Design and Control of Roller Grasper V2 for In-Hand Manipulation

Shenli Yuan[1,2], Lin Shao[2], Connor L. Yako[1,2], Alex Gruebele[1], and J. Kenneth Salisbury[2]

Abstract— The ability to perform in-hand manipulation still remains an unsolved problem; having this capability would allow robots to perform sophisticated tasks requiring repositioning and reorienting of grasped objects. In this work, we present a novel non-anthropomorphic robot grasper with the ability to manipulate objects by means of active surfaces at the fingertips. Active surfaces are achieved by spherical rolling fingertips with two degrees of freedom (DoF) – a pivoting motion for surface reorientation – and a continuous rolling motion for moving the object. A further DoF is in the base of each finger, allowing the fingers to grasp objects over a range of size and shapes. Instantaneous kinematics was derived and objects were successfully manipulated both with a custom handcrafted control scheme as well as one learned through imitation learning, in simulation and experimentally on the hardware.

I. INTRODUCTION

In an effort to bring robots from the laboratory into real world environments, researchers have endeavored to develop increasingly dexterous robots that can interact deftly with objects. In order for such robots to take on a wide range of everyday tasks, they need to be capable of sophisticated object manipulation. Many vital higher-level tasks will rely on a robot's capability to perform in-hand manipulation by reorienting objects while maintaining the grasp. Out of all the grasping and manipulation tasks, in-hand manipulation is among the ones that require the most dexterity.

A background of in-hand manipulation literature is presented in [1] and a more extensive review of robot hands and graspers is given in [2]. Common approaches to designing robotic grippers that can perform in-hand manipulation are: anthropomorphic hands which take advantage of intrinsic human dexterity, but due to the high number of degrees of freedom are complex and expensive [3] [4] [5]; underactuated hands which passively conform to objects, achieving good grasp stability, but at the cost of the controllability needed to perform many in-hand manipulation tasks [6] [7] [8] [9]; grippers with active surfaces (such as conveyors) which allow for the object to be manipulated without changing grasp pose, but with a fixed conveyor orientation limiting possible motions [10] [11] [12].

We developed a novel grasper design using articulated, actively driven spherical rollers located at the fingertips, shown in Fig. 1. By incorporating continuous rotating mechanisms,

[1]Department of Mechanical Engineering, Stanford University
[2]Stanford Artificial Intelligence Lab (SAIL), Stanford University
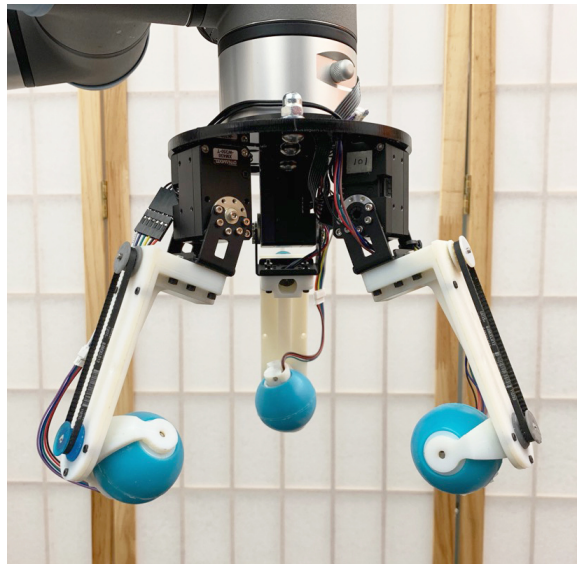{shenliy, lins2, clyako, agruebe2}@stanford.edu, jks@cs.stanford.edu

Fig. 1. The Roller Grasper V2 prototype mounted on a UR-5 robot arm. Three fingers, each with three degrees of freedom, can grasp and reorient an object using rolling contact with the rubber coated rollers

it is possible to create graspers that are highly capable but relatively simple by design. The active surface achieved by rolling and reorientation of the spherical rollers allow the grasper to perform in-hand manipulation without the need for finger gaiting. Rolling contact on the object can be viewed as a motion that continuously breaks contact with the object while simultaneously re-establishing contact at adjacent locations. The ability to reorient an object to any pose also lessens the need to use externally actuated degrees of freedom (e.g. actuation of the robotic arm and wrist) which simplifies the control scheme. More importantly, the spherical design of the finger tips allows for stable grasps independent from the roller orientations, eliminating the need to analyze grasping modes for different combinations of roller orientations.

Learning robust policies for in-hand manipulation has been a long-standing challenge in robotics due to the complexity of modelling the object and grasper contacts and the difficulty of controlling finger motion in long and complicated manipulation sequences. Deep Reinforcement Learning (DRL) has been used to learn dexterous manipulation [13] [14]. However, learning to hold the object firmly and stably while transforming the object with deep reinforcement learning requires many more training episodes and a carefully designed reward function. To overcome these issues, we used an imitation learning based approach to learn a control policy in order to arbitrarily transform an object while holding it.

We demonstrated the effectiveness of this learned policy both in simulation and in real world experiments.

Our in-hand manipulation system consisted of a 3-fingered grasper with spherical rollers at the fingertips, an overhead RGBD camera, and objects with QR-tags on all faces. A handcrafted control policy and an imitation learning policy were developed to perform complex in-hand object transformations. To the best of our knowledge, this work is the first attempt at developing a grasper with active surfaces at the fingertips that transforms grasped objects through an imitation learning policy. The paper is structured as follows: we first discuss our previous iteration of this robotic grasper as well as other design and algorithmic approaches to robotic grasping/in-hand manipulation (Section II). Section III then briefly describes the hardware. Section IV discusses the formulation of the handcrafted control policy as well as the imitation learning approach. The paper then provides an overview of the experimental setup in simulation and in real life, and concludes by reporting and discussing the results (Section V and Section VI).

