Object-Centered Hybrid Reasoning for Whole-Body Mobile Manipulation

Daniel Leidner, Alexander Dietrich, Florian Schmidt, Christoph Borst, and Alin Albu-Schäffer

***Abstract*— Many houseworks such as cleaning the floor or wiping the windows require to manipulate tools over wide areas. It is necessary to move along a path while manipulating a tool with the whole body and applying exactly the right amount of force to successfully accomplish the task. So mastering such a challenge demands detailed knowledge about the involved objects and the underlying process models. Reasoning about an appropriate parameterization of the task is thereby essential. In this paper we propose a combination of object-centered hybrid reasoning and compliant force control to solve complex whole-body mobile manipulation issues. Depending on the objects involved in the task, an appropriate controller is selected and automatically parameterized. The methods are validated in an elaborate experiment on the humanoid robot Rollin' Justin.**

I. Introduction

Mobile manipulation is one of the most challenging tasks for a service robot. Among others, it requires orientation and navigation in unstructured environments, simultaneous locomotion and manipulation, handling of objects with particular requirements (grasping strategies, handling instructions, force specifications, ...) as well as compliance with physical constraints (collision avoidance, actuator limits, singularity avoidance, ...). Hybrid reasoning is essential to autonomously solve such a complex task. Detailed knowledge about the environment and the involved objects is required. Especially when a task involves tool usage, handling instructions are mandatory to parameterize the behavior according to applied forces and the desired control behavior.

The most relevant example for such a task is cleaning, where a cleaning device needs to be guided along a dirty surface. Hess *et al.* [1] describe a generic approach to autonomously compute time- and effort-optimal cleaning trajectories. However, they disregard the process of cleaning itself, namely the alignment of the tool, the direction of motion, as well as the applied force and stiffness. In fact, these aspects are crucial for the cleaning result, but can only be taken into account when the reasoning level and the control level act jointly.

The combination of hybrid reasoning, which fuses symbolic with geometric planning, and control theory for whole-body mobile manipulation is a rare research topic. However, there exists some work to incorporate the different research fields. The most common combination is symbolic and geometric reasoning for mobile manipulation. Wolfe *et al.* directly integrate external geometric solvers into a symbolic *Hierarchical Task Network (HTN)* planner [2]. Dornhege *et al.* compute geometric effects during symbolic planning by calling semantic attachment modules [3] for navigation and manipulation [4]. Kaelbling *et al.* focus their work on symbolic and geometric planning under uncertainty for mobile manipulation [5]. *Semantic maps*, as generated from sensor input by Nüchter *et al.* [6], are used by Galindo *et al.* in combination with symbolic planning to command mobile robots to reach a desired goal location [7]. The works of Yamamoto *et al.* [8], Tan *et al.* [9], and us [10], [11] show how mobile manipulation is successfully tackled at the control level. The *Operational Space Formulation* [12] is probably the most popular method to implement force control in a reduced space [13], e. g. in the Cartesian coordinates of the end-effector. However, parameterizing controllers for different tasks without knowledge about the involved objects is difficult. Tenorth *et al.* [14] rely on a web-database for actions, objects and environments to parameterize tasks. Kallmann and Thalmann [15] store articulation trajectories to guide object manipulation. Levison [16] classifies objects by functionality and augments their symbolic domain with hierarchical properties. In our previous work we introduced an object-centered hybrid reasoning framework to solve manipulation tasks [17]. The concept is based on extensive integration of object knowledge during the reasoning process. Belta *et al.* compare environmental-driven discretization and control-driven discretization for automatic construction of robot control strategies from high-level task specifications [18]. Their concepts cover the scope of our work best but are limited to navigation strategies for mobile robots in predefined domains.

Fig. 1. A photo taken from behind a dirty window pane at Hanover Fair in April 2013. The robot Justin executes a cleaning task based on object-centered hybrid reasoning combined with whole-body control strategies.

All authors are affiliated with the Institute of Robotics and Mechatronics, German Aerospace Center (DLR), Wessling, Germany. Contact: daniel.leidner@dlr.de

