Whole-Body Motion Planning for Manipulation of Articulated Objects

Felix Burget Armin Hornung Maren Bennewitz

***Abstract*— Humanoid service robots performing complex object manipulation tasks need to plan whole-body motions that satisfy a variety of constraints: The robot must keep its balance, self-collisions and collisions with obstacles in the environment must be avoided and, if applicable, the trajectory of the end-effector must follow the constrained motion of a manipulated object in Cartesian space. These constraints and the high number of degrees of freedom make whole-body motion planning for humanoids a challenging problem. In this paper, we present an approach to whole-body motion planning with a focus on the manipulation of articulated objects such as doors and drawers. Our approach is based on rapidly-exploring random trees in combination with inverse kinematics and considers all required constraints during the search. Models of articulated objects hereby generate hand poses for sampled configurations along the trajectory of the object handle. We thoroughly evaluated our planning system and present experiments with a Nao humanoid opening a drawer, a door, and picking up an object. The experiments demonstrate the ability of our framework to generate solutions to complex planning problems and furthermore show that these plans can be reliably executed even on a low-cost humanoid platform.**

I. Introduction

Due to their dexterity, humanoid robots are well-suited for mobile manipulation tasks in complex environments designed for humans, e.g., containing stairs or slopes in domestic, industrial, or disaster recovery settings. Humanoid service robots need to interact with a variety of objects, i.e., not only objects to be picked up, but also articulated objects such as doors and drawers that have to be manipulated. Accordingly, coordinated motions of the lower and upper body parts need to be planned to successfully execute such complex tasks (see Fig. 1). Two challenges arise when generating whole-body motions for bipedal humanoids in this context. First, motions for a high number of degrees of freedom (DOF) have to be planned. Second, a variety of constraints have to be satisfied: Joint limits must be respected, self-collisions as well as collisions with obstacles in the environment must be avoided, and the robot must keep its balance. Furthermore, constraints induced from the objects to be manipulated must be taken into account.

In this paper, we present an extension of the RRT-Connect planner [1] that generates solutions satisfying all required constraints and applies inverse kinematics (IK) for single chains of joints within the randomized search. One focus of our work hereby lies in manipulating articulated objects. Thus, the robot must particularly keep contact with the object's handle to manipulate it. Here, the key idea of our approach is to force the hand poses of sampled configurations to follow the trajectory of the object handle.

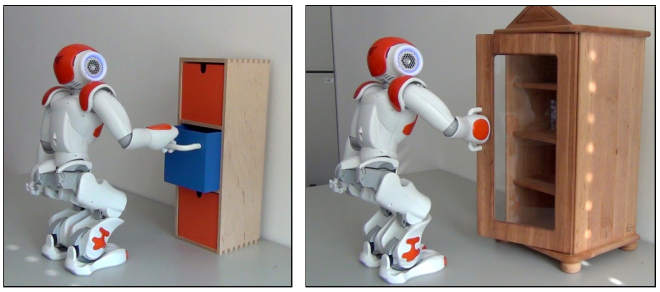

Fig. 1. A Nao humanoid manipulates articulated objects using our whole-body motion planning approach. The motion plan respects the constraints of the robot's stability while enforcing an end-effector trajectory given by the articulation model. By using all joints of the robot's body instead of just the arms, its manipulation range is much increased.

In contrast to techniques for manipulation of articulated objects using wheeled platforms equipped with a manipulator [2], [3], [4], the planning problem is more complex due to the high number of DOF and additional constraints regarding the balance of the humanoid system. While also Jacobian-based methods have been presented to generate whole-body motions [5], [6], [7], randomized, sampling-based approaches have the ability to cope with arbitrary collision environments also containing local minima.

The contribution of our work is a planning system for whole-body motions that considers general constraints during the search by combining the randomized RRT-based search with IK. We present experiments with a Nao humanoid illustrating that our approach can generate solutions to complex whole-body manipulation tasks. We particularly evaluate the manipulation of articulated objects. As the experiments show, the computed end-effector trajectories allow the robot to successfully open doors and drawers and can be reliably executed even on a low-cost platform. Our approach has been implemented in the *MoveIt!* framework in ROS [8] and is available open source at `http://www.ros.org/wiki/wholebody_planning`.

II. Related Work

Approaches to generate whole-body motions for complex, high-DOF robots can be classified into either Jacobian-based methods or randomized, sampling-based motion planning techniques. Although approaches belonging to the former category rather deal with online motion generation, many of their concepts can be inherited for planning motions offline.

A. Hornung and M. Bennewitz are with the Humanoid Robots Lab, University of Freiburg, Germany. F. Burget is with the Dipartimento di Informatica e Sistematica, Università di Roma "La Sapienza", Italy.

This work has been supported by the German Research Foundation (DFG) within the SFB/TR-8 *Spatial Cognition* and the Cluster of Excellence *BrainLinks-BrainTools* (EXC1086).

