

Model Predictive Control for Fluid Human-to-Robot Handovers

Wei Yang^{†*}, Balakumar Sundaralingam^{†*}, Chris Paxton^{†*}, Iretiayo Akinola[†],
Yu-Wei Chao[†], Maya Cakmak^{†,‡}, Dieter Fox^{†,‡}

Abstract—Human-robot handover is a fundamental yet challenging task in human-robot interaction and collaboration. Recently, remarkable progressions have been made in human-to-robot handovers of unknown objects by using learning-based grasp generators. However, how to responsively generate smooth motions to take an object from a human is still an open question. Specifically, planning motions that take human comfort into account is not a part of the human-robot handover process in most prior works. In this paper, we propose to generate smooth motions via an efficient model-predictive control (MPC) framework that integrates perception and complex domain-specific constraints into the optimization problem. We introduce a learning-based grasp reachability model to select candidate grasps which maximize the robot’s manipulability, giving it more freedom to satisfy these constraints. Finally, we integrate a neural net force/torque classifier that detects contact events from noisy data. We conducted human-to-robot handover experiments on a diverse set of objects with several users ($N = 4$) and performed a systematic evaluation of each module. The study shows that the users preferred our MPC approach over the baseline system by a large margin.

I. INTRODUCTION

Recently, remarkable advances have been made in generic human-to-robot handovers of arbitrary graspable objects [1], [2] thanks to development of learning-based grasp planners for unknown objects [3], [4]. In these works, grasps are selected based on grasp stability or task constraints, and the robot is driven towards the end-effector pose to complete the handover process.

While promising results have been demonstrated using this pipeline, how to select the best grasp and associated motion to execute is still an open challenge. Prior works select grasps based on grasp quality [1], [2], which neither guarantees the reachability and manipulability of the grasp nor the smoothness of the motion generated to reach the grasp. In this paper, we propose to learn a neural network to predict the manipulability of a grasp pose for motion-aware grasp selection. Together with the grasp stability, it enables us to select a stable grasp which is also reachable and manipulable. Additionally, compared with using inverse kinematics to check the reachability of each grasp sequentially, our model can evaluate the reachability and the manipulability of a batch of grasps in parallel, which is more efficient for real-time robotics tasks such as human-to-robot handovers.

Another challenge is how to plan smooth and natural motion for handovers. Given a selected grasp, the robot is

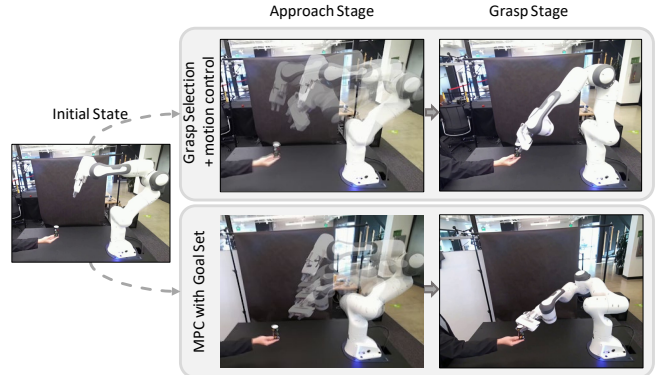


Fig. 1: How to responsively generate smooth motions for handovers is an open question. The pipeline of planning motion after selecting a grasp, such as in [2], does not always guarantee smooth motions (top). We propose an approach to use Model-Predictive Control (MPC) for motion generation with either a single selected grasp or a set of grasps (bottom), which results in a better, more fluid user experience. More results and videos are available at <https://sites.google.com/nvidia.com/mpc-for-handover>.

usually driven towards the end-effector pose by local (non-global) policies such as Riemannian Motion Policies [5] as in [2], [6] or visual servoing [1]. However, natural motions are not just accelerations towards a particular end-effector pose; they consider how to reach there while obeying complex task constraints. In addition, without planned motions, the robot might choose a grasp that leads to slower or less efficient motions. In this work, we employ a highly-parallelized Stochastic Model Predictive Control (MPC) framework [7] for real-time motion planning. This framework also allows us to incorporate a range of complex task-specific constraints proposed in [6] such as preventing hand occlusions.

To bridge the gap between grasp selection and motion planning, we further extend the MPC framework by incorporating the grasp selection procedure into the optimization given a set of candidate grasps. This unified MPC framework enables the system to select the grasp and plan motion simultaneously, while taking grasp reachability and domain specific constraints into account.

Finally, during the physical handover phase, *i.e.*, from the first contact of the receiver’s hand on the object till the release of the object, force feedback can be used to decide when to release the object (robot-to-human) or when to grasp the object (human-to-robot) [8]. However, the raw force/torque sensor readings are often noisy and hard to interpret directly [9]. To handle this, we collected a dataset of humans interacting with a moving robot, and trained a neural net to detect when the contact between hand and gripper happens. The detection is then coupled with vision during the physical handover phase to trigger the robot grasp action.

* Equal contribution.

[†]NVIDIA, USA {weiy, balakumars, cpaxton, iakinola, ychao, mcakmak, dieterf}@nvidia.com

[‡]University of Washington, USA

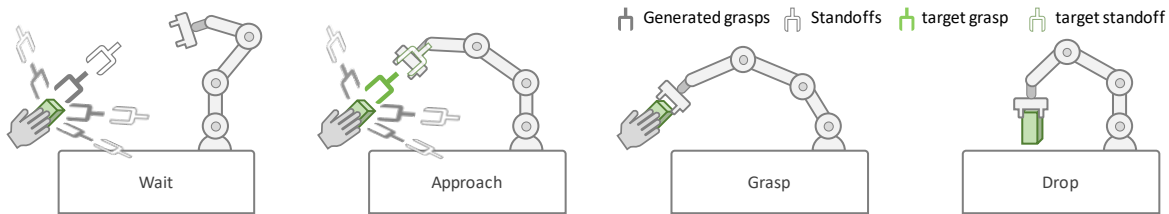


Fig. 2: Different stages of human-to-robot handovers. The robot is idle until it detects a human holding an object (*Wait*). Then it approaches to a standoff position using MPC (*Approach*). Once the robot reaches the standoff pose, the robot grasps the object with a blocking policy (*Grasp*). Finally, the robot places the object to a preset position (*Drop*).

