

Using a Model of the Reachable Workspace to Position Mobile Manipulators for 3-d Trajectories

Franziska Zacharias, Wolfgang Sepp, Christoph Borst and Gerd Hirzinger

Abstract— Humanoid robots are envisioned in general household tasks. To be able to fulfill a given task the robot needs to be equipped with knowledge concerning the manipulation and interaction in the environment and with knowledge about its own capabilities. When performing actions, e.g. opening doors or imitating human reach to grasp movements special 3-d trajectories are followed with the robot’s end-effector. These trajectories can not be executed in every part of the robot’s arm workspace. Therefore a task planner has to determine if and how additional degrees of freedom such as the robot’s upper body or the robot’s base can be moved in order to execute the task-specific trajectory.

An approach is presented that computes placements for a mobile manipulator online given a task-related 3-d trajectory. A discrete representation of the robot arm’s reachable workspace is used. Task-specific trajectories are interpreted as patterns and searched in the reachability model using multi-dimensional correlation. The relevance of the presented approach is demonstrated in simulated positioning tasks.

I. INTRODUCTION

Application scenarios for humanoid robots or mobile manipulators (Fig. 1) include general household tasks. Kitchen tasks for instance require fetching and carrying things, as well as manipulating and interacting with the environment. To accomplish these tasks, the robot has to use knowledge about the specific environment and knowledge about its own capabilities. It has to know how to open a cupboard or how well it can grasp objects. The robot also has to decide when to use which part of its body. We argue that not only the question when to include the upper body is important to accomplish tasks but also when to use the mobility of the base. To execute simple trajectories it is not always necessary to use the mobile base. On the contrary, if the mobile base is unnecessarily used, e.g., while opening a kitchen closet, additional forces have to be compensated. These forces are due to the fixed grasping of the door handle and the navigation errors of the mobile base. In this paper we propose an online method that determines where to place the robot. Furthermore, the method enables a planner to reason whether the mobile base is needed in a given task.

In a previous work [1], a mobile manipulator is positioned to execute linear constrained trajectories. However, more general types of trajectories, are needed in environments like a kitchen for opening doors and cupboards. Such specific, task-related trajectories might not only be imposed by the tool or object but also by requiring the robot to emulate human motions. In order to imitate prototypical movements



Fig. 1. The DLR robot *Rollin' Justin*.

of humans to reach an object [2] a task planner has to reason about where to place the robot.

We propose an algorithm that uses a model of the reachable workspace of a robot arm to determine where the robot can be placed or if a given task is solvable at all. We present results for 3-d trajectories using the example of opening a closet. Once the mobile manipulator is positioned, the trajectory is executed without using the mobile base.

II. RELATED WORK

The use of models encapsulating robot specific knowledge was recently taken up by several research groups. Pettré et al. [3] make the animation of a digital actor more efficient by dividing the large number of degrees of freedoms (DOF) of a humanoid into functional units providing the locomotion and the manipulation capabilities. Diankov et al. [4] use a similar functional structure for their humanoid robot to plan a path from a given start position to an object to be manipulated. In the process they furthermore consider a model of the reachable workspace of the robot arm to decide where the robot may stand to grasp an object and thus focus the search. Gienger et al. [5] use an object-specific model of the grasping capabilities of their humanoid robot to optimize the whole body motion to reach and grasp an object.

Most approaches to constrained trajectory planning for mobile manipulators combine the positioning of the robot with the search for feasible trajectories for the robot arm in the configuration space (C-space). Optimization and path planning techniques are used. Optimization techniques are

All authors are affiliated with the Institute of Robotics and Mechatronics, German Aerospace Center (DLR), Germany, franziska.zacharias@dlr.de

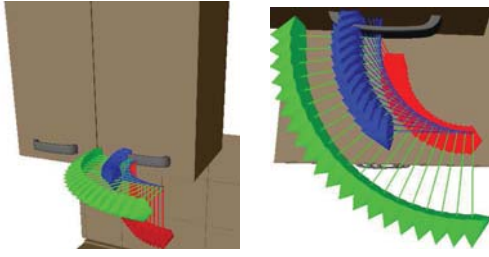


Fig. 2. (Left) Trajectory for opening a closet. (Right) A zoomed view.

applied to the whole kinematic chain. Multi-criteria optimization can also be used for positioning a mobile manipulator to reach a point. However choosing criteria weights, competing criteria, and the resulting local minima pose a great challenge [6]. When planning constraint motions for a mobile manipulator Stilman [7] uses the Jacobian transpose to project a given sample configuration into the subspace of configurations valid for the motion. However, the system always moves the mobile base to accomplish a task. For simple tasks this may not always be necessary. Furthermore, due to positioning inaccuracy of the mobile base additional forces will act at the fixed connection of the TCP, e.g., when opening a drawer. These have to be compensated. Diankov et al. [8] claim that fixed grasps on objects limit manipulation capabilities. They propose to use a set of caging grasps during path planning and execution to extend the possibilities of a mobile manipulator to fulfill the task. Whether or not a trajectory exists is determined by a timeout for the path search. In contrast, we aim at providing a decision module that can predict with high probability whether or not and where a trajectory is possible. The module is based on a reachability model for a robot arm and not on traditional path planning techniques.

