Task Maps in Humanoid Robot Manipulation

Michael Gienger, Marc Toussaint, and Christian Goerick

***Abstract*— This paper presents an integrative approach to solve the coupled problem of reaching and grasping an object in a cluttered environment with a humanoid robot. While finding an optimal grasp is often treated independently from reaching to the object, in most situations it depends on how the robot can reach a pregrasp pose while avoiding obstacles. We tackle this problem by introducing the concept of *task maps* which represent the manifold of feasible grasps for an object. Rather than defining a single end-effector goal position, a task map defines a goal hyper volume in the task space. We show how to efficiently learn such maps using the Rapidly exploring Random Tree algorithm. Further, we generalise a previously developed motion optimisation scheme, based on a sequential attractor representation of motion, to cope with such task maps. The optimisation procedure incorporates the robot's redundant whole body controller and uses analytic gradients to jointly optimise the motion costs (including criteria such as collision and joint limit avoidance, energy efficiency, etc.) and the choice of the grasp on the manifold of valid grasps. This leads to a preference of grasps which are easy to reach. The approach is demonstrated in two reach-grasp simulation scenarios with the humanoid robot *ASIMO*.**

I. Introduction

Consider the problem of grasping the handle of a basket in a cluttered environment. There are multiple criteria to be fulfilled for a successful reaching and grasping motion. First, the initial robot posture may be far from a feasible grasp posture. Hence we need to plan a collision-free reaching motion towards a feasible pregrasp. A classical solution to this is a motion planning technique towards a predefined goal posture. Second, the handle of the basket can be grasped at many different positions. The best grasp choice clearly depends on the initial posture of the robot and the obstacles in the environment. Hence, predefining a grasp position and using a classical motion planner is a suboptimal solution. In essence, we are faced with the *coupled problem of grasp choice and reaching motion planning*. In this paper we solve this problem by proposing an object representation in terms of an object-specific task map which can be learnt from data and, during movement generation, efficiently coupled into a movement optimisation process.

Most existing literature on grasp optimisation focuses on the grasp itself, isolated from the reaching movement. For instance, [1] reviews the various literature on defining grasp quality measures, [2] learn which grasp positions are feasible for various objects, [3] efficiently compute good grasps depending on how the objects shall be manipulated, and [4] simplify the grasp computation based on abstracting objects into shape primitives. The coupling to the problem of reaching motion optimisation is rarely addressed. A recent approach [5] makes a step towards solving the coupled problem by including a "environment clearance score" in the grasp evaluation measure. In that way, grasps are preferred which are not prohibited by immediate obstacles directly opposing the grasp. Still, the full reaching motion is neglected in the grasp evaluation.

Our approach is based on directly coupling the grasp choice problem into the motion optimisation procedure, and thereby solving both coupled problems in an integrated framework. The motion optimisation is based on recent work which optimises trajectories based on a sequence of task space attractors [6]. A more probabilistic view on this is given in [7]. If we could predefine a desired grasp posture as the goal of the reaching movement, this optimisation technique could readily compute optimal reaching motions. However, given the uncertainty of the choice of grasp we need to adapt the method.

A key pre-requisite for this is to learn object-specific *task maps* which represent the set of feasible grasps for an object. This is similar to the approach in [8] where an explicit (grid-based) representation of the feasible workspace is learnt, and to the object-specific "valid grasp sets" in [5]. Once such a map is learnt, it defines a *goal set* in a certain task space of the reaching motion rather than an explicit goal position. Hence, we generalise the motion optimisation method to cope with general goal sets in arbitrary task spaces.

The previous approaches to learning maps of the workspace or of feasible grasps are based on exhaustive sampling over a grid. If the grasp space is high dimensional (e.g., 6-dimensional when grasps are parametrised by the preshape position and orientation) a grid-based sampling approach is rather inefficient. We propose to use a more efficient technique to learn task maps based on Rapidly exploring Random Trees (RRTs, [9]). While this technique was so far used for exploring feasible trajectories from goal to start through a configuration space, we employ it for rapidly building a tree of feasible pregrasps for a specific object.

