2014 14th IEEE-RAS International Conference on
Humanoid Robots (Humanoids)
November 18-20, 2014. Madrid, Spain

Handover planning for every occasion

Ana Huamán Quispe   Heni Ben Amor   Mike Stilman†

Abstract— **In this paper we explore shared manipulation tasks involving collaboration between two agents: a bimanual manipulator and a movable humanoid. We propose a deterministic planner that outputs a handover pose for the object manipulated, the grasps to be used for both agents, and arm trajectories that allow the agents to reach, grasp, interchange and place the object at its final goal pose. Most existing approaches to shared manipulation assume that both participating agents are (or can be) positioned face-to-face. In this paper, we consider a more general set of scenarios in which the relative pose between agents is not tightly constrained to facing each other. To accomplish a successful interaction between the agents, careful planning must be done in order to select the best possible handover pose such that the movable agent should not need to move its torso, only its arm, to interact with the object. We present 3 simulation experiments with manipulation tasks performed by a bimanual fixed manipulator and a humanoid robot.**

I. Introduction

Useful home robots should be capable to perform simple tasks that make human lives easier. In particular, we consider collaborative manipulation tasks in which a robot acts as an assistant, handing objects to a user. While inherently simple, these types of tasks are more likely to happen in a household in a regular basis, so we consider it a key robot capability.

Shared manipulation tasks can be studied from different perspectives. From the side of human-robot interaction, there is a vast amount of work that analyses the information exchanged between robot and user during the handing over of the manipulated object [2]. On the other hand, the manipulation planning community frames the problem mostly from the point of view of the robot in order to select a robust grasp on the object and then find a feasible arm trajectory that allows the robot to perform its share of the task successfully.

In this paper, our goal is to enable a fixed robot to effectively perform a handover task while *keeping the movable receiver agent in mind*. We acknowledge the movable receiver agent by considering the receiver's dexterity while placing the object at its goal pose. Furthermore, we address the fact that for different tasks, the movable receiver might not be ideally located with respect to the fixed robot (i.e. facing it), so it is the fixed robot's task to select an arm trajectory that places the manipulated object in a pose such that the receiver can easily grasp it. Depending on the specific situation, this could be naively simple or notoriously difficult. As an illustration, consider the examples shown in Figure 1. All of them involve a bimanual fixed manipulator handing over an object to a movable humanoid whose relative pose with respect to the fixed manipulator varies drastically

Institute for Robotics and Intelligent Machines, Georgia Institute of Technology, Atlanta, GA 30332, USA.ahuaman3@gatech.edu, hbenamor@cc.gatech.edu

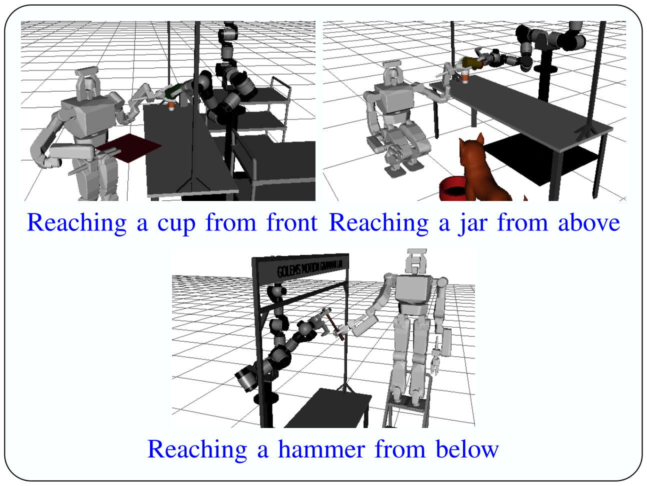
Reaching a cup from front
Reaching a jar from above
Reaching a hammer from below

Fig. 1: Shared manipulation tasks with non face-to-face interaction between agents

between examples (i.e. the humanoid does not face the fixed manipulator in neither of these examples, a fact that is commonly assumed for shared manipulation tasks). In this paper, we intend to address situations as the ones depicted, in which the movable receiver can vary its pose with respect to the fixed robot. Among some questions that must be answered to solve this problem are: A) Which arm should the fixed robot use to hand the object? B) How should the object be grasped such that the movable receiver can grasp it easily? and C) Which handover pose is good enough as to reduce the overall effort of the movable receiver?.