III. PROBLEM ANALYSIS

In service robotic applications, unconstrained pick and place tasks are often encountered. In these tasks, the robot moves an object from a start to a goal position. To perform these tasks, standard path planning algorithms can be queried for a suitable path. However, not only unconstrained, freely plannable paths are needed. Interaction with the environment is subject to physical constraints. For opening a closet, the tool center point (TCP) attached to the last link of a robot arm is constrained to move on a circular path (Fig. 2). For a frame attached to the handle of the closet, the orientation of the z-axis (blue arrow) constantly changes. The radius and orientation of this path are connected with the design of the closet. Due to the robot arm kinematics and link limits, executing constrained trajectories will not be possible at arbitrary positions in a robot arm's workspace. Depending on a robot arm's capabilities or the arm's attachment to an upper body, some mobile manipulators may not be able to perform certain tasks at all, like opening a closet at a certain height. A method is needed to analyze the capabilities of a robot given an environment and typical tasks performed therein. Therefore, we are interested in a method that works for generic 3-d

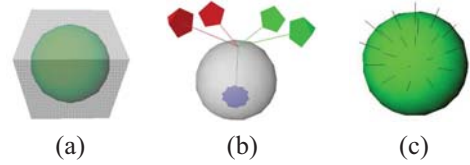


Fig. 3. Shows a sphere inscribed into the cube (a), exemplary frames for a point on the sphere (b), valid inverse kinematics solutions on a sphere (c).

trajectories of the robot arm TCP and determines where these trajectories are situated in a robot arm's workspace. Given this information, a mobile manipulator can be placed easily. In [1], we presented a method to search for linear constrained trajectories. However, this search method cannot be extended to cover 3-d trajectories. In this paper, we present a method to determine for given 3-d trajectories, where these trajectories are possible in a robot arm's workspace. We use a model of a robot arm's reachable workspace which is briefly introduced in Sec. IV. In the first stage, the model is analyzed. Sec.V describes how regions are extracted where the given 3-d cartesian space trajectory is possible. These are used to infer placements for the mobile manipulator. In the second stage, the placements are checked for collision-free execution of the trajectory. Results are reported in section VI.

IV. CAPABILITIES OF THE ROBOT ARMS

For a robot arm, the information which regions of the workspace are reachable from what directions can be discretized and described in a model [9]. We briefly summarize its main points in the following paragraph. The theoretically possible robot arm workspace is enveloped by a cube and subdivided into equally-sized sub cubes. Into each cube a sphere is inscribed and on this sphere n points are uniformly distributed (Fig. 3 (a), (c)). Frames are generated for each point on the sphere and serve as the target tool center point (TCP) for the inverse kinematics of the robot arm (Fig. 3 (b)). The normal to the sphere at a sphere point determines the z-axis (blue arrow) of the TCP frame. The orientation of the x- and y-axis is sampled equidistantly. If a solution is found for the specific frame then the frame is reachable. This fact is visualized by a black line perforating the sphere at the corresponding point. The spheres visualize the reachability for a region and are therefore called reachability spheres. The reachability sphere map is the aggregation of all spheres. This model can be used to visualize and inspect the orientation dependent reachability across the workspace and to approximate the shape of the robot arm workspace. Furthermore, the underlying data structure can be used to gather statistics about the workspace as well as perform task planning as shown in the following. Fig. 4 shows the reachability sphere map for the right arm of the robot *Rollin' Justin*. The map was cut in half for better inspection of the structure. The spheres are colored with respect to their reachability index D [9] which is the percentage of points on the sphere that are reachable (Fig. 5). The reachability index has a mean of 53 %, meaning that 53 % of the points on the sphere are

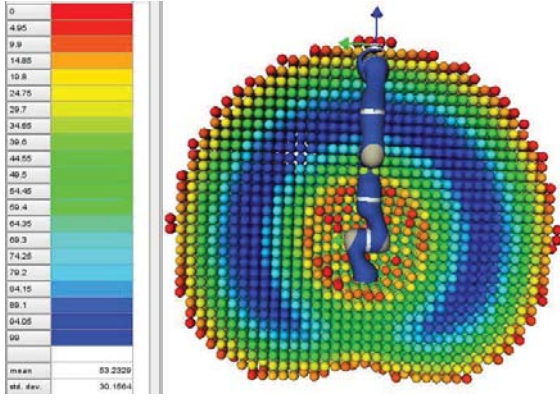


Fig. 4. The reachable workspace of the 7 DOF DLR light weight robot arm.