To summarise, the key novelties of our approach are:

- We solve the coupled problem of grasp choice and reaching motion optimisation. This leads to a preference of grasps which are easy to reach (reachable with little cost) and thereby a disambiguation of grasp choice for manifolds of feasible grasps.
- We learn object-specific task maps as a representation of a goal manifold in a task space, which can either be coupled to the motion optimisation methods or to

Michael Gienger and Christian Goerick are with the Honda Research Institute Europe, Carl-Legien-Strasse 30, 63073 Offenbach/Main, Germany. Marc Toussaint is with the Machine Learning group at Technical University Berlin, Franklinstr. 28/29, 10587 Berlin, Germany.

standard reactive task-space control.

- We propose an efficient learning scheme of task maps using RRTs to rapidly explore the set of feasible grasp for an object.

This paper is organised as follows. In the next section we introduce task maps and focus on the problem of learning them. Section III addresses our motion generation framework, which is based on ASIMO's whole body motion control framework [10]. Section IV revisits briefly the sequential attractor representation of motion. In Section V we detail the generalisation of the motion optimisation to general goal sets in arbitrary task spaces. This will allow us to incorporate the learnt task maps into the optimisation framework. Finally Section VI presents simulation studies for a long stick and for the handle of a basket. We find that the resulting choice of grasp nicely depends on the efficiency of the reaching motion towards the grasp from the initial robot posture.

II. Learning Task Maps

During the learning phase, the robot needs to build a map of feasible grasps. We assume that we parametrize a grasp in terms of the hand position and orientation before the fingers are closed – in coordinates relative to the object frame. The closing of the fingers is controlled by a feedback loop coupled to the haptic sensors. Given a vector $g \in \mathbb{R}^6$ of grasp parameters we can execute and evaluate the grasp in a simulation environment, resulting in a quality measure $f \in \mathbb{R}$. In abstract terms, we are given an evaluation function $C: \mathbb{R}^6 \rightarrow \mathbb{R}$.

In detail, grasps are evaluated as follows. The hand is positioned at a proximate pregrasp pose in front of the object, the fingers are opened. The 6 parameters $g \in \mathbb{R}^6$ determine the initial position and orientation of the preshaped hand *relative* to the object frame. The evaluation now follows a feedback loop as illustrated in Figure 1. First the hand moves forward. If contact is detected, the fingers are closed ("power grasp"). The finger movement is controlled according to the coupled 1-dof spring-tendon hand mechanism of the humanoid robot *ASIMO*. Once a sufficient enclosing force is detected, the object is lifted up and slightly rotated. A control sequence is considered to be successful if the finger segments still measure an enclosing contact state after the object is lifted. Otherwise, its fitness is zero.

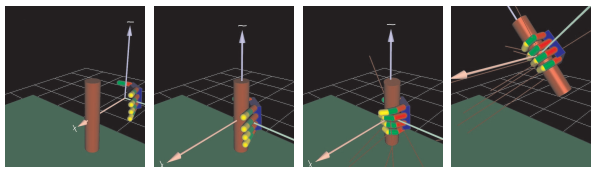


Figure 1: Grasp sequence

The problem of grasp exploration is to build a map of feasible grasps by efficiently collecting points $g \in \mathbb{R}^6$ for which $C(g)$ is above some threshold θ. A popular approach is to sample the grasp space along a predefined grid [3], [5]. However, in the case of a 6-dimensional grasp space this is rather inefficient. Instead we use the idea of Rapidly exploring Random Trees (RRTs, [9]) which so far were only used in the realm of planning for the rapid exploration of feasible paths through a configuration space. Even though we are not interested in paths through the task map, RRTs provide an efficient means to explore the map of feasible grasps. More concretely, we assume we are given one feasible grasp $g_0 \in \mathbb{R}^6$ to start with. We add g_0 to a set $T = \{g_0\}$. Next we sample a uniform random point x in the greater space $\mathbb{R}^6$, find the point $g_i \in T$ which minimizes the distance to x, and define the next exploration point g as $g = g_i + \epsilon \frac{x - g_i}{|x - g_i|}$. In words, this new exploration point is a step of length ϵ away from g_i in the direction of x. We evaluate the grasp g. If $C(g) > \theta$ we add g to the set T (and memorize a tree link from g_i to g), otherwise we discard it.

