Dynamic Regrasping with Asynchronous Vision Feedback using a Minimalist Robotic System

Yuxiang Ma, Edward Adelson
Massachusetts Institute of Technology
yxma20@mit.edu, adelson@csail.mit.edu

***Abstract*— Robotic manipulation has advanced significantly in quasi-static tasks like pick-and-place, peg-in-hole, and object reorientation, yet dynamic manipulation, exploiting motion-driven forces such as inertia and momentum, remains a critical challenge. While humans effortlessly perform dynamic regrasping by tossing and catching objects mid-air, robots typically require complex hardware (e.g., dexterous hands, bimanual setups) or computationally intensive planning with extrinsic contacts. Prior dynamic regrasping methods rely on specialized hardware, such as high-speed vision systems or multi-fingered hands, limiting their practicality. This work introduces a minimalist framework for dynamic regrasping using a single robotic arm, a Our approach decomposes the task into two phases: (1) a throwing phase, where the object is propelled into a ballistic trajectory, and (2) a catching phase, where computed torque control enables the gripper to dynamically regrasp the object mid-freefall. To accurately throw the object into the expected trajectory, we refine the throwing policy iteratively with asynchronous vision feedback. By integrating motion planning, computed torque control, and asynchronous visual tracking, we achieve dynamic regrasping without high-speed vision and expensive robot hardware. We present some preliminary experiments here to show the efficacy of the method as well as failure cases.**

I. INTRODUCTION

Robotic manipulation has emerged as a prominent research area in recent decades, with significant focus on non-dynamic tasks like pick-and-place, peg-in-hole assembly, and object reorientation. Traditional approaches to these tasks often rely on trajectory planning. Recent advances in perception systems, vision language models, and various robot learning algorithms have expanded the capabilities of robotic arms and grippers, enabling multi-contact, long-horizon operations such as kitchen sink cleaning, dishwasher loading, fabric folding, and so on [1–3].

Despite these advancements, robotic manipulation remains far inferior to human dexterity. A critical limitation lies in the inability of robots to perform dynamic tasks - actions that take advantage of motion-driven forces (e.g., inertia, momentum). This gap stems from a reliance on quasi-static assumptions, where interactions are slow and position-controlled. Dynamic manipulation exploits object dynamics to achieve high-speed motions and extend workspace boundaries, enabling tasks like throwing with precision, catching moving objects, or striking balls. Such capabilities are critical for sports applications, where robots must handle fast-moving objects and sometimes react to accidental events, and for industrial-automation scenarios that demand rapid, contact-rich workflows.

Humans, on the other hand, are able to manipulation objects and tools in a highly dynamic and reactive manner. And one key ability in human daily activities is regrasping an object to get a better grip or orientation of objects. Humans can easily perform regrasping by tossing and catching the object in a short interval, as shown in Fig. 1. However, robotic regrasping remains challenging, often requiring complex hardware (e.g., dexterous hands, bimanual setups) or extrinsic contacts, resulting in quasi-static operations. Prior work in dynamic regrasping, such as high-speed multifingered hands with kHz-rate vision systems [4, 5] or computed torque control with high-speed tracking [6], relies on expensive, specialized hardware. In contrast, we propose a minimalist approach using a single arm, a parallel-jaw gripper, and a standard RGB camera. By integrating motion planning, torque control, and asynchronous vision feedback, our method achieves dynamic regrasping without costly hardware, advancing toward human-like adaptability in unstructured environments.

Our proposed dynamic regrasping process can be decomposed into a throwing phase, at the end of which the object is released into mid-air, and a catching phase, during which the object follows the free-falling trajectory while the robot gripper reaches and regrasps the object, as shown in Fig. 1. The approach is composed of motion planning, computed torque control, and asynchronous visual feedback, presented in Section III. In Section IV, we elaborate on the experiment setup and experimental results, including discussions about failure cases. Lastly, we conclude the approach and preliminary results.

II. RELATED WORK

Regrasping, the process of repositioning an object within a robotic gripper, has evolved through diverse hardware and algorithmic approaches, each addressing specific challenges in stability, speed, and adaptability. The most trival approach is the pick-and-place method, which occupies a lot of space and is not efficient.