The rest of this paper is organized as follows: Section II summarizes relevant literature on shared manipulation tasks. Section III and IV explains our algorithm, the assumptions considered and our current limitations. Section V presents simulation results in 3 different arbitrary environments and finally in Section VI we discuss our approach, its advantages and shortcomings as well as future work.

II. Previous Work

In [5] Edsinger et al. argue that fluent handover between a robot and a human allows for a range of cooperative manipulation tasks. They present a set of perception and control modules which can be used to realize a simple object exchange between interaction partners. Kemp and colleagues argue that the complimentary skills of humans and robots can be exploited in such a scenario. More specifically, the challenging task of robot grasp planning can be simplified by having the human place the object in the robot's end effector.

Mainprice and colleagues [8] investigated how the spatial preferences of a human partner can be incorporated into the robot decision making during handover tasks. They introduced a planning algorithm which can be parametrized by specifying a mobility parameter. The mobility parameter reflects the ability or willingness of the human partner to move to a new position to obtain an object. In case of a high mobility parameter, the robot can plan handover actions that require some effort from the human partner. In contrast, a low mobility parameter means that the robot should reduce the human's effort by moving to the exchange site location. In [9], they also investigated how gaze direction and postural comfort can be used to identify a transfer point

In [4] Cakmak and colleagues investigated human preferences in robot handover configurations. In particular, they analyzed how humans responded to planned robot joint configurations during handover tasks. In their study they also allowed human subjects to provide positive and negative feedback to the robot on which configurations are more appropriate. This feedback was, then, used to improve the naturalness and legibility of the robot handover motion. Their experiments showed that planned handover motion can provide higher reachability of the object, while learned handover behavior can improve the naturalness and appropriateness of robot joint configurations.

Koene et al. [7] showed that temporal precision plays an important role during object exchange. For fluent interaction, a robot needs to synchronize the timing of the arm movement with the arm movement of the human partner. While we acknowledge that fluency and naturalness are important characteristics for robot interactions with humans, this paper focuses on planning interactions that allow two robot agents to keep dexterous arm configurations during the handover, placing less priority to the factors mentioned above (although a high dexterity is usually correlated to smooth, human-looking arm configurations).

III. Preliminaries

In this section, we formally define the scenario and the shared manipulation task to be addressed through this paper. We also provide an intuitive overview of our solution approach, to be expanded in the following section.

A. Problem definition

Our shared manipulation task can be defined with the following elements:

- A fully known 3D environment
- A target object O to be manipulated.
- Two agents: a fixed robot giver (R_g) and a movable robot receiver (R_r).

In this and the following section, we will use as an example our *butler scenario*, whose problem definition is shown in Figure 2.

B. Task definition

The shared manipulation task consists in transporting the target object O from its start pose T_s to a goal pose T_g. Furthermore, we assume T_s to be only reachable by R_g and T_g reachable by R_r, hence the object must be picked up by R_g, handed off to R_r and finally placed in its final pose T_g by R_r. An example of T_s and T_g for our butler test scenario is shown in Figure 3.

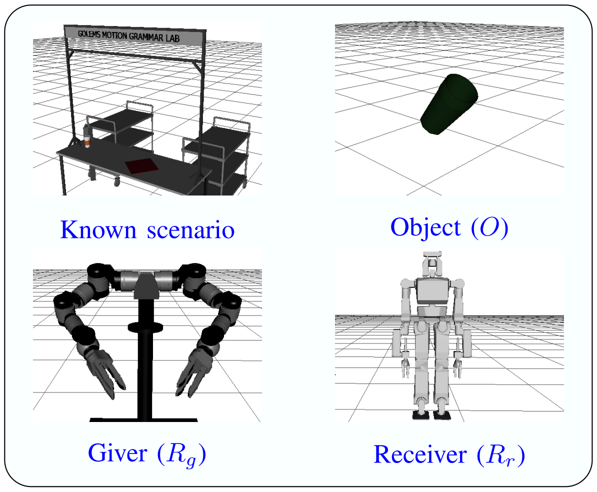

Fig. 2: Problem definition for butler scenario

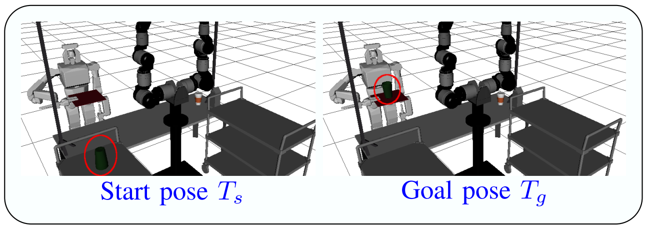

Fig. 3: Task definition for butler scenario

C. Algorithm Overview

At high level, our algorithm can be summarized as follows:

1) Decide which arm R_g will use to grasp the object.
2) Calculate a set of feasible grasps for R_r ($\mathcal{G}_r$) that allows it to place O at T_g
3) Calculate a set of feasible grasps for R_g ($\mathcal{G}_g$) that allows it to reach O at its start pose T_s
4) Calculate a set of feasible handover solutions $\mathcal{H}$ using the information provided by the set of feasible grasps $\mathcal{G}_g$ and $\mathcal{G}_r$ and the pose of the two robotic agents R_g and R_r.
5) Select the handover solution from $\mathcal{H}$ that maximizes the dexterity of the arms in the shared manipulation task and plan the corresponding arm trajectories.

In section IV we will expand and explain each point above in more detail. Any solution from $\mathcal{H}$ should have information such as Figure 4: The handover pose T_h and the grasps to be executed by R_g and R_r.

IV. Algorithm

In this section we will go in detail over the points briefly addressed in III-C:

A. Arm selection for R_g

Our planner must first determine if O is located within the reach of R_g. For this, the *reachability space* [6] of R_g ($\mathcal{R}_g$) is generated offline. We fill $\mathcal{R}_g$ by sampling a large number of random joint configurations for each arm of R_g and storing the average manipulability [11] of each sample in the voxel corresponding to the Tool Center Point (TCP) position of the end effectors (an entry of 0 means that the

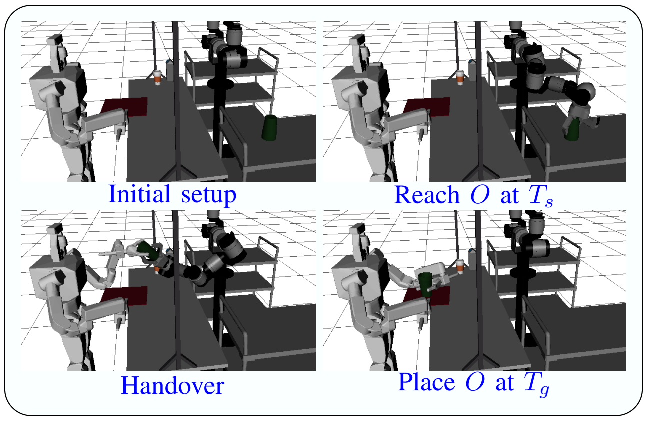

Fig. 4: Expected solution for butler scenario

arm cannot reach that location) . Since R_g is a (fixed) dual manipulator, $\mathcal{R}_G$ will provide us with a manipulability metric for both left and right arms at each voxel.

The object O is considered within reach of R_g if the voxel corresponding to T_s in $\mathcal{R}_G$ has a non-zero entry for either the left arm, right arm or both. If both entries are non-zero, then the arm selected will be the one with the highest entry value, since this suggests that the corresponding arm will have more dexterity.

If only one entry is non-zero, then the corresponding arm is selected. If both entries are zero then the task is considered infeasible and no further action is required.

B. *Generating Grasp candidates for R_r ($\mathcal{G}_r$)*

The next step consists on calculating a set of grasps and arm configurations that allow R_r to place O at its goal location T_g.

For this step we use a set of precomputed grasps for O. After placing the object O at T_g, we test each of the grasps in $\mathcal{G}_r$ and prune the candidates that either not kinematically feasible or that present collision with the environment. This procedure can be seen in Algorithm 1. For our butler scenario, some of the non-pruned grasps ($\in \mathcal{G}_r$) are depicted in Figure 5.

```
Algorithm 1: PruneGrasps($R$, $O$, $\mathcal{G}$)
  Input: Robot $R$, Object $O$ and its corresponding grasp
         set $\mathcal{G}$
  Output: A feasible subset of $\mathcal{G}$ ($\mathcal{G}^*$) and its arm
          configurations ($\mathcal{Q}^*$)
1 foreach g $\in \mathcal{G}$ do
2     q $\leftarrow$ $R$.executeGrasp(g, $O$)
3     if checkCollision() is false then
4         $\mathcal{G}^* \leftarrow \mathcal{G}^* \cup$ g
5         $\mathcal{Q}^* \leftarrow \mathcal{Q}^* \cup$ q
6     return $\mathcal{G}^*$, $\mathcal{Q}^*$
```

C. *Generating Grasp candidates for R_g ($\mathcal{G}_g$)*

Similar to the step above but now the object O is placed in T_s and the generated grasps are stored in $\mathcal{G}_g$. An example of these feasible grasps for our butler scenario example can be seen in Figure 6.