Our contributions are summarized as follows: 1) We integrate a stochastic MPC framework for efficient motion planning with complex task-specific objectives, and proposed a unified framework for grasp selection and motion planning. 2) We introduce a learned reachability/manipulability model to facilitate motion-aware grasp selection. 3) We collected a dataset and trained a neural net to detect the contact events for improving action triggering in the physical handover phase. 4) We evaluate handovers with the proposed systems on three objects from different handover locations and orientations, as well as dynamically changing the object orientation after the robot starts to move. We further conducted a user evaluation ($N = 4$) with a diverse set of household objects comparing the proposed system with the baseline.

II. RELATED WORK

Human-robot object handover is essential for assistive and collaborative robots in environments such as retirement homes and factories. Many efforts have been made in human-robot communication, grasp planning, perception, motion planning, and control to enable natural and fluent handovers [10]. Among them, grasp planning is a crucial component in pre-handover phase to ensure a success physical handover. For example, human gaze was used to determine the contact point on the object to direct a grasp in [11]. Cini *et al.* [12] chose grasps based on the object constraints and the receiver’s task. For human-to-robot handovers, object properties such as shape, function and safety were considered when planning grasp in [13]. Yang *et al.* [6] decided robot grasps based on how the object is holding by a human giver. Recent advances [1], [2] used deep-learning based grasp planners to generate grasps for unknown objects in human hand. These works selected the grasp to be executed based on the grasp quality or the stability associated with the grasps, which could not guarantee the selected grasps are all manipulable. Our reachability model directly evaluates the robot reachability/manipulability given a grasp and is complementary to the grasp stability and quality metrics.

Model predictive control (MPC) is being increasingly used for reactive motion generation in manipulators [7], [14], [15], [16], [17]. Kuindersma *et al.* [17] leveraged MPC to generate dynamic motions for humanoid robots, that can navigate varying terrains. Faroni *et al.* [14] explored leveraging MPC to dynamically slow down the robot when a human is in the workspace of the robot. Minniti *et al.* [15] used MPC in combination with an observer to handle contact with the environment such as contact during opening of doors. Kramer *et al.* [16] showed their approach avoiding

dynamic obstacles leveraging non-linear programming to solve MPC. Bhardwaj *et al.* [7] took a different approach from the above methods for MPC, specifically they explore leveraging sampling-based MPC for real-time reactive manipulator control. By leveraging sampling, their approach does not require cost terms to be smooth or differentiable. We leverage [7] in this paper to solve MPC for human-robot handovers as it does not require 1) extensive engineering effort to tune the many heuristics and 2) the cost terms to be smooth/differentiable.

III. TASK MODEL

Our high-level task model largely follows the one in [2], [6]. As illustrated in Fig. 2, there are four states the robot can be in: (1) waiting for human or object, (2) approaching the object standoff position, (3) grasping the object, and (4) placing the object when it is done.

(1) Wait. During this stage, the robot is idle. It will move to a home position, and not otherwise move around. This step will continue until the robot detects a human holding an object in the workspace.

(2) Approach. During this phase, the robot approaches the object and reaches a standoff pose from the object. This standoff pose is 15 cm from the final grasp on the object. Getting to this standoff position is the role of the MPC system discussed in Sec.V.

(3) Grasp. Once the robot reaches the standoff pose, we begin our grasping policy. Unlike the *Approach* or *Wait* policies, this is a *blocking* policy because as the robot gets closer to the object, body tracking, segmentation, and grasp estimation all become less reliable, as discussed in [2]. After moving the end-effector forward to the grasp position, the robot closes its gripper and retreats to the standoff pose.

We detect missed grasps by observing the distance between the gripper fingers after closing. After a failure grasp, the robot goes back to the *Approach* phase, and attempts to grasp the object again.

We trained a classifier given raw force/torque signal to detect the contact with the gripper. If the human pushes the object into the robot’s hand, the grasp motion will terminate early and the gripper will be closed. The force classifier is described in more detail in Sec. IV-C.

(4) Drop. If the robot has the object, it will place it on the table at a preset position.

IV. PERCEPTION PIPELINE

In this section, we first briefly review the modules we used for grasp generation in Section IV-A. Then we present

our reachability model which measures the manipulability of a grasp pose in IV-B. Finally, a contact event detector is introduced in IV-C to improve the robustness of the system when the contact happens (e.g., when humans push/pull the object into/from the robot gripper or when the robot gripper touches the object) or when vision modules fail.

A. Grasp Generation

Our system generates a set of candidate grasps for human-to-robot handovers by following [2]. Specifically, we track the robot pose by DART [18] and the human body by the Azure body tracking SDK at 15 Hz, then segment the hand and the object in hand by a pretrained hand segmentation model at 9 Hz. Given the segmented object point cloud, we use a temporally consistent version of 6-DoF GraspNet [2], [4] to generate grasps at 5 Hz.

B. Grasp Reachability Model

While GraspNet [4] estimates a score for each predicted grasp to measure the grasp stability, not all the generated stable grasps can be realized by the robot. To reduce the set of grasps being considered, we introduce a learned ranking/cost function that predicts the manipulability of grasp pose without an intermediate Inverse Kinematics (IK) step. The manipulability of a grasp pose is a measure of how far the robot is to a singular configuration [19]. This metric is particularly valuable when trying to reach a moving target; a highly manipulable grasp is likely to stay reachable as the object moves. While previous works have developed different methods for encoding reachability of grasps, their approaches use simple heuristics [20], proxy representations [21] or require interpolation of precomputed data [22], [23]. Our approach directly learns the manipulability metric from data to quickly predict scores.

