Robot Placement based on Reachability Inversion

Nikolaus Vahrenkamp, Tamim Asfour and Rüdiger Dillmann
Institute for Anthropomatics
Karlsruhe Institute of Technology
Adenauerring 2, 76131 Karlsruhe, Germany
Email: {vahrenkamp,asfour,dillmann}@kit.edu

***Abstract*— Having a representation of the capabilities of a robot is helpful when online queries, such as solving the inverse kinematics (IK) problem for grasping tasks, must be processed efficiently in the real world. When workspace representations, e.g. the reachability of an arm, are considered, additional quality information such as manipulability or self-distance can be employed to enrich the spatial data. In this work we present an approach of inverting such precomputed reachability representations in order to generate suitable robot base positions for grasping. Compared to existing works, our approach is able to generate a distribution in SE(2), the cross-space consisting of 2D position and 1D orientation, that describes potential robot base poses together with a quality index. We show how this distribution can be queried quickly in order to find oriented base poses from which a target grasping pose is reachable without collisions. The approach is evaluated in simulation using the humanoid robot ARMAR-III [1] and an extension is presented that allows to find suitable base poses for trajectory execution.**

I. Introduction

The capabilities of a robot can be expressed in several ways. Usually a set of basic parameters, such as joint limits or maximum torque, is given to describe the capabilities in terms of reaching and manipulation. This information can be used offline to build a representation of the robot's operational workspace in order to support online tasks. Since performance is crucial during online processing, a beneficial representation can clearly improve the efficiency of inverse kinematics (IK) queries [2], [3], [4]. Usually such a representation is given by a spatial grid which covers the 3D (position) or 6D (position and orientation) workspace. Either a binary value is stored, indicating whether the grid cell (voxel) is reachable, or each cell holds additional information, such as quality values, which allows to compare different voxels.

The IK-problem is usually given by a requested workspace pose $p \in SE(3)$, for which a joint configuration $q \in \mathbb{R}^n$ is searched that results in an end effector pose euqal to p. In case multiple targets are available, e.g. a set of predefined grasps are given for an object, the selection which grasp should be considered for IK solving can be supported by reachability data. Further, these data structures can be used to efficiently solve bimanual IK queries, where a grasping configuration for both arms of a humanoid robot are searched [5], [6].

A workspace representation of the robot's capabilities can also be used to support the search for suitable robot base poses for grasping. In [4] equivalent classes, representing the potential poses of the robot's base, are built and the resulting data structure is queried to find suitable placements. Experience-based representations of robot poses from which grasping was successful is presented in [7]. The so-called action related places (ARPlaces) implicitly cover any uncertainties that affect the grasping task. The ARPlace representation has to be trained and it gets invalid when internal parameters (e.g. camera calibration) change. In [8] an approach for finding suitable robot placements for execution of workspace trajectories is presented. Therefore, trajectories are interpreted as patterns and searched using multi-dimensional correlation. In earlier work, we showed how a 2D distribution can be built that describes potential base poses for bimanual grasping [9].

Compared to these works, we present a generic and efficient approach for inverting spatial reachability data in order to generate a distribution in SE(2) of potential base poses for grasping. With the proposed reachability inversion approach we present a generic formalism for transforming a robot's capability representation to an object-centered view.

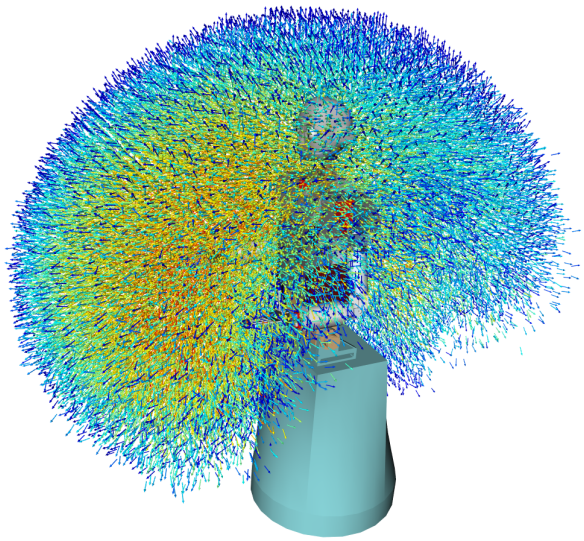

Fig. 1. The reachability distribution of the robot's right TCP. The kinematic chain covering hip and right arm with 10 degrees of freedom (DoF) was used for generation while manipulability, self-distance and joint limits were incorporated for determining the quality.