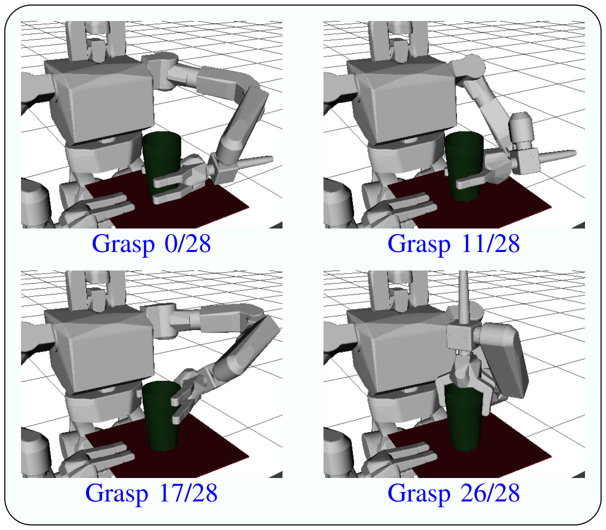

Fig. 5: Candidate grasps at T_g executed by R_r

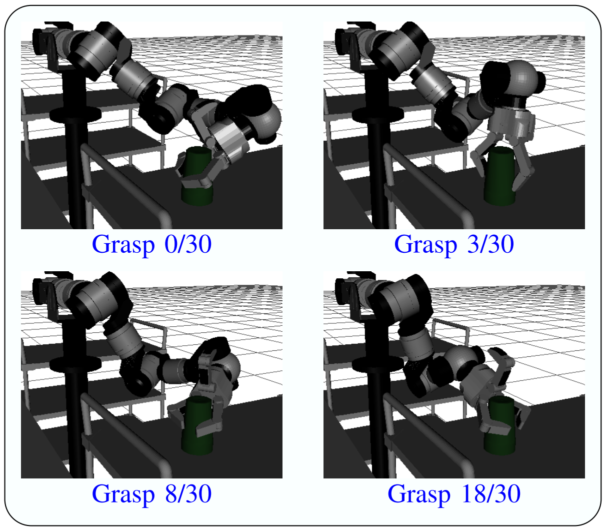

Fig. 6: Candidate grasps at T_s executed by R_g

D. *Calculation of feasible handover poses*

There exists an infinite number of possible poses for the object O to be handed from R_g to R_r. To perform an exhaustive search in this space would demand significant computation time.

Instead, we approach the problem by reducing the possible handover poses space by considering 2 simple observations from human grasping:

- When transporting an object, humans tend to reduce orientation changes in the object being grasped, unless necessary.
- Whenever possible, we keep our wrist in a resting pose while manipulating an object.

Given the points above, we generate a set of handover tuples (stored in $\mathcal{H}$) using Algorithm 2. Each tuple contains a handover pose for the object O, the grasps for R_r and R_g with an arm configuration that allows R_r and R_g to execute the grasp.

The specific procedure to calculate a handover grasp pose T_h is shown in Algorithm 3. The translation component of T_h is obtained from the set of near neighbours of the middle point between the shoulders of R_r and R_g. The voxel that can be reached from both arms and that has the highest average manipulability is chosen as the position for T_h. Regarding the rotation, we take advantage of the information provided by the grasp set $\mathcal{G}_r$: We calculate the pose by applying a minimum rotation between the R_r's hand approach vector at T_g and the direction between the shoulders of R_r and R_g.

A couple of example poses for our butler scenario can be seen in Figure 7.