Concretely, our manipulability metric \mathcal{M} is defined as $|J^T J|$ where J is the Jacobian of the IK of the grasp g . If no IK solution exists, we use the negative twist-distance between the closest achievable end-effector pose and the target:

$$\mathcal{M}(g) = \begin{cases} |J^T J|, & \text{if IK exists,} \\ -\text{dist}_{\text{twist}}(g, e), & \text{otherwise,} \end{cases}$$

where $|\cdot|$ is the determinant operator, $\text{dist}_{\text{twist}}(g, e)$ is the twist distance between grasp g and the closest achievable pose of the end-effector pose e .

We further incorporate joint limits of the robot into the manipulability score, we use a joint-limit performance metric derived in previous work [24] to form a weighting matrix. The metric for each joint $\theta = \{\theta^i : i = 1, 2, \dots, \text{dof}\}$:

$$\mathcal{P}_{\text{joint.Limit}}(\theta) = \frac{(\theta_{\max} - \theta_{\min})^2 * (2\theta - \theta_{\min} - \theta_{\max})}{4(\theta_{\max} - \theta)^2 * (\theta - \theta_{\min})^2}. \quad (1)$$

Note that this metric is minimum (0) in the middle for the joint ranges and maximum (infinity) at the extremes of the

joint limits. The weighting matrix is a diagonal matrix with entry on the diagonal corresponding to each joint given as

$$\mathcal{W}^{ii} = \frac{1}{\sqrt{1 + |\mathcal{P}_{\text{joint.Limit}}(\theta^i)|}}. \quad (2)$$

The joint-aware manipulability is then computed as $|J^T \mathcal{W} J|$ for grasp poses that have IK solutions. This is used to generate the dataset that is used to train the joint-limit-aware reachability model.

To facilitate training, we generated a dataset of grasp poses and the corresponding manipulability scores, and trained a multi-layer perception with the mean squared error loss. To account for IK redundancies during data generation, the manipulability of a given grasp pose is the maximum of the manipulability scores over the IK solutions that realize that grasp. The input of the multilayer perceptron (MLP) is the 6-DOF grasp pose $(x, y, z, \text{roll}, \text{pitch}, \text{yaw}) \in \mathbb{R}^6$ and the output is the manipulability score. The median inference time of the trained model is 1.3 milliseconds in experiments.

C. Physical Contact Detection

To better coordinate the timing for the physical handover phase, when the visual perception becomes noisier due to the close proximity between the hand and the robot gripper, we trained a feed-forward neural network to detect the contact event between the hand/object and the robot gripper. The network takes the raw sensor data including joint velocities, efforts, force, and torque for the last $T = 5$ steps as input, and predicts a probability of a contact event.

Our model is a temporal convolutional neural net which encodes each individual timestep through a two-layer MLP, then performs two 1D convolutions over history, before another MLP (with dropout) predicts a single output indicating if a force event was detected within the window.

We collected a physical contact dataset mimicking the handover procedures: we moved the robot gripper to a random position in the workspace. When the robot was about to reach the target pose, we pushed/pulled on the robot gripper, and pressed a key to label the physical contact. We recorded joint positions, velocities, efforts, forces, and torques during the whole procedure. With $T = 5$, our dataset consists of 20K samples with about 8% positive samples.

Our model achieved 93.6% accuracy on a held-out non-overlapping test set of 5k examples collected.

V. MODEL PREDICTIVE CONTROL FOR GRASPING

For human-to-robot handover, the robot needs to move to one of the grasp poses $X_{g \in G}$, and then grasp the object from the human, as shown in Fig. 2. During this robot motion, we also want to encode heuristics that promote fluid human-robot handovers, specifically: 1) encouraging the robot's gripper to move in straight lines; 2) avoiding collisions between robot and human; 3) having the human hand always in view; and 4) reducing jerk of the robot during motion. As some heuristics are in the joint space and some in Cartesian space of the links, we formulate the problem as a kinematic joint space trajectory optimization problem [25],

[26] (Sec. V-A). We then solve the formulated problem in real time leveraging stochastic MPC [7] (Sec. V-B).

A. Trajectory Optimization Formulation

Given the robot’s joint position θ_0 and velocity $\dot{\theta}_0$ at timestep 0, we want to compute the joint accelerations $\ddot{\theta}_{t \in [0, H-1]}$ across the horizon H timesteps that minimizes cost terms $C(\cdot)$ while also satisfying constraints formulated below,

$$\min_{\ddot{\theta}_{t \in 0, H-1}} C_g(G, \theta_t) + C_{sl}(\theta_t, \dot{\theta}_t) \quad (3)$$

$$+ C_{manip} + C_{stop}, \quad (4)$$

$$\text{s.t. } S_e(\theta_t) < 0.0 \quad (5)$$

$$S_r(\theta_t) < 0.0 \quad (6)$$

$$\dot{\theta}_t = \dot{\theta}_{t-1} + \ddot{\theta}_t dt \quad (7)$$

$$\theta_t = \theta_{t-1} + \dot{\theta}_t dt \quad (8)$$

where (3) contains the cost terms formulated for human-robot handover, and (4) lists the manipulability and stop cost terms from Bhardwaj *et al.* [7] that aid MPC in avoiding local minima and overshooting respectively (refer [7] for more details). Constraints (5-6) are collision avoidance constraints that prevent the robot from colliding with the environment, human, and itself. Eq. (7-8) are euler integration equations to obtain joint position, and joint velocity from joint acceleration. We also have box constraints on the robot’s joint position, velocity, and acceleration to satisfy joint limits.

