Exploiting Object Symmetry for Efficient Grasping

Paper XXX

ABSTRACT

In this, paper we introduce an efficient representation for robot grasping that exploits symmetry properties of objects. The new representation forms a low-dimensional manifold, which can be used to identify the set of feasible grasps during a sequential manipulation task. We analyse the properties of this low-dimensional manifold and show that some of these properties can be used for fast manipulation planning. We apply the introduced representation and planner to bi-manual manipulation in humanoid robots.

1. INTRODUCTION

Robots with advanced dexterous capabilities have the potential to revolutionize important application domains such as healthcare, security, or manufacturing. Whether in structured environments such as factories, or unstructured environments such as homes, grasping is often the first step in the physical interaction between a robot and its surroundings. The generation of grasps on objects is therefore at the heart of plan generation for physical tasks. However, to date most grasp planning methods are mainly focused on generating physically stable grasps without incorporating immediate or future task constraints.

Many well established algorithms assume only a single action and do not include foresight and reasoning about next actions. However, when picking up a mug, it is important to select a grasp that facilitates the next action to be performed. For example, in case the next action is a pouring motion, the robot needs avoid grasps that would cover the rim of the object. In contrast, if any of the next actions involves placing the mug on the table, the robot cannot choose any grasp in which fingers touch the mug's base.

Many grasp representations are based on a floating hand, thereby representing only the end-effector and ignoring the embodiment of the entire robot in environment. As a result, infeasible grasps are generated and have to be pruned out in a post-processing step. This can be particularly limiting in bi-manual, sequential, and co-worker scenarios, in which the actions of one agent are likely to violate the constraints of another agent. In these scenarios, grasps need to be carefully chosen, such that they do not violate constraints imposed by a second arm, a human interaction partner, or a future action.

In this paper, we address the issue of manipulation planning with task constraints. We are particularly interested in tasks that involve several subtasks or several interacting agents. Planning grasps for such tasks can rapidly become computationally infeasible due to the large search space. The key insight of this paper is that re-occurring patterns in the geometry of shapes can be exploited to drastically reduce the space of solutions. We will introduce a low-dimensional, object-centered representation for grasp planning which is based on rotational symmetries. The symmetric nature of the objects allows us to update our representation as the object is rotated around the axis of symmetry. *Since the object is symmetric any stable grasp can be rotated around the axis yielding a family of feasible grasps.* We will show that this basic property leads to a significant reduction of the search complexity in grasp planning. Specifically, this property allows us to generate multiple grasps around the object which is useful for bi-manual tasks and cooperative robot hand over tasks. In addition, it allows us to identify a single grasp that is useful for a sequence of tasks.

Appears in: *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015), Bordini, Elkind, Weiss, Yolum (eds.), May, 4–8, 2015, Istanbul, Turkey.*

2. RELATED WORK

Robot grasping is a heavily studied sub-field of robotics. Until recently, most approaches have focused on the problem of generating stable grasps on objects. Given a model of an object, a hand shape is synthesized that allows the physically stable manipulation of the object [?, ?]. A prevalent approach is to modify the grasp parameters until a hand shape is found which is optimal w.r.t. a given criterion, such as force closure [?], or grasp wrench space [?]. For recent surveys of the grasp synthesis literature the reader is referred to [?] and [?].

To efficiently synthesize a variety of grasps, objects are often modelled using primitives, such as bounding boxes [?], medial axis representations [?], superquadrics [?], or simple geometrical shapes including cylinders, boxes and spheres [?]. Another way of increasing efficiency is by reducing the dimensionality of the space of possible hand configurations. In [?] and [?], dimensionality reduction techniques are used to project the space of hand shapes onto a low-dimensional manifold of two to three dimensions. However, in the above approaches, reachability and possible collisions during motion execution are only analyzed in a post-processing step after grasps are generated. Hence, possibly unreachable or unsafe grasp configurations are produced. Recent approaches have tried to bridge the divide through simultaneous grasp synthesis and motion planning [?]. Still, as noted in [?], looking solely at the stability of the grasp, or other immediate criteria ignores task constraints which are induced by the subsequent manipulations of the object. A grasp is often highly dependent on the task to be performed and should be chosen with next actions in mind.