As we will show, with this transformation extended IK queries covering the robot's joints and base placement can be solved efficiently.

II. Representing the Robot's Workspace

In order to generate a representation of the robot's capabilities in terms of reaching and grasping, a spacial data structure is generated and filled in an offline step. Therefore, the 6D workspace (position and orientation of the end effector) is voxelized and the resulting voxel grid is filled with pose quality information. Each voxel represents all possible spatial poses of the end effector for which the tool center point (TCP) lies within the extends of the voxel. Note, that position and orientation is considered here, while other approaches exist where the 3D workspace is voxelized and the orientation is additionally encoded [3].

To build up a representation of the robot's workspace, the voxel grid is filled with quality information describing the capabilities of the corresponding kinematic chain (e.g. an arm). Usually reachability information is used [3], [5], [4], but other performance indices, such as the manipulability, can be used as showed in [2], [10].

When using a discretized workspace representation, the reachability of a voxel v cannot be reliably expressed as a binary value, since there might be poses $p \in v$ that are reachable while other poses $p' \in v$ are not. Hence, the reachability of a voxel can just give a hint (or a probability) that a pose inside that voxel is reachable. By considering the reachability of a voxel as a probability that a pose is reachable, the discretized reachability data can be interpreted as a frequency distribution known from descriptive statistics [11]. Hence, we use the term reachability distribution in this work to name the discretized reachability data that is represented by entries of 6D workspace voxels [12].

A. Extended Manipulability Measurement

In this work are using an extended manipulability measurement in order to get a good approximation of the robot's ability to maneuver in workspace. Compared to the classical manipulability index [13], [14] where the distance to singular configurations is determined by analyzing the manipulability ellipsoid that is spanned by the singular vectors of the Jacobian, we are explicitly considering joint limits under redundancy. Additionally, the distance between body parts of the robot (self-distance) is incorporated in order to penalize configurations that result in a limited maneuverability. Further details can be found in [10].

In Fig. 1 the distribution of the manipulability is shown for a 10 degrees of freedom (DoF) kinematic chain covering hip and right arm of the humanoid robot ARMAR-III [1].

B. Reachability Distribution

The reachability in workspace can be constructed by going through all 6D voxels around the robot while trying to solve the inverse kinematics (IK) problem. If an IK solution can be determined, the voxel can be marked as reachable. This results in a binary representation of the reachability, having the disadvantage that no quality comparisons between voxels can be performed (except reachable or not reachable). To get a better representation, sampling can also be performed in joint space and forward kinematics can be used to determine the corresponding voxels. By doing so, the number of *hits* of a voxel can be counted and a voxel's quality entry gives then information about the volume in joint space that maps to the voxel. This approach would serve information about how large the volume in joint space is for which the TCP pose ends up in the corresponding voxel. This can be of interest, e.g. when a variety of potential IK-solutions are helpful in finding collision-free configurations, but singular configurations are not handled appropriately. Hence, we use the approach of [10] to compute a distribution of the robot's extended amnipulability.