Using generalized IK with task-specific Jacobians, joint velocities that minimize given task functions can be iteratively computed. For example, Kanoun *et al.* [5] plan local motions for a humanoid robot based on the prioritized kinematic control scheme. The approach has been applied successfully on the humanoid robot platform HRP-2 for the task of reaching a ball underneath an object. The authors have recently extended the approach to locomotion by modeling footsteps as a chain of rotational and prismatic joints in an extended virtual kinematic structure [9]. This approach, however, bears the risk of the virtual chain being trapped in local minima, as generally encountered with numerical optimization methods. This especially occurs when planning collision-free motions in narrow environments with cavities.

Yoshida *et al.* [7] presented an approach that couples generalized IK with a dynamic walking pattern generator, also based on prioritized tasks. Mansard *et al.* [6] proposed to implement visual servoing to generate whole-body motions for a humanoid robot. The authors split the control into several sensor-based control tasks that are executed simultaneously. With control at the task level, the HRP-2 was able to grasp an object while walking. However, avoiding obstacles and self-collisions has not been taken into account.

As an alternative, rapidly-exploring random trees (RRTs) and probabilistic roadmaps are proven methods to plan collision-free paths in high-dimensional configuration spaces [10], [11]. Kuffner *et al.* [12] were the first who applied probabilistic motion planning to generate whole-body motions for humanoids. The authors presented a variant of the RRT-Connect planner [1] and introduced a new sampling strategy. We extend this work towards manipulation of articulated objects, where the end-effector trajectory is induced by the object motion constraints.

Also Stilman [13] presented an approach to deal with task constraints as restrictions on the end-effector motion. The local planner used in the tree expansion process is enhanced by projecting the sampled configurations on a submanifold that solves the task. The technique was applied to a manipulator on a mobile base opening doors and drawers. A limitation of this work is that it only considers constraints regarding the relative motion of the end effector with respect to a fixed world frame and does not consider constraints regarding stability. Berenson *et al.* [14] follow a similar approach. In their work, they introduce the Constrained Bidirectional RRT planner that considers constraints as task space regions (TSRs), a concept previously used for goal-specification. To describe more complex constraints, these TSRs can be chained together. Results for motion planning considering stability, manipulation and closure constraints are presented. Kanehiro *et al.* [16] present an approach composed of a planning and execution phase. A coarse plan is quickly generated with RRT-Connect and analytical IK. This plan is executed online while compensating approximation errors in real-time and maintaining constraints. Although capable of reaching a target object, the approach does not consider a subsequent manipulation of the object at this stage. Oriolo and Vendittelli [17] proposed a control-based approach to task-constrained motion planning. As opposed to previous methods, their RRT planner makes use of a Jacobian-based motion generation scheme that allows for continuous constraint satisfaction. Closure or stability constraints are currently not considered. Also Dalibard *et al.* [18] combined local Jacobian-based methods with randomized motion planning for humanoids. Their local planner employs a prioritized pseudo-inverse technique [7] to generate goal configurations prior to planning and to project randomly generated samples onto the constraint manifold. In a latter work [19], the authors planned door opening actions with a humanoid in simulations. The global planner used an RRT to plan for the three DOF of the robot's bounding box.

Nozawa *et al.* [20] integrated a constrained end-effector trajectory into the walking controller of the humanoid HRP-2. This is suitable for pushing objects, however, it is unclear how collision avoidance can be integrated.

Our approach plans whole-body motions for complex object manipulation tasks with RRT-Connect in combination with IK. The system takes into account all required constraints, e.g., stability, collision-freeness, and keeping contact with the articulated object to be manipulated, that are necessary to carry out the task successfully.

III. Whole-Body Motion Planning for Object Manipulation

Our system builds upon the RRT-Connect algorithm [1], which has already demonstrated the ability to efficiently find solutions to planning problems in high-dimensional domains. The basic idea of RRT-Connect is to grow two search trees, one from the start and one from the goal configuration. The search trees are iteratively connected by randomly sampling configurations and extending the trees. In the following, we describe our variant of RRT-Connect for whole-body motion planning under stability and manipulation constraints.

A. The Extended RRT-Connect Algorithm

When whole-body motions for manipulation actions have to be planned, several constraints need to be considered during the search. For example, stability of the humanoid needs to be ensured and manipulation constraints must be taken into account such that the robot's hand is attached to an articulated object to be manipulated. These constraints define a subspace in the search space, also referred to as constraint manifold. When randomly sampling configurations during the search, most of them will not be on the constraint manifold and have to be rejected, e.g. because they cause the robot to lose its balance. This results in a poor coverage of the constrained configuration space. One solution is to compute an approximation of the constraint manifold offline, and then directly sample from this approximation during planning [15]. Collision avoidance with the environment and the end-effector trajectory for an articulated object are usually task-specific while a humanoid's geometry and weight distribution are not. Thus, our system precomputes the constraint manifold as a set of statically stable configurations that are free of self-collisions. All other constraints