1) *Reaching Grasp Pose:* Given a target pose X_g and the current gripper pose $X_t = FK(\theta_t)$ computed using the forward kinematics of the robot at the current joint configuration θ_t , we define a pose distance metric $\text{dist}(X_t, X_g)$,

$$\text{dist}(X_t, X_g) = \|\alpha_1(I - {}^w R_g^T {}^w R_t)\|_2 + \|\alpha_2({}^w R_g^T {}^w d_t - {}^w R_g^T {}^w d_g)\|_2, \quad (9)$$

where ${}^w R_g$ and ${}^w R_t$ are the rotation matrices of pose X_g and X_t respectively. The translation vectors of X_g and X_t are represented by ${}^w d_g$ and ${}^w d_t$ respectively.

We can reach a given gripper pose by using the above distance metric as the goal cost $C_g(\cdot) = \text{dist}(\cdot)$. We can also optimize for reaching one pose from a set of grasp poses G by writing the goal cost as the distance between the current gripper pose X_t and the closest pose in the goal set,

$$C(\theta_t, G) = \min(\text{dist}(FK(\theta_t), X_{g \in G})). \quad (10)$$

The above formulation of reaching a goal set enables MPC to reason about all the grasp poses inside the optimization while also considering all the other heuristics and constraints. This would enable MPC to move fluidly to another grasp from the current grasp given new constraints (*e.g.*, human hand moves to current grasp region). We discuss results from both these formulations in Sec. VI.

2) *Straight Line Cost:* We found the robot to take very circular trajectories in the Cartesian space due to the presence of many revolute joints in the manipulator. These circular trajectories are hard to predict by users, especially those with minimal domain knowledge. We hence formulate a cost term that penalizes gripper’s linear velocity in directions that

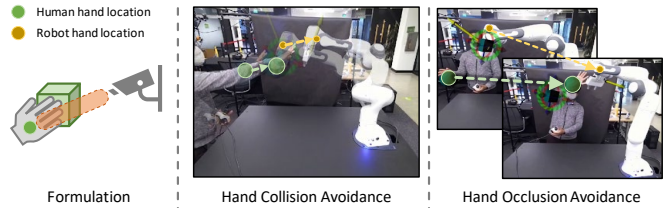


Fig. 3: Left: the formulation of the hand collision and occlusion avoidance. The robot maintains an end-effector position marked by the green cube until the human hand either comes near the robot (collision, middle) or gets occluded by the robot (right).

are not parallel to the vector connecting the current gripper position to the goal position $\hat{d}_g = \frac{\hat{d}_g - d_t}{\|\hat{d}_g - d_t\|}$. We compute the gripper’s current velocity leveraging the kinematic Jacobian $J(\theta_t)$ at the current joint configuration θ_t and the current joint velocity $\dot{\theta}_t$ as $\hat{d}_t = J(\theta_t)\dot{\theta}_t$ and then normalize this vector to get the current gripper motion direction \hat{d}_t . The cost is then written as,

$$C(\hat{d}_t, \hat{d}_g) = 1 - \hat{d}_t \cdot \hat{d}_g. \quad (11)$$

This cost becomes zero when both vectors are parallel.

3) *Collision Avoidance:* During gripper motion, the robot needs to avoid colliding with the table, and also the human hand. In addition, we want to keep the hand from being occluded by the robot during the robot motion. To do so, we represent the table as a cuboid, the human hand as a sphere, and use the line connecting the camera’s origin with the human hand position to build a capsule as shown in Fig. 3 (left). We also represent the robot’s links with spheres similar to [7] and then compute the signed distance between the robot and the environment via an analytic function $S_e(\theta_t)$. We additionally avoid robot self collisions by using a neural network trained by [7]. The neural network outputs the signed distance¹ between the two closest links given a joint configuration θ_t . We then use this as $S_r(\theta_t)$ in Eq.(6). As demonstrated in Fig. 3, the robot avoids colliding with human hand and keeps the hand from occlusion.

B. Solving as Stochastic MPC

We solve the optimization problem by leveraging stochastic MPC (also known as sampling-based MPC) as it has shown to work with many cost terms [7]. Stochastic MPC optimizes by sampling action sequences for many particles from a distribution, rolling out these actions, computing the cost incurred by each particle, and then using these costs to update the distribution. By iterating through this process, given large number of particles, stochastic MPC can generate motions at 50-100 Hz which is comparable to gradient based MPC methods while having the benefit of not requiring differentiability of the cost terms or the dynamics [7]. Stochastic MPC cannot handle constraints, hence we setup the constraints in our optimization problem as cost terms with large weights which has shown to be sufficient for satisfying constraints on realistic settings [7], [27]. We also leverage this optimization problem for motion generation in other stages of the handover, *e.g.*, “Drop”,

¹Our signed distance functions output negative values when there is no collision and positive when there is a collision.

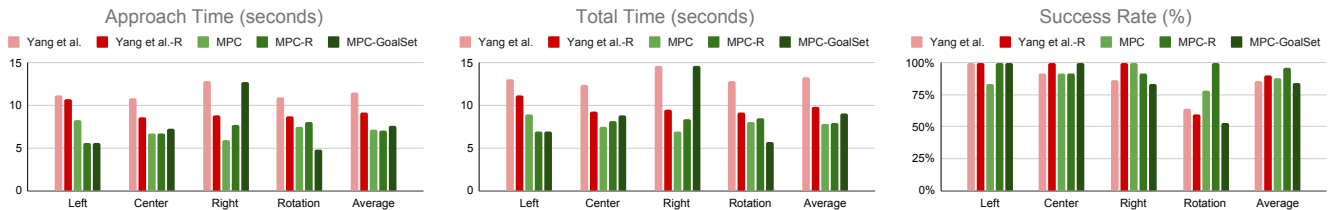


Fig. 4: Systematic evaluation. *-R denotes methods with the reachability model. Our MPC framework generates faster and smoother motions compared with Yang *et al.* [2].