II. RELATED WORK

A. Previous Grasper Design

Our previous work [1] used articulated, actively driven cylindrical rollers at the fingertips of a grasper to explore imparting motion within a grasp using active surfaces. The grasper used three modular 3-DoF fingers, and demonstrated full 6-DoF spatial manipulation of objects including a sphere, cube, and cylinder, as well as various grasping modalities. One limitation of the previous roller grasper is the grasp stability. Due to the cylindrical design of the finger tips, several grasping configurations are unstable, resulting in undetermined manipulation behaviors. The redundant combinations of grasping configurations also complicates the control scheme as the configuration used is dependent on specific manipulation tasks and the object being manipulated.

B. In-Hand Manipulation

In-hand manipulation is an age-old question in robotics with a rich literature, from two-fingered grippers [15] [16], to dexterous hands [13]. We briefly review the relevant in-hand manipulation works in this subsection. To achieve in-hand manipulation, multi-fingered dexterous hands utilize the redundancy of the fingers to move the object while holding it. Under-actuated hands are able to leverage model based control [17] for in-hand manipulation tasks. There are also task-specific designs of under-actuated hands which enable a limited set of repositioning skills [18] [19].

Other approaches to in-hand manipulation have been explored which rely on gravity with controlled slip [20] [21], induced accelerations [22] [23], or the environment [15] [24] [25] to reduce the dexterity required of the hand. However, such approaches require complex control and modeling schemes or dependency on available environmental geometry.

Contrary to modeling the complex dynamics involved in grasping and object reorientation, some researchers have opted to use reinforcement learning (RL) to search for optimal policies. This is especially useful when using underactuated graspers or graspers with high DoF's. In [26] an underactuated grasper with tactile sensors on the fingertips was used to horizontally slide a wooden cylinder back and forth by rolling it along each finger. The learned policy was evaluated on cylinders of different masses, sizes, and friction coefficients. They found that the policy performed better than a hard-coded control policy, which was used as a baseline, but still struggled with cylinders with low-friction properties. DRL has also been implemented successfully on the 24 DoF Shadow Hand for dynamically moving a cylinder in hand and for arbitrarily reorientating a cube using visual information [13] [14]. However, our grasper needs to maintain hold of the object solely through friction at the roller contact during the manipulation process. This means that a tiny perturbation of the fingertip could possibly break the contact and lead to a dropped object. Therefore, the space of successful policies is incredibly small relative to the entire policy space. Since exploration is necessary in any DRL problem, it is difficult for the algorithm to converge to the optimal policy. To avoid this problem, we instead adopted an imitation learning method, which will be discussed in the next section.

C. Imitation Learning Methods

Imitation learning aims to learn control policies by observing expert demonstrations. There are in general two types of approaches to tackle an imitation learning problem. *Behaviour cloning* aims to train the agent to learn a mapping from observations to actions given demonstrations, in a supervised learning fashion [27] [28]. Another approach is *Inverse Reinforcement Learning* [29] [30], which attempts to learn a reward function that describes the given demonstrations. Our method falls under the scope of *Behavior Cloning* which has led to many successes in robotics [31] [32]; our approach is based on one *Behaviour Cloning* method called DAgger [28]. To tackle the problem of generating expert demonstrations, we also develop a method to accumulate the demonstration examples iteratively starting from a few expert demonstrations.

III. DESIGN

A. Hardware Design

The gripper (Fig. 2(a)) consists of three fingers, each having three degrees of freedom (DoF). The first DoF is at the base of each finger and consists of a revolute joint directly driven by a Robotis Dynamixel XM430-W350 smart actuator. The other two DoF are located at each fingertip, and are responsible for steering and rolling. The second joint shown in Fig. 2(a) is orthogonal to the first DoF, and is driven by a micro DC motor with built-in gearbox and quadrature encoder (Servocity No.638099). For a compact form-factor, this actuator is located remotely from the axis of rotation through a timing belt (Fig. 2(b)), and allows the roller assembly to be pitched up to 180 degrees. The final DoF is actuated using the same type of geared motor but housed inside the roller assembly (Fig. 2(c)), allowing it to
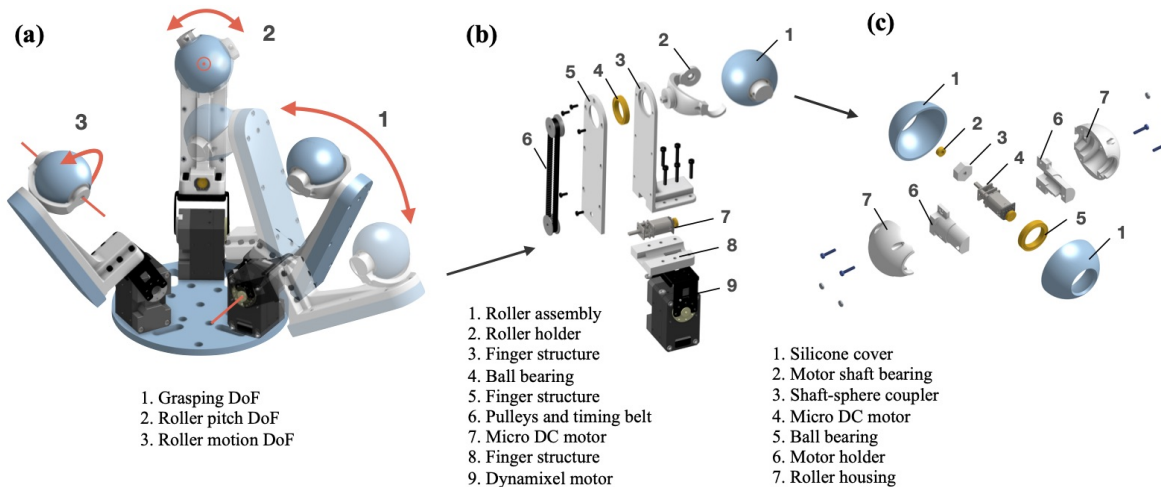
Fig. 2. A CAD model of the grasper: (a) the three degrees of freedom each finger has (b) an exploded view of each finger (c) an exploded view of the roller assembly that contacts the object being manipulated