Algorithm 1: Extended RRT-Connect (q_{start}, q_{goal}, *object_params*)

```
1  n_s.q ← q_start
2  n_g.q ← q_goal
3  if object_grasped = true then
4  |   ⟨h_0, ..., h_k⟩ ← GEN_HAND_TRAJ(q_start, object_params)
5  |   n_s.h ← 0
6  |   n_g.h ← k
7  T_a.init(n_s); T_b.init(n_g);
8  for i = 1 to max_iter do
9  |   q_rand ← RAND_DS_CONFIG(DS_DATABASE)
10 |   if not (EXTEND(T_a, q_rand) = TRAPPED) then
11 |   |   if (EXTEND(T_b, T_a.last.q) = REACHED) then
12 |   |   |   if object_grasped = false then
13 |   |   |   |   PATH(T_a, T_b) ← SMOOTH_PATH(T_a, T_b)
14 |   |   |   return PATH(T_a, T_b)
15 |   SWAP(T_a, T_b)
16 return FAILURE
```

Algorithm 2: EXTEND ($\mathcal{T}$, q_{ref})

```
1  n_near ← FIND_NEAREST_NEIGHBOR(T, q_ref)
2  n_new.q ← NEW_CONFIG(q_ref, n_near.q)
3  if object_grasped = true then
4  |   n_new ← ENFORCE_MANIP_CONSTRAINT(n_new, n_near)
5  if IS_CONFIG_VALID(n_new.q) then
6  |   T.add_node(n_new)
7  |   T.add_link(n_near, n_new)
8  |   if n_new.q = q_ref then return REACHED
9  |   else return ADVANCED
10 return TRAPPED
```

are considered in the planning process, which samples from the constraint manifold. In the following, we first explain the basic functionality of the planner in the absence of manipulation constraints.

Alg. 1 shows the pseudocode of our algorithm. Each element of the search trees contains a whole-body configuration q and additional information related to the desired object handle trajectory (that is ignored for now). We will refer to the elements of a tree as nodes n. As input, the planner takes two collision-free statically stable double support configurations q_{start} and q_{goal}. While the start configuration usually corresponds to the current state of the robot, the goal configuration is generally not known in advance but has to be generated from the task constraints as described in Sec. III-C. The third input argument *object_params* is only required when an interaction with an object is desired (Lines 3-6 in Alg. 1, see Sec. III-D).

After initialization (Line 7 of Alg. 1), two search trees $\mathcal{T}_a$ and $\mathcal{T}_b$ are grown from q_{start} and q_{goal}. Hereby, at each iteration the function RAND_DS_CONFIG returns a random statically stable pose q_{rand} from the precomputed constraint manifold database *DS_DATABASE*. First, $\mathcal{T}_a$ is expanded by the EXTEND function, described in Alg. 2. This function finds q_{near}, the nearest configuration to q_{rand} in $\mathcal{T}_a$, and tries to extend $\mathcal{T}_a$ by a new configuration q_{new} generated by NEW_CONFIG at a distance ε from q_{near} towards q_{rand} (Line 2 of Alg. 2). An example for a single expansion step is illustrated in Fig. 2. The algorithm then checks whether q_{new} satisfies all required constraints in IS_CONFIG_VALID. If this is not the case, q_{new} is rejected and Alg. 1 proceeds by swapping the two trees, i.e., tree $\mathcal{T}_a$ becomes $\mathcal{T}_b$ and vice versa. In the next iteration, a new configuration is randomly chosen from *DS_DATABASE* and $\mathcal{T}_a$ is expanded. Otherwise, i.e., if q_{new} is valid, it is added to the search tree and $\mathcal{T}_b$ is extended towards the newly added node (Line 11 of Alg. 1). If q_{new} of $\mathcal{T}_a$ (which is now denoted as $\mathcal{T}_a.last.q$ in Alg. 1) is reached from $\mathcal{T}_b$, a valid path has been found. If an invalid configuration has been encountered, the algorithm proceeds with the next iteration as explained above.

This procedure is repeated until a path has been found,

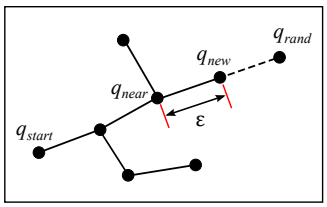

Fig. 2. Example of a tree expansion step.

i.e., the two trees are connected, or a maximum number of iterations has been reached. When the search was successful, the solution path is smoothed by removing extraneous nodes while respecting collision-freeness. This is, however, only admissible for motion plans regarding posture changes, because smoothing of a manipulation motion plan could cause a violation of the manipulation constraints defined by the articulated object in Cartesian space.

Besides checking for self-collision and collision with obstacles, IS_CONFIG_VALID (Alg. 2, Line 5) verifies whether a configuration is statically stable and the manipulation constraints are satisfied.