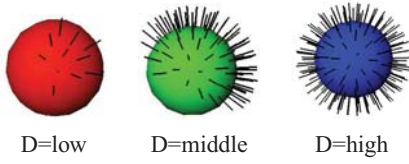


Fig. 5. Spheres with different reachability indices D from the reachability map for the right arm of *Rollin' Justin*.

reachable. This index gives a general impression of the robot arm's capabilities and does not provide information about directional preferences. However, it can be seen that the capability of *Rollin' Justin* to reach regions varies strongly across the workspace. Hence, specific trajectories are not possible everywhere in the workspace. In the next section, we show how the reachability representation can be used to position the robot for 3-d trajectories. Note that the representation is computed offline only once. In the current implementation its size is 62 MB.

V. PLACEMENT FOR 3-D TRAJECTORIES

A. The search pattern

We use the example of opening a closet introduced in Sec. III to illustrate our method. If a closet has to be opened, the end-effector grasps the handle and moves on an arc (Fig. 2). We assume the trajectory template followed by the robot arm TCP to be given as a sampled sequence of frames F_l , $l > 0$ with respect to a local reference system. A frame F_l is represented as a homogenous matrix $F_l = \begin{pmatrix} R_l & \vec{t}_l \\ \vec{0}^T & 1 \end{pmatrix}$ with $R_l = (\vec{x}, \vec{y}, \vec{z}) \in SO(3)$ and $\vec{t}_l \in \mathbb{R}^3$ describing rigid body rotation and translation. The frames are mapped to their discrete representations in the model. The set of accordingly mapped frames F_l is called search pattern p hereafter. The mapping works in the following manner. For each position of the Cartesian trajectory we first determine the sphere it maps to. Fig. 6 (left) exemplarily shows a trajectory superimposed on the workspace discretization underlying the reachability sphere maps. The 2-d projection was chosen for illustration. The filled spheres symbolize those spheres the trajectory is

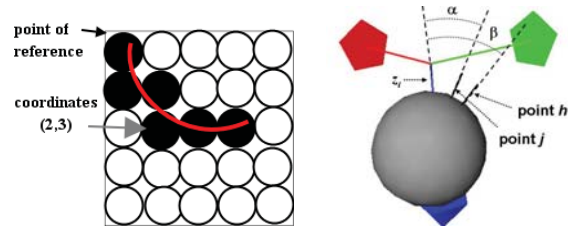


Fig. 6. Discretization of a trajectory. (Left) 2-d view of mapping translations on the sphere map grid. (Right) Mapping of the frame orientation to a point on the sphere.

mapped to.

$$f(\vec{t}_l) : \mathbb{R}^3 \rightarrow \mathbb{N}^3 \text{ with } f(\vec{t}_l) = (p[l].x, p[l].y, p[l].z) \quad (1)$$

Let f be the function that maps \vec{t}_l to a sphere in the pattern given the discretization, i.e. the sphere diameter, also underlying the reachability sphere map. Each sphere is represented by an offset in $p[l].x, p[l].y, p[l].z$ of the sphere space with respect to the point of reference of the pattern. The frame F can equally well be interpreted as a coordinate

system base $F = \begin{pmatrix} \vec{x} & \vec{y} & \vec{z} & \vec{t} \\ 0 & 0 & 0 & 1 \end{pmatrix}$ with $\vec{x}, \vec{y}, \vec{z}, \vec{t} \in \mathbb{R}^3$.

In a second step, the z-axis \vec{z}_i of frame F_l is mapped to the best fitting point on the sphere and determines the pattern element $p[l].k \in [1, n]$. Let $F_{i,0}$ be the coordinate system attributed to point i and nominal orientation 0 around $\vec{z}_{i,0}$. Note that $\vec{z}_{i,0} = \vec{z}_{i,m} = \vec{z}_i$ and $|\vec{z}_i| = |\vec{z}_i| = 1$. Eq. 2 describes the mapping which is illustrated in Fig. 6 (right).

$$p[l].k = \operatorname{argmin}_{i \in [1, n]} (\operatorname{acos}(\vec{z}_i^T \cdot \vec{z}_l)) \quad (2)$$

For the computation of the reachability sphere map the orientation around the z-axis of a frame (Fig. 3(b)) was discretized into m steps. In the last part of the mapping process the orientation around the z-axis of frame F_l has to be mapped to the sphere data structure, i.e. to one of these m orientations. Let $F_{k,0}$ be the frame belonging to the previously computed sphere point k and nominal orientation 0 . The x-axis \vec{x}_l of frame F_l is projected onto the xy-plane of the coordinate system defined by the frame $F_{k,0}$ (Eq. 3). Let P_{xy} be the projection matrix for projection onto the xy-plane.