(a)
1. Grasping DoF
2. Roller pitch DoF
3. Roller motion DoF

(b)
1. Roller assembly
2. Roller holder
3. Finger structure
4. Ball bearing
5. Finger structure
6. Pulleys and timing belt
7. Micro DC motor
8. Finger structure
9. Dynamixel motor

(c)
1. Silicone cover
2. Motor shaft bearing
3. Shaft-sphere coupler
4. Micro DC motor
5. Ball bearing
6. Motor holder
7. Roller housing

TABLE I

PHYSICAL PROPERTIES

Property	Value
Link a (referred in Fig. 4)	$48mm$
Link b (referred in Fig. 4)	$122mm$
Link r (referred in Fig. 4)	$21.5mm$
Whole grasper weight	$700g$
Maximum normal force at the fingertip	$33.6N$
Maximum roller shear force	$16.4N$

Fig. 3. System architecture

perform continuous rotation of the spherical contact surface without its cables winding. The roller is encased in a pair of 2mm-thick semi-spherical silicone covers to provide a high-friction surface for grasping and manipulation. The silicone covers are molded from Mold Star 16 Fast (SmoothOn), cast in a 2-part 3D printed mold (Objet Vero white, glossy surface finish), with a coat of Ease Release 2000 release agent. They are adhered to the 3D printed roller surfaces using silicone compatible tape (McMaster No.7213A31) The reference frame of the grasper is shown in Fig. 4 and the key physical parameters are listed in Table I.

The design allows for unbounded reorientation of the grasped object, while the range of translation is determined by various factors such as the physical dimensions, shape, in-hand orientation, mass, and resulting friction coefficient of the grasped objects.

B. System Architecture

The system architecture used to operate the gripper is shown in Fig. 3. A high-level API was developed to interface between the low-level (hardware) information and the manipulation algorithm. Information transferred during the bidirectional communication includes positions for each joint of the fingers, the current limit of the base joint, as well as the control parameters for controlling the motors. Current to the Dynamixel motors is used to set the stiffness of the base joints and estimate the force exerted by the object on each finger during manipulation.

A Teensy 3.6 microcontroller is used to handle communication with the high-level API as well as low-level control
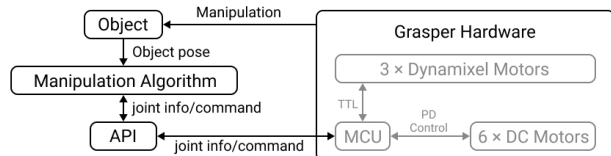
of the motors. The six micro gearmotors located at the intermediate joints and within the rollers are controlled by PD position controllers handled by the microcontroller. The Dynamixel motors each run local PD control and communicate with the Teensy microcontroller through a TTL half-duplex asynchronous serial communication protocol.

IV. TECHNICAL APPROACH

In this section we begin by describing the analytical approach that led to development and implementation of a handcrafted control policy. We then discuss how this control policy was fine-tuned based on simulation results in order to act as an expert trajectory generator for the imitation learning. The final subsection describes the imitation learning formulation.

Initially, several DRL methods were used in place of imitation learning. From these preliminary experiments we learned that the space of successful policies, not necessarily optimal, was especially small. This becomes a critical problem in DRL given large policy spaces since significant exploration is necessary. Instead, we elected to pursue imitation learning which proved to be both effective and efficient.

A. Analysis

Manipulating an object through rolling contact can be viewed as navigating the rollers on the object. Therefore, an object transformation can be achieved by navigating the rollers from their initial contact locations to the desired final contact locations. While there is no unique solution for the paths the rollers take during their navigation for given initial and final grasping poses, it is possible to solve, on this
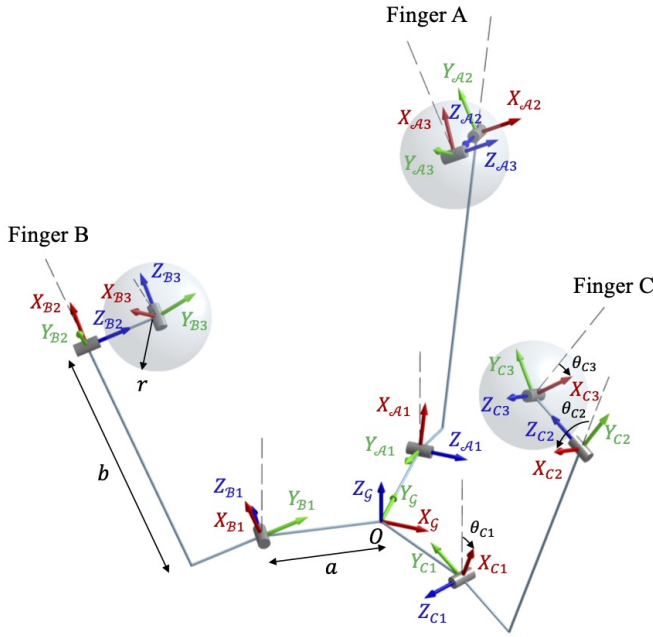
Fig. 4. Reference frames of each of the three joints within each finger, relative to the base coordinate system. Dashed lines indicate neutral positions from which joint angles are measured.

non-holonomic system, for the instantaneous joint velocities based on known object geometry, object pose, object velocity, and contact conditions (rolling without slipping).