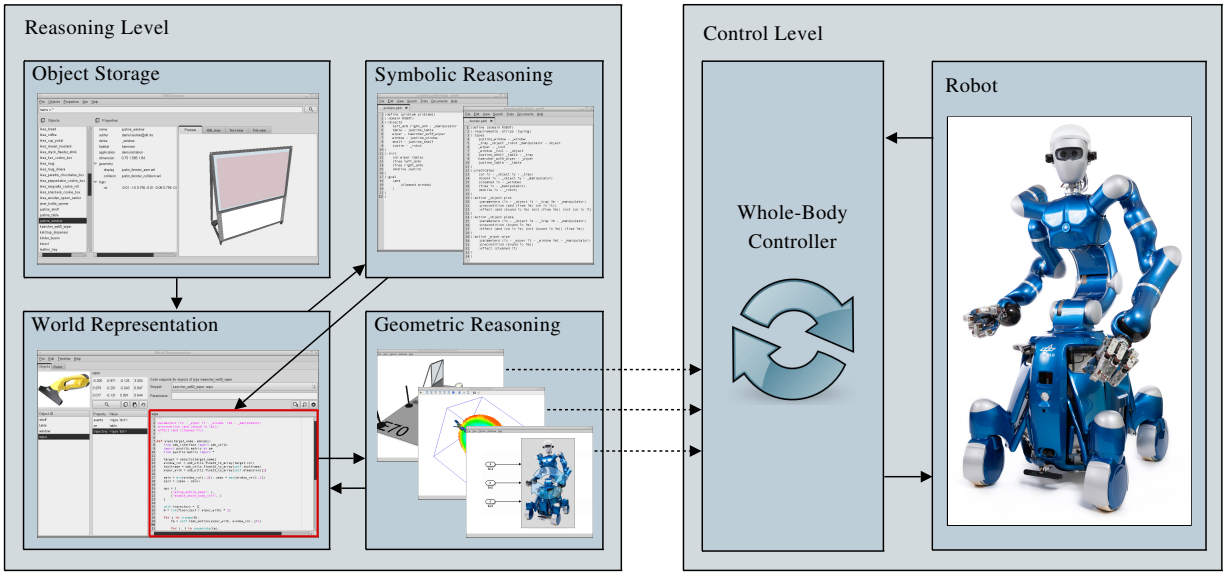

Fig. 2. Schematic of the combined hybrid reasoning and whole-body control methods. The geometric simulations are parameterized according to the object knowledge. The outcome is utilized for whole-body control. Thereby, the redundancy of the robot can be exploited to satisfy additional constraints such as applying object-specific contact forces.

The contribution of this work is a generic combination of hybrid reasoning and compliant control strategies to solve whole-body mobile manipulation tasks w. r. t. given object knowledge. This knowledge is used to parameterize the task on the symbolic level, the geometric level, and the executive control level at the same time. We extend our previous work [17] by modules for reachability analysis and navigation to position a mobile robot optimally according to the given task. Furthermore, whole-body manipulation as well as low-level control features [11] such as impedance control, force-based tool usage, and a hierarchical stack of tasks, are integrated. The whole body of the robot, namely the manipulators, the torso, and the mobile base, act jointly this way. That combination makes it possible for the humanoid Rollin' Justin [19] to autonomously execute a complex service task where mobility and particular applied forces are mandatory, see Fig. 1. The robot identifies a window wiper on the table, grasps it, moves to the window, and cleans the pane while following tool-based force specifications from the object database.

The paper is organized as follows: After a more detailed introduction to preceding work in Sec. II, we extend the methods in Sec. III and implement them on a real humanoid robot in Sec. IV. A discussion on the concept is presented in Sec. V.

II. Preliminary Work

Tool usage tasks are very common in human environments. Most of the tools have to be handled along a particular Cartesian workspace trajectory to accomplish the task. Additionally, a certain amount of force has to be applied along the trajectory. It is quite natural to describe these parameters in the coordinates of the object. Therefore, most robotic control frameworks addressing object manipulation are defined by the desired Cartesian behavior (motion, force, stiffness). Our previous work on whole-body manipulation [11] describes such a framework. An issue is that the parameterization of the low level control behavior depends on high level process models. The combination of both worlds is not straightforward. However, our earlier approach on object-centered hybrid reasoning [17] perfectly fits into this issue. Since the complete action parameterization takes place in the object context, force specifications and other low level control behaviors can be considered within the reasoning process. The underlying methods as illustrated in Fig. 2 are briefly reviewed in this section.

A. Preliminary Work on Hybrid Reasoning

In our previous work [17] we argued that all information needed to solve manipulation tasks can be stored within the descriptions of the involved objects. Therefore, we categorize objects in a hierarchical structure according to their functionality and additionally store process models to define arbitrary manipulation instructions.

An *object storage* provides prior knowledge for all available objects. The objects are hierarchically arranged in the object-oriented paradigm and categorized by functionality. Objects of the same class share the same process models to handle them and can therefore be manipulated in the same way but under consideration of their specific properties such as size and shape. The *world representation* holds the current state for the environment of the robot. Objects as described in the object storage are instantiated here with specific symbolic and geometric properties.

The process models within the object context are described in so-called *action templates* (red in Fig. 2). Action templates consist of two segments. The first segment provides symbolic action definitions for symbolic planning. The second segment

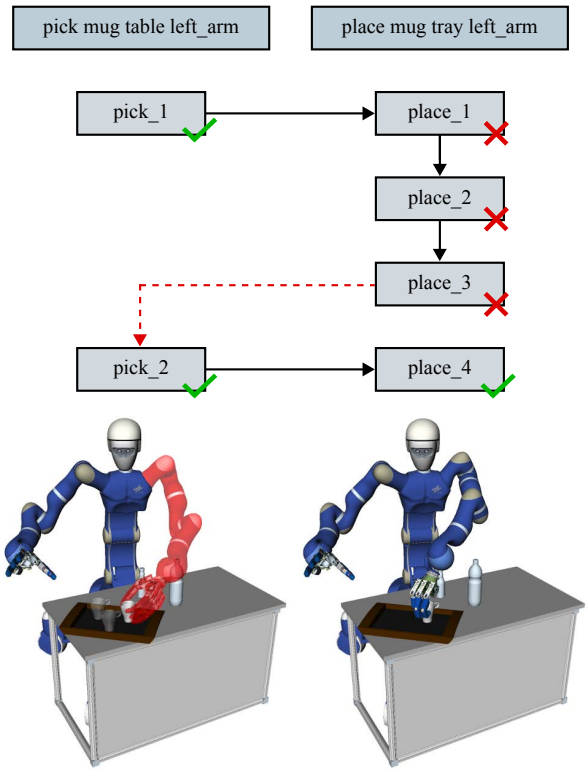

Fig. 3. The geometric reasoning procedure illustrated in a simple pick-and-place example. The procedure is outlined in a block diagram where the symbolic transition is shown at the top (darker blocks) while different geometric alternatives are depicted below (lighter blocks). A side grasp is not suitable to place the mug on the tray since all possible put down alternatives (red crosses) are colliding with the bottles (lower left). After geometric backtracking to the previous pick action (red, dashed arrow), a top grasp is chosen to pick the mug so that putting it down is feasible (lower right). Note that no symbolic alternative is illustrated, but could be envisioned as a pick operation with the right arm.