Dang et al. [?] incorporate task constraints by semantically annotating grasps from a database. These grasps are then mapped onto a new object using local optimization. While this approach is a substantial step towards task-based grasp synthesis, it still requires prior human annotations of a grasp dataset. In a similar vein, Antanas et al. [?] use a probabilistic logic module to semantically reason about pre-grasp configurations with respect to the intended tasks. By exploiting symbolic world knowledge stored in object/task ontologies, they can infer appropriate grasps for a given task. A machine learning approach with a similar goal was introduced by Song et al. [?]. Their approach uses a Bayesian Network to learn task constraints for object manipulation.

The need for sufficient *foresight* during action selection has recently been discussed in the hierarchical planning literature [?]. Levihn and colleagues argue that look-ahead into future plans is important to avoid making short-sighted choices. While close in spirit to the work presented in our paper, the work of Levihn and colleagues concentrates on the task of moving among obstacles and does not include grasp synthesis.

In this paper, we show how inherent shape properties, specifically rotational symmetries, can be used to efficiently generate task-constrained grasps. Object symmetries have long been studied in the computer vision literature []. Recently, various studies have indicated that symmetries can be exploited for grasping [?, ?]. However, these studies have mostly focused on how to generate complete meshes from partial point clouds.

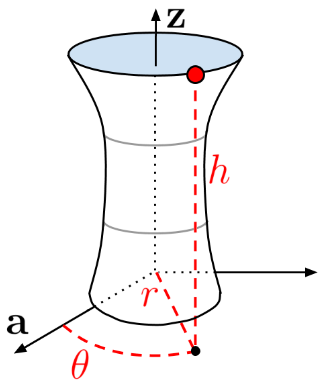


Figure 1: Cylindrical coordinate system to parameterize points on a rotationally symmetric object

3. MANIFOLD REPRESENTATIONS FOR ROTATIONALLY SYMMETRIC OBJECTS

Projecting manipulation constraints such as collisions with the environment and robot manipulator limitations (e.g. reachability) onto object surfaces can facilitate the search for feasible grasps. In this work, we demonstrate that particularly for rotationally symmetric objects, which do not have any surface protrusions or cavities, a number of grasping subproblems such as finger placements, wrist pose, and collision-free inverse kinematics are simplified. Inspired by the texture mapping literature in computer graphics, we begin with the two-dimensional manifold representation of rotationally symmetric objects whose surfaces can be unwrapped into simple planes. The goal is to use this low-dimensional representation to accumulate multiple task constraints in the same local object space and then, efficiently identify grasps that satisfy all the future actions.

3.1 Cylindrical Surface Parametrization

Coordinate spaces characterized by rotations around an axis can be parametrized by cylindrical parameters where a vector $\mathbf{z}$ represents the principal orientation and the polar axis $\mathbf{a}$ captures the secondary direction in the reference plane perpendicular to $\mathbf{z}$. Using their intersection $\mathbf{o}$ as the origin, any point in the Euclidean coordinate system can be represented in this coordinate system by computing the projection of the point onto the $\mathbf{z}$ axis, its distance to the axis, and the angle between the polar axis $\mathbf{a}$ and its projection onto the reference plane.

Figure 1 demonstrates the representation of a point on a rotationally symmetric object in terms of the parameters $[h, r, \theta]^T$ that correspond to the height along $\mathbf{z}$, the radius r of the circle parallel to the reference plane that includes the point, and the angle θ between $\mathbf{a}$ and the point projection onto the reference plan. Note that the reference plane is arbitrarily placed on the bottom surface of the object and the polar axis can be freely chosen within this plane.

A significant observation about the representation of rotationally symmetric objects in cylindrical coordinates is the dependency between the radius r and the height h parameters. For a given shape S, all the points on S, at some height h along the principal axis $\mathbf{z}$, have the same distance r to the axis: $\forall \mathbf{p} \in S$ s.t. $\mathbf{p} \cdot \mathbf{z} = h$, $dist(\mathbf{p}, \mathbf{z}) = r_S(h)$. The function $r_S(h)$ represents the inherent curvature of the shape - for instance, for a cylinder, $r = c$ for some constant c. Based on this observation, we can conclude that any contact point $\mathbf{p}$ on an object shape S can in fact be parameterized minimally by two variables, h and θ. In this work, we show that analyzing the task constraints, robot limitations and contact models in this low-dimensional subspace leads to significant efficiency gains in grasp planning.