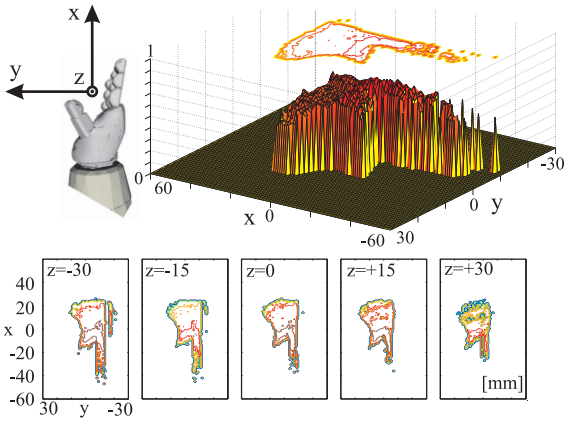


Figure 2: Task map for object-hand relations

This exploration strategy explores a contiguous region of valid parameters, forming a cluster of solutions in the task map. To illustrate the proposed concept, we generated a rather complete map for grasping a cylinder using an exhaustive search algorithm. Figure 2 shows the parameters of the lateral (y-direction) and frontal (x-direction) pregrasp offset between hand and object vs. the quality of the grasp. The inclination of the hand was upright. The lower part of the figure shows the contours when the cylinder was grasped at different heights (z-direction). In this example, the region of grasping success is contiguous. However, valid parameters don't fall into only one contiguous region, but may rather be clustered in several disjoint clusters. It is for instance very likely that a similar region exists if the hand is rotated about 180°.

We applied the presented RRT-based exploration scheme to the problem of grasping a stick and the handle of a basket. The exploration has been initialized with a successful grasp at the center of each object. Figure 3 shows the resulting

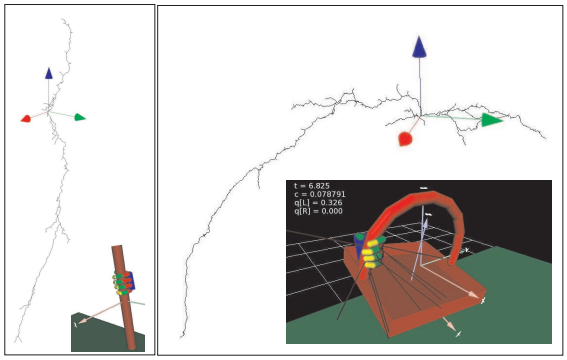


Figure 3:
Left: RRT of valid pregrasp poses for a cylinder, right: for the handle of a basket

RRTs after exploration, using the presented grasp control sequence. To illustrate the learnt RRT, only the position parameters of the pregrasp pose are indicated in the graphical visualization. However, the task map is defined over the full 6-dimensional grasp space and includes parameters for the orientation. It can very nicely be seen that the tree explores the geometry of both objects. The RRT-based exploration results in a significant speed-up as compared to an exhaustive parameter search, but has the disadvantage to only find solutions that are within one contiguous region.

While the chosen objects have a rather simple shape, the proposed scheme can also be applied to more complex objects without loss of generality. In such cases, the rather simple control sequence should be extended to more elaborate strategies. The task map would then relate the more sophisticated control parameters to a chosen quality criterion.

III. Movement Representation and Control

The learnt task maps are the basis to integrate the problem of grasp choice in our motion optimisation framework. This optimisation framework is based on representing motion as a sequence of task space attractors [6]. In this section we briefly review to the basics of this motion generation process: the general definition of task spaces and attractor dynamics to generate whole body motion for high-dimensional humanoid robots.

