Learning and Optimizing Bimanual Regrasping Behaviors

Benjamin Balaguer and Stefano Carpin
School of Engineering, University of California, Merced, CA, USA

Although unimanual regrasping has been studied extensively, either by regrasping in-hand or by placing the object on a surface for each regrasping phase, bimanual regrasping has been less popular. Unimanual regrasping algorithms are inadequate for human-like service robots, however, since they require complex high DoFs robotic hands or perform time-consuming operations (e.g., placing the object on a table to regrasp it). Bimanual regrasping, which consists in regrasping the object in the air using two manipulators, can alleviate the problems found in unimanual regrasping techniques and is used frequently by humans but rarely by robots. Bimanual regrasping is most often exploited when an object in one of the manipulator's reachability space needs to be moved to a location in the other manipulator's reachability space. Even though bimanual regrasping has seen little attention, it is a crucially beneficial behavior for service robots since it not only saves time performing certain tasks but also mimics human behavior. Additionally, bimanual regrasping does not require complex end-effectors and does not impose restrictions on the manipulators used.

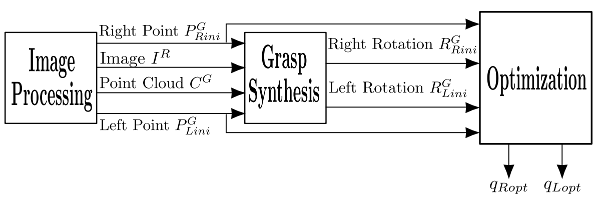

Fig. 1. High-level overview of proposed bimanual regrasping algorithm.

As shown in Fig. 1, the algorithm is composed of three components: Image Processing, Grasp Synthesis, and Optimization. The Image Processing, the purpose of which is to find two good grasping points in image space, exploits a stereo camera along with a state-of-the-art supervised machine learning algorithm that we modify to accommodate bimanual regrasping. The two points correspond to the points that each manipulator will use when regrasping. We use a single stereo image as input, I^R, from which we can calculate the corresponding point cloud, C^G, thanks to stereo vision. Using the image, a binary classifier exploiting logistic regression extracts two good grasping points, one for each arm, and converts them to Cartesian coordinates, P^G_{Rini} and P^G_{Lini}. The Grasp Synthesis component takes I^R, C^G, P^G_{Rini}, and P^G_{Lini} as input and outputs appropriate orientations for the right and left end-effectors, R^G_{Rini} and R^G_{Lini}, to grasp the object at the points P^G_{Rini} and P^G_{Lini}. This process works with a supervised learning unimanual grasp planer that combines multi-class SVMs, image moments, orthogonal regression, and nearest neighbor searches. The Optimization component searches the reachability spaces of the arms to find the most efficient regrasping configurations, outputting the manipulators' configuration, q_{Ropt} and q_{Lopt}. Specifically, the optimization performs a six-dimensional search space (i.e., manipulator's X, Y, Z, Roll, Pitch, Yaw) with the Nelder-Mead algorithm, which does not require an objective function with a corresponding derivative. Indeed, we simply use a cost function that minimizes the manipulators' joint movements, consequently reducing the task's execution time.

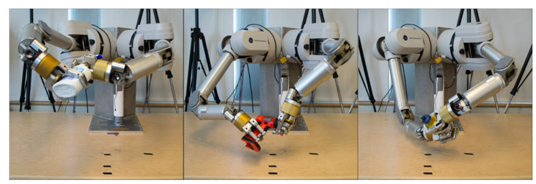

Fig. 2. Screenshots of our robot regrasping different objects.

For the experiments, 10 varying configurations of 4 objects were regrasped by our robot, consisting of a static torso with two 7 DoFs Barrett WAM arms and 4 DoFs Barrett hands. First, we manually dictate the grasping positions and orientations of the manipulators (i.e., P^G_{Rini}, P^G_{Lini}, R^G_{Rini} and R^G_{Lini}), consequently removing potential errors from the Image Processing and Grasp Synthesis components and investigating the Optimization component on its own. Being successful 87.5% of the time, with the cause for every failure being positional errors from the manipulator, it is clear that the Optimization component yields valid results. Second, we analyze the end-to-end algorithm by incorporating the Image Processing and Grasp Synthesis components back into the algorithm, a few snapshots of which are shown in Fig. 2. Overall, the end-to-end algorithm performed very well, successfully completing 75% of the experiments. The majority of the errors were attributed to the Grasp Synthesis component failing to provide good orientations for one of the manipulators. We additionally compared the Optimization algorithm with four grid-based search algorithms (Brute Force, Random Grid Search, Reachability Subspace, Hierarchical Search). The Optimization algorithm is not only extremely efficient, providing solutions more than six times faster than the second-fastest algorithm, but also competitive with the Brute Force approach. In fact, the Optimization algorithm finds a better solution than Brute Force and the other algorithms 84.61% and 100% of the time, respectively. The Brute Force algorithm is not a viable solution, however, because it takes 72.5 hours on average. Conversely, our entire algorithm, including the Image Processing, Grasp Synthesis, and Optimization, runs in 329ms.