$$\vec{x}_l^{new} = R_{k,0} \cdot P_{xy} \cdot R_{k,0}^{-1} \cdot \vec{x}_l \quad (3)$$

$$\alpha = \operatorname{acos} \left(\frac{\vec{x}_l^{new T} \cdot \vec{x}_{k,0}}{|\vec{x}_l^{new}| \cdot |\vec{x}_{k,0}|} \right) \quad (4)$$

The angle between the projected axis and the x-axis of frame $F_{k,0}$ is then computed (Eq. 4) and discretized. It determines the pattern element $p[l].o \in [1..m]$ with $p[l].o = \lfloor \frac{\alpha}{\Delta o} + 0.5 \rfloor$. Δo denotes the discretization step width of the orientation around \vec{z} . Fig. 7 shows an example trajectory for opening a cupboard and the search pattern in the space of spheres. In Fig. 7 (left) the large coordinate frames represent the original trajectory frames. The smaller frames represent the mapped frames in the sphere data structure. In Fig.7 (right) the red

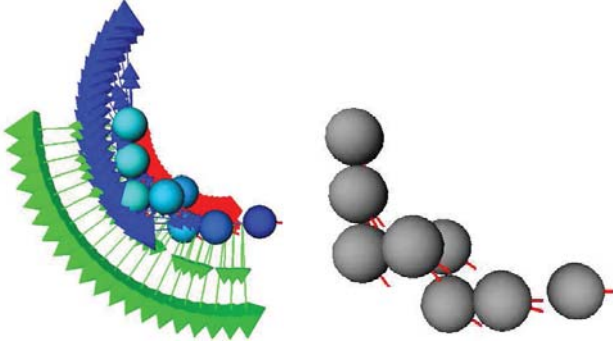


Fig. 7. (Left) The trajectory mapped to spheres. (Right) A zoomed view showing the sphere points mapped.

lines show to which lines on the sphere the frames were mapped.

B. The search for the trajectory in the workspace

Cross-correlation is a standard technique in signal processing to determine the shift between two signals. The signals are specified over the same domain, e.g. $\mathbb{R} \rightarrow \mathbb{R}$ for audio signals over time or $\mathbb{R}^2 \rightarrow \mathbb{R}$ for static grey images. The result of a correlation is a signal in the same domain, showing peaks at those locations where the two signals best match each other. We use this idea to find the search pattern obtained in Sec.V-A in the reachability sphere map. The search is done by correlating the sphere data structure with the given search pattern. Figuratively speaking the pattern is moved across the 3-d data structure and compared with the data present. Equation 5 implements the correlation between the two signals. Let D be the 3-d data structure which represents the reachability sphere map. The search pattern p is obtained as described in the last section.

$$(D * p) = \sum_{ix} \sum_{iy} \sum_{iz} \sum_{l=0}^{p.length} S(ix_l, iy_l, iz_l)[p[l].k][p[l].o] \quad (5)$$

$$S(ix, iy, iz) = D[ix]D[iy]D[iz] \quad (6)$$

$$ix_l = ix + p[l].x, iy_l = iy + p[l].y, iz_l = iz + p[l].z \quad (7)$$

$S(ix, iy, iz)$ in Eq. 6 describes the location of a sphere in the 3-d reachability sphere map. ix, iy, iz iterate over the whole workspace D minus the dimension of the 3-d search pattern. In Eq. 7 the sphere offset of the pattern element l is added to the current starting point of the pattern in the 3-d sphere space. Given a number of discretized orientations around the z-axis (Fig. 3(b)) the value of $S(ix, iy, iz)[p[l].k][p[l].o]$ encodes for point $p[l].k$ on the sphere whether the orientation $p[l].o$ around the z-axis is reachable. The variable is 1 if the orientation is reachable and 0 if it is not reachable. As a result $(D * p)$ is a representation of how well the trajectory fits across the map. We search those places in the robot arm workspace where the pattern fits completely. Fig. 8 exemplifies the correlation result for opening a closet at a certain height. Justin's torso is in its zero position and the trajectory is at about shoulder height (Fig. 8 (top)). Since

