$). In the *binary* framework we only

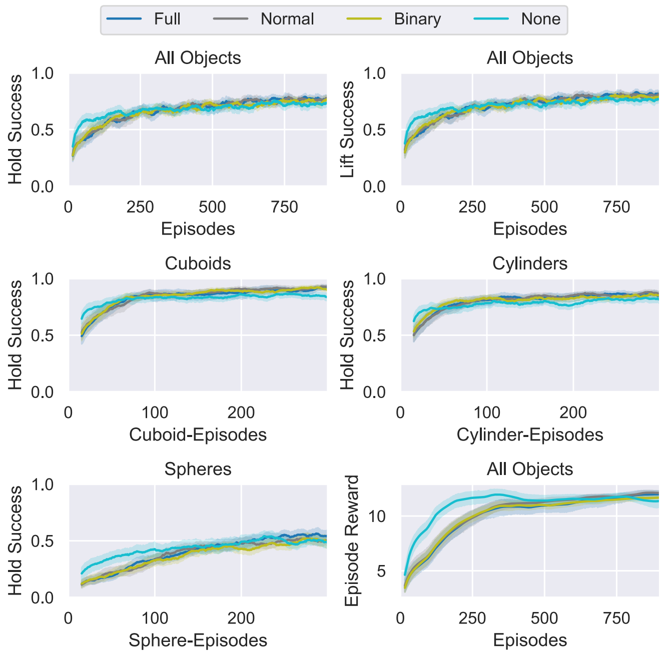


Fig. 10: Training results for contact sensing frameworks.

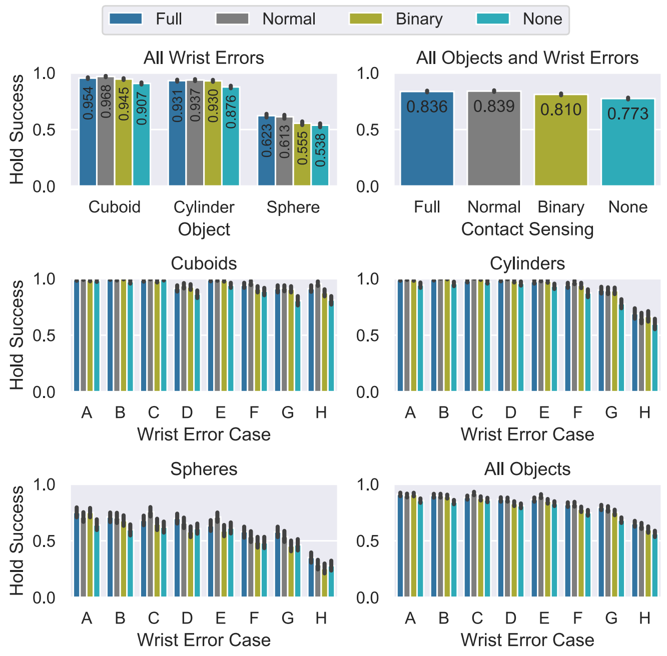


Fig. 11: Test results for contact sensing frameworks.

give a binary signal whether a link is in contact (1) or not (0) ($\boldsymbol{s} \in \mathbb{R}^{56}$). Finally, we solely provide the joint positions in the *none* framework ($\boldsymbol{s} \in \mathbb{R}^{7}$). We adjust the size of the input layer of the neural network from section III-C to match the size of the state vector of each framework. We keep the rest of the network's architecture fixed to allow a fair comparison. The reward function in these experiments is ϵ *and* δ from Fig. 1. Hence, all contact sensing frameworks receive contact information indirectly via the reward.

B. Results

Fig. 10 shows the training performance of the contact sensing frameworks. Note that the *full* framework is the same as the ϵ *and* δ framework from section III. We can observe that the *none* framework initially learns faster than the other frameworks. However, after approximately 250 episodes, the frameworks that receive contact feedback outperform the *none* framework, which plateaus at a lower success rate.

Fig. 11 compares the test results of the different contact sensing frameworks. We observe that the frameworks which receive contact feedback (*full*, *normal*, *binary*) outperform the *none* framework by 6.3%, 6.6% and 3.7%, respectively. Providing the algorithm with *normal* force information yields a performance increase of 2.9% compared to the *binary* contact sensing framework. However, training with the *full* contact force vector only increases the performance by 2.6% compared to the *binary* framework. Furthermore, the success rates for cuboids and cylinders are higher than for spheres (for the *normal* force framework the success rates are 96.8%, 93.7%, 61.3%, respectively). We show the results of a one-sided, paired t-test in Table IV.