The robotic regrasping literature can be divided into four categories. First, bimanual systems employ two robotic arms to transfer objects between grippers, enabling regrasping through handovers [7–10]. Though this avoids the complexity of in-hand manipulation, it introduces hardware costs and two-agent collaboration challenges. Second, multi-finger hands allow a third or fourth finger to switch contact

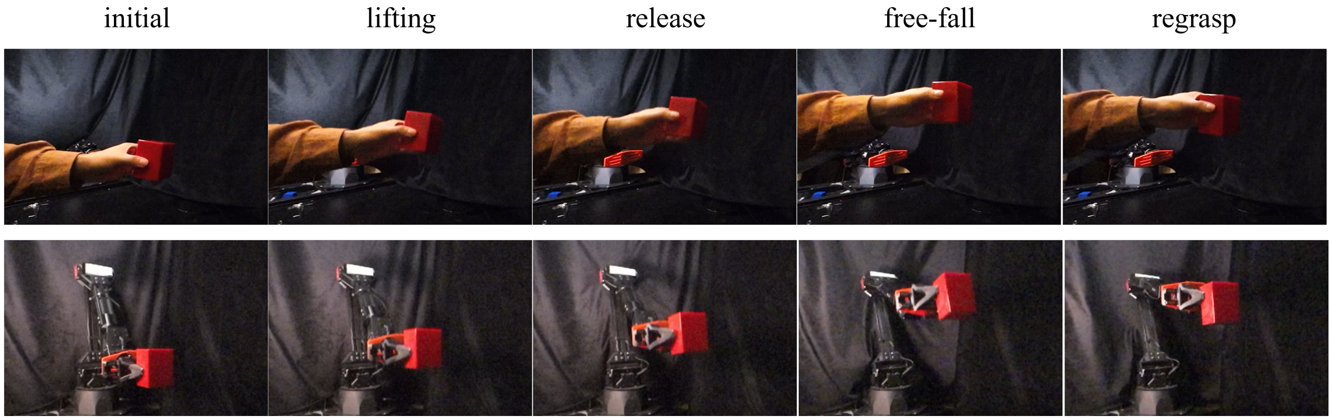

Fig. 1. Snapshots of a human hand and a robotic gripper performing dynamic regrasping on a red cube.

points while maintaining force closure on the object [11–15]. Expensive dexterous hands are needed for the quasi-static finger gaiting approach, and the corresponding control policies often require plenty of effort and resources to train or plan. Third, it has been shown that extrinsic contact can help to stabilize the object while repositioning the gripper's contact points [16, 17]. While effective in structured environments, this method depends heavily on predefined contact geometries and may struggle in dynamic or cluttered settings. Lastly, dynamic regrasping distinguishes itself from the above methods, by foregoing the quasi-static assumption of interaction. Instead, it utilizes object and robot dynamics to switch grasp poses in an agile way.

Unlike humans, who can perform throwing and catching with little effort, dynamic regrasping is non-trivial for robots. There only have been a few works using dynamical movements to switch between grasp poses. Furukawa et al. developed a dynamic regrasping system with a high-speed multifingered hand and a high-speed vision system at the rate of 1 kHz, which focuses mainly on motion planning of throwing and catching strategy [4]. A similar regrasping approach on a two-dimensional plane using a high-speed robot hand and high-speed vision is presented in [5]. The above two methods replies on a multi-dof hand to catch the object more easily, while Sintov and Shapiro proposed a dynamic regrasping algorithm based on both motion-planning and computed torque control, which works with a simple two-jaw gripper but still needs high-speed visual tracking. The existing work all have very high hardware requirements, which greatly limit the potential applications. In this work, we present a dynamic regrasping method with minimal hardware requirements. We show the efficacy of our algorithm using a single arm with a parallel-jaw gripper and a standard RGB camera.

III. Approach

Our proposed approach is composed of motion planning, computed torque control, and asynchronous visual feedback. A workflow of the proposed dynamic regrasping method is shown in Fig. 2.

A. Motion planning

The dynamic regrasping process, as shown in Fig. 1, can be decomposed into two phases. During the first throwing phase, the robot arm lifts the object to a specified speed and then releases the object. In the second catching phase, the gripper adjusts the pre-grasping poses and approaches to a target position to grasp the object. The target position is set to be the highest point of the object's free-falling trajectory, which makes visual feedback and error analysis easier.