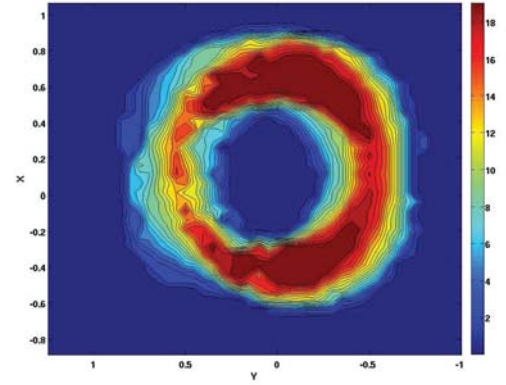
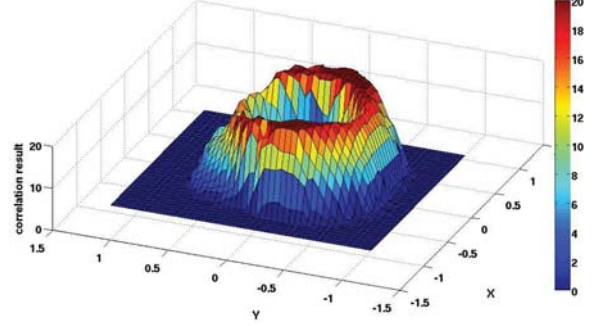
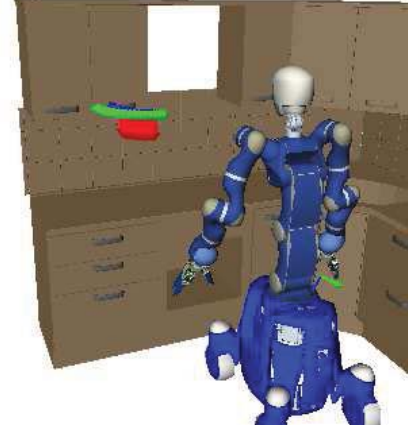


Fig. 8. (top) Justin in the Kitchen. (middle) Correlation result for the trajectory. (bottom) Contour view of correlation result.

the trajectory is composed of 20 frames, the correlation result ranges from 0 to 20 (Fig. 8 (middle), (bottom)). Note that the positive x-axis indicates the front of the robot. A value of 20 (dark red) means all frames of the trajectory are predicted to be reachable if the trajectory is started at the corresponding point in the robot arm workspace. It can be seen that the region in which the trajectory can be performed completely is quite small in this example.

Note that with this method, the pattern will not be found if it occurs in the image in a different orientation. The standard solution to this is to rotate the pattern using a fixed stepsize. Accordingly, the original cartesian space trajectory can be rotated before being mapped into the sphere space.

C. Computing the robot base position

The reachability map is computed with respect to the robot arm base. Thus if the robot arm base is moved, the map moves accordingly. Once we have the position of the trajectory in the robot arm workspace with respect to the robot arm base, the position of the mobile manipulator with respect to the world can be determined easily.

The z-axis of the world system is assumed to point upwards. The transformation from the old to the new robot base position involves only a rotation R around the z-axis of the world system and a translation t in the xy plane. Let $\vec{t} = (x, y, z)^T$ be the translation and R be the rotation of the search pattern with respect to the map space for which the correlation of Eq. 5 reaches the maximum. Eq. 8 gives the target position of the arm base in a reference frame F_0 , e.g. the world base. The placement of the mobile base follows directly.

$$T_{arm}^0 = T_{object}^0 \cdot (T_{object}^{arm})^{-1} = T_{object}^0 \begin{pmatrix} R & \vec{t} \\ \vec{0}^T & 1 \end{pmatrix}^{-1} \quad (8)$$

D. Computational complexity

In this section we present the examination of complexity that led us to perform the search in position space as opposed to in frequency domain. Cross-correlation in image processing customarily transforms two images into frequency domain using fourier transformation. The reachability sphere map and the search pattern are the correspondences of the image. In frequency domain the spectra are multiplied and the result is transformed back using inverse fourier transformation. It is known that the fast discrete fourier transform (FFT) using Cooley-Tuckey's radix-2 algorithm has a time complexity of $O(N \log(N))$, where N is a power of factor 2. Let N_D^3 denote the volume of the discretized robot workspace. The complexity is highest if the search pattern has the dimension of the workspace, i.e. $N_P = N_D$. According to the number of multiplications for a single FFT [10], the total cost for two fourier transformations and for the multiplication in frequency domain involved are

$$\text{Cost}_{\text{freq}} = |O|N_D^3 \log_2(N_D^3) + |O|N_D^3. \quad (9)$$

where $|O| = n \cdot m$. n is the number of discretized orientations, i.e. points on the sphere and m the number of discretized orientations around \vec{z} . Note that the cost of the fourier transformation of the reachability map is neglected because it can be computed once and used for different planning tasks. In the case of the discretization of Justin's arm workspace with side length $N_D = 40$ spheres and $|O| = 200 \cdot 12$ orientations, the total number of multiplications $\text{Cost}_{\text{freq}}$ amount to $2.6 \cdot 10^9$.