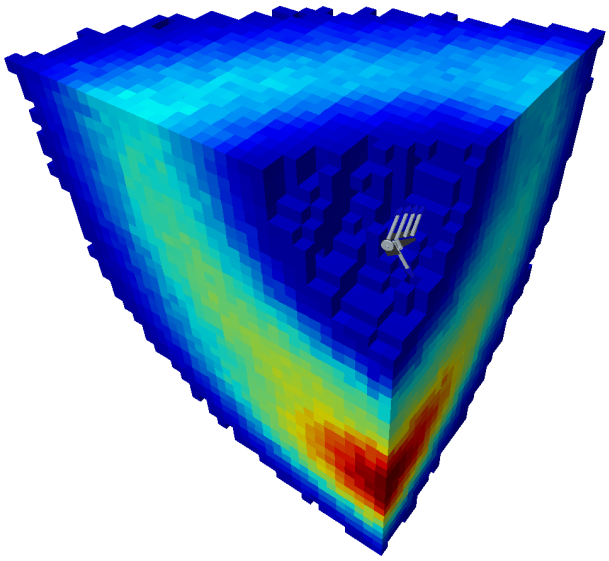

Fig. 2. One octant of the IRD's 3D visualization. Each voxel is colorized by the maximum entry of any potential rotation at the corresponding workspace position.

Depending on the involved number of degrees of freedom (DoF), complete sampling can be time consuming, but since the generation is performed in an offline step, performance is not critical here. Nevertheless, random sampling can be used in order to generate an approximated representation more quickly.

III. Reachability Inversion

Once a workspace representation of the robot's reachability is generated, the data can be inverted in order to generate an object-centered reachability representation: Instead defining the reachability of the TCP w.r.t. a fixed robot base pose, the inverted data gives information about potential robot base poses when a fixed TCP pose in workspace, e.g. a grasp to be applied on an object, is given. This data can be generated once in an offline step and during online processing it can be quickly queried to find suitable robot base poses for grasping.

Note, that the inverse reachability data contains all possible base poses in workspace together with their according

probability. Hence, any additional constraints, e.g. when the robot should be located upright on the ground plane, will be considered during online processing. This allows to store the inverse reachability data in the most flexible way.

A. Building the Inverse Reachability Distribution (IRD)

A given reachability distribution RD is usually defined in a base coordinate system of the robot. This can be located in the waist for legged robots or in the platform for platform-based humanoid and service robots. Hence, the data provides information about the reachability of the TCP in the base frame.

In order to prepare the generation of the inverted reachability distribution IRD, in a first step all tuples (t_i, e_i) are built by going through all 6D voxels v_i of RD, while determining the corresponding base-to-TCP transformation $t_i \in SE(3)$ and the voxel entry e_i.

Then, a second voxelized data grid IRD is filled by the tuples (t_i^{-1}, e_i), which are generated by inverting the transformations t_i. For each t_i^{-1} the corresponding IRD voxel v_i' is determined and its entry is set to e_i.

The IRD has to be computed only once in an offline step and the resulting data can be stored in a file. During online processing, the IRD is positioned at a target location in workspace (e.g. a grasping pose) in order to obtain the distribution of potential poses of the robot's base frame.

As shown in Fig. 2, the reachability inversion results in an approximated spherically shaped volume around the investigated pose in workspace. The figure shows one octant of the IRD's 3D visualization, which was built by colorizing the voxels according to the maximum entry of any potential rotation at the corresponding voxel position in workspace.

B. Building the Oriented Reachability Map (ORM)

The inverse reachability data can be used to efficiently query suitable robot base poses w.r.t. a grasping pose. Since the IRD holds information about all reachable poses of the robot's base frame, constraints must be applied in order to define the subset of possible poses. Such constraints will usually be defined by the surface on which the robot is allowed to stand on. In the following, we assume that the surface is flat and the robot's base frame is located in an upright direction. Hence, the floor $F \cong SE(2) \cong \mathbb{R}^2 \times \mathbb{S}^1$ is defined as the subspace of SE(3) that goes through the origin and is parallel to the xy plane:

$$F = \{ \begin{pmatrix} R_z & 0 & p_{x,y} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \in SE(3) | p_{x,y} \in \mathbb{R}^2, R_z \in \mathbb{S}^1 \}. \tag{1}$$

By cutting F with the IRD volume located at a grasping pose $p \in SE(3)$, one can obtain the three dimensional distribution $ORM \cong SE(2)$ which gives information about potential robot base positions and orientations.