Object geometry and pose are necessary to calculate the contact locations, which determine the relationship between joint motions and given contact motions on the rollers (the contact jacobian matrix). This information can subsequently be used to map the desired object motion to the motions at the contact point on the object. Applying the contact condition of rolling without slipping means that the contact velocity on the object and the contact velocity on the roller are equal. Thus, inverse kinematics can be used to calculate the joint velocities required for a desired object motion, a process similar to the calculation of geometric stiffness for grasping presented in [33].

The problem is formulated as follows: given the object's initial and target position and orientation, compute the contact location and contact motion on the roller, and then compute the pivot joint orientation and motions of the base and roller joints.

Given the desired motion of the object and known contact locations, we can obtain the contact motion by
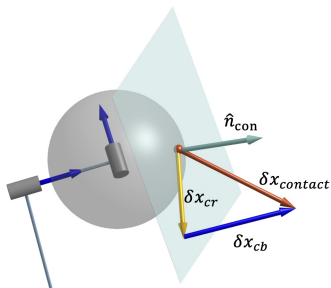

Fig. 5. Contact motion breaks down to two components

$$\delta x_{contact} = J_{obj} \delta x_{obj} \tag{1}$$

where J_{obj} is the jacobian matrix mapping object motion to the motion at the contact frame.

On the other hand, with the contact motion $\delta x_{contact}$, the object position x_{obj}, and roller position x_{roller} all known, the contact motion can be interpreted as the motion at the contact location due to the movement of the 3 joints:

$$\delta x_{contact} = J_\theta \delta\theta \tag{2}$$

Where J_θ is the contact jacobian matrix for mapping finger joint motions to motions in the contact coordinates, and $\delta\theta$ is a vector of joint motions of the finger. In many robotics applications $\delta\theta$ is determined directly by solving (2). However, this method is not particularly applicable to this gripper design for the following reasons:

1) The contact locations are very close to the pivot joint so the finger is always close to a singularity.
2) The pivot joint has a joint limit of $\left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$ meaning that in many cases the instantaneous velocity required from the pivot joint to achieve the desired contact velocity cannot be reached.

Therefore, instead of carrying out the full inverse kinematics, we divide the contact motion into two components: the component resulting from the motion of the base joint, δx_{cb}, and the component resulting from the rolling, δx_{cr}. The pivot joint is used to reorient the roller so that the rolling direction at the contact point is aligned with δx_{cr}. This approximation is sufficient because (1) when the contact locations are close to the pivot axis, the pivot motion does not have a significant impact on the object motion, and (2) the object being grasped is over constrained by the three fingers, so using the soft finger model [34] to approximate roller contacts, the torque exerted by pivoting is compensated by the lateral frictions from the two other fingers. Thus, the singularity is advantageous in that it enables sole use of the pivot joint to align the roller in order to achieve δx_{cr}.

We developed a handcrafted control strategy based on the above formulation with some modifications that fit the gripper design. The handcrafted control policy tested in simulation and on the hardware assumes that the object geometry is spherical with radius R. Specifically, the below calculations are for a single finger, but can be applied to the two other fingers as well. For the simplicity of notation, we use subscript 1, 2, and 3 to represent the base joint, pivot joint, and roller joint, respectively, and leave out the subscript $\mathcal{A}$, $\mathcal{B}$ or $\mathcal{C}$ because the following derivation would be identical for each finger.

In order to obtain δx_{cb} and δx_{cr}, we need to project $\delta x_{contact}$ onto Z_2 and the contact plane (Fig. 5). The contact plane can be described by its normal vector $\vec{n}_{con}$:

$$\hat{n}_{con} = \frac{x_{obj} - x_{roller}}{||x_{obj} - x_{roller}||_2} \tag{3}$$

Using the fact that δx_{cr} is orthogonal to $\hat{n}_{con}$ and is also in the plane formed by δx_{cb} and $\delta x_{contact}$, we can compute

the direction of δx_{cr} as:

$$\widehat{\delta x_{cr}} = \frac{(\delta x_{cb} \times \delta x_{contact}) \times \hat{n}_{con}}{||(\delta x_{cb} \times \delta x_{contact}) \times \hat{n}_{con}||_2} \qquad (4)$$

It is also easy to find that the direction of δx_{cb} aligns with Z_2:

$$\widehat{\delta x_{cb}} = Z_2 \qquad (5)$$

Projecting $\delta x_{contact}$ onto $\widehat{\delta x_{cb}}$ and $\widehat{\delta x_{cr}}$ gives δx_{cb} and δx_{cr}. It is equivalent to solving α and β in equation:

$$\delta x_{contact} = \alpha \widehat{\delta x_{cb}} + \beta \widehat{\delta x_{cr}} \qquad (6)$$

By cross multiplying $\widehat{\delta x_{cb}}$ and $\widehat{\delta x_{cr}}$ to (6), respectively, we can solve for α and β, and the resulting δx_{cb} and δx_{cr} are shown below. Note that $\hat{n}_z$ is only used to extract the magnitude from the cross products.

$$\hat{n}_z = \frac{\widehat{\delta x_{cr}} \times \widehat{\delta x_{cb}}}{||\widehat{\delta x_{cr}} \times \widehat{\delta x_{cb}}||_2} \qquad (7)$$

$$\delta x_{cb} = \frac{\hat{n}_z \cdot (\widehat{\delta x_{cr}} \times \delta x_{contact})}{\hat{n}_z \cdot (\widehat{\delta x_{cr}} \times \widehat{\delta x_{cb}})} \widehat{\delta x_{cb}} \qquad (8)$$

$$\delta x_{cr} = \frac{\hat{n}_z(\widehat{\delta x_{cb}} \times \delta x_{contact})}{\hat{n}_z \cdot (\widehat{\delta x_{cb}} \times \widehat{\delta x_{cr}})} \widehat{\delta x_{cr}} \qquad (9)$$

The joint velocity of base joint (ω_1) and roller joint (ω_3) can be calculated using inverse kinematics.