Algorithm 2: Get set of candidate handover poses

```
1 foreach $\mathbf{g}_r \in \mathcal{G}_r$ do
2     $T_h$, $\mathbf{q}_r$ ← GetHandoverPose($R_g$, $R_r$,$O$,$\mathbf{g}_r$,$T_g$)
3     if $T_h \neq NULL$ then
4         $O$.set($T_h$)
5         $R_r$.set($\mathbf{q}_r$, $\mathbf{g}_r$)
6         $\mathcal{G}_g^* \leftarrow$ PruneGrasps($R_g$, $\mathcal{G}_g$)
7         if $\mathcal{G}_r^* \neq \emptyset$ then
8             foreach $\mathbf{g}_g \in \mathcal{G}_g^*$ do
9                 $\mathcal{H} \cup (T_h, \mathbf{g}_r, \mathbf{g}_g, \mathbf{q}_r, \mathbf{q}_g)$
```

Algorithm 3: GetHandoverPose(R_g, R_r, O, $\mathbf{g}_r$, T_g)

```
Input: Robots, object, goal pose and grasp for $R_r$
Output: Handover pose $T_h$ and arm conf. for $R_r$ ($\mathbf{q}_r$)
1 $\mathbf{z}_{\text{shoulder}}$ ←getVec($R_r$.shoulder(),
  $R_g$.shoulder())
  /* Get translation for $T_h$                */
2 $\mathbf{p} \leftarrow R_r$.shoulder() + $0.5 \cdot \mathbf{z}_{\text{shoulder}}$
3 $\mathcal{S}_r \leftarrow$ GetClosestVoxels($\mathcal{R}_r$, $\mathbf{p}$)
4 $\mathcal{S}_g \leftarrow$ GetClosestVoxels($\mathcal{R}_g$, $\mathbf{p}$)
5 $\mathcal{S} \leftarrow \mathcal{S}_r \cap \mathcal{S}_g$
  /* Sort according to manipulability         */
6 $\mathcal{S}^* \leftarrow$ Sort($\mathcal{S}$)
7 $T_h$.trans() ← $\mathcal{S}^*$.front()
  /* Get rotation for $T_h$                   */
8 $O$.setPose($T_g$)
9 $\mathbf{q}_r \leftarrow R_r$.executeGrasp($\mathbf{g}_r$, $O$)
10 $T_{\text{hand}} \leftarrow R_r$.getHandPose()
11 rot ← minRot($T_{\text{hand}}$.z(), $\mathbf{z}_{\text{shoulder}}$)
12 $T_h$.rotation() ← rot × $T_{\text{hand}}$ × $\mathbf{g}_r.T_{ho}()$
13 $O$.setPose($T_h$)
14 $\mathbf{q}_r \leftarrow R_r$.executeGrasp($\mathbf{g}_r$, $O$)
15 return $T_h$,$\mathbf{q}_r$
```

E. Selection of dexterous handover pose and arm trajectory generation

After the step described before, our planner has a set $\mathcal{H}$ of handover poses with the corresponding grasps for R_r and R_g. The only remaining task is to select one of these tuples

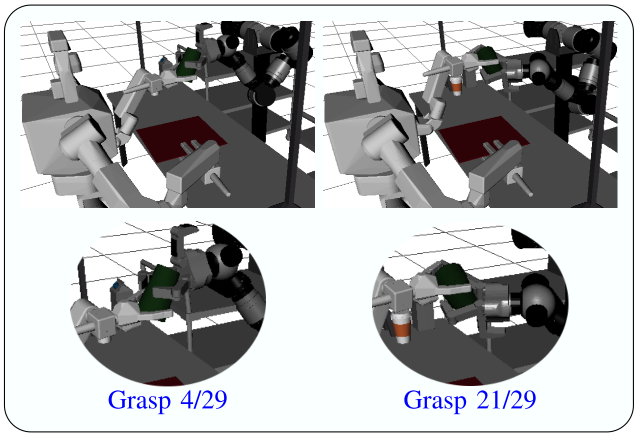

Fig. 7: Candidate handover poses during transfer phase

and generate arm trajectories between them in order to reach, pick, handover and place the object. For this we use an standard IKBiRRT [3] approach.

V. Simulation Results

We set 3 simulation environments, as shown in Figure 1. All three scenarios involve 2 robots, which will be briefly described in the following subsection:

A. Generalities

1) Crichton: Crichton is a bimanual fixed manipulator consisting on 2 Schunk LWA4 arms with a Schunk Dexterous Hand (SDH) attached to each last arm link. Each arm has 7 DOF, whereas the SDH possess 3 fingers and 8 DOF. A pseudo-analytical IK solver for the LWA4 arm was used for IK queries.

2) Hubo: Hubo is a humanoid robot with two 7-DOF arms. Each arm has a hand at their last link. The left hand has 3 fingers and 6 DOF whereas the right hand has 4 fingers and 8 DOF. A inverse-Jacobian IK solver for the Hubo arms was used for IK queries.