B. Precomputing Stable Configurations

Inspired by the idea of Kuffner *et al.* [12], we use a precomputed set of stable whole-body configurations as constraint manifold approximation, from which the planner then draws samples [15]. Our approach builds this set by sampling whole-body configurations respecting the joint limits. In the sampled configurations, the leg joint angles are adjusted so that the robot is in double support mode. To do so, our system chooses the frame of the right foot $\mathcal{F}_{rfoot}$ as the root of the kinematic model (see Fig. 3) and adapts the left leg configuration to reach the desired pose of the left foot. Note that the choice of the foot frame is arbitrary in double support mode. From forward kinematics of the right leg chain, the 6D rigid body transform of the hip frame ${}^{rfoot}T_{hip}$, expressed in $\mathcal{F}_{rfoot}$ can be obtained. From the fixed transformation ${}^{rfoot}T_{lfoot}$, which denotes the desired location of the left foot relative to the right, ${}^{hip}T_{lfoot}$ expressed in the hip frame $\mathcal{F}_{hip}$ can be computed. Inverse kinematics for the left leg chain is solved numerically, using the right leg joint values as an initial guess. Finally, if a solution exists, the modified whole-body configuration is added to the database if it is free of self-collisions and statically stable. Generally, building this database of statically stable configurations needs to be performed only once since it is independent of the planning

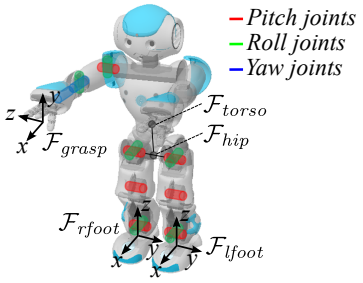

Fig. 3. Kinematic model used in our work.

Fig. 4. Examples of valid goal configurations for a given pose of the robot's right hand. The feet of the robot remain fixed, and the 5D goal pose for the hand is identical in all configurations. In this case, the left arm was set to a safe configuration adjacent to the robot's body.

scenario or environment. Sets for single support mode (left or right) can be constructed in a similar fashion, without the second leg adjustment step.

C. Goal Pose Generation

When planning whole-body motions for manipulation, the goal configuration has to fulfill several requirements. For manipulation actions, the hand pose of the goal configuration used within RRT-Connect is dependent of the object to be manipulated. In order to obtain a valid goal configuration with a specific hand pose, our approach first generates double support configurations according to the method presented in Sec. III-B and then adapts the arm configuration as explained in the following.

In our work, we assume that the grasping goal, i.e., a 3D world coordinate and the z-axis direction of the grasp frame $\mathcal{F}_{grasp}$ located at the tip of the chosen hand, is given according to the object to be manipulated. Accordingly the roll and pitch angles of the grasp frame are fixed. By means of basic kinematics computations, $\mathcal{F}_{grasp}$ can be expressed w.r.t $\mathcal{F}_{torso}$ (see Fig. 3), for each double support configuration generated. Our system then computes the 5D IK for the arm chain in closed form. In general, it is also possible to solve the IK for a full 6D hand pose. However, in our application we do not constrain the yaw angle of $\mathcal{F}_{grasp}$ to allow for a larger set of possible grasp poses.

The analytic IK computation may result in a number of solutions, which we evaluate with

$$\text{eval}(q^{arm}) = q^{arm}[0] \cdot |\det\left(J(q^{arm})\right)|. \qquad (1)$$

Here, $\det\left(J(q^{arm})\right)$ is the determinant of the Jacobian associated with the arm configuration q^{arm} and denotes a measure of manipulability [21]. We combine this with the shoulder pitch angle $q^{arm}[0]$ to prefer elbow-down configurations which are considered to be more natural than elbow-up configurations. The best IK solution maximizing Eq. (1) is assigned to the arm joints of the initially sampled double support configuration.

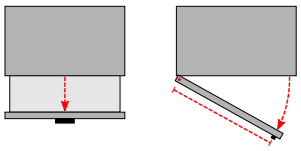

Fig. 5. Two examples of articulated objects: A drawer and a door.

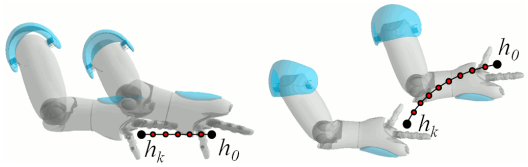

Fig. 6. Example end-effector trajectories, i.e., desired hand poses for opening a drawer (left, side view) and a door (right, top view).

Fig. 4 shows a set of example goal configurations generated for a desired grasp frame position and z-axis direction. Currently, our planner uses only the overall best whole-body configuration as goal pose, but generally one can think of rooting multiple trees at the generated goal configurations and initializing different RRT searches from which the best solution is chosen afterwards.