TABLE IV: Results of t-test for contact sensing comparison.

Result	$\mu_{normal} > \mu_{full}$	$\mu_{normal} > \mu_{binary}$	$\mu_{normal} > \mu_{none}$
p-value	0.2232	$7.0177\ 10^{-11}$	$1.3087\ 10^{-46}$

C. Discussion

This experiment studies how contact sensing resolution in the policy's state vector is related to grasp success when training with fully contact informed rewards. Thereby, we investigate the viability of our hypothesized training and deployment workflow in Figure 2. The training curves of the *full*, *normal* and *binary* frameworks in Fig. 10 are hard to distinguish, which indicates a similar training performance in all cases. Each data point in the training curves includes the outcome of only one grasp refinement episode per model (one object O and one wrist error W). This punctual evaluation poorly reflects on the *overall* model performance. Therefore, we should focus our analysis on the test results from the 240 experiments per model over multiple objects and wrist errors which provide a more comprehensive model evaluation. In the test results, we observe statistically significant improvements for the *normal* force framework when compared to the *binary* and *none* frameworks (p-values in Table IV < 0.01). However, these improvements are small, and the results suggest that an affordable *binary* contact sensor suite may be suitable if a small decrease in performance is tolerable. The surprisingly good performance of the *none* framework means that agents can refine grasps solely based on the crude contact feedback of finger joint position data when trained with rewards that encode grasp stability. These results support our hypothesis that RL grasping algorithms are deployable to hands with reduced contact sensor resolution at little performance decrease when incorporating rich tactile feedback at train time. These results also have implications for systems trained in simulation that usually suffer from a sim-to-real gap. The gap may be reduced by choosing only a feature set in the state vector that is cheaply and accurately available on the real hand (e.g., joint positions) and integrating harder-to-obtain information in the reward, which is easily computable in simulation.

Interestingly, the algorithms trained with the *full* force vector perform approximately on par with the ones that receive the *normal* force information (the small difference in success rates of 0.3% is not statistically significant because p-value > 0.01 in Table IV). This observation is counterintuitive since tangential forces are prominent in the designed grasping task. This result could be due to three reasons. (1) The *full* force framework has the most network parameters and requires even longer training times. (2) The model fails to represent the concept of the friction cone internally. An alternative representation of the tangential forces could be a solution (e.g., providing a margin to the friction cone instead of a tangential force vector). (3) Simulated contact forces are prone to numerical instability [25], especially when simulating robotic grasping [26]. Hence, since simulated contact forces are not always physically meaningful, they may not constitute a good proxy of grasp success in simulation.

We relate the differences in learning speed to the size of the state vector. The *none* framework has a smaller state vector and can hence learn faster, while the frameworks that process contact information require more training data to converge. The relative performance decrease when reducing contact sensor resolution is approximately the same across all objects, even though they have different geometries and grasping strategies. This result suggests that our conclusions are representative of a variety of object geometries.

V. Conclusion

This paper investigated the importance of tactile signals in the reward and the policy's state vector to identify the tactile sensing needs in RL-based grasping algorithms. We found that rewards incorporating contact positions, normals, and forces are the most powerful optimization objectives for RL grasp refinement controllers. While this tactile information is essential in the reward function, we uncovered that reducing contact sensor resolution in the policy's state vector decreases algorithm performance only by a small amount. This result has implications for the design of physical grippers and their training and deployment workflows.

In future work, we aim to build physical robotic hands with advanced sensing capabilities to calculate grasp metrics. Secondly, we want to test the proposed training and deployment workflow, providing only limited contact information in the state vector and testing the algorithm on other robotic hands.

References

[1] M. R. Cutkosky and W. Provancher, "Force and tactile sensing," in *Springer Handbook of Robotics*. Springer, 2016, pp. 717–736.

[2] A. M. Dollar, L. P. Jentoft, J. H. Gao, and R. D. Howe, "Contact sensing and grasping performance of compliant hands," *Autonomous Robots*, vol. 28, no. 1, pp. 65–75, 2010.