In contrast, the number of multiplications for general cross-correlation in the space domain amount to $|O| \cdot N_D^3 \cdot N_P^3 = 1.5 \cdot 10^{11}$, whereas a side length $N_P = 10$ of the trajectory map is assumed. Contrary to the general case, significant optimizations can be applied here because only a few entries in the trajectory volume are non-zero. Let $|p|$ denote the

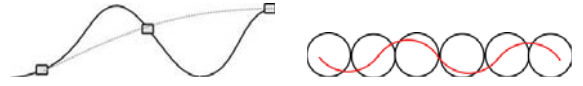


Fig. 9. (Left) The aliasing effect when sampling a 2-d trajectory. (Right) A trajectory with a low amplitude.

length of the discretized trajectory, then the multiplications amount to

$$\text{Cost}_{\text{space}} = (N_D - N_P)^3 \cdot |p|. \quad (10)$$

Only a subvolume of the robots reachability map is considered, because the trajectory is requested to be completely within the workspace of the robot. Whenever $|p| = N_P$, the costs reach its maximum at $N_P = N_D/4$. In the case of Justin's arm, this corresponds to only $2.7 \cdot 10^5$ multiplications¹, which are four orders of magnitude fewer than assessed for the correlation with the fourier transformations.

E. Discretization issues

In this section we describe the requirements for the data representation and the trajectory representation that have to be met to ensure the success of our approach. The requirements for the reachability map concern the workspace discretization step width, the number of points on the spheres and the orientations around the z-axis. In general it can be stated the greater the workspace discretization step width, i.e. the sphere diameter, the worse the prediction performance. A corner stone of our approach is that the spheres inscribed in a subcube describe the reachability for this region. Due to memory consumption the spheres cannot be arbitrarily small. Therefore a sphere diameter of 0.05 m was empirically chosen. If too few points are distributed on the sphere, then direction-specific reachability is not represented well anymore. In this paper the orientation of the z-axis is captured by uniformly distributing 200 points across a sphere. In this case the minimum angle between two points is 8.12° . The orientation around the z axis was discretized into 12 steps. To unambiguously represent the task-specific trajectory template, the trajectory has to be sampled according to the Nyquist-Shannon sampling theorem [11]. If this theorem is violated and too few frames characterize the trajectory, the trajectory cannot be correctly reproduced, i.e. the pattern does not correctly represent the trajectory. In this case the trajectory is aliased with a less frequent one (Fig. 9 left). The search results are not valid for the original trajectory. In this case the ratio of predicted to actually reachable trajectories is low. For trajectories with low amplitude i.e. $\text{amplitude} < \text{sphere radius}$ (Fig. 9 right) we assume that the interpolation assumption holds which underlies the discretization of the robot arm workspace. It is assumed that at each point of a subcube the same orientations are reachable as on the sphere located at its center.

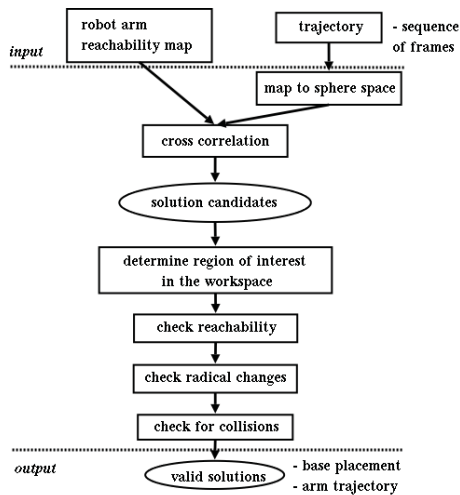


Fig. 10. The algorithm at a glance.

VI. EVALUATION

In this section we evaluate the presented approach for different types of tasks and corresponding trajectories. For reference the approach is summarized in Fig. 10. The mobile humanoid robot *Rollin' Justin* [12] is used. It consists of an humanoid upper body system mounted on a mobile base with variable footprint. The humanoid upper body is composed of a 3-DOF torso, two 7-DOF arms, two 4-finger hands and a head. The 7-DOF DLR light weight robot arm serves as the left and the right arm of the robot *Rollin' Justin*. Inverse kinematics solutions were computed as detailed in [13].

A. Naive search vs. model-based search

Without a reachability map, a brute force search could be thought of to determine whether a given trajectory can be performed by the robot. The start point is randomly sampled using a uniform distribution. The trajectory is attached to this start point. However the sample only contributes to the computed success rate if the position of the first and last frame of the trajectory lies within the hull of the reachable workspace of the robot arm. The trajectory is checked for reachability using inverse kinematics. A relative success measure is computed from the number of successes. In Tab. I we report the likelihood to find the arc for opening a closet at a certain height in the workspace. Results are reported for the brute force search and the reachability model based search for the trajectory in one orientation. The results for the brute force approach show that a lot of effort is wasted to find valid trajectories. While our efficient approach needs 1.6 s to find and verify the 96 trajectories, the naive approach finds the same number of valid trajectories in 20 s. Considering that the trajectories may still be inconsistent or colliding, the advantages of our model-based approach are emphasized.