During the throwing phase, the robotic arm imposes a desired velocity to the object, ensuring it follows a predictable ballistic trajectory after release. The process begins with trajectory planning, where our motion planner calculates the required release velocity to position the object's apex—the highest point in its parabolic path within the robot's workspace. Actuators then execute a rapid, dynamic motion to accelerate the object to the predetermined velocity profile. Upon reaching the target speed, the gripper releases the object at a precisely timed instant, initiating its free fall. This phase prioritizes repeatability and accuracy in velocity delivery, as errors in the release object states propagate during free flight.

The catching phase focuses on regrasping the object at the apex of its trajectory, where its instantaneous vertical velocity approaches zero. This strategic choice simplifies synchronization by eliminating the need to match lateral or vertical velocities during contact. As the object ascends post-release, the gripper initiates a closed-loop tracking routine, combining vision-based pose estimation (e.g., via high-frame-rate cameras or motion capture systems) with predictive models of ballistic motion to anticipate the apex location. Concurrently, the robotic arm reorients the gripper to align with the object's orientation at the apex. Once the object reaches the apex, the gripper executes a regrasp maneuver, capitalizing on the momentary kinematic stability of the object to minimize contact forces and slip risks.

Theoretically, the gripper is able to regrasp the object at any point during the fall. However, grasping the object at the non-apex point introduces a speed-matching problem

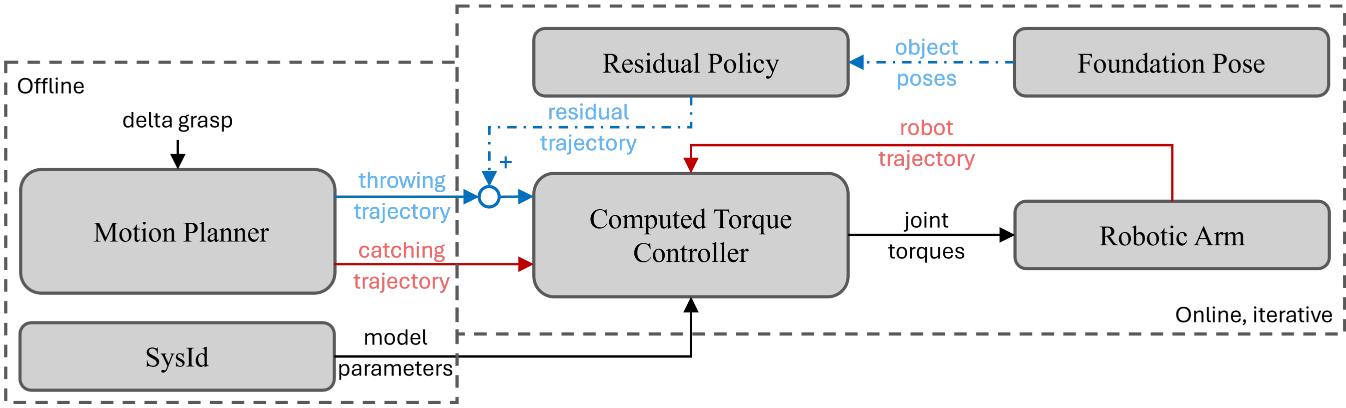

Fig. 2. Workflow of the proposed dynamic regrasping approach. Blue arrows show processes related to the throwing phase, and dashed blue arrows stand for asynchronous vision feedback. Red arrows show processes related to the catching phase.

between the gripper and the object, like in [6]. The non-zero object speed introduces more cumulative errors in the object trajectory and also makes error analysis harder. Meanwhile, in this agile manipulation scenario, measuring speed is much more challenging than measuring positions and orientations, which actually stopped [6] from deploying their policy on real hardware.

The details of the motion planner are shown in Algorithm 1. The desired ballistic object trajectory $\boldsymbol{x}_d^O(t)$ only has two independent factors, the initial position $\boldsymbol{x}_d^O(0)$ and initial vertical speed $v_{O,d}(0)$, which can be determined manually according to the robotic arm's workspace. First, end effector trajectories are computed for the two phases, based on the object trajectory $\boldsymbol{x}_d^O(t)$ and the desired delta grasp $\Delta \boldsymbol{x}_{EE}^O$. The generated trajectory includes positions, velocities, and accelerations. Then, the end effector trajectories are further converted into joint-level commands at every time step, which will be fed to the computed torque controller.