D. RRT-Connect for Manipulation of Articulated Objects

We now describe how to consider motion constraints for manipulating articulated objects once the handle has been grasped, using the generated goal poses and RRT-Connect for planning.

Articulated objects are represented by a volumetric model, a joint with the corresponding position, and a handle position. From this, a generative model can predict the trajectory of the articulated parts and the handle while the object is being manipulated. Articulation models can be learned from previous experience while carefully manipulating the environment [22]. For the scope of this work, we assume the model to be given. Here, we are particularly interested in prismatic (e.g., drawers) and revolute models (e.g., doors), as illustrated in Fig. 5. A prismatic model has one degree of freedom as a translation along a fixed axis. The handle trajectory is constrained on a 3D vector denoting the opening direction. A revolute model has one degree of freedom around a specified 3D axis of rotation. The handle trajectory is constrained on the circular arc described by the axis of rotation and the radius of the handle position. When planning for manipulation of articulated objects, the respective handle trajectory and the robot's grasp frame need to be expressed w.r.t the same reference coordinate system.

1) Considering Manipulation Constraints for Tree Initialization: Given a start configuration q_{start} with a hand attached to the object handle and the object parameters, our approach computes a sequence of hand poses $\langle h_0, \ldots, h_k \rangle$ along the handle trajectory (GEN_HAND_TRAJ in Line 4, Alg. 1). For the initialization of the search trees, the nodes n_s and n_g that

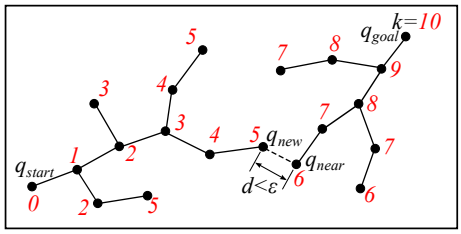

Fig. 7. Nodes of the start tree growing from the left and the goal tree growing from the right, with their respective hand pose indices when manipulating an articulated object. A connection between the two trees can be established when two nodes with adjacent hand pose index can be connected.

contain q_{start} and q_{goal} are assigned the hand pose indices 0 and k, respectively (Lines 5 and 6 in Alg. 1). Fig. 6 illustrates examples for linear and circular object trajectory with the generated hand poses.

2) Extending the Trees Under Manipulation Constraints: It is required that the hand remains on the object trajectory for each new configuration generated in EXTEND. Given a configuration q_{new} generated by NEW_CONFIG by stepping from q_{near} towards q_{rand}, the manipulating hand needs to move along the handle trajectory. This requirement is satisfied by ENFORCE_MANIP_CONSTRAINT (Line 4 in Alg. 2). In practice, this means that the hand needs to move from the hand pose with index $n_{near}.h$ to the hand pose with index $r = n_{near}.h + 1$ or $r = n_{near}.h - 1$, depending on the tree to be expanded. To obtain this desired hand pose, our approach again uses IK for the arm as in Sec. III-C. In the set $IKsol$ of IK solutions found for the hand pose, our system selects q_c^{arm} that minimizes the configuration space distance to the arm configuration of q_{near}

$$q_c^{arm} = \underset{q_i^{arm} \in IKsol}{\arg\min} \|q_i^{arm} - q_{near}^{arm}\| \tag{2}$$

and sets q_{new}^{arm} to q_c^{arm} and $n_{new}.h = r$.

It remains to describe how a connection between the two trees is detected (see Fig. 7). At each iteration, the forward kinematics (FK) for the kinematic chain from $\mathcal{F}_{rfoot}$ to $\mathcal{F}_{grasp}$ in the configuration q_{new} is computed. A path has been found when the FK determines the hand to be already in the correct pose. This case occurs exactly once, namely when the hand pose indices of q_{near} and q_{new} are already adjacent and q_{new} of $\mathcal{T}_b$ has reached $\mathcal{T}_a.last.q$.

IV. Implementation Details

Our planner is implemented in the *MoveIt!* framework in ROS [8] and uses FCL [23] for collision checks. When the planner checks a configuration for validity, the collision mesh model of each robot link is tested for self-collisions and collisions with the environment. For the stability constraint, the zero moment point generally indicates a humanoid's dynamic stability [24]. In this work, we use the simplification of static stability which is a valid approximation for slow motions. Thus, stability can be evaluated by checking whether the robot's center of mass (CoM) projected to the ground plane is within the support polygon. In practice, we scale down the actual support polygon by a small safety margin to avoid the

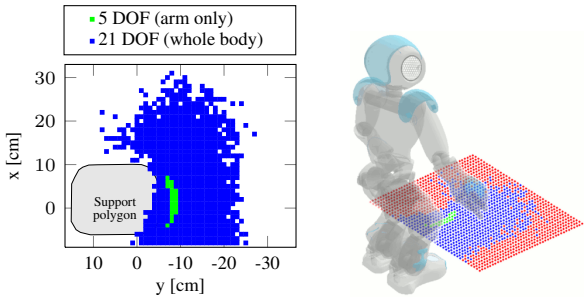

Fig. 8. Reachability map of the right hand for 5D end-effector goals at height z=0.2 m as 2D projection (left) and relative to the robot model (right). Green squares denote success for arm-only and whole-body planning, while blue squares are only reachable with whole-body planning.

critical boundary areas, thus accounting for sensor noise and a mismatch between the real robot and its CAD model. To analytically solve IK, we use *IKfast* [25]. In our case, this results in 5D goal poses with a 5 DOF arm.