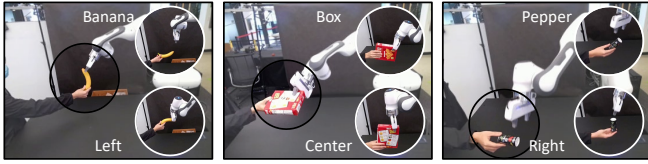


Fig. 5: Systematic evaluation on different handover locations. For each location, we handed over the object three times with three different ways of holding the object.

where we change the goal cost to be a $L2$ loss on the joint configuration. Our MPC uses a horizon of 2 seconds while optimizing the motion. All constraints have a weight of 5000. We set α_1 and α_2 in Eq. (9) as 70 and 220 respectively and 30 for the straight line cost weight. by starting with a small value and slowly increasing until the motion reached sufficient precision (a total of 30 minutes was spent to tune these parameters).

VI. SYSTEMATIC EVALUATION

System Setup. We conducted experiments with a Franka Emika Panda arm, and an externally mounted Azure Kinect RGBD camera with 1280×720 resolution. The system is distributed on 3 desktop computers with an additional realtime desktop for Franka control. Four NVIDIA RTX 2080 Ti and one 3090 GPUs were used.

Protocols and Metrics. We evaluated the system performance with three objects with different shapes and sizes, including a banana from the YCB object [28], a cracker box, and a pepper bottle, as shown in Fig. 5. For all the experiments in this section, we repeated the handover trials until three successful handovers have been recorded. Then we measured the system performance by computing the *success rate*, *approach time*, and the *total time* for a successful handover. We also recorded the *velocity*, *acceleration* and the *jerk* during the handover.

Baselines. We considered Yang *et al.* [2] as the baseline method. In addition to the grasp selection criterion proposed in [2], we use the learned reachability metric $R(X_g)$ described in Sec. IV-B to determine which grasps are reachable and restrict the set to ones that are highly manipulable:

$$C = \omega_s \min(S_g - S_{min}, 0) + \omega_{prev} d(X_g, X_{prev}) + \omega_{home} d(X_g, X_{home}) + \omega_R R(X_g), \quad (12)$$

where S_g and S_{min} is the score for grasp g and the minimum acceptable score, X_g and X_{prev} denote the pose of grasp g and the pose of previous chosen grasp respectively. X_{home} denotes the end-effector pose at the home position. $\omega_s, \omega_{prev}, \omega_{home}, \omega_R$ are weights. $d(\cdot, \cdot)$ is a distance metric with both position and rotation components.

We compared the above baselines with the following variations of the proposed approach: 1) *MPC/MPC-R*: using MPC for motion planning towards one selected grasp (without/with reachability metric); 2) *MPC-GoalSet*: using MPC to generate motion given a set of grasps. Noted that reachability metric is inherently embedded in the cost function Eq.(4).

A. Handover Reactivity

We first investigate the performance of our approach by rotating the object along it's standing axis 45 degrees after the robots starts moving. As shown in Fig. 4 (Rotation), all *MPC*-based systems have lower approach/total time for successful handovers since they can react quickly to the changing of the object orientation. The reachability model promotes lower approach time while maintaining a similar success rate for the baseline method [2], and improves the success rate for the *MPC* variant. We observe that while the *MPC-GoalSet* achieves the lowest approach time for successful handovers, it has a reduced success rate especially when handling changes in orientation. We suspect this to be because the number of grasps sufficiently reduces after the filtering by the reachability model (Sec. IV-B), and at least one grasp is consistent across time.

B. Handover Location

We further evaluate the efficiency of our approach in response to H2R handovers at different locations and different ways of holding objects. We handed over three objects at three locations: *left*, *center*, and *right*. For each location, we handed over the object three times with three ways of holding the object, as shown in Fig. 5. Results are reported in Fig. 4.

The *MPC* is comparable to Yang *et al.* [2], where we used *MPC* for motion planning while keeping the robot velocity (Fig. 6 left) and all the other modules the same. *MPC* reduces the approach time (11.5s to 7.1s) and total time (13.3s to 7.9s) and improve the success rate from 85.7% to 88.3%.

We also investigate integrating the reachability model with both Yang *et al.* [2] and *MPC* (denoted as *-R). In this setting, the reachability metric allows for better grasp selection. It improves the approach time of [2] by two seconds and also improves the success rate of MPC by 7.5%. This is because, with the reachability model, the robot tends to choose more reachable/manipulable grasp poses. Our *MPC-GoalSet* variant has slightly longer approach time and slightly lower success rate compared to *MPC/MPC-R*, but it achieves smoother motion with less jerk (see Fig. 6 and Sec. VI-C).

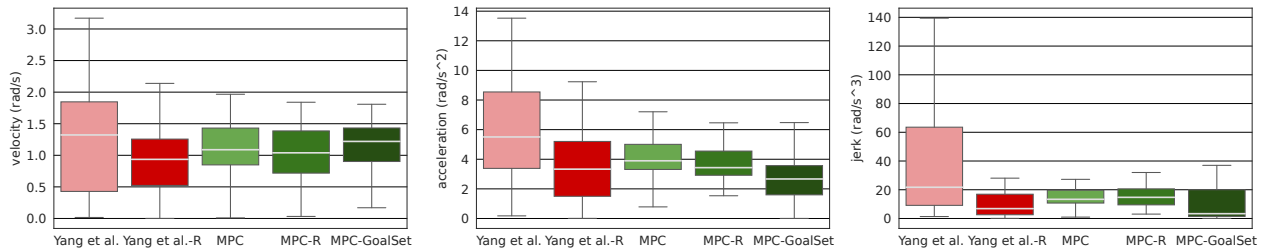


Fig. 6: Our proposed approach reduces jerk compared with prior work [2], a possible explanation for the better user experience.