Algorithm 1 Motion plan for dynamic regrasping

Input: The desired object trajectory $\boldsymbol{x}_d^O(t)$, and the grasp change $\Delta \boldsymbol{x}_{EE}^O$.
Output: The desired joint trajectory and its derivatives $\boldsymbol{q}_d(t), \dot{\boldsymbol{q}}_d(t), \ddot{\boldsymbol{q}}_d(t)$.

1: $\boldsymbol{x}_d^{EE}(t), \boldsymbol{v}_d^{EE}(t), \boldsymbol{a}_d^{EE}(t) = \text{ee\_traj}(\boldsymbol{x}_d^O(t), \Delta \boldsymbol{x}_{EE}^O)$
2: **for** all time steps **do**
3: $\quad q_d = \text{inverse\_kinematics}(\boldsymbol{x}_d^{EE})$
4: $\quad \dot{q}_d = J^\dagger \boldsymbol{v}_d^{EE}$
5: $\quad \ddot{q}_d = J^\dagger (\boldsymbol{a}_d^{EE} - \dot{J}\dot{q}_d)$
6: **end for**

B. Computed torque control

The dynamic regrasping framework employs computed torque control to achieve precise trajectory tracking during both throwing and catching phases. This method leverages the robot's dynamic model to compute feedforward torque commands that compensate for inertial, Coriolis, and gravitational forces, enabling accurate tracking of time-varying trajectories under dynamic conditions.

The robot arm's rigid-body dynamics are modeled :

$$H(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q}, \boldsymbol{q})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{u} \tag{1}$$

where $H, C, \boldsymbol{g}$ represent the inertia matrix, centripetal and Coriolis term, and gravity term. In a real system, these values are hard to estimate in advance because some terms, like inertia and friction, might vary in different scenarios. Therefore, we apply model identification to estimate the parameters in the dynamic terms beforehand, and we assume the parameters remain the same during the whole task. We use the same symbol $\tilde{H}, \tilde{C}, \tilde{\boldsymbol{g}}$ to stand for the estimated terms.

The computed torque control scheme applied to the system is as follows [6]:

$$\boldsymbol{u} = \tilde{H}\left(\ddot{\boldsymbol{q}}_d + K_d(\dot{\boldsymbol{q}}_d - \dot{\boldsymbol{q}}) + K_p(\boldsymbol{q}_d - \boldsymbol{q})\right) + \tilde{C}\dot{\boldsymbol{q}} + \tilde{\boldsymbol{g}} \tag{2}$$

where K_p, K_d are diagonal gain matrices for proportional and derivative terms, and the desired configuration, velocity, and acceleration $\boldsymbol{q}_d(t), \dot{\boldsymbol{q}}_d(t), \ddot{\boldsymbol{q}}_d(t)$ are generated in the former motion planning step. The closed-loop system guarantees trajectory convergence, assuming the estimated dynamic model is accurate [6].