Findings from the field of biology impressively reveal how efficiently movement is represented in living beings. Besides the well-known principle of movement primitives, it is widely recognized that movement is represented in various frames of reference, such as in eye centered, reach and grasp centered or object centered ones [11].
We borrow this principle and represent robot motion in a suitable task representation. For this, the robot control model is described in the form of a tree structure as depicted in Figure 4. The individual links are connected by degrees of freedom (joints) or fixed transformations. Further, the tree may also comprise objects from the environment. This

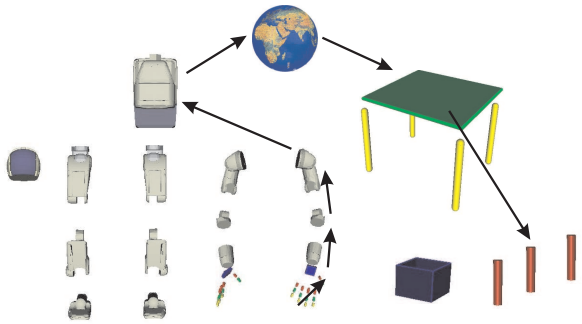


Figure 4: Robot control model including environmental objects

allows to derive the inverse kinematics equations not only with respect to a heel or world reference frame, but also to formulate task descriptors accounting for robot-object relations. We define a task as the relative movement of two tree nodes[1] and such can compute the task velocity as $\dot{x}_{task} = \dot{x}_{ef} - \dot{x}_{base}$. Indices *ef* and *base* denote the effector body and its reference, respectively.

The choice of effector and reference yields some interesting aspects. This is illustrated in Figure 5 for a simple planar example. Representing the movement of the hand with respect to the cylinder results in the left part of Figure 5. A coordinated hand-object movement has to consider three task variables $(x\ y\ \varphi)$. Switching the frame of reference and representing the object movement with respect to the hand leads to a description of the movement in hand coordinates. In this example, this might be advantageous, since the object is symmetric and can be approached from any side. While in the first case the task variables are dependent, in the second case φ and y are invariant and can be set to zero. There are many other examples, such as representing a gazing controller as an object in head-centered coordinates which is "pointed" to by the focal axis, or a pointing controller in a similar way.

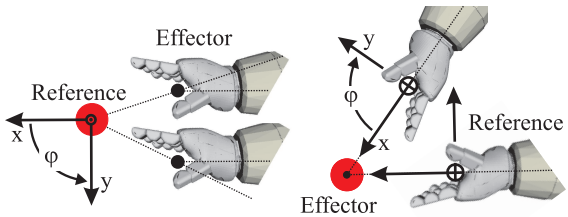


Figure 5: Relative hand-object task description

A task can be described in different ways, for instance

[1]There are other special cases for tasks, for instance the overall linear and angular momentum, etc.

as linear position, inclination, spherical and Euler angles, etc. A task element may comprise just individual parts of such a description, such as the vertical element of a 3-D position. Based on this, we derive an augmented Jacobian holding all controlled task elements (see also [10], [12]). The underlying whole body motion control is based on the scheme by Liegeois [13] [14], the equations are given in Table I, eq. (A.2). Redundancies are resolved by mapping the gradient of an optimisation criterion (joint limit avoidance, etc.) into the null space of the motion.

The trajectories are generated using a dynamical systems approach. This is closely related to the biological findings, and yields further advantages like robustness against perturbations and dynamical environments [15]. We apply a simple attractor system [6], [10] to the task elements to be controlled. This is depicted in Figure 6, where a sequence of 3 attractor potentials leads to the indicated smooth trajectory.