specifies geometric instructions to implement the symbolic effects by the use of modular geometric simulations such as navigation, motion planning or dynamics simulations. Action templates constitute the main element for our approach on object-centered hybrid reasoning in a two step approach:

First, all symbolic action definitions are gathered from the action templates of the object types currently in the world state. Additionally the matching symbolic properties are collected from the knowledge base. Hence, the symbolic domain is only filled with information of the current state. With this information, a symbolic planner is able to generate a symbolic transition leading to the desired goal state.

In the second reasoning step, the main part of the action templates is revisited to resolve the geometric grounding. Modules to simulate the geometric parameterization can be integrated according to different requirements. If one simulation step succeeds, the next action is simulated until the symbolic transition is processed. Should one step of the geometric simulation fail, the action template is reviewed for geometric alternatives. Should the predefined set of alternatives turn out to be insufficient, geometric backtracking is initiated to find a prior action with remaining alternatives to start over. If the given symbolic transition is not feasible at all, it is removed from the symbolic domain and the symbolic planner is called once again in order to find a symbolic alternative. Fig. 3 illustrates the selection of geometric alternatives as well as the geometric backtracking mechanism in a pick-and-place example. Alternatives for the pick action consist of different grasps while the place action samples different put down positions.

B. Preliminary Work on Whole-Body Control

Although the so far proposed method is capable of reacting on failures during planning time, it is not aware of errors during execution on the real system. The *global* motion planning methods are implemented based on a static initial world state. To react on changes and unforeseen events in the environment and to circumvent errors in the robot perception, it is necessary to integrate *local* reactive behavior during execution time. We have developed whole-body control frameworks [11], [20] matching these requirements. They are characterized by compliant task execution which is well suited for contacts and interaction tasks in dynamic environments.

Multi-redundancy is resolved by applying hierarchy-based control algorithms that involve control subtasks such as self-collision avoidance, Cartesian impedance control of the end-effectors, and joint impedance control of the whole body. A combination of this local behavior with global planning methods (for the trajectories of the control tasks) allows to fuse the advantages of the local approaches (real-time capable, interaction) and the benefits of the global approaches (no local minima, optimized paths) while simultaneously evading the drawbacks of both worlds.

III. Methods

In general, service tasks are distributed over human environments. Involved objects may be located in different rooms and different storage positions. Furthermore, some tasks may require to manipulate an object across wider areas. These tasks can only be fulfilled by coordinated whole-body motions. However, parameterizing the tasks is difficult. We examine the issues in this section by the use of hybrid reasoning which includes impedance-based whole-body manipulation. The questions to be answered are:

- Where and when should a mobile robot move to fulfill a given task?
- How can whole-body motions be parameterized to solve force-sensitive tasks in a generic way?

The first question covers the reasoning about the task on symbolic and geometric level regarding the optimal position and time to move a mobile base. The reasoning is performed according to Sec. III-A. An optimal base placement has direct influence on the second question. It allows for a better parameterization of the control level w. r. t. the task-related objects. Whole-body motions including the manipulators, the torso, and the mobile base, can be executed in a coordinated way to accomplish the given task plus considering additional control tasks as described in Sec. III-B.

A. Mobile Manipulation

Manipulating objects with a mobile robot raises the question of finding a suitable position to place the robot base so that task-related objects are reachable. Zacharias *et al.* suggested to use so-called *capability maps* as external module for symbolic planners to answer this question with geometric background knowledge [21]. Feasible positions to reach for an object under uncertainty are calculated w. r. t. the reachability of the robot by Stulp *et al.* [22]. Furthermore, Cartesian task trajectories such as opening a cupboard can be queried for reachability to get feasible base positions [23]. We extend this idea to not just cover workspace trajectories but whole regions in the robot workspace such as table planes or window panes. The *inverse reachability* approach by Vahrenkamp *et al.* [24] is comparable to our method, but designed to find suitable base positions for a particular 6D target position, rather than a target region. Fig. 4 illustrates the intersection between the capability map of the right manipulator of the robot and the task-related objects. The colors indicate the *reachability index* $r = \frac{R}{N}$ for an individual voxel in the *region of interest (ROI)*, where N is the maximum number of hypothetically reachable discrete positions in the voxel, and R is the number of positions which are actually reachable for the manipulator [21]. Dark red is unreachable ($r = 0.0$) and dark blue is fully reachable ($r = 1.0$). The maximum reachability index r_{max} for the arms of the humanoid robot Rollin' Justin is 0.833. The ROI for the window wiper is defined by the object bounding box, while the ROI for the window is defined by the area of the window pane as it is defined in the object database. The mean reachability r_{roi} for such a region is maximized by adapting the base position $\boldsymbol{p}_{\text{base}}$ and the torso configuration $\boldsymbol{q}_{\text{torso}}$, where $\boldsymbol{x} = (\boldsymbol{p}_{\text{base}}^T, \boldsymbol{q}_{\text{torso}}^T)^T$.