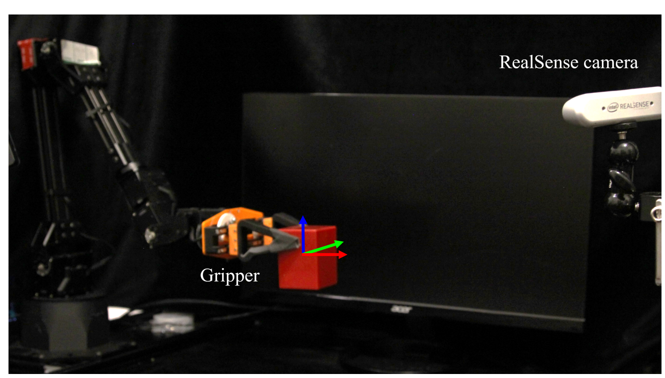

Fig. 3. Experimental setup of the dynamic regrasping algorithm

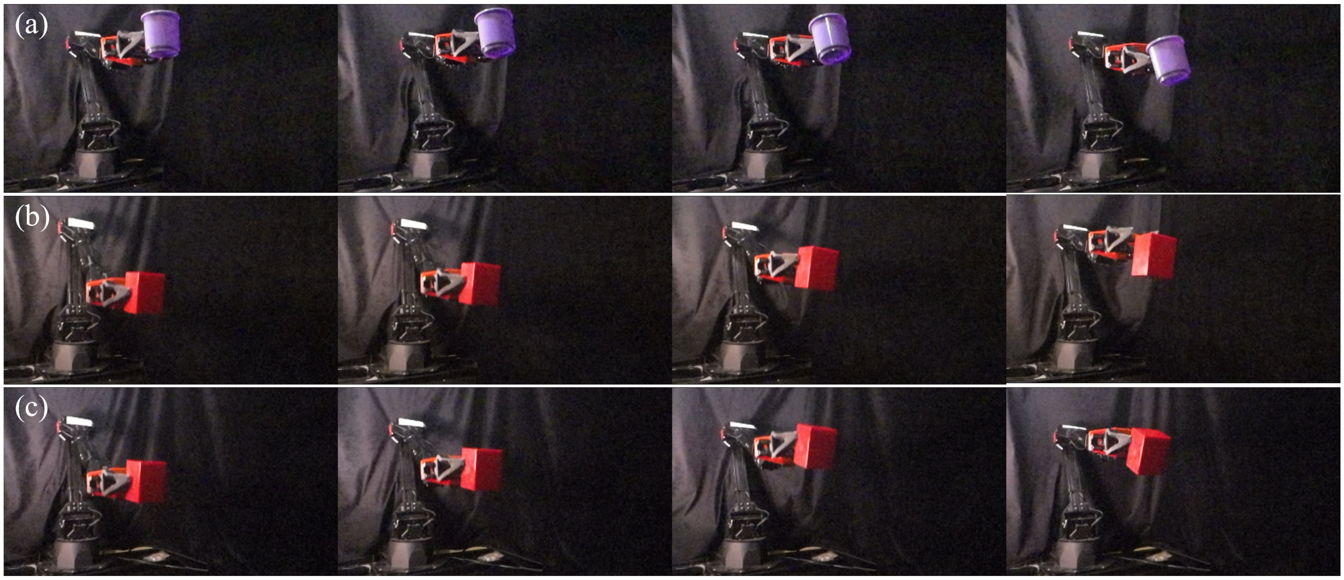

Fig. 4. Failure cases analysis. Failing reasons: (a) the cup drifted along the x axis; (b) the box did not reach the target height along the z axis; (c) the object drifted along the x axis and rotated along the y axis.

C. Asynchronous vision feedback

Our dynamic regrasping strategy throws the object towards a target apex point, where the object is regained by the gripper. Computed torque control can make sure that the planned trajectory is executed. However, the throwing policy is very sensitive to both hardware errors and the object properties, such as inertia, geometry, and friction, which vary with objects and initial grasping positions. And some of the properties are hard to characterize. Therefore, we develop an asynchronous feedback algorithm to improve the throwing policy.

We use FoundationPose [18] to track the initial and apex poses of the object ($\boldsymbol{X}^O(0), \boldsymbol{X}^O(t_{\text{apex}})$). The two poses are chosen because the object has a non-zero velocity elsewhere, leading to blurry images and adding noise to the tracking. The error in the throwing process is

$$\boldsymbol{e}_{\text{throw}} = \boldsymbol{X}^O(t_{\text{apex}})^{-1}\boldsymbol{X}^O(0)\boldsymbol{X}^O_d(0)^{-1}\boldsymbol{X}^O_d(t_{\text{apex}}).$$

In our experimental setup, the main error sources are the translation along the x and z axes and the y-axis rotation. The setup and coordinate frame definition is shown in 3. According to the amount error, we iteratively generate new throwing trajectories by combining a residual policy. For most common objects, i.e. with regular shape and inertia distribution, we are able to achieve successful grasps within 5-10 iterations.

IV. Results

Our experiment platform is shown in Fig. 3.The hardware includes a RealSense D435 RGBD camera (only RGB images are used), a ViperX 6-DoF robot arm, and a parallel-jaw gripper originally designed for ALOHA 2 [19]. The video is processed with FoundationPose [18]. We show both successful and failure cases for several objects here.

Some failure cases are shown in Fig. 4 with reasons described in the caption. Although successful regrasps can usually be achieved in several iterations, the converged policies still fail sometimes due to errors in the initial setup. Among different trials, the initial grasp spots change, which affects the policy performance. Another reason is that the ALOHA 2 gripper is too compliant and struggles to make stable grasps, which also adds sensitivity to the policy.

V. Conclusion

This work demonstrates a minimalist algorithm for dynamic regrasping using a single robotic arm and parallel-jaw gripper, eliminating the need for high-speed vision systems, dexterous hands, or bimanual setups. By decomposing the task into a throwing step and a catching step, we leverage computed torque control for precise trajectory execution and asynchronous vision feedback to iteratively tune the throwing policy. Preliminary experiments validate the approach's ability to handle variations in object properties, though challenges remain in the sensitivity to initial grasping conditions. This work advances robotic dexterity toward human-like agility, offering a cost-effective solution for applications requiring rapid, contact-rich manipulation, such as industrial automation, sports robotics, and disaster response.