		box	newspaper	plate	mug	remote	toothpaste	scissors	towel	pen	spoon	average
Appr. Time	Yang <i>et al.</i> [2]	7.5±2.4	8.8±1.0	10.0±1.2	14.0±6.3	11.5±6.6	9.0±1.2	10.5±5.8	9.5±3.1	9.5±1.3	10.5±1.0	10.1±3.7
	Ours	6.5±2.5	6.3±1.0	5.8±2.1	5.3±1.3	8.8±7.1	10.8±6.7	15.3±4.9	4.3±1.3	6.5±3.9	5.8±2.1	7.5±4.7
Success Rate	Yang <i>et al.</i> [2]	66.7%	100.0%	66.7%	100.0%	80.0%	66.7%	66.7%	80.0%	50.0%	80.0%	75.7%
	Ours	66.7%	80.0%	100.0%	66.7%	50.0%	66.7%	80.0%	66.7%	100.0%	66.7%	74.3%

TABLE I: Quantitative results from the user evaluation with four users.

C. Motion Metrics

To better understand the motion pattern, we recorded the robot position, velocity, acceleration and jerk during the above experiments. The statistics are reported in Fig. 6. In general, all our MPC-based approaches have more consistent velocity and accelerations compared with the baseline approach [2]. Our approach has less sudden accelerations, resulting in less jerk. In particular, our MPC-GoalSet achieves the least jerk, as in this formulation MPC optimizes to reach a single grasp from the grasp set, while considering the robot’s current position, velocity, and acceleration.

VII. USER EVALUATION

We also conducted a user evaluation to compare the proposed system (*MPC-R* as *Ours*) with the prior system Yang *et al.* [2] to validate that our system enables fluid H2R handovers. We recruited four participants from the lab². Each participant did two rounds of handovers to interact with two systems without knowing the details of the systems. Participants were instructed to hand over ten objects from Household-A objects used in [2] to the robot one at a time. They were asked to rate the systems with a set of Likert scale questions that are commonly asked in prior works [10] immediately after each round of interaction with the system. We additionally asked the users which system is less jerky to evaluate the subjective motion smoothness. We counterbalanced the order in which participants interacted with the two systems to avoid the bias. After completing all interactions, participants were asked to rank the two systems in different dimensions and share their opinions and comments through open-ended questions.

Objective Evaluation. We report the approach time and the success rate in Table. I. Our approach was able to take over the object with less time except for *scissors* and *toothpaste*, and the overall approach time is shorter than the prior system [2]. The success rate of our system is on par with prior system [2].

²We were not able to recruit outside participants due to the ongoing COVID-19 pandemic.

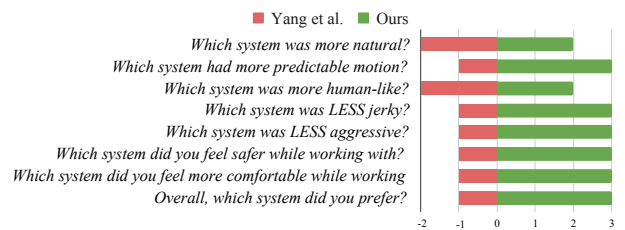


Fig. 7: Participants’ ranking of the two systems across eight questions in the questionnaire. Length of the bar denotes the number of participants.

Subjective Evaluation. Fig. 7 reports user ratings over the two systems. 3 out of 4 users preferred our proposed system. Particularly, they thought our system was more predictable, less jerky, less aggressive, safer, and more comfortable to work with. The main reasons for choosing our systems are “I was able to predict the robot motion better”, “it felt less aggressive”, and “it did not crash” and “required less motion before picking the object”. One user preferred [2], commenting “[2] had a better approach for some of the items and changed trajectories for less items, thus providing a better experience for me”. But this same user also commented “some of these questions are really A=B”, which proves the user experience of our system is better or on par with the prior system [2] in these aspects. For questions “which system was more natural” and “human-like”, half of the users chose [2] and the other half preferred our system.

VIII. DISCUSSION AND CONCLUSION

We proposed a method for fluid human-to-robot handovers by incorporating more knowledge about the robot motion planning, specifically using MPC to find smooth, consistent trajectories. To facilitate grasp selection, we learned a reachability model to prioritize robot grasps with higher manipulability scores. Our approach was demonstrated more efficient through a systematic evaluation and more favorable by users through a user evaluation compared to prior work. Nevertheless, the reachability model and the grasp generation are independent, which sometimes leads to robot oscillations when the reachability scores and the candidate grasps are in disagreement, especially when using MPC given a set

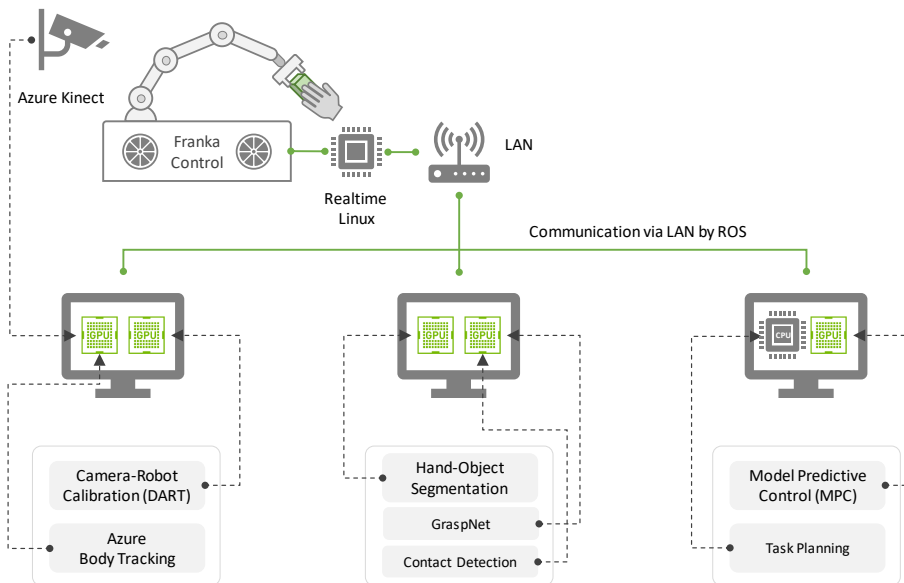


Fig. 8: The structure of the proposed handover system.

of grasps (MPC-GoalSet). In the future, we plan to better incorporate grasp and motion planning as well as reachability modeling to improve the robustness of the system.