$$\boldsymbol{x}_{\text{max}} = \arg\max_{\boldsymbol{x} \in \mathcal{D}} \; r_{\text{roi}}(\boldsymbol{x}) \tag{1}$$

The solution vector $\boldsymbol{x}_{\text{max}}$ is obtained by maximizing the mean reachability within the region of interest in the restricted domain $\mathcal{D}$, in order to cover a large set of possible geometric alternatives. The restrictions in $\mathcal{D}$ originate from kinematic and dynamic constraints such as collisions, minimum clearance, joint limits and torque limitations. The current state including grasped objects is considered thereby. The initial hypothesis $\boldsymbol{x}_0$ is chosen w. r. t. these restrictions. The initial base position is set to the closest possible position towards the object ROI and the initial torso configuration is set to be upright. Once the optimal solution vector is determined, an A* planning module [25] is applied to plan a trajectory towards the optimal base position.

However, the approach so far reasons only about the topology of a task by defining *where* to move a mobile robot but it does not consider the chronology, *when*. A common solution to tackle this problem on the symbolic level is to augment every symbolic action definition by spatial preconditions. The PDDL description for picking up an object would then look as follows:

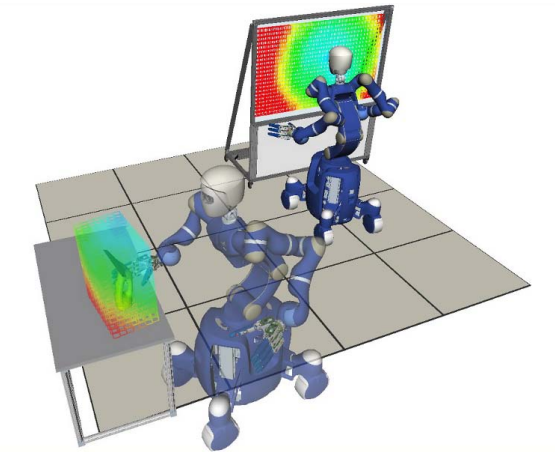

Fig. 4. Optimal intersection of the capability map for the right manipulator and the ROI of a window wiper on the left, and the optimal intersection for the ROI of a window on the right.

```
:action _object.pick
  :parameters (?o - _object ?t - _tray ?m - _manipulator)
  :precondition (and (free ?m)
                     (on ?o ?t)
                     (reachable ?o ?m))
  :effect (and (bound ?o ?m)
               (not (free ?m))
               (not (on ?o ?t)))
```

The symbolic domain can be extended towards mobility by including preconditions for reachability within the related actions (red). This will force a symbolic planner to schedule additional moving actions to satisfy the precondition. However, there is no geometric feedback in pure symbolic planning to reason about this decision. Therefore, we make the assumption that the relevant objects can always be reached by the robot and do *not* add the *reachable* precondition to the symbolic domain. Hybrid reasoning allows to postpone the decision on moving the mobile base to the geometric planning step, in order to only calculate the optimal base position when it is required due to the geometric state. This is done by inserting a call to the capability map module before processing the geometric reasoning part of the action templates. The extended procedure is shown in the pseudocode below.

Algorithm III.1: PROCESSAT($robot, object, action\_template$)

$operations \leftarrow$ PARSEAT($action\_template$)
if $robot.mobile = True$
then {
  $r_{min} \leftarrow 0.5 * robot.r_{max}$
  if CHECKREACHABILITY($object.roi$) $\leq r_{min}$
  then {
    $op0 \leftarrow (\text{'}NavigateToObject\text{'}, object)$
    INSERT($0, op0, operations$)
  }
}

$l \leftarrow$ SIZEOF($operations$)
for $i \leftarrow 0$ **to** $l-1$
do {
  $op \leftarrow operations(i)$
  SIMULATEOP(op)
}

The function *ParseAT* parses the action templates to generate a list of operations. The operation *('NavigateToObject', object)* is inserted in the beginning of the operations list, if the *mobile* attribute of a robot is set. This function plans a

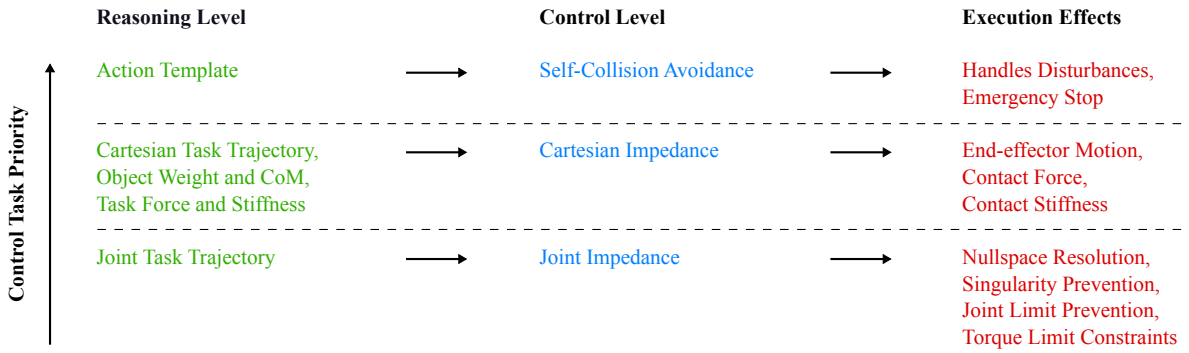

Fig. 5. The control task priorities as used for the window cleaning task. The object properties to parameterize the control level are listed on the left.

feasible base trajectory towards the optimal base position, regarding the optimal object reachability as described earlier in this section. If the actual reachability is higher than the minimal reachability threshold $r_{\text{min}} = 0.5\, r_{\text{max}}$ (more than 50% of the discretized voxelspace covered by the ROI is reachable), the operation to reach the goal will be skipped. Finally, the function *SimulateOP* simulates the operations by calling the external geometric modules which are individually defined by the robot.