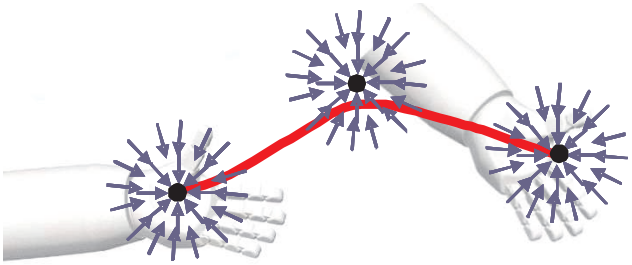


Figure 6: Attractor sequence and resulting trajectory

The same attractor dynamics are applied to other controllers that are not related to the inverse kinematics, such as "closing the fingers to a power grasp", etc.

IV. Optimisation-based motion generation

To generate a joint-limit and collision free reaching motion, we apply the attractor-based optimisation scheme presented in [6]. It incorporates the robots redundant controller (eq. (A.2)) into the optimisation process, and finds a sequence of task space attractors describing the movement. The key idea is to optimise a scalar cost function by finding an optimal sequence of task space attractor vectors which determines the robots motion.

We consider an integral cost function over the movement in the general form of eq. (A.1). Here, $q \in \mathbb{R}^n$ is the joint state vector of the n-DoF robot, $x \in \mathbb{R}^d$ is the task state vector in the augmented task space, $x^*_{1:K} \in \mathbb{R}^d$ is a series of attractor points, and $r \in \mathbb{R}^d$ an additional smoothing variable (see [6] for details). The function g subsumes cost criteria in the q-space which depend on single time steps. It is suited to account for costs that depend on the posture of the robot. We formulate criteria to account for the offset of the final end-effector state to a target, collisions and proximities between collidable objects throughout the trajectory, and joint limit proximities. The function h subsumes costs for transitions in q-space and depends on the current and the previous time steps. It is suited to formulate criteria like the global length of the trajectory in q-space and for the end effector velocity at the end of the trajectory.

The movement generation process can be summarized by equations (A.2) - (A.5). Since the dependencies between attractor points and the task space trajectories are determined by the attractor dynamics (Eqs. (A.13)-(A.17)) and the dependencies between task and joint space is determined by the whole body motion control, we can derieve analytical gradients to relate the attractor point location to the chosen cost function:

$$\frac{dC}{dx^*} = \sum_{\text{children } y_i \text{ of } x^*} \frac{\partial y_i}{\partial x^*} \frac{dC}{dy_i} \,. \tag{1}$$

The partial derivatives are given in part d) of Table I. The gradient computation is carried out in a forward and a backward propagation step. In the *forward propagation step* we start with a given set of current attractor points $x^*_{1:K}$, then compute the task space trajectory $x_{0:T}$, then the $q_{0:T}$-trajectory, and finally the global cost C. In the *backward propagation step* we propagate the cost function gradients backward through the network using the chain rules. This involves first computing gradients dC/dq_t, then dC/dx_t, and finally $dC/dx^*_{1:K}$. Since all computations in the forward and backward propagation are local, the overall complexity is $O(T)$.

We use a gradient-based optimisation method (*RPROP* [16]). Our technique provides feasible solutions within the range of 0.5 to 2 seconds – which is below the critical "patience" threshold for the interaction with humans.

V. Task map integration

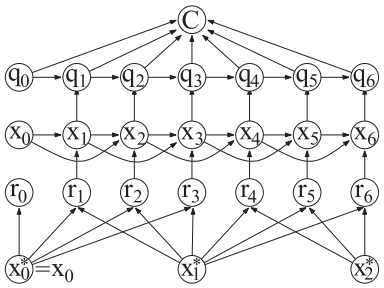
In the following, we extend this scheme to incorporate the learnt task maps. The key idea is to define a cost function and its gradient to account for the proximity to the nearest solution in the task map. It will replace the offset of the target end-effector state and contribute to the cost function g during motion optimisation. Given a current state x^{rel}_t in the task space (e.g., the current hand position and orientation relative to the object), we compute the nearest element $x_{\text{map}} \in T$ in the task map. Now we can define a cost

$$c_{\text{map}} = (x^{\text{rel}}_t - x_{\text{map}})^T W (x^{\text{rel}}_t - x_{\text{map}}) \tag{18}$$