The final step is to calculate the pivot angle $\hat{\theta}_2$ to align the rolling direction with $\widehat{\delta x_{cr}}$:

$$Z_3 = \pm \frac{Z_2 \times \widehat{\delta x_{cr}}}{||Z_2 \times \widehat{\delta x_{cr}}||_2} \qquad (10)$$

Note that because there is a joint limit at the pivot, we are limiting Z_3 to always have a component along the Z_0 direction. The pivot angle θ_2 can be calculated by

$$\theta_2 = \arccos\left(\frac{Z_1 \cdot Z_3}{||Z_1||_2 ||Z_3||_2}\right) \qquad (11)$$

The above process only describes how each joint should move given the instantaneous desired velocity of the object; no path planning is used for navigating the rollers given the initial and target poses of the object. In our case, we simply compute the difference between the initial and target pose, and set the desired velocity equal to a scaling factor multiplied by this computed difference: $\delta x_{obj} = \lambda \Delta x_{obj}$. This method works very well for convex objects whose radii of curvature do not change drastically. Experimental validation of manipulating a cube is shown in Section V.

B. Handcrafted Control Policy

The handcrafted control policy is formulated according to the results from the previous section. Given the current object position and orientation, the target object position and orientation, and the current joint values, the positions of all nine joints are calculated for the following time step. One difference between the implemented policy and the theoretical policy is that the base joint velocities are not controlled based on the derivation above. Instead, they are position controlled to a setpoint in order for the rollers to maintain contact with the object. This is because we are focusing on the rotation of the object instead of translation. The main purpose of the base joint in our case is to keep the rollers in contact with the object. The analytical approach presented in the previous section performs both translation and rotation. The translation capability of the grasper is demonstrated in the complementary video through scripted movements. However, the translation capability is relatively limited compared to the rotation capability, which is why we decided to focus on object rotation in the control policy.

C. Imitation Learning Algorithm

We adopted imitation learning, specifically *Behavior Cloning* in order to learn how to transform an object. The optimal policy was learned through expert demonstrations characterized by the above handcrafted control policy. *Behavior Cloning* is particularly useful when it is difficult to explicitly specify the reward function.

In our case, the state space is defined as $|\mathcal{S}| \in \mathbb{R}^{35}$, and consists of the following: the current state of the grasper ($s_1 \rightarrow s_9$), current object position ($s_{10} \rightarrow s_{12}$), current object quaternion ($s_{13} \rightarrow s_{16}$), previous object position ($s_{17} \rightarrow s_{19}$), previous object quaternion ($s_{20} \rightarrow s_{23}$), object termination position ($s_{24} \rightarrow s_{26}$), object termination orientation in angle-axis representation ($s_{27} \rightarrow s_{29}$), object initial position ($s_{30} \rightarrow s_{32}$), and object initial orientation in angle-axis representation ($s_{33} \rightarrow s_{35}$). The action space is defined as $|\mathcal{A}| \in \mathbb{R}^9$ and contains the nine joint positions ($a_1 \rightarrow a_9$).

We constructed a deep neural network to determine the actions for each of the nine gripper joints. The network consisted of three fully connected hidden layers with leaky ReLU activations, except at the output layer, and 256 nodes in each hidden layer.

The handcrafted control policy from the previous section was used to generate N expert trajectories, which are simply a series of state-action pairs. The ith trajectory is defined as:

$$T^{(i)} = [(s_0^{(i)}, a_0^{(i)}), (s_1^{(i)}, a_1^{(i)}), \dots] \qquad (12)$$

We first trained our policy $\pi^0(s^i)$ to predict the expert action by minimizing the loss L between $\pi^0(s_j^i)$ and a_j^i in a supervised approach:

$$L_j^i = ||\pi^0(s_j^i) - a_j^i||_2, \forall s_j \in |\mathcal{T}^i|, i \in N \qquad (13)$$

Subsequent policy updates are computed according to DAgger [28].

We also implemented a method to increase the number of expert demonstrations iteratively. For a given object, every object transformation trajectory is specified by a 12-dimensional vector containing the object starting and target position and orientation: $(x_{obj,s}^i, q_{obj,s}^i, x_{obj,t}^i, q_{obj,t}^i)$. By imitating the expert demonstration examples, we learn a policy supported by these expert demonstrations. This control embedding is able to interpolate between known trajectories

using a nearest neighbor policy in order to generate trajectories for previously unseen transformations.

Based on the learned policy, we accumulate nearby transformations. Let $\mathcal{D} = (x^i_{obj,s}, q^i_{obj,s}, x^i_{obj,t}, q^i_{obj,t})$ represent the transformations which our policy already knows. If an interpolated trajectory transformation is nearby to one of the transformations in $\mathcal{D}$, we add the transformation to $\mathcal{D}$, and then continue to train our policy based on the transformation demonstrations in $\mathcal{D}$. Through this fashion, we kept growing and maintaining a traversable graph of transforming the object between various poses, similar to [35]. The learned control policy could also be treated as an efficient initialization for a deep reinforcement learning approach or motion planning for in-hand manipulations.

V. EXPERIMENTS

A. Simulation Experiments

Mujoco 2.0 was used to simulate both the handcrafted control policy and the learned policy before transferring them to the physical setup (see Fig. 6). By default, Mujoco assumes a soft-contact model which leads to relatively slippery contacts, but the solver parameters (*solref* and *solimp*) were changed to have a harder contact model in order to better model the friction between the object and the spherical rollers. An elliptical friction cone was used along with the Newton solver to evaluate these constraints.