A visualization of this *Oriented Reachability Map (ORM)* can be seen in Fig. 3. For each grid point on the floor, the potential orientations of the base frame are depicted. The color indicates the quality of the corresponding ORM entry.

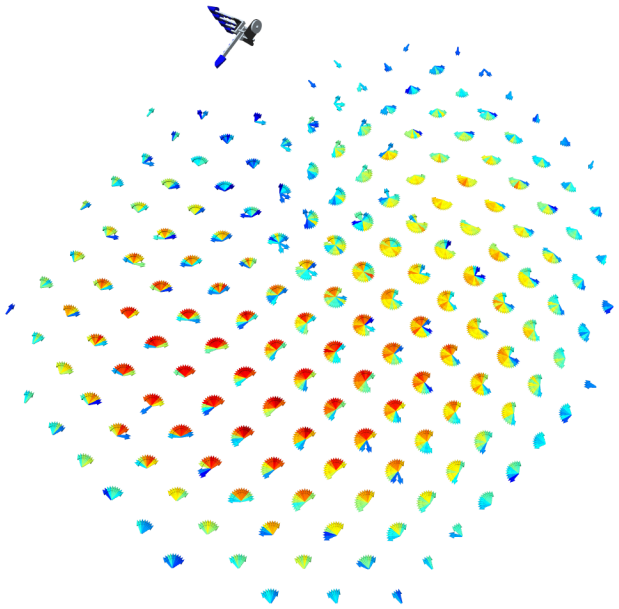

Fig. 3. The reachability inversion for the given grasp visualized on the ground plane. The potential platform orientations are depicted by arrows which are colored according to their inverse reachability (red:high, blue:low).

IV. Robot Placement

In this section we show how the inverted reachability data can be used to efficiently find suitable robot base poses.

A. Robot Placement for Grasping

The ORM can be queried in order to find suitable robot base poses for which the target pose p is reachable by the end effector. This can be done by choosing $(x, y, \gamma)_{max} \in SE(2)$ for which the ORM-entry is at its maximum. Note, that due to discretization, a set $M \subset SE(2)$ of poses exists for which the entry is at the maximum:

$$M = \arg\max_{x,y,\gamma} ORM(x, y, \gamma). \tag{2}$$

Further, randomized techniques can be applied in order to query the inverse reachability data. Here, the probability of choosing an $(x, y, \gamma)_{sample}$ is proportional to its ORM entry. This can be useful when additional constraints, such as obstacles, have to be considered, which may lead to situations where no valid IK-solution exists for base poses $b \in M$.

B. Lazy ORM

Since the ORM computation introduces some computational overhead when only one IK-query has to be computed, a lazy evaluation of the ORM entries can speed up single-query IK tasks. Hence, the ORM data is not calculated in advance, but for every query it is checked whether the entry of the corresponding ORM cell has already been calculated or if it has to be computed. In the second case the entry of the corresponding ORM cell is updated for the query pose

$p = (x, y, \gamma)$. With $p_{x,y} = (x, y)^T$ and the rotation matrix $R_z = R(\gamma)$ the pose p can be rewritten to

$$p_2 = \begin{pmatrix} R_z & p_{x,y} \\ 0 & 1 \end{pmatrix} \in SE(2),$$

and

$$p_3 = \begin{pmatrix} R_z & 0 & p_{x,y} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \in SE(3).$$

Then the IRM entry of the workspace pose p_3 can be easily determined in order to compute the ORM cell entry.

By doing so we loose the ability of determining the maximum ORM entry in advance since we do not have knowledge about the complete data. However, robot placement queries can be processed by generating a fixed number of samples in order to determine the maximum of the corresponding ORM entries. Since sampling and the IRM query process can be performed efficiently, the performance of single-query tasks can be improved as shown by the evaluation in Section VI.