The metric W accounts for the difference in the linear and angular units. The nearest neighbor in the task map to the hand is computed with the approximate nearest neighbor algorithm described in [17], similar to kd-trees. For this, the hand position and orientation x^{rel}_t is represented in the reference frame of the object. The gradient is

$$\frac{\partial c_{\text{map}}}{\partial x^{\text{rel}}_t} = 2(x^{\text{rel}}_t - x_{\text{map}})^T W \tag{19}$$

However, since we might want to choose another task description than relative coordinates when controlling the preshaping motion of the robot, the gradient of eq. (19) has to be projected onto the task description of the controller.



a) cost function:

$$C = \sum_{t=0}^{T} g(q_t) + \sum_{t=0}^{T-1} h(q_t, q_{t+1}) \ , \tag{A.1}$$

b) movement generation:

$$q_{t+1} = q_t + J_t^{\#}(x_{t+1} - \phi(q_t)) - \alpha \ (I - J_t^{\#} J_t) \ W^{-1} \ (\partial_q H_t)^T \tag{A.2}$$

$$x_{t+1} = x_t + \pi(x_t, x_{t-1}, r_{t+1}) \tag{A.3}$$

$$\pi(x_t, x_{t-1}, r_{t+1}) = a(r_{t+1} - x_t) + b(x_t - x_{t-1}) \tag{A.4}$$

$$r_t = (1-\tau)x_k^* + \tau x_{k+1}^* \ , \quad k = \lfloor tK/T \rfloor \ , \quad \tau = \frac{t - kT/K}{T/K} \tag{A.5}$$

c) chain rules following (1):

$$\frac{dC}{dq_t} = \frac{\partial C}{\partial q_t} + \frac{\partial q_{t+1}}{\partial q_t} \ \frac{dC}{dq_{t+1}} \tag{A.6}$$

$$\frac{dC}{dx_t} = \frac{\partial q_t}{\partial x_t} \ \frac{\partial C}{\partial q_t} + \frac{\partial x_{t+1}}{\partial x_t} \ \frac{dC}{dx_{t+1}} + \frac{\partial x_{t+2}}{\partial x_t} \ \frac{dC}{dx_{t+2}} \tag{A.7}$$

$$\frac{dC}{dr_t} = \frac{\partial x_t}{\partial r_t} \ \frac{dC}{dx_t} \tag{A.8}$$

$$\frac{dC}{dx_l^*} = \sum_t \frac{\partial r_t}{\partial x_l^*} \ \frac{dC}{dr_t} \tag{A.9}$$

d) partial derivatives:

$$\frac{\partial C}{\partial q_t} = g'(q_t) + h'^1(q_t, q_{t+1}) + h'^2(q_{t-1}, q_t) \tag{A.10}$$

$$\frac{\partial q_{t+1}}{\partial q_t} = I - J_t^{\#} J_t + (\partial_q J_t^{\#})(x_{t+1} - \phi(q_t)) - \alpha \ (I - J_t^{\#} J_t) \ W^{-1} \ (\partial_q^2 H_t)^T + \alpha \ \partial_q (J_t^{\#} J_t) \ W^{-1} \ (\partial_q H_t)^T \tag{A.11}$$

$$\frac{\partial q_t}{\partial x_t} = J_{t-1}^{\#} \tag{A.12}$$

$$\frac{\partial x_{t+1}}{\partial x_t} = 1 + \pi'^1(x_t, x_{t-1}, r_{t+1}) \tag{A.13}$$

$$\frac{\partial x_{t+2}}{\partial x_t} = \pi'^2(x_{t+1}, x_t, r_{t+2}) \tag{A.14}$$

$$\pi'^1(x_t, x_{t-1}, r_{t+1}) = -a + b \ , \quad \pi'^2(x_t, x_{t-1}, r_{t+1}) = -b \tag{A.15}$$

$$\frac{\partial x_t}{\partial r_t} = \pi'^3(x_{t-1}, x_{t-2}, r_t) \tag{A.16}$$

$$\frac{\partial r_t}{\partial x_l^*} = (1-\tau)\delta_{l=k} + \tau \delta_{l=k+1} \ , \quad \tau \text{ and } k \text{ depend on } t \text{ as above} \tag{A.17}$$

TABLE I

FUNCTIONAL NETWORK OF THE CONTROL ARCHITECTURE.