Simple pick-and-place scenarios have been successfully evaluated in simulation with a preliminary version of the navigation module [26]. The use case for whole-body manipulation is, however, different: An optimal positioning of the mobile manipulator allows to satisfy additional constraints during task execution. Since the reachability along the task trajectory will be higher, there will be more space to avoid self-collisions, react on external forces, and other limitations as described in the following.

B. Whole-Body Control

Reaching objects for picking them up, carrying them around, and placing them at a different location is a common task in mobile manipulation. In our previous works we have shown how tasks like serving ketchup and more complex bi-manual actions like handling hedge shears can be described within the object context by using action templates [17]. However, some objects require mobility and specified applied forces for task execution. These additional requirements can be expressed within the context of the action templates or directly as object property. Depending on the tasks different control behaviors [11], [20] are activated and parameterized according to relevant object properties.

Wiping a window is a typical example where whole-body mobile manipulation is needed, especially when the window is very large. The workspace trajectory for this task can be predefined. This Cartesian path is parameterized by the type and size of the wiper and the window. It is used to determine *global* joint motions during the simulation step via fast, discrete inverse kinematics. The manipulator, the torso, and the mobile base are thereby synchronized to execute a coordinated, fluent motion to slide the wiper along the window pane. It is sufficient to generate a sparse joint path to verify the global feasibility and additionally provide the desired Cartesian force as input for the local control behavior. Therefore, the computational costs for the calculation are distinctly lower. One can imagine that a window wiper requires different contact forces than a feather duster, for example. Therefore, the knowledge about contact stiffness and force is specified in the object database. The Cartesian task trajectory is intentionally parameterized to traverse far behind the window pane, only the Cartesian force restriction prevents the robot from breaking the glass. Errors in localization and odometry can be corrected this way.

The abundant, remaining redundancy can be utilized to satisfy additional *local* control tasks simultaneously such as collision avoidance, singularity avoidance, desired postures, and so on. An efficient way is to impose a hierarchy among all those control tasks, for example by null space projections [27]. Three control tasks are illustrated in Fig. 5 and have been implemented as follows:

1) Self-Collision Avoidance: The control action $\boldsymbol{\tau}_{\text{coll}}$ [28], [20] derives from repulsive potential fields. It can handle unpredicted events and disturbances at highest priority to prevent otherwise unavoidable self-collisions. Self-collision avoidance can be disabled within the action template when that is required for an action, e. g. bi-manual manipulation of a small object.

2) Cartesian Impedance: It is applied with middle priority to realize the desired end-effector behavior, i. e. guiding the window wiper over the glass while maintaining desired contact stiffness $\boldsymbol{K}_{\text{Cart}}$ and contact force $\boldsymbol{f}_{\text{Cart}}$, which both originate from the object database. They influence the attractive potential denoted by $V_{\text{Cart,imp}}$. Additionally damping can be injected through the p. d. matrix $\boldsymbol{D}_{\text{Cart}}(\boldsymbol{q})$.

$$\boldsymbol{\tau}_{\text{Cart,imp}} = -\left(\frac{\partial V_{\text{Cart,imp}}(\boldsymbol{q}, t, \boldsymbol{K}_{\text{Cart}}, \boldsymbol{f}_{\text{Cart}})}{\partial \boldsymbol{q}}\right)^{T} - \boldsymbol{D}_{\text{Cart}}(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{2}$$

3) Joint Impedance: This control subtask is parameterized with the precomputed joint motion $\boldsymbol{q}_{\text{des}}(t)$, the joint stiffness $\boldsymbol{K}_{\text{joint}}$, and the p. d. damping matrix $\boldsymbol{D}_{\text{joint}}(\boldsymbol{q})$. This is one way to resolve parts of the null space, and several additional constraints such as joint limits and singularity avoidance can be considered at the lowest priority.

$$\boldsymbol{\tau}_{\text{joint,imp}} = \boldsymbol{K}_{\text{joint}}\left(\boldsymbol{q}_{\text{des}}(t) - \boldsymbol{q}\right) - \boldsymbol{D}_{\text{joint}}(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{3}$$

4) Whole-Body Control Law: The overall control law combines all control subtasks by applying null space projectors. The null space of the self-collision avoidance is defined by $\boldsymbol{N}_1(\boldsymbol{q})$ and the null space projector of the first two priority levels (self-collision avoidance and Cartesian impedance) is given by $\boldsymbol{N}_2(\boldsymbol{q})$. Gravity compensation is implemented via the model-based control action $\boldsymbol{g}(\boldsymbol{q})$.

$$\boldsymbol{\tau} = \boldsymbol{\tau}_{\text{coll}} + \boldsymbol{N}_1(\boldsymbol{q})\boldsymbol{\tau}_{\text{Cart,imp}} + \boldsymbol{N}_2(\boldsymbol{q})\boldsymbol{\tau}_{\text{joint,imp}} + \boldsymbol{g}(\boldsymbol{q}) \quad (4)$$

This way of combining global motion generation and reactive, local control task execution leads to a robust force-sensitive behavior. The Cartesian impedance controller is hereby parameterized to successfully implement the commanded high-level contact behavior while further disturbances are compensated by the joint impedance controller. Self-collisions are avoided at highest priority in case of emergency. All information needed to parameterize the control level is gathered from the object knowledge as illustrated in Fig. 5.