3.1.1 *Mapping Euclidean Coordinates to Reduced Cylindrical Space*

To reason about tasks traditionally expressed in Euclidean coordinates, we introduce a mapping between the representation of a point in the Euclidean world frame, p_w, to the $\mathbf{x} = [h, \theta]^T$ parameters in the reduced cylindrical space $\mathbb{S}$. First, we define the local object frame L in the Euclidean coordinate frames by using the principal axis $\mathbf{z}$, the polar axis $\mathbf{a}$, and their cross-product. Given the pose of an object in the world coordinates, T_l^w, we can first transform the point into the local object space, $p^l = T_w^l \, p^w$, and then map it to the surface representation $\mathbf{x}$ using the function $f : \mathbb{R}^3 \rightarrow \mathbb{S}$:

$$f(\mathbf{p}^l) = f([x, y, z]^T) = \begin{bmatrix} z \\ atan2(y, x) \end{bmatrix}. \tag{1}$$

Once a planner reasons about the environment and a contact point is determined in the cylindrical surface representation, we need to map it back to the Euclidean space using an inverse mapping $f^{-1} : \mathbb{S} \rightarrow \mathbb{R}^3$:

$$f^{-1}(\mathbf{x}) = f^{-1}([h, \theta]^T) = \begin{bmatrix} r(h) \; cos(\theta) \\ r(h) \; sin(\theta) \\ h \end{bmatrix}. \tag{2}$$

Fig. 3.2.1 depicts the forward mapping process starting from the representation of a contact point in the world coordinate frame, moving to the local object frame and finally transformed to the cylindrical surface representation. Observe that the surface of the example soda can be unwrapped into a smooth finite two-dimensional plane with minimal distortion because the object is rotationally symmetric, and does not have protrusions nor cavities.

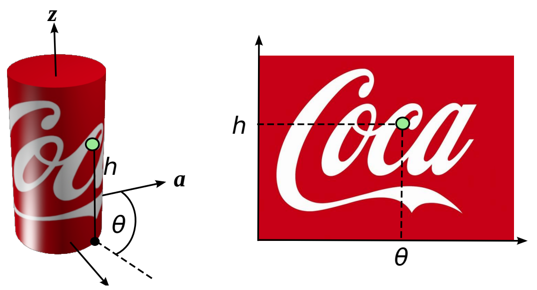


Figure 2: The points on the 3D surface (left) are projected onto a 2D cylindrical manifold.

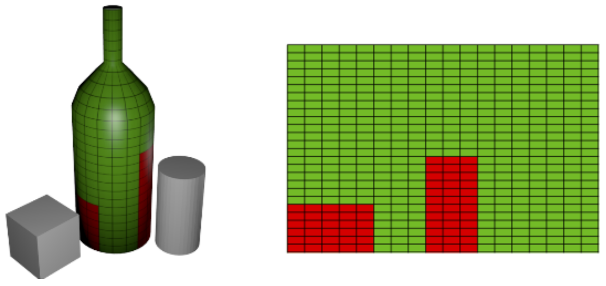


Figure 3: The points on the 3D surface (left) are projected onto a 2D cylindrical manifold.

3.2 Grasp Manifold

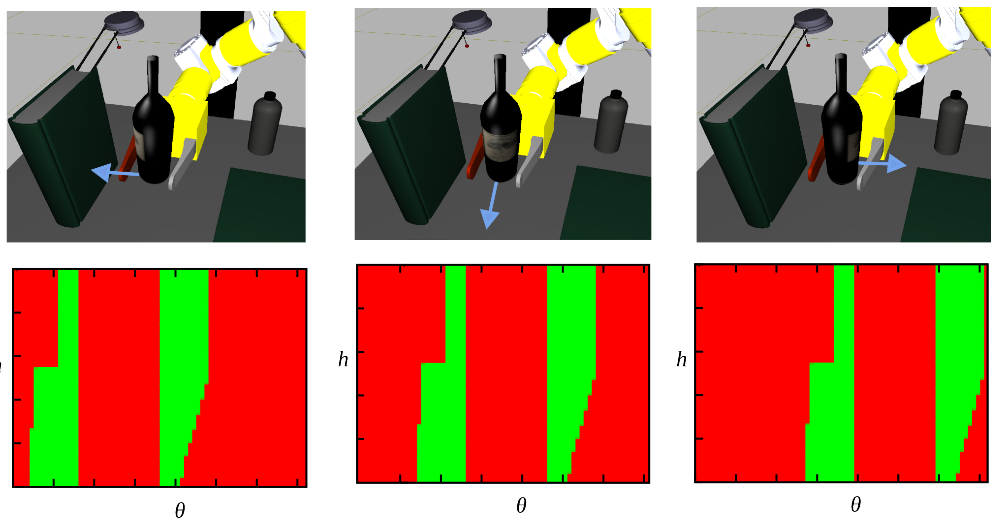
Stable grasps can be rotated around the principle axis of rotationally symmetric objects without having to modify the hand shape. Moreover, the relative rotation between a predefined grasp and an object can be captured with a simple linear motion model in the reduced 2D cylindrical manifold. To demonstrate the advantages of these key ideas, we begin our analysis with a simplified manipulation model where a point on the object surface corresponds to a wrist position during grasping. Moreover, the hand is assumed to be parallel to the base of the object such that the plane between the grasping fingers is perpendicular to the axis of symmetry.