¹since both volumes are binary, the multiplication can be replaced with a simple binary AND operation.

brute force search			reachability model		
samples	valid	%	predicted	valid	%
$1 \cdot 10^6$	39807	3.98	119	96	80.7

TABLE I
RESULTS OF THE BRUTE FORCE SEARCH ARE COMPARED WITH THE REACHABILITY MODEL-BASED APPROACH.

torso conf.	nr. of solutions				
	predicted	real	reconf.	colliding	valid
C1	119	96	70	82	7
C2	204	195	97	157	27

TABLE II
THE SET OF SOLUTIONS IS ANALYZED W.R.T. DIFFERENT CRITERIA.

B. Validation of solutions

The correlation process itself is very efficient. It needs 25 ms to find all occurrences of an arc trajectory (Fig. 2) with 20 frames in one orientation within the discretized workspace. Only those correlation results are considered that lie in an area of the workspace that is of interest for the given task. The solutions have to be validated because of the following:

1) *Generalization assumption*: The correlation process provides a number of starting points for trajectories. These need to be checked for actual reachability since the reachability sphere map assumes that entries in the map generalize over the complete subregion, i.e. subcube. It is assumed that at each point of a subcube the same orientations are reachable as on the sphere located at its center, i.e. that the reachability structure does not change for small increments in space. Normally, this assumption will hold. However at the inner and outer border of the reachable workspace or at the border of structurally different regions, this assumption may be violated.

2) *Collision-free and consistent trajectories*: Using the reachability sphere map, reachability of the trajectory is independently evaluated for the individual frames. It is not guaranteed that the robot is able to follow a smooth trajectory between frames, e.g., in the vicinity of singularities reconfigurations of the robot arm will occur. Therefore each trajectory is checked for consistency by setting a threshold on the allowed link-wise change of two configurations for two adjacent path steps. Additionally the trajectory is also checked for collisions for the robot with the environment and the robot arm with e.g. the head or the upper body.

C. Robots working in the kitchen

A robot working in a kitchen is above all required to be able to open cupboards to extract dishes or fill the dishwasher. Robot placements of the mobile manipulator (Fig. 1) are computed and validated for opening the door of a cupboard (Fig. 2). The trajectory search was only performed for the original orientation of the trajectory. Results in Tab. II are reported for two different configurations of the movable upper-body of the robot *Rollin' Justin*. The results show significant differences for the two torso configurations.

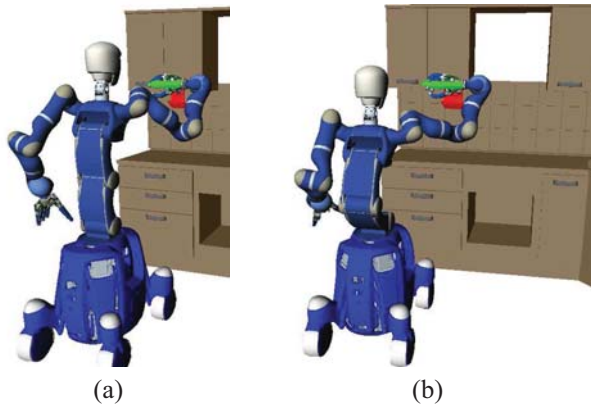


Fig. 11. Rollin' Justin is placed to open a closet. (a) The torso is in configuration C1. (b) The torso is in configuration C2.

correlation	reachability	reconfiguration	collision	total
25 ms	1564 ms	0.113 ms	31 ms	1620 ms

TABLE III
COMPUTATION TIMES FOR THE STEPS OF THE ALGORITHM.

For configuration C2 more reachable solutions are found. These results indicate that the presented approach is able to decide on a beneficial torso configuration for a given task. Fig. 11 exemplarily shows a robot placement for each torso configuration. After the validation step, a set of valid solutions remain, that a task planner can choose from. In both cases, the number of collisions is striking. Since the workspace of the arm is nearly symmetric (compare Fig.4) solutions are found in front of and behind the torso of Justin (Fig. 8). In the latter case Justin has to be placed inside the kitchen closets to execute the trajectories. Furthermore often collisions of the arm and the head were encountered. The results for the check for consistency and freedom of collision should be seen as a proof of concept. Our inverse kinematics currently computes independent solutions for the individual frames and does not exploit the null space of the 7-DOF robot arm to avoid collisions. For trajectories that are currently invalid, we expect that there exist alternative arm configurations that lead to consistent and collision-free solutions. We measured the time consumption for each part of the algorithm for the torso in configuration C1. The computation times are summarized in Tab. III. The test was performed on an Intel Pentium D 3 GHZ computer with 2 GB memory. The accompanying video illustrates the algorithm for the torso in configuration C1.