IV. Evaluation

This work describes the integration of high-level symbolic and geometric reasoning with low-level control behaviors by the use of a generic object-centered manipulation framework. The performance of the whole system is best expressed by the level of autonomy a service robot can reach, and the complexity of the task it can solve with the proposed framework. Therefore, we have chosen a whole-body tool usage scenario in which the mobile humanoid robot Rollin' Justin has to clean a window with a wiper that is initially located on a table (see Fig. 6). Additionally, statistics on the computation times for the individual planning steps are provided in Tab. I.

Task-relevant information such as size of the window pane, the size of the wiper, as well as the forces to be applied are stored within the object context of the involved objects. The initial positioning of the mobile base and the torso is optimized by the use of the capability map module and spatial object information. The task that is commanded to the robot is only defined as the desired symbolic goal state *"cleaned window"*. With the initial state and the resulting symbolic preconditions and effects, the symbolic transition to solve the task is determined as follows:

```
_object.pick wiper table right_arm,
_wiper.wipe window right_arm
```

The symbolic transition does not include geometric information about the position of the objects. Moreover, it is not explicitly mentioned that the robot has to navigate in between the actions to solve the task. The reasoning about this is done during the subsequent geometric simulation according to the action-relevant object knowledge.

With a width of $1.5\,\text{m}$ and a height of $1.0\,\text{m}$, the window pane is too large to follow the Cartesian task motion by only using the arm of the robot. Even with the aid of the torso it is not possible to maintain the contact with the window along the complete trajectory. Especially in the corners the reachability index decreases until the task gets unfeasible, not to mention the reduced capability to compensate for possible disturbances during task execution. Consequently, the mobile base is mandatory to accomplish the task. The additional degrees of freedom of the mobile base can be considered by the applied discrete inverse kinematics solver. This way, the base follows the lateral movement of the end-effector, respectively the window wiper, along the window pane to extend the workspace of the robot.

Two plots illustrate the task execution. The first plot (Fig. 7) shows the commanded six-dimensional task trajectory of the window wiper blade. It is represented as overlay over the simulated window pane. To demonstrate the varying control behaviors, the wiper is moved from left to right with applied forces on the window pane, while it is moved without contact but increased stiffness in the opposite direction. The second plot (Fig. 8) illustrates the reachability index r for the end-effector position of the right manipulator for all wiping motions along the window pane. It can be seen

Fig. 6. The experiment as shown at the Hanover Fair 2013. The table on which the wiper was initially located can be seen in the back while Justin is cleaning the window with the wiper in the front.

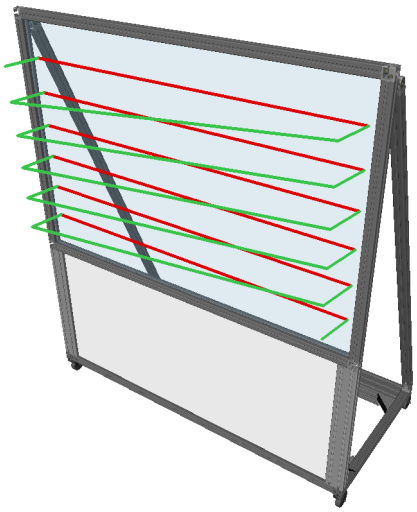

Fig. 7. The workspace trajectory for wiping the window. Red segments are touching the window and apply forces. Green segments have no contact.

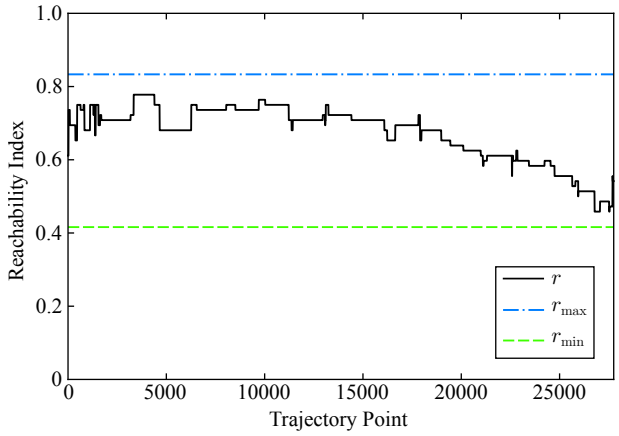

Fig. 8. The solid, black plot indicates the discrete reachability index r for the end-effector along the complete trajectory of Fig. 7. It is close to the maximum value $r_{\max}$ (blue, chain-dotted) and never below $r_{\min} = 0.5\, r_{\max}$ (green, dashed) along the trajectory due to optimal base position and torso configuration. For comparison, the colors for the thresholds correspond to the color scheme used for visualizing the capability map in Fig. 4.

that the reachability is relatively high along the complete trajectory. It is close to the maximum reachability $r_{\max}$ and never below $r_{\min} = 0.5\, r_{\max}$. Maintaining such a high reachability provides more space for reactivity, potential evasion movements, disturbances to be compensated, and additional control tasks that have to be executed. This is achieved due to the optimal positioning of the robot w. r. t. the maximized reachability r_{roi} before the actual execution. The reachability index decreases over time since the lower part of the window pane is harder to reach for the robotic manipulator. However, the task maintains feasible unless the reachability index converges to zero.