To approximate the constraint manifold for planning whole-body motions, we used a database of 463 statically stable double support configurations, generated within 10 000 iterations. The success rate of only 4.63% demonstrates the low probability of generating valid configurations, when the configurations space is sampled completely at random during the search. For generating goal poses, we allow a maximum number of 3 000 iterations. For efficiency, we stop searching when more than six goal poses have been found and choose the best one according to Eq. (1). The maximum number of iterations *max_iter* in Alg. 1 was set to 3 000. The step width ε was set to 0.1.

V. Experiments

A. Humanoid Robot Platform

For the experimental evaluation of our approach, we use a V4 Nao humanoid by Aldebaran Robotics. The robot is 58 cm tall and has 25 DOF: 2 in the neck, 6 in each arm (including one to open and close the hand), and 5 in each leg. In addition, the legs share a common (linked) hip joint that cannot be controlled independently. In total, our planner operates on 21 DOF since we do not articulate the head and use the hands only when grasping. To correct for backlash of the gears while executing joint angle trajectories, each angle is measured with Hall effect magnetic rotary encoders at a precision of 0.1°. Inertia, mass, and CoM of each link are known from CAD models. For efficient collision checks, we created a low-vertex collision mesh model for each of the robot's links from the CAD models.

In the following experiments, planning was performed off-board on a single core of a standard desktop CPU (Intel Core2 Duo, 3 GHz).

B. Goal Pose Generation

In the first experiment, we evaluated the performance of our approach to generate goal poses. The robot had to plan stable whole-body motions to reach a 5D manipulation goal with the right arm (similar as in Fig. 4). The z axis of the grasp frame was horizontal and perpendicular to the

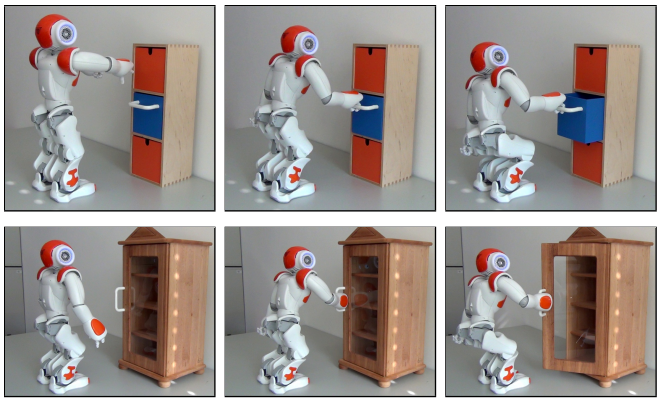

Fig. 9. Execution of a whole-body manipulation plan for a drawer and a door. First, the robot plans to grasp the handle, then it plans a motion to manipulate the object once the handle has been grasped. A video of the results is available at `http://hrl.informatik.uni-freiburg.de`.

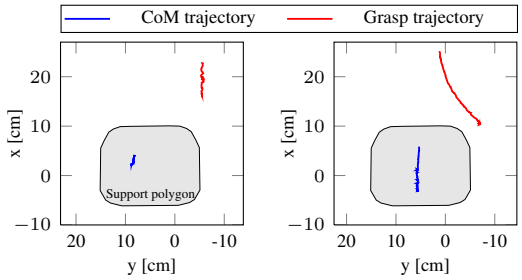

Fig. 10. Trajectory for the right hand and center of mass over the support polygon while opening a drawer (left) and a door (right)

robot's orientation, e.g., to grasp the handle of a drawer, at a height of z=20 cm. x and y were varied in intervals of 1 cm in a 40 cm $\times$ 40 cm area. Fig. 8 shows the resulting reachability map, in which squares denote successful results. In the other locations, the goal pose generation failed, i.e., they are not reachable. It is clearly visible that whole-body planning extends the manipulation range around the robot, particularly compared to using only the 5 DOF arm.

For the reachable locations, it took 6.62 s $\pm$ 6.33 s to generate the first goal pose, and 0.18 s $\pm$ 0.19 s for planning with RRT-Connect expanding 49.79 $\pm$ 29.15 nodes on average.