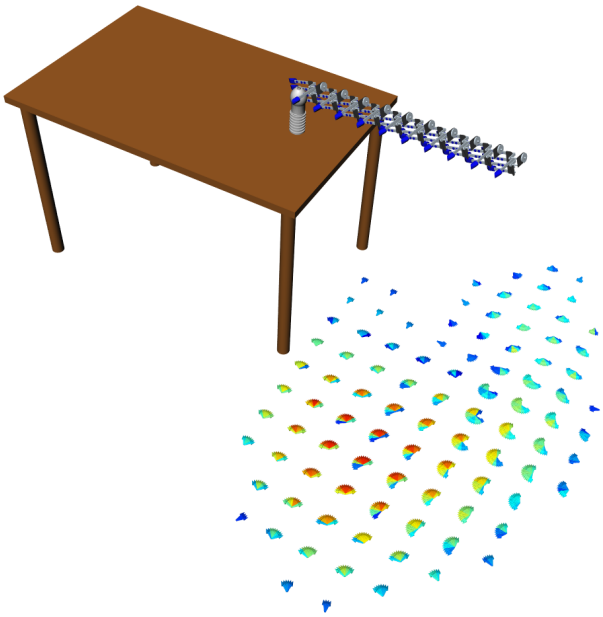

Fig. 4. A workspace trajectory with corresponding Oriented Reachability Map (ORM) visualized on the ground plane.

C. Robot Placement for Trajectory Execution

Reachability inversion can also be useful when multiple workspace poses should be reached by the TCP. This can be the case for pick and place operations or for trajectory execution, e.g. door opening tasks. In the following, trajectories are represented by a finite sequence $(s_0, \ldots, s_n)$ of workspace poses. For each s_i, the corresponding ORM_i is generated and the united ORM_{tr}, representing the capability of reaching the whole trajectory, is built by going through (x, y, γ) and determining the minimum entry of all ORM_i:

$$ORM_{tr}(x, y, \gamma) = min\{ORM_i(x, y, \gamma)\}_{i=0}^{n}. \quad (3)$$

The visualization of an exemplary trajectory with corresponding ORM_{tr} is depicted in Fig. 4.

V. IK Solving

Based on the ORM data structure the complete IK query for grasping tasks, that consists of finding a suitable robot base pose together with a configuration of the arm, can be realized as shown in Algorithm 1. Initially the ORM is built as described in Section IV. The IK search loop consists of sampling suitable robot base poses followed by a call to the robot's IK solver and a collision check. The IK solver for the arm can be realized by a generic Jacobian-based approach, or custom algorithms can be used. Once an IK result has been found, the resulting configuration is checked against self-collisions and collisions with the environment.

Algorithm 1: IK solver covering the robot's base pose and arm configuration.

Input: Target pose $p \in SE(3)$,
Inverse Reachability Distribution IRD
Output: Robot base pose $r \in SE(2)$,
joint configuration $j \in \mathbb{R}^n$
$IRD.setPose(p)$;
$ORM \leftarrow IRD.cut(F)$;
while *(!timeOut)* **do**
  $r \leftarrow ORM.sampleRobotPose()$;
  $robot.apply(r)$;
  $j \leftarrow solveIKArm(p)$;
  if (j) **then**
    $robot.apply(j)$;
    if $(!robot.isInCollision())$ **then**
      return $\{r, j\}$;
    end
  end
end
return $NULL$;

An IK-solver that uses the $LazyORM$ approach can be built analogous by omitting the ORM initialization and performing the robot pose sampling as described in Section IV-B.

VI. Evaluation

In the following we evaluate the performance of the proposed approach in different setups. All evaluations have been carried out with the 43-DoF model of the humanoid robot ARMAR-III and the C++ robotics simulation package Simox [15]. All results were gained by averaging 100 queries which have been carried out on a 3.0 GHz Core i5 PC within a single-threaded application.