The experiment was repeatedly executed in simulation on an Intel Xeon CPU with 8 Cores at 2.80 GHz and 8192 KB cache. Fast Downward [29] was used for symbolic planning and OpenRAVE [30] for geometric planning and simulation. The resulting statistical computation times for the complete experiment are listed in Tab. I. Due to the two step hybrid reasoning approach, the symbolic planning time is measured 0.11 s in average, and thus not listed individually. The computational load is mainly generated during geometric reasoning. The navigation planning with the A* algorithm takes place on a 0.05 m grid. The computation times for the optimization of the final base and torso position arise from the restrictions in the domain $\mathcal{D}$ including numerous collision and clearance checks. The combined motion planning time for the pick action consists of the planning times for reaching for the wiper, lifting it, and planning to the default positions. The motion planning time for the wiping action is dominated by inverse kinematics computations in order to obtain a whole-body joint motion.

The experiment shows that the concept of action templates is suitable to describe the process model for wiping the window w. r. t. whole-body control behaviors of a humanoid robot. It is shown that various components can be addressed to solve even complex mobile and whole-body manipulation

TABLE I

COMPUTATION TIMES FOR THE GEOMETRIC REASONING STEPS.

	_object.pick wiper table right_arm	_wiper.wipe window right_arm
navigation planning	1.21 s	1.17 s
base optimization	1.25 s	0.84 s
motion planning	1.49 s	2.45 s

tasks rather than only mobile pick-and-place tasks. All information needed to describe the task was stored within the context of the involved objects. The reasoning was autonomously performed by the robot with the use of the provided action templates. The experiment was successfully demonstrated on a daily basis to a public audience during Hanover Fair in April 2013 as seen in Fig. 6. An extensive video of the application is attached to this paper.

V. DISCUSSION

Combining hybrid reasoning and whole-body control by the use of the proposed architecture has several advantages. The main benefit is that the way of programming manipulation tasks within the proposed architecture brings a shift from the robot point of view to the object point of view. This is very beneficial for the engineering process. The programmer gets a different view on the problem which results in a different way of solving the task. The functionality of the objects gets into focus and the task parameterization takes place in a natural way by using the properties of the related objects, such as task trajectories and forces to be applied.

Maintaining a high reachability along a trajectory indicates a high reactive potential. However, it is not guaranteed that joint limits and singularities can be circumvented while generating joint motions w. r. t. a predefined workspace trajectory. Using a discrete inverse kinematics to calculate global joint motions may thus still lead to local minima since both planning and execution are based on local methods. Huaman *et al.* [31] resolve joint motions out of a given Cartesian trajectory based on a discretized Jacobian null space and a backtracking mechanism to prevent local minima and additionally avoid obstacles. However, the approach has not yet been applied to a mobile manipulator. It has to be investigated if the approach can be extended by the additional degrees of freedom of a mobile base and a torso, without getting impractical due to increased planning times.

A generic parameterization of robotic control behaviors is desirable. Depending on the task different behaviors such as variable stiffness or local collision avoidance strategies may be required. However, although the proposed methods allow for such a parameterization, action templates tend to get more complex the more complex the task itself is. The process models as well as the object information need to be defined carefully. A knowledge-based, fully autonomous selection of control strategies and their parameterization is thus subject to future work. We plan to integrate additional modules for dynamic simulation and more accurate robot models to collect the required information.

VI. Summary

In this paper we presented a generic approach to combine object-centered hybrid reasoning and compliant control strategies for whole-body mobile manipulation. Object knowledge was used to parameterize mobile manipulation tasks on a symbolic and geometric level. Capability maps have been introduced to the framework to autonomously solve for the optimal positioning of a mobile robot. Furthermore, impedance-based control strategies for whole-body manipulation have been integrated and parameterized according to a given task and the related object information. An area-wide tool-usage experiment was realized to evaluate the approaches. The attached video clearly demonstrated the potential of the proposed concept.

VII. Acknowledgments

The authors would like to thank Werner Friedl and Thomas Wimböck for their support during Hanover Fair 2013.

References

[1] J. M. Hess, G. D. Tipaldi, and W. Burgard, "Null space optimization for effective coverage of 3d surfaces using redundant manipulators," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 1923–1928.

[2] J. Wolfe, B. Marthi, and S. J. Russell, "Combined task and motion planning for mobile manipulation." in *Proc. of the International Conference on Automated Planning and Scheduling (ICAPS)*, 2010, pp. 254–258.

[3] C. Dornhege, P. Eyerich, T. Keller, S. Trüg, M. Brenner, and B. Nebel, "Semantic attachments for domain-independent planning systems," in *Towards Service Robots for Everyday Environments*. Springer, 2012, pp. 99–115.

[4] C. Dornhege and A. Hertle, "Integrated symbolic planning in the tidyup-robot project," in *AAAI Spring Symposium Series*, 2013.

[5] L. P. Kaelbling and T. Lozano-Pérez, "Integrated task and motion planning in belief space," *International Journal of Robotics Research*, vol. 32, pp. 1–60, 2013.

[6] A. Nüchter and J. Hertzberg, "Towards semantic maps for mobile robots," *Robotics and Autonomous Systems*, vol. 56, no. 11, pp. 915–926, 2008.