C. Manipulating Articulated Objects

We further evaluated the manipulation of articulated objects. From its initial configuration, the robot first had to plan for reaching the known handle location, grasp it, and then plan an end-effector trajectory given by the parameterization of the articulated object. Fig. 9 shows snapshots from an accompanying video in which the robot executes the whole-body plans. It successfully grasped the handle and opened the drawer and door while keeping its balance. Fig. 10 shows the trajectories of the right hand and CoM, as measured by the joint angle sensors during execution. As can be seen, our approach leads to hand trajectories that closely follow the given model, while the CoM remains safely within the support polygon.

To quantitatively evaluate the reliability of our randomized

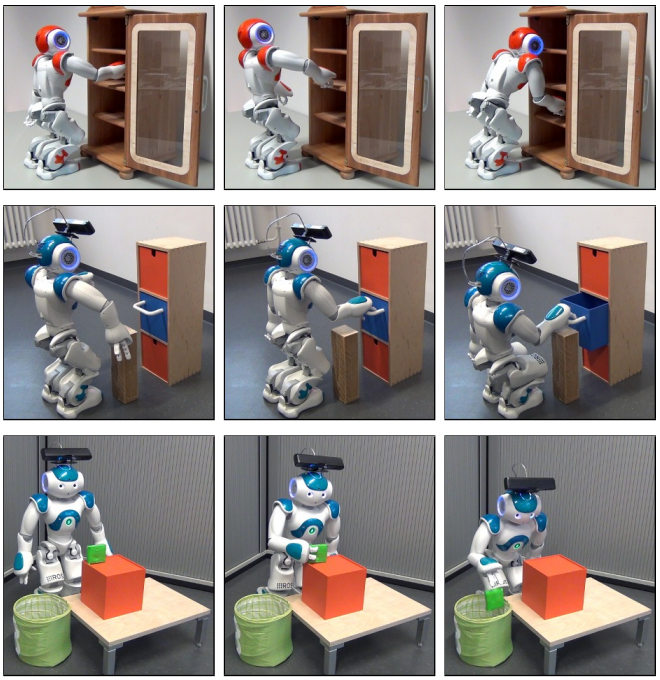

Fig. 11. Execution of collision-free whole-body plans: Reaching into different shelfs of a cabinet (top), manipulating a drawer while avoiding an obstacle (middle), and picking up a small object (bottom). A video of the results is available at `http://hrl.informatik.uni-freiburg.de`.

planning approach, we carried out planning in both scenarios 100 times. The results are shown in Table I. With the chosen parameters of our algorithm, planning for the entire manipulation task could be solved successfully in 89% of all attempts for the drawer, and 78% for the door. These rates certainly increase as more planning time is allowed.

D. Planning Collision-Free Motions

In the next scenario, we evaluated the performance of the planner in the presence of obstacles seriously constraining the possible motions. Here, the robot needed to reach into one shelf of a cabinet and then into a second one without colliding (Fig. 11, top row). As before, the task was planned 100 times with results following from Table I. With the chosen parameters of our algorithm, the entire task of reaching the grasping goals located inside the upper and lower shelf could be planned successfully in 77% of all attempts. The comparably longer planning times result from the highly constrained workspace. Note that this scenario is usually problematic for Jacobian-based optimization techniques due to the local minima induces by the cavities.

E. Manipulation with Collision Avoidance

We then extended the scenario with the drawer from Sec. V-C with an additional obstacle that the robot had to avoid (Fig. 11, middle row). Results for 100 planning attempts can be found in Table I. The obstacle close to the handle caused an increased computation time for goal generation and planning. The entire task succeeded in 83% of the runs. The planned motions were executed on the real

TABLE I

AGGREGATED RESULTS OVER 100 PLANNING ATTEMPTS

Scenario	Goal generation [s]	Planning time [s]	Expanded nodes
Drawer (reach)	5.42±5.17	0.08±0.04	32.81±8.41
Drawer (open)	4.14±3.94	0.11±0.07	30.36±12.69
Door (reach)	8.59±7.54	0.09±0.05	32.16±9.43
Door (open)	2.63±2.47	0.15±0.08	35.17±14.51
Shelf (upper)	16.4±13.74	0.09±0.27	19.84±30.06
Shelf (lower)	19.06±16.75	10.44±0.83	1164.89±98.99
Drawer w. obst. (reach)	16.24±12.64	7.71±3.0	1067.93±333.54
Drawer w. obst. (open)	8.53±7.21	0.2±0.19	32.16±21.09

robot platform multiple times collision-free and statically stable.

F. Pick and Place Task

In the last experiment, the task of the robot was to pick up an object and place it into a basket. After picking up the object, its mesh model becomes attached to the robot's end-effector for collision checking with the environment. Fig. 11 (bottom) and the accompanying video show the resulting plan.