The gripper was modeled in the software, and the following setup was specified. The base joints were position controlled, and had their force output clamped; these settings allowed the fingers to act compliantly and conform to an object's shape as it was reoriented, and stabilized the grasp. The pivots had a rotation range from $[-\frac{\pi}{2}, \frac{\pi}{2}]$ (with the zero position corresponding with the roller axis aligned along the length of the finger) in order to represent the limited range in the physical setup due to motor wires. A pivot rotation threshold of $3°$ per time step was used to prohibit quick jumps between the two rotation limits in favor of smoother motion that stabilized the simulation.

Sensors were placed at all nine joints to read joint positions and two sensors were placed on the object to obtain its orientation (quaternion) and 3D position relative to the global frame. The learned policy outputted values for all nine joints. The handcrafted control policy only used the latter two
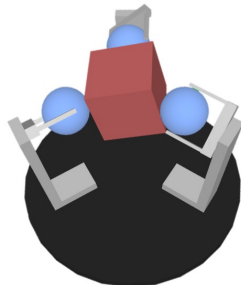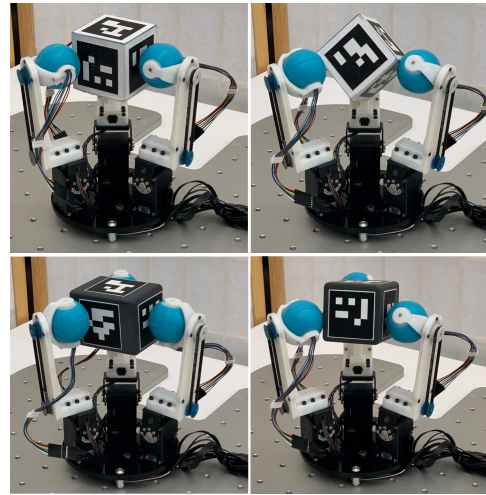

Fig. 7. Top row: the cube being manipulated from a starting position shown on the left. Bottom row: the novel object, a rectangular prism being similarly manipulated

sensors and the base joint positions to calculate the output velocities for the pivots and rollers at every time step.

Experiments were run by specifying a desired orientation in angle-axis representation $([x, y, z]^T, \theta)$. Most experiments were carried out with a 6 cm cube with a mass of 32.4 grams that had a starting quaternion of $q_0 = [1, 0, 0, 0]$. Object rotation angles were specified as at least $90°$ in the majority of experiments to ensure that the rollers had to traverse across the edges of the cube. Rotation axes ranged from more vertical axes – which were generally easier for the simulation to realize – to purely horizontal axes which were more difficult.

B. Experimental Setup

The experimental setup included the grasper, an overhead Intel Realsense D400series RGBD camera, and various 3D printed objects including a cube, a cube with filleted edges, and spheres of various masses and sizes. The handcrafted control policy was able to be run both open-loop and closed-loop. Since object orientation and position were used as input to the control policy, only the cube was run in the closed-loop operation with QR-tags on all 6 faces. Open-loop runs with the spheres were used to qualitatively verify the handcrafted control policy on the hardware.

At the start of each experiment the fingers would spread out, allowing a human operator to hold the cube at approximately the initial position specified for that particular trial. The fingers would then move inwards and grasp the object. At every time step the joint positions, object orientation, and position would be read-in, from which the corresponding joint output would be calculated and sent to the actuators.

C. Evaluation Metric

We adopted the orientation error metric suggested in [36] which can be computed by using the sum of quaternions and normalizing. Given the desired object orientation, $q_{obj,d}$, and the current object orientation, $q_{obj,c}$, the error is calculated as the following:

Fig. 6. A visualization of our simulation in Mujoco. In simulation the gripper maintains the same physical design as the real gripper.
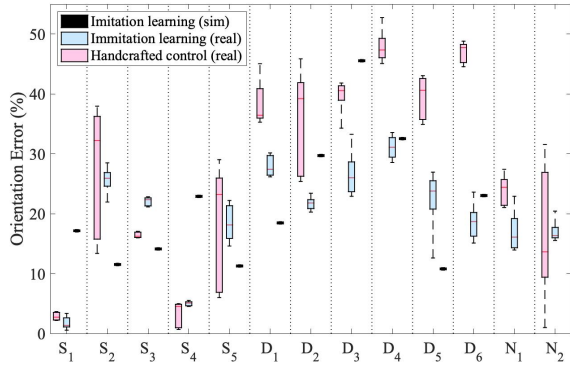
Fig. 8. Orientation error for real world experiment results. The vertical axes represents the orientation error defined in V-C. The horizontal lines represents various experiments. S D and N denote simple manipulations tasks, difficult manipulation tasks and novel object transformations,respectively. Notes: (1) The handcrafted control is designed to stop at a pre-defined error threshold, therefore its orientation error is not meaningful to shown. (2) Imitation learning policy in simulation is deterministic. (3) The novel object transformation was only tested in real-world experiments, thus no simulation results were available.

$$e_\omega = 100 \frac{\min(||q_{obj,d} - q_{obj,c}||_2, ||q_{obj,d} + q_{obj,c}||_2)}{\sqrt{2}} \quad (14)$$

VI. RESULTS AND ANALYSIS

The results of the experiments carried out on the hardware are presented in Fig. 8. All reported percent average errors were calculated according to (14). The experiments were divided up as follows: simple transformations, which had axes of rotations with strong $Z_\mathcal{G}$ components, difficult transformations, which had axes of rotations with strong x and y components, and novel object transformations, which consisted of transforming an elongated rectangular prism with filleted edges (6 cm × 6 cm × 8 cm) (Fig. 7). The reason transformations with strong $Z_\mathcal{G}$ components for the rotation axis were easier was due to the three-finger design of the grasper, leading to a much better force closure around $X_\mathcal{G} - Y_\mathcal{G}$ direction than the $Z_\mathcal{G}$ direction. All experiments specified the target as an axis and a 90° rotation about that axis. This choice of a rotation angle is described in *Simulation Experiments*.