[7] C. Galindo, J.-A. Fernández-Madrigal, J. González, and A. Saffiotti, "Robot task planning using semantic maps," *Robotics and Autonomous Systems*, vol. 56, no. 11, pp. 955–966, 2008.

[8] Y. Yamamoto and X. Yun, "Coordinating locomotion and manipulation of a mobile manipulator," in *Proc. of the IEEE Conference on Decision and Control (CDC)*, 1992, pp. 2643–2648.

[9] J. Tan, N. Xi, and Y. Wang, "Integrated task planning and control for mobile manipulators," *The International Journal of Robotics Research*, vol. 22, no. 5, pp. 337–354, 2003.

[10] A. Dietrich, T. Wimböck, and A. Albu-Schäffer, "Dynamic Whole-Body Mobile Manipulation with a Torque Controlled Humanoid Robot via Impedance Control Laws," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011, pp. 3199–3206.

[11] A. Dietrich, T. Wimböck, A. Albu-Schäffer, and G. Hirzinger, "Reactive Whole-Body Control: Dynamic Mobile Manipulation Using a Large Number of Actuated Degrees of Freedom," *IEEE Robotics & Automation Magazine*, vol. 19, no. 2, pp. 20–33, 2012.

[12] O. Khatib, "A Unified Approach for Motion and Force Control of Robot Manipulators: The Operational Space Formulation," *IEEE Journal of Robotics and Automation*, vol. RA-3, no. 1, pp. 43–53, 1987.

[13] L. Sentis and O. Khatib, "Synthesis of Whole-Body Behaviors through Hierarchical Control of Behavioral Primitives," *International Journal of Humanoid Robotics*, vol. 2, no. 4, pp. 505–518, 2005.

[14] M. Tenorth, A. Perzylo, R. Lafrenz, and M. Beetz, "The roboearth language: Representing and exchanging knowledge about actions, objects, and environments," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2012, pp. 1284–1289.

[15] M. Kallmann and D. Thalmann, "Modeling objects for interaction tasks," in *Computer Animation and Simulation 98*. Springer, 1999, pp. 73–86.

[16] L. Levison, "Connecting planning and acting via object-specific reasoning," Ph.D. dissertation, University of Pennsylvania, 1996.

[17] D. Leidner, C. Borst, and G. Hirzinger, "Things are made for what they are: Solving manipulation tasks by using functional object classes," in *Proc. of the IEEE/RAS International Conference on Humanoid Robots (ICHR)*, 2012, pp. 429–435.

[18] C. Belta, A. Bicchi, M. Egerstedt, E. Frazzoli, E. Klavins, and G. J. Pappas, "Symbolic planning and control of robot motion [grand challenges of robotics]," *Robotics & Automation Magazine, IEEE*, vol. 14, no. 1, pp. 61–70, 2007.

[19] C. Borst, T. Wimbock, F. Schmidt, M. Fuchs, B. Brunner, F. Zacharias, P. R. Giordano, R. Konietschke, W. Sepp, S. Fuchs, *et al.*, "Rollin'justin-mobile platform with variable base," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2009, pp. 1597–1598.

[20] A. Dietrich, T. Wimböck, A. Albu-Schäffer, and G. Hirzinger, "Integration of Reactive, Torque-Based Self-Collision Avoidance Into a Task Hierarchy," *IEEE Transactions on Robotics*, vol. 28, no. 6, pp. 1278–1293, 2012.

[21] F. Zacharias, C. Borst, and G. Hirzinger, "Capturing robot workspace structure: representing robot capabilities," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2007, pp. 3229–3236.

[22] F. Stulp, A. Fedrizzi, L. Mösenlechner, and M. Beetz, "Learning and reasoning with action-related places for robust mobile manipulation," *Journal of Artificial Intelligence Research*, vol. 43, no. 1, pp. 1–42, 2012.

[23] F. Zacharias, W. Sepp, C. Borst, and G. Hirzinger, "Using a model of the reachable workspace to position mobile manipulators for 3-d trajectories," in *Proc. of the IEEE/RAS International Conference on Humanoid Robots (ICHR)*, 2009, pp. 55–61.

[24] N. Vahrenkamp, T. Asfour, and R. Dillmann, "Robot placement based on reachability inversion," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2013, pp. 1962–1967.

[25] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[26] D. Leidner and C. Borst, "Hybrid reasoning for mobile manipulation based on object knowledge," in *Workshop on AI-based Robotics at IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[27] B. Siciliano and J.-J. Slotine, "A General Framework for Managing Multiple Tasks in Highly Redundant Robotic Systems," in *Proc. of the International Conference on Advanced Robotics (ICAR)*, 1991, pp. 1211–1216.

[28] A. Dietrich, T. Wimböck, H. Täubig, A. Albu-Schäffer, and G. Hirzinger, "Extensions to Reactive Self-Collision Avoidance for Torque and Position Controlled Humanoids," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 3455–3462.

[29] M. Helmert, "The fast downward planning system," *Journal of Artificial Intelligence Research*, vol. 26, pp. 191–246, 2006.

[30] R. Diankov, "Automated construction of robotic manipulation programs," Ph.D. dissertation, Carnegie Mellon University, Robotics Institute, 2010.

[31] A. Huaman and M. Stilman, "Deterministic motion planning for redundant robots along end-effector paths," in *Proc. of the International Conference on Humanoid Robots (ICHR)*, 2012, pp. 785–790.