3) Simulation environment: For our simulation experiments we used DART [1]. The environment used, as well as the object and robot models, are replicas of their real hardware counterparts in order to make the transition to hardware experiments easier.

4) Grasp generation: The grasp datasets for each object were generated offline and loaded online for all the experiments shown. The numbers of grasps generated per each object are of a few hundreds, as it can be seen in Table I, hence, the selection of a handover pose and the corresponding grasps to use per each robot is not trivial.

Since the grasp set generation is independent of the planning algorithm presented here, any method to generate grasps can be used. In particular, we used the common approach of evenly sampling the object's surface and set the hand's palm just above each sample, with the approach direction parallel to the normal of the object at each of these points. Other methods could be used to generate the set of precomputed grasps, such as GraspIt! [10] or other similar open source tools.

Finally, all experiments were ran in an Intel Core i7 machine (1.6GHz).

TABLE I: Number of grasps per object

Object	SDH Hand	Hubo left hand	Hubo right hand
Cup - Butler scenario	316	368	368
Hammer - Stool scenario	445	590	644
Food jar - Kneeling scenario	448	448	448

TABLE II: Results for 100 randomized butler scenarios

Parameter measured	Values
Average Planning time	2.31 seconds
Successful plans	100/100
Average number of grasps for R_g	32.36
Average number of grasps for R_r	39.77
Average number of handover poses	16.64

B. Experiment 1: Butler scenario

Our first test problem is depicted in Figure 8: The target object O (a green cup) is initially resting - upside down - on a cart located to the left of the fixed manipulator (Crichton). The goal is to place O upright on the middle of the tray carried by Hubo's right hand.

We performed 100 tests for randomized scenarios similar to the one depicted in Figure 8. The variables randomized were:

- Hubo's translation along an axes parallel to the table's rim (total range of movement 1.6 m).
- Hubo's rotation around its Z axis (pointing up) (total range: 30 degrees with respect to orientation directly facing Crichton).
- Tray's height: The tray held by Hubo's right hand was randomly moved up and down in a total range of 6 cm.

Some statistics (such as planning time and number of average handover poses generated) are shown in Table II and 2 handover poses for different randomized examples are shown in Figure 9

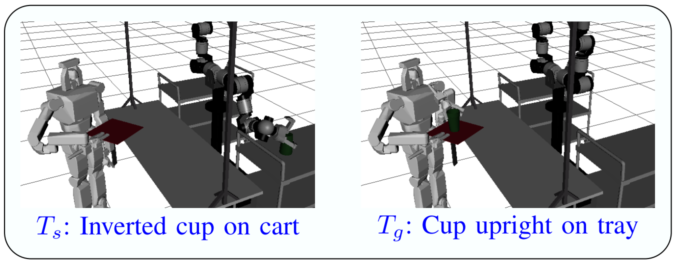

Fig. 8: Start and goal configurations for butler test case

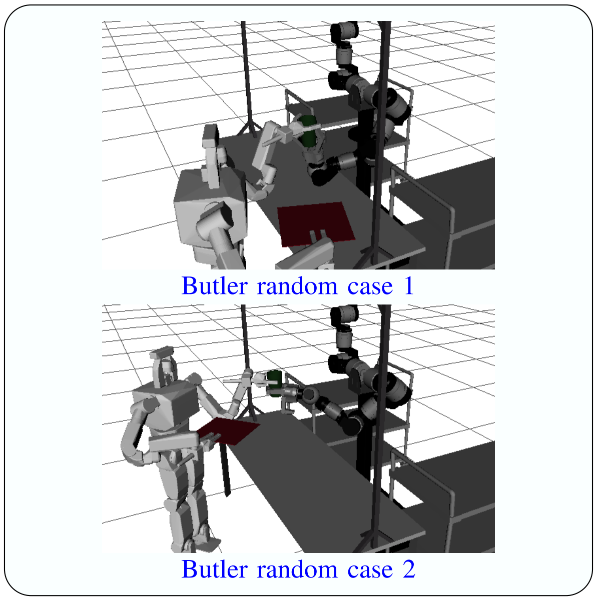

Fig. 9: Sample solutions for random butler test cases

C. Experiment 2: Stool scenario

Our second test scenario consists on Hubo standing on top of a 3-step stool while Crichton hands it a hammer from its start pose on the table in order for Hubo to lift it in front of it with its head pointing to the right (see Figure 10).