VII. CONCLUSIONS AND FUTURE WORKS

We presented an algorithm to position a mobile manipulator to execute 3-d trajectories using a model describing the capabilities of a robot arm. Trajectories are localized in the reachability map using correlation and then validated. Once a trajectory is found, deriving the corresponding mobile manipulator position is straight forward. The method was illustrated using the example of opening a closet using the

humanoid robot *Rollin' Justin*. The proposed algorithm is not only relevant for service robotic tasks. It can also be used for online positioning industrial robots e.g. for welding tasks or to compare the capabilities of different robot arms. The presented method can be used to evaluate how well a mobile manipulator is suited for specific environments. The determined number of solutions for the constrained task can be assumed to correlate with the ability of the robot to cope with disturbances e.g. objects left behind by a human, or a human standing in the way. It could be argued that the mobile base can always be used for compensation. However, then the robot cannot operate in tight spaces and has to compensate forces occurring at the TCP.

The method is especially suited to decide whether or not a task e.g., opening a door, can be done without using the mobile base. This information could be used by a task planner to decide which planner or execution component to trigger. Navigation controllers on a real robot accumulate navigation errors. Therefore a planned position will never be exactly reached. In future work the proposed method should be extended to cope with uncertainty in navigation and object localization.

VIII. ACKNOWLEDGMENTS

The research has been funded by the EC Seventh Framework Programme (FP7) under grant agreement no. 216239 as part of the IP DEXMART.

REFERENCES

- [1] F. Zacharias, C. Borst, M. Beetz, and G. Hirzinger, "Positioning mobile manipulators to perform constrained linear trajectories," in *Proc. IEEE Int. Conf. on Intelligent Robots and Systems (IROS)*, 2008.
- [2] J. Vaughan, D. Rosenbaum, and R. Meulenbroek, "Planning reaching and grasping movements: The problem of obstacle avoidance," *Motor Control*, vol. 5, no. 2, 2001.
- [3] J. Pettré, J.-P. Laumond, and T. Siméon, "A 2-stage locomotion planner for digital actors," in *Proc. Eurographics symposium on Computer Animation (SCA'03)*, 2003.
- [4] R. Diankov, N. Ratliff, D. Ferguson, S. Srinivasa, and J. Kuffner, "Bispace planning: Concurrent multi-space exploration," in *Proc. Int. Conf. on Robotics: Science and Systems*, 2008.
- [5] M. Gienger, M. Toussaint, N. Jetchev, A. Bendig, and C. Goerick, "Optimization of fluent approach and grasp motions," in *Proc. IEEE Int. Conf. on Humanoid Robots*, 2008, pp. 111–117.
- [6] F. G. Pin and J.-C. Culioli, "Optimal positioning of combined mobile platform-manipulator systems for material handling tasks," *J. Intelligent and Robotic Systems*, vol. 6, no. 2-3, pp. 165–182, 1992.
- [7] M. Stilman, "Task constrained motion planning in robot joint space," in *Proc. IEEE Int. Conf. on Intelligent Robots and Systems (IROS)*, 2007, pp. 3074–3081.
- [8] R. Diankov, S. Srinivasa, D. Ferguson, and J. Kuffner, "Manipulation planning with caging grasps," in *Proc. IEEE Int. Conf. on Humanoid Robots*, 2008.
- [9] F. Zacharias, C. Borst, and G. Hirzinger, "Capturing robot workspace structure: Representing robot capabilities," in *Proc. IEEE Int. Conf. on Intelligent Robots and Systems (IROS)*, 2007, pp. 3229–3236.
- [10] I. N. Bronstein, K. A. Semendjajew, G. Musiol, and H. Mühlig, *Taschenbuch der Mathematik*, 6th ed. Harri Deutsch, 2005.
- [11] C. E. Shannon, "Communication in the presence of noise," *Proc. Institute of Radio Engineers*, vol. 37, no. 1, pp. 10–21, 1949.
- [12] M. Fuchs, C. Borst, P. R. Giordano, A. Baumann, E. Krämer, J. Langwald, R. Gruber, N. Seitz, G. Plank, K. Kunze, R. Burger, F. Schmidt, T. Wimboeck, and G. Hirzinger, "Rollin justin design considerations and realization of a mobile platform for a humanoid upper body," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2009.
- [13] R. Konietzschke and G. Hirzinger, "Inverse kinematics with closed form solutions for highly redundant robotic systems," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2009.