Given a predefined hand shape and a task definition, the planner needs to choose a contact point, $\mathbf{x} = [h, \theta]$, that represents its wrist position. Feasible contact points are characterized by collision-free inverse kinematic solutions where at least one manipulator pose should exist that both reaches the desired point and does not collide with the environment. In addition to the physical feasibility of a contact point, its functionality towards accomplishing tasks also needs to be taken into account in grasp selection. For instance, if an object is picked with the purpose of placing it down in another cluttered environment, collisions in the second scene also need to be accounted for. Similarly, a task may require an object to be configured at a specific pose, i.e. pouring water from a bottle to a cup, in which case the planner should account for the reachability of that pose with the *foresight* that the same grasp would have to be utilized.

3.2.1 *Modeling Task Constraints in Cylindrical Space*

Most everyday grasps require reasoning about multiple tasks even though the tasks may take place in substantially different environments with varying constraints on the pose of the manipulated object. Although the locations and requirements of the tasks may vary, they all impose constraints on the same gripper-object system since once an object is picked, the grasp cannot be changed. We exploit the *invariance* of the gripper-object relationship in manipulation tasks with rotationally symmetric objects, where the task descriptions naturally account for all the degrees of freedom but the rotation around axis of symmetry since the rotation does not effect object properties. The goal is to autonomously exploit the under-constrained task specifications by planning for object rotations and grasp positions.

By only modeling grasps that are perpendicular to the axis of symmetry, e.g. grasp rotates around object, we induce a coupling between gripper and object poses. Now,



given a single pair of gripper and object poses that is known to satisfy task constraints, an infinite number of new pairs can be generated by rotating the gripper around the object by some θ degrees and rotating the object around itself by $-\theta$. Moreover, by determining the feasible set of contacts for an initial object pose which satisfies task requirements, we can extrapolate to a family of object rotations and corresponding contact placements. This extrapolation is only feasible because the rotation of the object around its axis and the rotation of the grasp around the object are coupled by the θ variable around the principle axis $\mathbf{z}$ and thus, we reason about task constraints in the cylindrical parameter space.

The idea is as follows. For each task.... To sample this space, we propose discretizing the manifold by fixed step sizes for the height h and the angle θ. Refer to picture. For each patch of the discretized manifold, using the center point as the reference for the wrist, we attempt to compute a collision-free inverse kinematics solution. Refer to picture again showing red/blue and which collisions were caused.

A crucial property of this representation is that the produced map can be efficiently updated when the object is rotated around its axis of symmetry. More specifically, a rotation around the z-axis corresponds to a shift along the θ dimension.

3.2.2 *Extrapolating task constraints to different object rotations*

The analysis of grasp feasibility for a given object pose and task description can be utilized to reason about grasp selection if the object is rotated only around its axis of symmetry. The idea is that the coupling between the rotation of the grasp around the object pose and the rotation of the object around its own principle axis can be utilized to generate previously examined grasps for new object rotations and to adopt the successful examples. The invariance of task constraint effects on gripper-object systems to interactions between the gripper and the object poses that do not change gripper configuration in the task space can be analyzed formally.