[3] Q. Wan and R. D. Howe, "Modeling the effects of contact sensor resolution on grasp success," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1933–1940, 2018.

[4] H. Merzić, M. Bogdanović, D. Kappler, L. Righetti, and J. Bohg, "Leveraging contact forces for learning to grasp," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 3615–3621.

[5] A. Melnik, L. Lach, M. Plappert, T. Korthals, R. Haschke, and H. Ritter, "Tactile sensing and deep reinforcement learning for in-hand manipulation tasks," in *IROS Workshop on Autonomous Object Manipulation*, 2019.

[6] ——, "Using tactile sensing to improve the sample efficiency and performance of deep deterministic policy gradients for simulated in-hand manipulation tasks," *Frontiers in Robotics and AI*, vol. 8, p. 57, 2021.

[7] Y. Chebotar, K. Hausman, Z. Su, G. S. Sukhatme, and S. Schaal, "Self-supervised regrasping using spatio-temporal tactile features and reinforcement learning," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 1960–1966.

[8] B. Wu, I. Akinola, J. Varley, and P. Allen, "Mat: Multi-fingered adaptive tactile grasping via deep reinforcement learning," *arXiv preprint arXiv:1909.04787*, 2019.

[9] W. Hu, C. Yang, K. Yuan, and Z. Li, "Reaching, grasping and re-grasping: Learning multimode grasping skills," 2020.

[10] D. Silver, S. Singh, D. Precup, and R. S. Sutton, "Reward is enough," *Artificial Intelligence*, vol. 299, p. 103535, 2021.

[11] C. Ferrari and J. Canny, "Planning optimal grasps," in *Proceedings 1992 IEEE International Conference on Robotics and Automation*, 1992, pp. 2290–2295 vol.3.

[12] B. Mirtich and J. Canny, "Easily computable optimum grasps in 2-d and 3-d," in *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*. IEEE, 1994, pp. 739–747.

[13] M. Buss, H. Hashimoto, and J. Moore, "Dextrous hand grasping force optimization," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 3, pp. 406–418, 1996.

[14] Z. Li and S. Sastry, "Task-oriented optimal grasping by multifingered robot hands," *IEEE Journal on Robotics and Automation*, vol. 4, no. 1, pp. 32–44, 1988.

[15] N. Pollard, "Synthesizing grasps from generalized prototypes," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 3, 1996, pp. 2124–2130 vol.3.

[16] C. Borst, M. Fischer, and G. Hirzinger, "Grasp planning: how to choose a suitable task wrench space," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, vol. 1, 2004, pp. 319–325 Vol.1.

[17] M. A. Roa and R. Suárez, "Grasp quality measures: review and performance," *Autonomous robots*, vol. 38, no. 1, pp. 65–88, 2015.

[18] D. Prattichizzo and J. C. Trinkle, "Grasping," in *Springer Handbook of Robotics*. Springer, 2016, pp. 955–988.

[19] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, Sep 2004, pp. 2149–2154.

[20] J. Lee, M. X. Grey, S. Ha, T. Kunz, S. Jain, Y. Ye, S. S. Srinivasa, M. Stilman, and C. K. Liu, "Dart: Dynamic animation and robotics toolkit," *Journal of Open Source Software*, vol. 3, no. 22, p. 500, 2018.

[21] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "Ros: an open-source robot operating system," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA) Workshop on Open Source Robotics*, Kobe, Japan, May 2009.

[22] A. Raffin, A. Hill, M. Ernestus, A. Gleave, A. Kanervisto, and N. Dormann, "Stable baselines3," https://github.com/DLR-RM/stable-baselines3, 2019.

[23] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," 2018.

[24] A. Y. Ng, D. Harada, and S. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," in *In Proceedings of the Sixteenth International Conference on Machine Learning*. Morgan Kaufmann, 1999, pp. 278–287.

[25] J. M. Hsu and S. C. Peters, "Extending open dynamics engine for the darpa virtual robotics challenge," in *Proceedings of the 4th International Conference on Simulation, Modeling, and Programming for Autonomous Robots - Volume 8810*, ser. SIMPAR 2014. Berlin, Heidelberg: Springer-Verlag, 2014, p. 37–48.

[26] J. R. Taylor, E. M. Drumwright, and J. Hsu, "Analysis of grasping failures in multi-rigid body simulations," in *2016 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR)*, 2016, pp. 295–301.