A. IK Performance

The setup of this test case can be seen in Fig. 5. For this experiment we use a precomputed grasp that is defined relatively to the target object. During the tests the bottle is randomly placed on the table and the target pose $p \in SE(3)$ is derived by applying the object related grasping transformation. The IK task consists of finding a suitable robot base pose (3 DoF) and a configuration of hip and right arm (10 DoF). As shown in Algorithm. 1 the result is checked against self-collisions and collisions with the environment.

TABLE I

PERFORMANCE EVALUATION OF THE 13 DOF IK TASK.

	Complete Query	ORM Setup	Base Pose #	Base Pose Time	IK Arm #	IK Arm Time
Full ORM						
Rand. target	**732.3ms**	726.9ms	23.0	5.5ms	1.8	2.2ms
Fixed target	**2.4ms**	0.0ms	2.6	0.6ms	1.4	1.7ms
Lazy ORM						
Rand. target	**7.0ms**	0.0ms	19.9	2.9ms	4.5	3,9ms
Fixed target	**4.4ms**	0.0ms	6.9	2.5ms	1.6	1.9ms

An overview of the results is given in Table I. The performance of the full (Section IV-A) and the lazy (Section IV-B) ORM approach is shown in the upper and the lower part of the table. The average runtime of one complete IK query is shown in the first column, followed by the ORM setup time and details about the ORM and arm-IK query. The IK solver that was used for this experiment is based on a Pseudoinverse Jacobian gradient descent algorithm. The rows labeled with *Rand. target* show the results which were measured when placing the target randomly on the table (with upright orientation). The rows with label *Fixed target* show the average results for a fixed target position when multiple IK requests have to be performed. The setup time is not considered in this case, since the ORM has to be created only once for all queries.

The results show, that the query time can be clearly reduced when the lazy ORM approach is chosen. The complete 13 DoF IK query including robot placement and collision detection can be realized in less than 10 ms for the investigated setup.

B. Searching Base Poses for Trajectory Execution

In this test case the ability of finding suitable robot base poses for door opening tasks is evaluated. We assume that the trajectories are given as a vector of workspace poses which must be reached in order to open the door. Hence, the extended IK task consists of sampling an potential robot base pose according to the ORM followed by determining, if a collision-free IK solution for the hip-arm system can be found for all trajectory points. In Fig. 6 three workspace trajectories in a kitchen environment are depicted. The left figure shows a trajectory for opening the dishwasher together with the corresponding ORM that was computed as described in Section IV-C. The figure in the middle shows an opening trajectory related to a drawer. Again, the ORM is depicted and additionally the robot is set to the IK solution

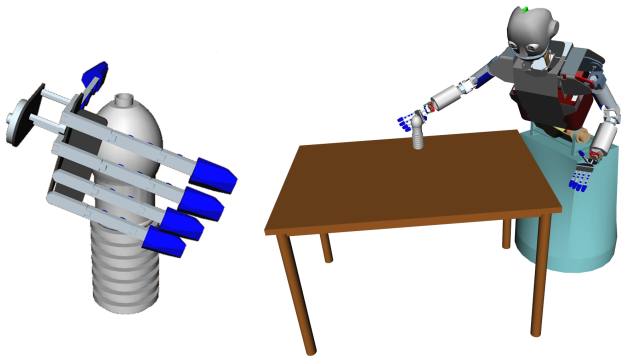

Fig. 5. The target object with grasp (left). An exemplary IK result with 13 DoF (right).

for the final trajectory position. On the right of Fig. 6 the same setup is depicted for a third trajectory that can be used to open the fridge.

TABLE II

DOOR OPENING PERFORMANCE EVALUATION.