Let the function $m : \mathbb{S}, SE(3) \rightarrow \{0,1\}$ represent a mapping from a 6-DOF object pose $T^w \in SE(3)$ and a contact pose on the object surface, $\mathbf{x} = [h,\theta]^T \in \mathbb{S}$, to a binary evaluation of whether a grasp is feasible. The feasibility of a grasp may depend on a variety of criteria such as inverse kinematics, collisions, torque limits, manipulability and etc. In this work, we focus on collision-free inverse kinematics and define the operator $ik : SO(3), \mathbb{R}^3 \rightarrow \{0,1\}$ that evaluates whether an arm configuration exists that can achieve the end-effector rotation $R_e^w \in SO(3)$ and location $t_e^w \in \mathbb{R}^3$.

We claim that given the mapping of feasibility grasps for an initial object pose T_o^w, the feasibility of any grasp $x \in \mathbb{S}$, for a new pose $T_{o_2}^w$, can be deduced by using the previous mapping:

$$m([h,\theta], T_{o_2}^w) = m([h, \theta+\delta], T_o^w) \tag{3}$$

where $T_{o_2}^w = T_o^w R^z(\delta)$ where $R^z(\delta)$ is a rotation matrix around the local $z-axis$ by angle δ (the notation $T^z(\delta)$ also represents a transformation matrix with the rotation $R^z(\delta)$ but no translation). The claim is proven by showing that the gripper and object rotations counteract each other:

$$\begin{aligned}
&m([h,\theta]^T, T_{o_2}^w) = \\
&\quad = ik(R_{o_2}^w R^z(\theta), \quad T_{o_2}^w f^{-1}([h,\theta]^T)) && (4)\\
&\quad = ik(R_o^w R^z(\delta) R^z(\theta), \; T_{o_2}^w T^z(-\delta) f^{-1}([h,\theta+\delta]^T)) && (5)\\
&\quad = ik(R_o^w R^z(\theta+\delta), \quad T_o^w f^{-1}([h,\theta+\delta]^T)) && (6)\\
&\quad = m([h,\theta+\delta]^T, T_o^w). && (7)
\end{aligned}$$

The proof starts by reducing the mapping to the inverse kinematics problem where the wrist orientation is computed in the world coordinate frame by composing the second object rotation $R_{o_2}^w$ with the contact angle offset θ. Similarly, the end-effector position is computed by composing the second object transform $T_{o_2}^w$ with the contact position in the local coordinates obtained with inverse cylindrical mapping $T^{o_2} = f^{-1}([h,\theta]^T)$. The key insight takes place in Equation 5 where the cylindrical object representation is rotated by δ degrees all the while the pose of the object is rotated in the opposite direction by composing with $T^z(-\delta)$. Equations 6 and 7 simply reorganize the terms to show that the initial feasibility mapping can be reduced from the inverse kinematics formulation.