For this, we need to consider the differential kinematics of the task space control:

$$\frac{\partial c_{\text{map}}}{\partial x_t^{\text{ctrl}}} = \frac{\partial c_{\text{map}}}{\partial x_t^{\text{rel}}} \frac{\partial x_t^{\text{rel}}}{\partial q_t} \frac{\partial q_t}{\partial x_t^{\text{ctrl}}} \tag{20}$$

which corresponds to

$$\frac{\partial c_{\text{map}}}{\partial x_t^{\text{ctrl}}} = 2(x_t^{\text{rel}} - x_{\text{map}})^T W J_{\text{rel}} J_{\text{ctrl}}^{\#} \tag{21}$$

Since the task map comprises data vectors describing contiguous regions, gradient (21) will strongly force the pregrasp pose toward the valid region. However, there is only little influence of the gradient in the tangential directions of the task map parameter surface. These directions will be exploited by other criteria that are subject to the optimisation, so that the final state of the movement may travel along the parameter surface. We therefore don't have a fixed attractor, but rather an attractor hyper volume in which the optimal solution may be found. This is similar to the task relaxation method proposed in [12].

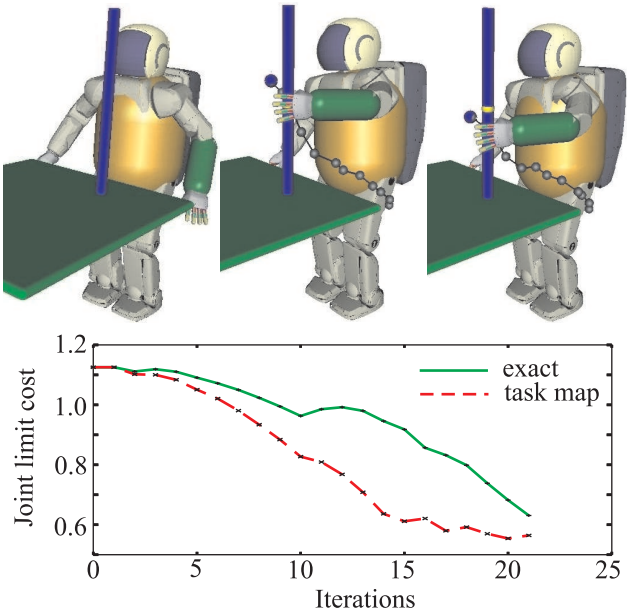


Figure 7: Grasping a long cylinder from a table

VI. SIMULATION STUDIES

We applied the proposed method to two simulated scenarios. Learning the task maps was done in dynamic multibody simulations with a reduced hand - table - object model.

We evaluated different simulation packages. It turned out to be difficult to resolve a stable contact simulation. The engines provided by Ageia (PhysX) and Cm-Labs (Vortex) produced stable and realistic results. In the following simulations, the PhysX engine was used. The simulation was set up with estimated material properties, and accounted for inertia and gravity effects. Contacts and friction are handeled by