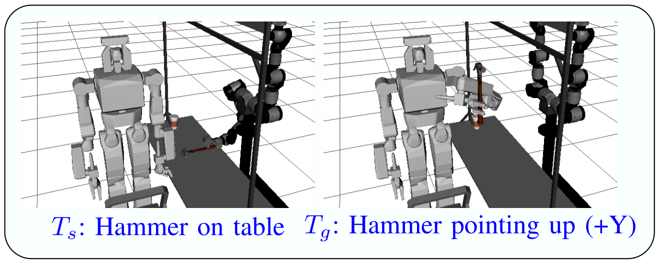

Fig. 10: Start and goal configurations for stool test case

In a similar way as the butler case, here we also performed 100 randomly varied runs. The varied parameters were the location of Hubo (first, second or third step on the stool) and the pose of the stool (some translation along an axis parallel to the table longest side and a small rotation around the up axis, no larger than 20 degrees). Results regarding number of possible handover solutions are shown in Table III and some solutions for the random cases are shown in Figure 11

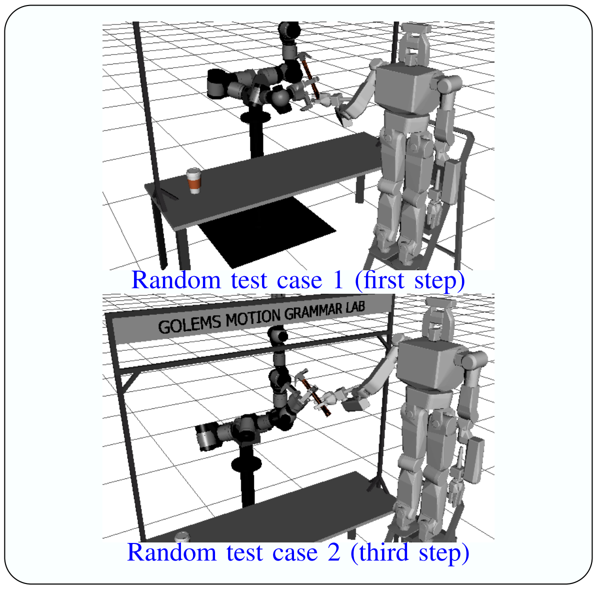
Random test case 1 (first step)
GOLEMS MOTION GRAMMAR LAB
Random test case 2 (third step)

Fig. 11: Sample solutions for random stool test cases

Notice that the planning time increases for this example, this probably due to the fact that the size of the grasp sets for the hammer object is bigger than for the cup, hence the search space increases in size.

TABLE III: Results for randomized stool scenarios

Parameter measured	Values
Average Planning time	4.62 seconds
Successful plans	100/100
Average number of grasps for R_g	56.87
Average number of grasps for R_r	143.28
Average number of handover poses	89.49

TABLE IV: Results for 100 randomized kneeling scenarios

Parameter measured	Values
Average Planning time	3.1845 seconds
Successful plans	100/100
Average number of grasps for R_g	30.45
Average number of grasps for R_r	72.91
Average number of handover poses	39.89

D. Experiment 3: Kneeling scenario

Our final test scenario is depicted in Figure 12: Hubo is kneeling and needs to be handed an object from the table, in this example O is a dog food jar whose goal pose is above the dog's plate and slightly tilted (rotated with respect to the X axis going from table towards Hubo).

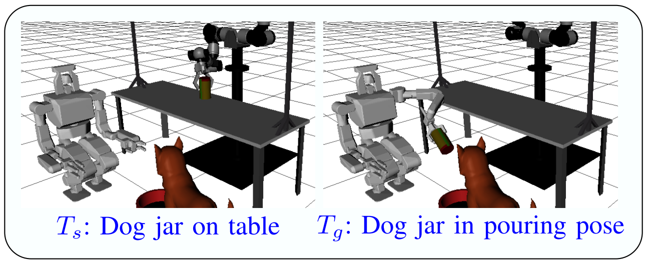

Fig. 12: Start and goal configurations for kneeling test case

As in the cases mentioned before, the average results of running our algorithm 100 times on this test case are shown in Table IV. The randomized parameters for these cases were the goal location of the dog food jar (position change of no more than 3 cm and roll rotation changes of up to 20 degrees).