-

-

4. TASK PLANNING WITH GRASP MANIFOLDS

```
Algorithm 1: CreateMap(): Generates feasibility maps
Input: $\{\hat{h}, \check{h}, \hat{\theta}, \check{\theta}\}$: min/max height and angle limits,
       $\{\delta h, \delta\theta\}$: increments, $R$: robot, $W$: world
Result: $M$: Binary 2D array with feasibility indicators
1  for $h \leftarrow \hat{h}$ to $\check{h}$ do
2      for $\theta \leftarrow \hat{\theta}$ to $\check{\theta}$ do
3          $p^l \leftarrow [rcos(\theta), rsin(\theta), h]^T$;   // Inverse mapping
4          $p^w \leftarrow W.objectPose() * p^l$;   // Goal wrist pose
5          for $\phi \leftarrow 0$ to $2\pi$ do   // Sample redundancy
6              $R.analyticalIK(W.objectPose(), p^w, \phi)$;
7              if $W.collisionFree()$ then
8                  $M[(h - \hat{h})/\delta h][(\theta - \hat{\theta})/\delta\theta] \leftarrow 1$;
9                  jump to line 2;
10         $\theta \leftarrow \theta + \delta\theta$;
11     $h \leftarrow h + \delta h$;
12 return $M$;
```

```
Algorithm 2: Single Grasp for Sequential Tasks
Input: $\{\hat{h}, \check{h}, \hat{\theta}, \check{\theta}\}$: min/max height and angle limits,
       $\{\delta h, \delta\theta\}$: increments, $R$: robot, $W$: world,
       $\mathbb{T}$: object poses for tasks
Result: $\{h, \theta\}$: grasp pose, $\delta_i$: task rotations
1  for $T \in \mathbb{T}$, $i \leftarrow 0$ do   // Generate maps for each task
2      $W.setObjectPose(T)$;
3      $M[i]$ =CreateMap($\hat{h}, \check{h}, \hat{\theta}, \check{\theta}, R, W$);
4      $i \leftarrow i + 1$;
5  for $h \leftarrow \hat{h}$ to $\check{h}$ do   // Iterate through object height
6      $\theta^* \leftarrow -1$;
7      for $\theta \leftarrow \hat{\theta}$ to $\check{\theta}$ do   // Find grasp pose on 1st object
8          if $M[(h - \hat{h})/\delta h][(\theta - \hat{\theta})/\delta\theta] = 1$ then
9              $\theta^* \leftarrow \theta$;
10             break;
11         $\theta \leftarrow \theta + \delta\theta$;
12     if $\theta^* = -1$ then continue;

13     $foundAll \leftarrow true$;
14     for $T \in \mathbb{T}$, $i \leftarrow 0$ do   // Find grasps for each task
15         for $\theta \leftarrow \hat{\theta}$ to $\check{\theta}$ do
16             if $M[(h - \hat{h})/\delta h][(\theta - \hat{\theta})/\delta\theta] = 1$ then
17                 $\delta_i \leftarrow \theta - \theta^*$;
18                 break;
19             $\theta \leftarrow \theta + \delta\theta$;
20         if $\delta_i = \emptyset$ then
21             $foundAll \leftarrow false$;
22             break;
23     if $foundAll$ then return $\{h, \theta^*, \delta_i\ \forall i\}$;
24     $h \leftarrow h + \delta h$;
25 return $\emptyset$;
```

```
Algorithm 3: Collaborative Grasps on Single Task
Input: $\{\hat{h}, \check{h}, \hat{\theta}, \check{\theta}\}$: min/max height and angle limits,
       $\{\delta h, \delta\theta\}$: increments, $\mathbb{R}$: robots, $W$: world,
       $\mathbb{H}$: hand prints
Result: $\{h_i, \theta_i\}$: grasp poses for each robot
1  for $R \in \mathbb{R}$, $i \leftarrow 0$ do   // Generate maps for each robot
2      $M[i]$ =CreateMap($\hat{h}, \check{h}, \hat{\theta}, \check{\theta}, R, W$);
3      $i \leftarrow i + 1$;
4  for $restart \leftarrow 0$ to $MAX\_RESTARTS$ do
5      for robot $R \in \mathbb{R}$, $i \leftarrow 0$ do   // Sample contacts
6          $[h_i, \theta_i] \leftarrow random(\hat{h}, \check{h}, \hat{\theta}, \check{\theta}, \delta h, \delta\theta)$;
7          $i \leftarrow i + 1$;

       // Minimize collisions by moving random grasp
8      for $iters \leftarrow 0$ to $MAX\_ITERS$ do
           // Stop if all the grasps are collision free
9          if $W.collisionFree()$ then return $\{h_i, \theta_i\}$;
10         $R \leftarrow \mathbb{R}[random()]$;   // Choose random robot
11         $dirs \leftarrow [\{\delta h, 0\}, \{0, \delta\theta\}, \{-\delta h, 0\}, \{0, -\delta\theta\}]$;
12         $minNumColls \leftarrow \infty$, $bestDir \leftarrow \emptyset$;
13         for $dir \in dirs$ do
14             $h^* \leftarrow h_R + dir[0]$, $\theta^* \leftarrow \theta_R + dir[1]$;
15             if $M[(h^* - \hat{h})/\delta h][(\theta^* - \hat{\theta})/\delta\theta] = 0$; then
               continue;
16             $numColls \leftarrow$countCollisions($W$);
17             if $numColls < minNumColls$ then
18                 $minNumColls \leftarrow numColls$;
19                 $bestDir \leftarrow dir$;
           // Update grasp with minimum collision direction
20         $h^* \leftarrow h_R + dir[0]$, $\theta^* \leftarrow \theta_R + dir[1]$;
21         $h_r \leftarrow h^*, \theta_r \leftarrow \theta^*$;
22 return $\emptyset$;
```

4.1 Planning for Task Sequences

4.2 Planning for Bi-Manual and Cooperative Tasks

5. EXPERIMENTS

6. DISCUSSION

7. CONCLUSIONS