A. Simple Cases & Sim-to-Real Transferring

The imitation learning method performed better for 3 out of the 5 simple transformation cases ($S_1 \rightarrow S_5$). The average orientation error across these 5 trials shows that the imitation learning method (15.3%±2.2%) and the handcrafted control policy (13.6% ± 7.2%) are comparable when performing object rotations about more vertical axes. However, the handcrafted control policy has more than 3 times the standard deviation of the imitation learning.

As with any physical system there is a performance difference when compared to the simulation setup. Fortunately, this difference was small with the imitation learning's results in simulation reporting an average error of 15.4%. We believe this is due to inclusion of sensor noise in the simulation, as

well as fine tuning of the contact model to more closely align with observations of the physical system performance.

B. Untrained Target Pose Test Cases

For the difficult transformations ($D_1 \rightarrow D_6$), the imitation learning outperformed the handcrafted policy across all 6 trials by an average percent average error of 16.0%. Three of these trials, $D_1 \rightarrow D_3$, including target poses not trained in simulation. Overall, the imitation learning achieved a percent average error of 25.3%±4.2%, while the handcrafted control policy achieved 41.3%±5.8%. Again, the imitation learning was able to demonstrate more stable trajectories despite the sensor noise.

C. Novel Object Test Cases

Tests with the novel object ($N_1 \rightarrow N_2$), a rectangular prism, demonstrated similar results to the simple transformation case. The percent average errors were comparable between imitation learning (17.1%±3.9%) and the handcrafted control policy (20.3% ± 10.3%). However, the handcrafted control policy had a much higher standard deviation.

D. Discussion

A potential explanation for the poorer performance of the handcrafted control policy, especially for the difficult transformations, was that it always generated linear trajectories from the current position to the target. However, in many cases, especially for rotation axes with dominating horizontal components, taking a linear path can lead to the gripper dropping the object. As the object is rotated closer to the desired axis of rotation, one or more of the rollers can roll onto a surface where the contact normal has a $-Z_\mathcal{G}$ component; the contact normal can overcome the frictional support provided by another roller and result in dropping the object. This problem can be mitigated by increasing the positional error compensation in the closed-loop control method in order to more actively control the object height. Unfortunately, placing a greater emphasis on controlling the object position was seen to cause previously successful trajectories to fail. No combination of gains was seen to work across all observed successes. On the other hand, the imitation learning was better for a couple of potential reasons: (1) the policy was learned from noisy sensor data which increased its robustness, and (2) the discovery of safe and repeatable trajectories.

VII. CONCLUSION AND FUTURE WORK

This paper presents Roller Grasper V2, a new design for a grasper based on steerable spherical rollers located at the finger tips. The hardware design choices and engineering details were provided. A handcrafted control policy was constructed that utilized the active surfaces of the rollers to transform an object to an arbitrary target pose. This control policy was used to generate expert trajectories in order to develop an imitation learning based policy. This learned policy was able to interpolate between learned trajectories in order to successfully reach new targets; these new trajectories

were placed in a buffer of known trajectories in order to develop a traversable graph. Experiments for the handcrafted policy and the learned policy demonstrated the utility of this graph for generating safe trajectories, as the learned policy outperformed the handcrafted policy for all difficult target poses. For simple cases, the two policies performed comparably, except the learned policy had much lower variance in the trajectory. Future work includes reducing the size of the rollers to allow for manipulation of smaller objects, developing a novel mechanism that solves the non-holonomic constraints and incorporating tactile sensing on the rollers to provide high-fidelity feedback.

REFERENCES

[1] S. Yuan, A. D. Epps, J. B. Nowak, and J. K. Salisbury, "Design of a roller-based dexterous hand for object grasping and within-hand manipulation," in *2020 IEEE International Conference on Robotics and Automation*, May 2020.

[2] C. Piazza, G. Grioli, M. Catalano, and A. Bicchi, "A century of robotic hands," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 2, pp. 1–32, 2019.

[3] M. A. Diftler, J. S. Mehling, M. E. Abdallah, N. A. Radford, L. B. Bridgwater, A. M. Sanders, R. S. Askew, D. M. Linn, J. D. Yamokoski, F. A. Permenter, B. K. Hargrave, R. Platt, R. T. Savely, and R. O. Ambrose, "Robonaut 2 - the first humanoid robot in space," in *2011 IEEE International Conference on Robotics and Automation*, pp. 2178–2183, May 2011.

[4] Shadow Robot Company, "Design of a dextrous hand for advanced CLAWAR applications," in *Proceedings of the 6th International conference on climbing and walking robots and the supporting technologies for mobile machines*, pp. 691–698, September 2003.

[5] M. Grebenstein, A. Albu-Schäffer, T. Bahls, M. Chalon, O. Eiberger, W. Friedl, R. Gruber, S. Haddadin, U. Hagn, R. Haslinger, H. Höppner, S. Jörg, M. Nickl, A. Nothhelfer, F. Petit, J. Reill, N. Seitz, T. Wimböck, S. Wolf, T. Wüsthoff, and G. Hirzinger, "The dlr hand arm system," in *2011 IEEE International Conference on Robotics and Automation*, pp. 3175–3182, May 2011.

[6] R. R. Ma and A. M. Dollar, "An underactuated hand for efficient finger-gaiting-based dexterous manipulation," in *2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014)*, pp. 2214–2219, Dec 2014.

[7] A. M. Dollar and R. D. Howe, "The highly adaptive sdm hand: Design and performance evaluation," *The International Journal of Robotics Research*, vol. 29, no. 5, pp. 585–597, 2010.