For the first plan to reach the small green box, our algorithm required 3.28 s to generate a goal pose and 0.37 s for the whole-body motion plan, hereby expanding 62 nodes. For the second plan to place the box over the basket, our algorithm generated a goal pose within 9.16 s and a motion plan within 0.18 s, expanding 38 nodes.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we presented an approach to whole-body motion planning. Our RRT-Connect based algorithm uses IK to satisfy constraints resulting from the task specification, in addition to collision-freeness and stability of the configurations. One focus hereby is the manipulation of articulated objects, which imposes constraints on the end-effector. Based on the models of the articulated objects, our approach generates hand poses of sampled configurations that follow the object handle trajectory. The experiments show that our planner generates solutions to complex object manipulation problems that satisfy all the desired constraints, e.g. to open doors and drawers, or to pick up objects. As we demonstrated in the evaluation on a real Nao humanoid, the whole-body motion plans can be reliably executed even on such a low-cost platform. Our approach is generally applicable to other (full-size) humanoids and is available as open source implementation.

In future work, we will integrate the perception of the articulated objects, particularly their handles. We are also working on changing the stance leg while manipulating through a shift of the center of mass, which can be easily integrated into our planner and will enable an even greater manipulation range.

REFERENCES

[1] J. J. Kuffner and S. M. Lavalle, "RRT-Connect: An efficient approach to single-query path planning," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2000.

[2] T. Rühr, J. Sturm, D. Pangercic, M. Beetz, and D. Cremers, "A generalized framework for opening doors and drawers in kitchen environments," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2012.

[3] S. Chitta, B. Cohen, and M. Likhachev, "Planning for autonomous door opening with a mobile manipulator," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2010.

[4] R. Diankov, S. Srinivasa, D. Ferguson, and J. Kuffner, "Manipulation planning with caging grasps," in *Proc. of the IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2008.

[5] O. Kanoun, F. Lamiraux, P.-B. Wieber, F. Kanehiro, E. Yoshida, and J.-P. Laumond, "Prioritizing linear equality and inequality systems: application to local motion planning for redundant robots," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2009.

[6] N. Mansard, O. Stasse, F. Chaumette, and K. Yokoi, "Visually-guided grasping while walking on a humanoid robot," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2007.

[7] E. Yoshida, O. Kanoun, C. Esteves, and J.-P. Laumond, "Task-driven support polygon reshaping for humanoids," in *Proc. of the IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2006.

[8] S. Chitta, I. Sucan, and S. Cousins, "MoveIt! [ROS topics]," *Robotics Automation Magazine, IEEE*, vol. 19, no. 1, pp. 18 –19, 2012.

[9] O. Kanoun, J.-P. Laumond, and E. Yoshida, "Planning foot placements for a humanoid robot: A problem of inverse kinematics," *Int. Journal of Robotics Research (IJRR)*, vol. 30, no. 4, pp. 476–485, 2011.

[10] L. E. Kavraki, P. Svestka, J. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[11] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge University Press, 2006, available at http://planning.cs.uiuc.edu/.

[12] J. J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba, and H. Inoue, "Dynamically-stable motion planning for humanoid robots," *Autonomous Robots*, vol. 12, pp. 105–118, 2002.

[13] M. Stilman, "Global manipulation planing in robot joint space with task constraints," *IEEE Trans. on Robotics and Automation*, vol. 26, no. 3, pp. 576–584, 2010.

[14] D. Berenson, J. Chestnutt, S. Srinivasa, J. Kuffner, and S. Kagami, "Pose-constrained whole-body planning using task space region chains," in *Proc. of the IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2009.

[15] I. A. Şucan and S. Chitta, "Motion planning with constraints using configuration space approximations," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2012.

[16] F. Kanehiro, E. Yoshida, and K. Yokoi, "Efficient reaching motion planning and execution for exploration by humanoid robots," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2012.

[17] G. Oriolo and M. Vendittelli, "A control-based approach to task-constrained motion planning," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2009.

[18] S. Dalibard, A. Nakhaei, F. Lamiraux, and J.-P. Laumond, "Whole-body task planning for a humanoid robot: a way to integrate collision avoidance," in *Proc. of the IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2009.

[19] ——, "Manipulation of documented objects by a walking humanoid robot," in *Proc. of the IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2010, pp. 518–523.

[20] S. Nozawa, Y. Kakiuchi, K. Okada, and M. Inaba, "Controlling the planar motion of a heavy object by pushing with a humanoid robot using dual-arm force control," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2012.

[21] T. Yoshikawa, "Manipulability of robotic mechanisms," *The International Journal of Robotics Research*, vol. 4, no. 2, pp. 3–9, 1985.

[22] J. Sturm, C. Stachniss, and W. Burgard, "A probabilistic framework for learning kinematic models of articulated objects," *Journal on Artificial Intelligence Research (JAIR)*, vol. 41, pp. 477–626, August 2011.

[23] J. Pan, S. Chitta, and D. Manocha, "FCL: A general purpose library for collision and proximity queries," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2012.

[24] M. Vukobratovic and B. Borovac, "Zero-moment point – thirty five years of its life," *Int. Journal of Humanoid Robots*, vol. 1, 2004.

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