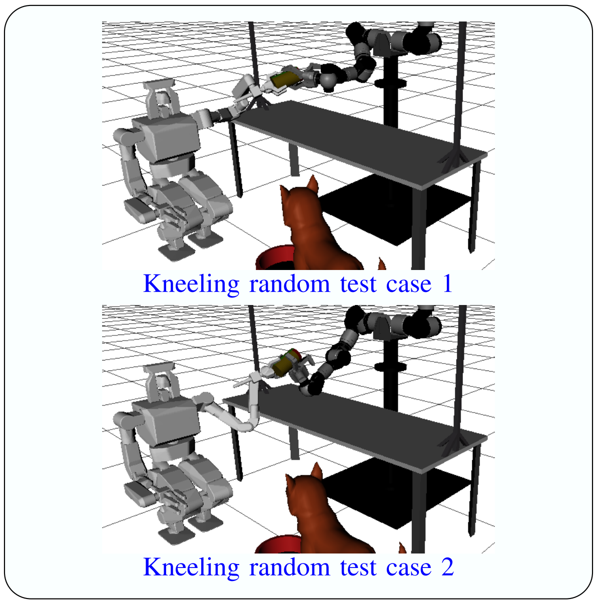

Fig. 13: Sample solutions for random kneeling test cases

VI. Discussion and Conclusion

In this paper we have presented a planner to solve shared manipulation tasks. Given an object that must be passed on from one robot R_g to another R_r, our planner calculates a set of possible handover solutions $\mathcal{H}$ and selects the one that maximizes the dexterity of R_r while performing the final object placing.

Our approach reduces its computation time by considering only a subset of all the possible handover poses. For this, it takes advantage of the grasps calculated for R_r. By using them and the direction between the shoulders of both robots, our handover pose search space is greatly reduced. We presented 3 simulated test cases in which our approach works across diverse randomized situations.

While our method shows early indications of efficiency, it should be noted that it is not complete; hence, the possibility exists that our algorithm might not find a handover pose even if one exists. As future work, we are transitioning our simulation setup to our real hardware in order to validate our preliminary simulation results with experimental tests.

References

[1] Dynamic Animation and Robotics Toolkit. `http://dartsim.github.io/dart/`. Accessed: 2014-07-21.

[2] H. Admoni, A. Dragan, S. Srinivasa, and B. Scassellati. Deliberate delays during robot-to-human handovers improve compliance with gaze communication. In *Proceedings of the 2014 ACM/IEEE international conference on Human-robot interaction*, pages 49–56. ACM, 2014.

[3] D. Berenson, S. Srinivasa, D. Ferguson, A. Collet, and J. Kuffner. Manipulation planning with workspace goal regions. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 618–624. IEEE, 2009.

[4] M. Cakmak, S. Srinivasa, M.K. Lee, J. Forlizzi, and S. Kiesler. Human preferences for robot-human hand-over configurations. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 1986–1993. IEEE, 2011.

[5] A. Edsinger and C. Kemp. Human-robot interaction for cooperative manipulation: Handing objects to one another. In *Robot and Human interactive Communication, 2007. RO-MAN 2007. The 16th IEEE International Symposium on*, pages 1167–1172. IEEE, 2007.

[6] D. Kee and W. Karwowski. Analytically derived three-dimensional reach volumes based on multijoint movements. *Human factors: the journal of the human factors and ergonomics society*, 44(4):530–544, 2002.

[7] A. Koene and M. Remazeilles. Relative importance of spatial and temporal precision for user satisfaction in human-robot object handover interactions.

[8] J. Mainprice, M. Gharbi, T. Siméon, and R. Alami. Sharing effort in planning human-robot handover tasks. In *RO-MAN, 2012 IEEE*, pages 764–770. IEEE, 2012.

[9] J. Mainprice, E. Sisbot, T. Siméon, and R. Alami. Planning safe and legible hand-over motions for human-robot interaction. In *IARP Workshop on Technical Challenges for Dependable Robots in Human Environments*, volume 2, page 7, 2010.

[10] A. Miller and P.K. Allen. Graspit! a versatile simulator for robotic grasping. *Robotics & Automation Magazine, IEEE*, 11(4):110–122, 2004.

[11] Y. Nakamura and H. Hanafusa. Inverse kinematic solutions with singularity robustness for robot manipulator control. *Journal of dynamic systems, measurement, and control*, 108(3):163–171, 1986.