Future Work includes: (1) evaluating and compensating for different initial grasping conditions ; (2) redesigning the gripper transmission mechanism and finger configurations to offer sturdier grasps; (3) evaluating the policy error in desired delta grasp on more objects; (4) introducing learning-based approaches, such as diffusion policy, and/or simulation to better tune the throwing policy.

References

[1] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning fine-grained bimanual manipulation with low-cost hardware," *arXiv preprint arXiv:2304.13705*, 2023.

[2] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song, "Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots," *arXiv preprint arXiv:2402.10329*, 2024.

[3] J. Zhang, J. Huang, S. Jin, and S. Lu, "Vision-language models for vision tasks: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[4] N. Furukawa, A. Namiki, S. Taku, and M. Ishikawa, "Dynamic regrasping using a high-speed multifingered hand and a high-speed vision system," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* IEEE, 2006, pp. 181–187.

[5] R. Higo, T. Senoo, and M. Ishikawa, "Dynamic in-hand regrasping using a high-speed robot hand and high-speed vision," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 9796–9801, 2020.

[6] A. Sintov and A. Shapiro, "Dynamic regrasping by in-hand orienting of grasped objects using non-dexterous robotic grippers," *Robotics and computer-integrated manufacturing*, vol. 50, pp. 114–131, 2018.

[7] J.-P. Saut, M. Gharbi, J. Cortés, D. Sidobre, and T. Siméon, "Planning pick-and-place tasks with two-hand regrasping," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 4528–4533.

[8] Z.-Y. Chiu, F. Richter, E. K. Funk, R. K. Orosco, and M. C. Yip, "Bimanual regrasping for suture needles using reinforcement learning for rapid motion planning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 7737–7743.

[9] B. Balaguer and S. Carpin, "Bimanual regrasping from unimanual machine learning," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 3264–3270.

[10] 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.

[11] K. Tahara, K. Maruta, A. Kawamura, and M. Yamamoto, "Externally sensorless dynamic regrasping and manipulation by a triple-fingered robotic hand with torsional fingertip joints," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 3252–3257.

[12] B. Sundaralingam and T. Hermans, "Geometric in-hand regrasp planning: Alternating optimization of finger gaits and in-grasp manipulation," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 231–238.

[13] T. Chen, M. Tippur, S. Wu, V. Kumar, E. Adelson, and P. Agrawal, "Visual dexterity: In-hand reorientation of novel and complex object shapes," *Science Robotics*, vol. 8, no. 84, p. eadc9244, 2023.

[14] Z.-H. Yin, B. Huang, Y. Qin, Q. Chen, and X. Wang, "Rotating without seeing: Towards in-hand dexterity through touch," *arXiv preprint arXiv:2303.10880*, 2023.

[15] T. Pang, H. T. Suh, L. Yang, and R. Tedrake, "Global planning for contact-rich manipulation via local smoothing of quasi-dynamic contact models," *IEEE Transactions on robotics*, vol. 39, no. 6, pp. 4691–4711, 2023.

[16] N. C. Dafle, A. Rodriguez, R. Paolini, B. Tang, S. S. Srinivasa, M. Erdmann, M. T. Mason, I. Lundberg, H. Staab, and T. Fuhlbrigge, "Regrasping objects using extrinsic dexterity," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 2560–2560.

[17] N. Chavan-Dafie and A. Rodriguez, "Regrasping by fixtureless fixturing," in *2018 IEEE 14th international conference on automation science and engineering (CASE)*. IEEE, 2018, pp. 122–129.

[18] B. Wen, W. Yang, J. Kautz, and S. Birchfield, "Foundationpose: Unified 6d pose estimation and tracking of novel objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 17 868–17 879.

[19] J. Aldaco, T. Armstrong, R. Baruch, J. Bingham, S. Chan, K. Draper, D. Dwibedi, C. Finn, P. Florence, S. Goodrich, *et al.*, "Aloha 2: An enhanced low-cost hardware for bimanual teleoperation," *arXiv preprint arXiv:2405.02292*, 2024.