[8] H. Stuart, S. Wang, O. Khatib, and M. R. Cutkosky, "The ocean one hands: An adaptive design for robust marine manipulation," *The International Journal of Robotics Research*, vol. 36, no. 2, pp. 150–166, 2017.

[9] N. Rojas, R. R. Ma, and A. M. Dollar, "The gr2 gripper: an underactuated hand for open-loop in-hand planar manipulation," *IEEE Transactions on Robotics*, vol. 32, no. 3, pp. 763–770, 2016.

[10] N. Govindan and A. Thondiyath, "Design and Analysis of a Multimodal Grasper Having Shape Conformity and Within-Hand Manipulation With Adjustable Contact Forces," *Journal of Mechanisms and Robotics*, vol. 11, 07 2019. 051012.

[11] V. Tincani, M. G. Catalano, E. Farnioli, M. Garabini, G. Grioli, G. Fantoni, and A. Bicchi, "Velvet fingers: A dexterous gripper with active surfaces," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1257–1263, Oct 2012.

[12] R. R. Ma and A. M. Dollar, "In-hand manipulation primitives for a minimal, underactuated gripper with active surfaces," in *proceedings of the ASME 2016 International Design Engineering Technical Conferences (IDETC), Computers and Information in Engineering Conference (CIE)*, p. V05AT07A072, 2016.

[13] V. Kumar, E. Todorov, and S. Levine, "Optimal control with learned local models: Application to dexterous manipulation," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 378–383, IEEE, 2016.

[14] O. M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, *et al.*, "Learning dexterous in-hand manipulation," *The International Journal of Robotics Research*, vol. 39, no. 1, pp. 3–20, 2020.

[15] N. Chavan-Dafle, R. Holladay, and A. Rodriguez, "In-hand manipulation via motion cones," *arXiv preprint arXiv:1810.00219*, 2018.

[16] S. Cruciani, C. Smith, D. Kragic, and K. Hang, "Dexterous manipulation graphs," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2040–2047, IEEE, 2018.

[17] M. Liarokapis and A. M. Dollar, "Deriving dexterous, in-hand manipulation primitives for adaptive robot hands," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1951–1958, IEEE, 2017.

[18] N. Rahman, L. Carbonari, M. D'Imperio, C. Canali, D. G. Caldwell, and F. Cannella, "A dexterous gripper for in-hand manipulation," in *2016 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, pp. 377–382, July 2016.

[19] W. G. Bircher, A. M. Dollar, and N. Rojas, "A two-fingered robot gripper with large object reorientation range," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3453–3460, May 2017.

[20] Y. Karayiannidis, K. Pauwels, C. Smith, D. Kragic, *et al.*, "In-hand manipulation using gravity and controlled slip," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5636–5641, IEEE, 2015.

[21] D. L. Brock, "Enhancing the dexterity of a robot hand using controlled slip," in *Proceedings. 1988 IEEE International Conference on Robotics and Automation*, pp. 249–251, IEEE, 1988.

[22] N. C. Dafle, A. Rodriguez, R. Paolini, B. Tang, S. S. Srinivasa, M. Erdmann, M. T. Mason, I. Lundberg, H. Staab, and T. Fuhlbrigge, "Extrinsic dexterity: In-hand manipulation with external forces," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1578–1585, IEEE, 2014.

[23] J. Shi, J. Z. Woodruff, P. B. Umbanhowar, and K. M. Lynch, "Dynamic in-hand sliding manipulation," *IEEE Transactions on Robotics*, vol. 33, no. 4, pp. 778–795, 2017.

[24] N. Chavan-Dafle and A. Rodriguez, "Prehensile pushing: In-hand manipulation with push-primitives," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6215–6222, IEEE, 2015.

[25] C. Eppner, R. Deimel, J. Alvarez-Ruiz, M. Maertens, and O. Brock, "Exploitation of environmental constraints in human and robotic grasping," *The International Journal of Robotics Research*, vol. 34, no. 7, pp. 1021–1038, 2015.

[26] H. Van Hoof, T. Hermans, G. Neumann, and J. Peters, "Learning robot in-hand manipulation with tactile features," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pp. 121–127, IEEE, 2015.

[27] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," in *Advances in neural information processing systems*, pp. 305–313, 1989.

[28] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635, 2011.

[29] A. Y. Ng, S. J. Russell, *et al.*, "Algorithms for inverse reinforcement learning.,"

[30] L. Shao, T. Migimatsu, Q. Zhang, K. Yang, and J. Bohg, "Concept2robot: Learning manipulation concepts from instructions and human demonstrations," in *Proceedings of Robotics: Science and Systems (RSS)*, July 2020.

[31] A. Billard, Y. Epars, S. Calinon, S. Schaal, and G. Cheng, "Discovering optimal imitation strategies," *Robotics and autonomous systems*, vol. 47, no. 2-3, pp. 69–77, 2004.

[32] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal, "Learning and generalization of motor skills by learning from demonstration," in *2009 IEEE International Conference on Robotics and Automation*, pp. 763–768, IEEE, 2009.

[33] M. R. Cutkosky and I. Kao, "Computing and controlling compliance of a robotic hand," *IEEE transactions on robotics and automation*, vol. 5, no. 2, pp. 151–165, 1989.

[34] M. T. Mason and J. K. Salisbury Jr, "Robot hands and the mechanics of manipulation," 1985.

[35] S. Cruciani, K. Hang, C. Smith, and D. Kragic, "Dual-arm in-hand manipulation and regrasping using dexterous manipulation graphs," *arXiv preprint arXiv:1904.11382*, 2019.

[36] D. Q. Huynh, "Metrics for 3d rotations: Comparison and analysis," *Journal of Mathematical Imaging and Vision*, vol. 35, no. 2, pp. 155–164, 2009.