APPENDIX

A. System Diagram

Fig. 8 illustrates the structure of our system. We used a Franka Emika Panda arm for all experiments. The Franka control box was connected to a realtime Ubuntu desktop. An Azure Kinect RGBD camera was mounted next to the robot to capture the full scene. The perception, planning and control modules are distributed on three desktop computers. Specifically, the Azure Kinect software, including depth image processing and body tracking, and the camera-robot calibration using DART [18] ran on two NVIDIA RTX 2080 Ti GPUs from the same desktop. The hand-object segmentation, GraspNet [4], and the physical contact detection net ran on another two NVIDIA RTX 2080 Ti GPUs from a different desktop. Finally, the stochastic MPC is computed from an NVIDIA RTX 3090 GPU and the task planning is computed without GPUs. All the desktops are connected through a local area network (LAN) and communicated through the Robot Operating System (ROS).

B. Neural Network Architectures

1) *Physical Contact Detection Model*: Our contact detection model takes a concatenation of joint velocities (7 dim), efforts (7 dim), force (3 dim) and torque (3 dim) for the past $T = 5$ steps as input, which is a $T \times 20$ tensor. Each individual timestep is first encoded through a two-layer perception. Each layer consists of a 1D convolutions with kernel size one and a ReLU activation (the number of channels is given by 20–256, 256–512). Then the model performs two 1D convolutions with kernel size one and 512 channels with ReLU activation and Batch Normalization over

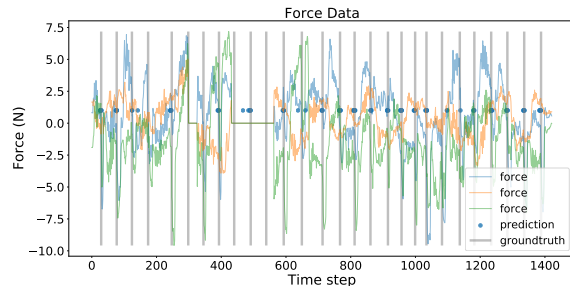


Fig. 9: An example of the captured force data. Gray vertical lines denote the moment when a contact occurs. Blue circles illustrate the detected contact by our model.

history. Finally, another two-layer perception (1D convolutions with kernel size 1 and number of channels as 512–256, 256–1, ReLU activation and dropout) predicts a single output indicating if a force event was detected within the window. The model is trained with the binary cross-entropy loss. Fig. 9 visualizes the recorded force, the ground truth label and our prediction of a test sequence.

2) *Reachability Model*: The reachability prediction model takes in the 6-dim grasp pose as input, which is passed through five fully-connected layers, each with 64 neurons and ReLU activation. The final output is the predicted reachability value and the model is trained with the mean-squared-error loss.

3) *Hand Segmentation Model*: The backbone of the hand segmentation model is a Feature Pyramid Network [29] based on ResNet-50 [30] pretrained on the ImageNet [31]. The four convolutions with upsampling operations are used to recover the feature map with the original resolution and a binary segmentation mask is predicted to indicate whether a pixel is hand or background. Please see [2] for details of the training dataset.

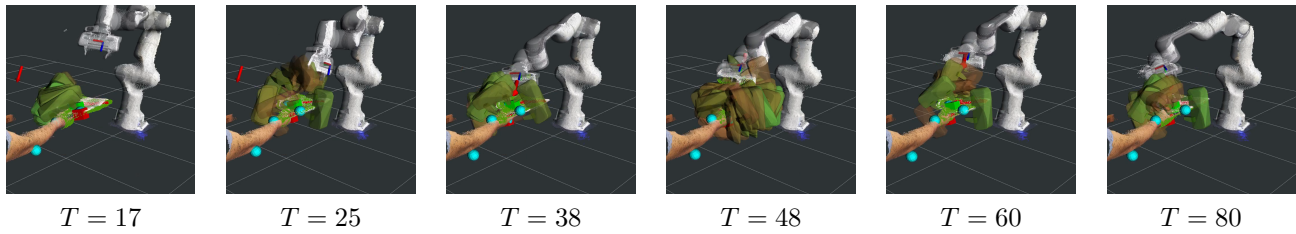


Fig. 10: The oscillation of robot motion caused by inconsistent grasps.

C. Failure cases

Graspnet [4] and reachability together do not guarantee consistent grasps. As a result, we do see a number of cases where no grasps are possible – for example, GraspNet might predict only grasps that the reachability model says are very low quality. One problematic case sees a number of generated grasps that are right on the threshold. Due to the noise from one step to the next, the method might be trying to reach very different results, leading to oscillations that are unnerving to a human user, as shown in Fig. 10.