	Compl. Query	ORM Setup	Base Pose #	Base Pose Time	IK Arm #	IK Arm Time
Full ORM						
Dishwasher	**1161ms**	1141ms	5.7	3.3ms	21.6	16.3ms
Drawer	**1158ms**	1154ms	3.0	1.1ms	4.1	3.1ms
Fridge	**1209ms**	1203ms	1.6	0.3ms	6.8	5.0ms
Lazy ORM						
Dishwasher	**33ms**	0ms	4.5	14.5ms	21.2	18.1ms
Drawer	**10ms**	0ms	3.1	6.7ms	4.2	3.1ms
Fridge	**7ms**	0ms	1.7	2.1ms	6.7	4.9ms

The results of this test case are shown in Table II. The first three rows show the evaluation when the complete ORM is computed for every query. It can be seen that the biggest part of the query time is spent for ORM calculation (column *ORM Setup*: ∼1.2 sec.) while the query itself can be processed quickly. The columns labeled with *Base Pose* show the detailed analysis of the robot base pose search. For every successful query 1.6 to 5.7 base poses have to be sampled until a valid solution can be determined. The runtime for this step was measured with 0.3 to 3.3ms. Once a potential base pose is sampled the IK query for the hip-arm kinematic chain is called 4.1 to 21.6 times on average (column *IK Arm*). If the IK query is successful for all trajectory poses, a valid solution has been found. The accumulated runtime for this test is 3.1 to 16.3ms on average.

The three rows on the bottom of Table II show the evaluation for the same setup when the lazy ORM approach is used. Since the ORM is not completely calculated, no setup time has to be considered and hence the IK processing can be performed more efficiently. The runtime for the three trajectories varies from 7 to 33ms on average. This performance measure includes all necessary steps to generate a valid and collision-free base pose together with the corresponding IK solutions for the hip-arm system.

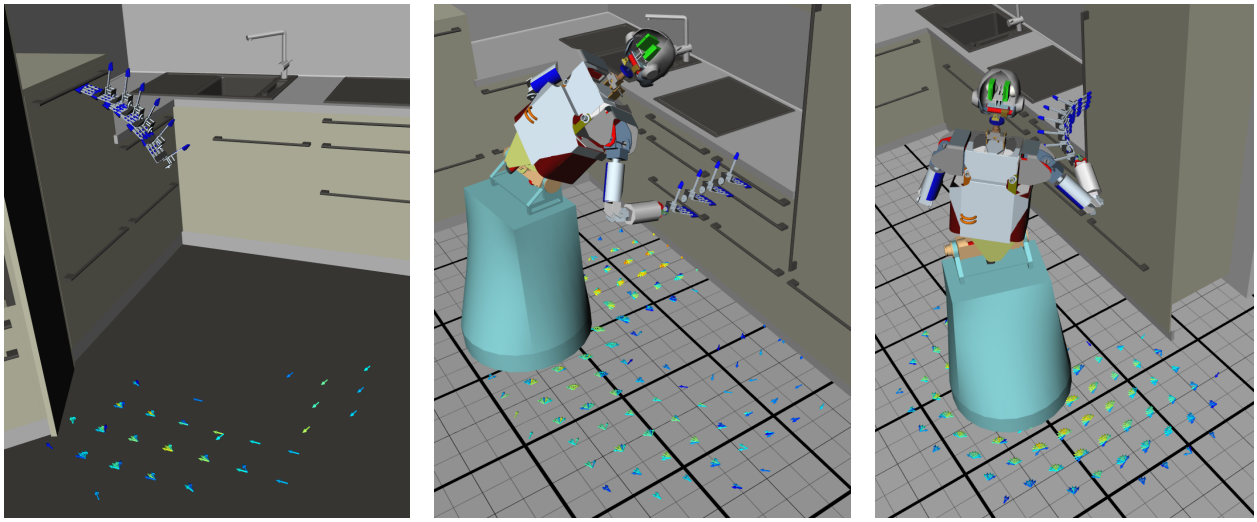

Fig. 6. A trajectory for opening the dishwasher with depicted ORM (left). The trajectory for opening the drawer with exemplary IK solution for the final pose (center). A trajectory for opening the fridge with robot configured at the final trajectory point (right).

VII. Conclusions

In this work we presented a generic approach for determining the inverse reachability related to a kinematic chain of a robot. We showed how this reachability inversion can be used to find suitable base poses in order to place the robot w.r.t. an object that should be grasped. Therefore, the *Oriented Reachability Map (ORM)* was introduced which can be built by placing the inverse reachability data at the target pose and constraining the allowed poses (e.g. upright on the floor). Based on the ORM data structure we showed how an efficient IK solver can be implemented which generates valid robot base poses together with correct joint configurations while avoiding self-collisions and collisions with the environment. Further, the *Lazy ORM* was introduced that can be used to avoid the computation of the complete ORM data structure by determining the required data on the fly while procesing the IK query. It was shown that the resulting data structure can be queried efficiently and several tasks such as placement for grasping or trajectory execution can be solved with high performance. The evaluation showed that the *Lazy ORM* approach is suitable for online IK processing in changing surroundings and hence it can be used for real world applications such as humanoid robots operating in human-centered environments.

Acknowledgment

The research leading to these results has received funding from the European Union Seventh Framework Programme under grant agreement 270273 (Xperience).

References

[1] T. Asfour, K. Regenstein, P. Azad, J. Schröder, A. Bierbaum, N. Vahrenkamp, and R. Dillmann, "ARMAR-III: An integrated humanoid platform for sensory-motor control." in *Proceedings of IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, December 2006, pp. 169–175.

[2] L. Guilamo, J. Kuffner, K. Nishiwaki, and S. Kagami, "Efficient prioritized inverse kinematic solutions for redundant manipulators," in *IEEE International Conference on Intelligent Robots and Systems (IROS)*, aug. 2005, pp. 3921 – 3926.

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

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

[5] N. Vahrenkamp, D. Berenson, T. Asfour, J. Kuffner, and R. Dillmann, "Humanoid Motion Planning for Dual-Arm Manipulation and Re-Grasping Tasks," in *IEEE International Conference on Intelligent Robots and Systems (IROS)*, St. Louis, USA, October 2009, pp. 2464–2470.

[6] F. Zacharias, D. Leidner, F. Schmidt, C. Borst, and G. Hirzinger, "Exploiting structure in two-armed manipulation tasks for humanoid robots," in *IEEE International Conference on Intelligent Robots and Systems (IROS)*, oct. 2010, pp. 5446–5452.

[7] F. Stulp, A. Fedrizzi, and M. Beetz, "Learning and performing place-based mobile manipulation," in *Development and Learning, 2009. ICDL 2009. IEEE 8th International Conference on*, Jun. 2009, pp. 1–7.

[8] 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 *Proceedings of IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, dec. 2009, pp. 55 –61.

[9] N. Vahrenkamp, E. Kuhn, T. Asfour, and R. Dillmann, "Planning multi-robot grasping motions," in *Proceedings of IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, Nashville, USA, 2010, pp. 593–600.

[10] N. Vahrenkamp, T. Asfour, G. Metta, G. Sandini, and R. Dillmann, "Manipulability analysis," in *Proceedings of IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, (accepted) 2012.

[11] S. T. Rachev, M. Hechstetter, F. J. Fabozzi, and S. M. Focardi, *Probability and Statistics for Finance*. John Wiley & Sons, 2007.

[12] N. Vahrenkamp, T. Asfour, and R. Dillmann, "Efficient inverse kinematics computation based on reachability analysis," *International Journal of Humanoid Robotics*, vol. 9, no. 4, 2012.

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

[14] M. Togai, "An application of the singular value decomposition to manipulability and sensitivity of industrial robots," *SIAM Journal on Algebraic and Discrete Methods*, vol. 7, no. 2, pp. 315–320, 1986.

[15] N. Vahrenkamp, M. Kröhnert, S. Ulbrich, T. Asfour, G. Metta, R. Dillmann, and G. Sandini, "Simox: A robotics toolbox for simulation, motion and grasp planning," in *International Conference on Intelligent Autonomous Systems (IAS)*, 2012.