REFERENCES

- [1] P. Rosenberger, A. Cosgun, R. Newbury, J. Kwan, V. Ortenzi, P. Corke, and M. Grafinger, “Object-independent human-to-robot handovers using real time robotic vision,” *IEEE Robotics and Automation Letters*, vol. 6, no. 1, pp. 17–23, 2021.
- [2] W. Yang, C. Paxton, A. Mousavian, Y.-W. Chao, M. Cakmak, and D. Fox, “Reactive human-to-robot handovers of arbitrary objects,” *IEEE-RAS International Conference on Robotics and Automation (ICRA)*, 2021.
- [3] D. Morrison, P. Corke, and J. Leitner, “Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach,” *arXiv preprint arXiv:1804.05172*, 2018.
- [4] A. Mousavian, C. Eppner, and D. Fox, “6-dof graspnet: Variational grasp generation for object manipulation,” in *ICCV*, 2019.
- [5] N. D. Ratliff, J. Issac, D. Kappler, S. Birchfield, and D. Fox, “Riemannian motion policies,” *arXiv preprint arXiv:1801.02854*, 2018.
- [6] W. Yang, C. Paxton, M. Cakmak, and D. Fox, “Human grasp classification for reactive human-to-robot handovers,” *IROS*, 2020.
- [7] M. Bhardwaj, B. Sundaralingam, A. Mousavian, N. Ratliff, D. Fox, F. Ramos, and B. Boots, “STORM: An Integrated Framework for Fast Joint-Space Model-Predictive Control for Reactive Manipulation,” *arXiv preprint*, 2021.
- [8] M. Costanzo, G. De Maria, and C. Natale, “Handover control for human-robot and robot-robot collaboration,” *Frontiers in Robotics and AI*, vol. 8, p. 132, 2021.
- [9] W. P. Chan, C. A. Parker, H. M. Van der Loos, and E. A. Croft, “Grip forces and load forces in handovers: implications for designing human-robot handover controllers,” in *HRI*, 2012.
- [10] V. Ortenzi, A. Cosgun, T. Pardi, W. Chan, E. Croft, and D. Kulic, “Object handovers: a review for robotics,” *arXiv preprint arXiv:2007.12952*, 2020.
- [11] R. S. Johansson, G. Westling, A. Bäckström, and J. R. Flanagan, “Eye-hand coordination in object manipulation,” *Journal of neuroscience*, vol. 21, no. 17, pp. 6917–6932, 2001.
- [12] F. Cini, V. Ortenzi, P. Corke, and M. Controzzi, “On the choice of grasp type and location when handing over an object,” *Science Robotics*, 2019.
- [13] J. Kim, “Advanced grasp planning for handover operation between human and robot: Three handover methods in esteem etiquettes using dual arms and hands of home-service robot,” in *Proc. Int. Conf. on Autonomous Robots and Agents December, Palmerston North, 2004*, 2004.
- [14] M. Faroni, M. Beschi, and N. Pedrocchi, “An mpc framework for online motion planning in human-robot collaborative tasks,” in *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2019, pp. 1555–1558.
- [15] M. V. Minniti, R. Grandia, K. Föh, F. Farshidian, and M. Hutter, “Model predictive robot-environment interaction control for mobile manipulation tasks,” *arXiv preprint arXiv:2106.04202*, 2021.
- [16] M. Krämer, C. Rösmann, F. Hoffmann, and T. Bertram, “Model predictive control of a collaborative manipulator considering dynamic obstacles,” *Optimal Control Applications and Methods*, vol. 41, no. 4, pp. 1211–1232, 2020.
- [17] S. Kuindersma, R. Deits, M. Fallon, A. Valenzuela, H. Dai, F. Permenter, T. Koolen, P. Marion, and R. Tedrake, “Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot,” *Autonomous robots*, vol. 40, no. 3, pp. 429–455, 2016.
- [18] T. Schmidt, R. A. Newcombe, and D. Fox, “DART: Dense Articulated Real-Time Tracking,” in *Robotics: Science and Systems*, vol. 2, no. 1, 2014.
- [19] N. Vahrenkamp, T. Asfour, G. Metta, G. Sandini, and R. Dillmann, “Manipulability analysis,” in *2012 12th IEEE-RAS international conference on humanoid robots (humanoids 2012)*. IEEE, 2012, pp. 568–573.
- [20] D. Berenson, R. Diankov, K. Nishiwaki, S. Kagami, and J. Kuffner, “Grasp planning in complex scenes,” in *2007 7th IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2007, pp. 42–48.
- [21] I. Akinola, J. Varley, B. Chen, and P. K. Allen, “Workspace aware online grasp planning,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 2917–2924.
- [22] N. Vahrenkamp, D. Berenson, T. Asfour, J. Kuffner, and R. Dillmann, “Humanoid motion planning for dual-arm manipulation and re-grasping tasks,” in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2009, pp. 2464–2470.
- [23] I. Akinola, J. Xu, S. Song, and P. K. Allen, “Dynamic grasping with reachability and motion awareness,” *arXiv preprint arXiv:2103.10562*, 2021.
- [24] T. F. Chan and R. V. Dubey, “A weighted least-norm solution based scheme for avoiding joint limits for redundant joint manipulators,” *IEEE Transactions on Robotics and Automation*, vol. 11, no. 2, pp. 286–292, 1995.
- [25] B. Sundaralingam and T. Hermans, “Relaxed-Rigidity Constraints: Kinematic Trajectory Optimization and Collision Avoidance for In-Grasp Manipulation,” *Autonomous Robots*, vol. 43, no. 2, pp. 469–483, February 2019.
- [26] M. Toussaint, “Newton methods for k-order markov constrained motion problems,” *arXiv preprint arXiv:1407.0414*, 2014.
- [27] M. Bhardwaj, B. Sundaralingam, A. Mousavian, N. Ratliff, D. Fox, F. Ramos, and B. Boots. (2021) STORM: An Integrated Framework for Fast Joint-Space Model-Predictive Control for Reactive Manipulation: Further experimental results. [Online]. Available: <https://sites.google.com/view/manipulation-mpc/further-experimental-results?authuser=0#h.ud1wqx4676yl>
- [28] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, “The ycb object and model set: Towards common benchmarks for manipulation research,” in *ICAR*. IEEE, 2015.
- [29] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature pyramid networks for object detection,” in *CVPR*, 2017.
- [30] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *CVPR*, 2016.
- [31] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al., “Imagenet large scale visual recognition challenge,” *International journal of computer vision*, 2015.