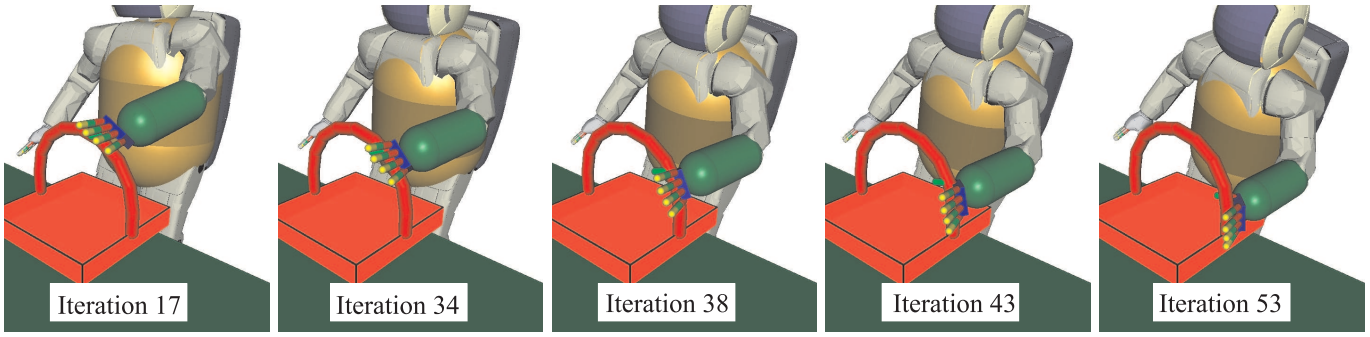


Figure 8: Pregrasp poses during an optimization run

the simulation engine. The palm is represented as a cuboid which is kinematically constrained to the transformation of the hand. The finger links are connected to the palm with revolute joints. The joints are modeled in series with spring-damper elements to account for the elasticity of the real robots hand mechanism.

In both simulation scenarios, the optimisation is initialized with 12 attractor vectors, which are linearly interpolated between initial hand pose and some point in the proximity of the object. The corresponding task elements that are subject to the optimisation comprise the hand position and the grasp axis attitude, represented in polar angles (2 dof). Further, we applied fixed equality constraints to the transformation of the feet and the lateral position of the center of gravity. In the figures, the position elements of the attractor vectors are indicated with small spheres.

Collisions between body and forearm of the robot, table and basket are considered. The collision model consists of simple primitive shapes (Line swept spheres for the robot segments and the baskets handle, rectangle swept spheres for the table and the lower part of the basket) and is also depicted in the images.

Grasping a long cylinder

In the first scenario, the task was to grasp a cylinder from a table. We performed two optimisation runs, one with the center of the cylinder as the ”exact” reaching target (indicated by a small sphere), and another one employing a task map. Both runs converged after 21 iterations. In each iteration, the movement of the robot is propagated from the initial state to the target, and the optimisation costs and gradients are computed based on the recursive scheme presented in Sections IV and V. The attractor vectors are then updated according to the chosen optimisation algorithm, and the next iteration is computed. The left image of Figure 7 shows the robots initial pose. The middle image depicts the pose found for the exact target, the right image the pose found by incorporating the task map. Both runs result in a smooth and collision-free movement around the table, finally reaching to the cylinder. As expected, the task map yields valid solutions along the cylinder axis, and the final grasp is located at a position lower to the center. The bottom part of Figure 7 shows the integral joint limit cost over the iterations of the simulation. The plot illustrates nicely that when employing the task map, the joint limit cost drops significantly faster and converges to a lower value.

Grasping a basket

In the second experiment, we incorporated the task map for the basket into the optimisation scheme. Three simulation runs with similar settings as in the previous example have been carried out. The initialization of the target attractor vector was in the proximity of the center of the handle. The basket has been put at three different locations with respect to the robot, see Figure 9. In Figure 8, we illustrate the progress of one optimisation run by showing intermediate results at certain iterations of the optimisation. The image sequence shows the pose at the end of the movement and illustrates how the pregrasp ”travels” along the geometry of the basket handle, and finally converges to the optimal pose. The image sequence in Figure 9 shows the different grasps that result from the optimisation. In the first image, the grasp targets to the lower left handle of the basket. Moving the basket to the left of the robot results in grasps at the upper left part of the handle, and finally to the center of the handle. This illustrates how the algorithm finds the most appropriate grasp considering the explored geometry of the basket handle.

VII. Conclusion

We presented a framework for reaching and grasping objects with a humanoid robot. The coupled problem of grasp choice and reaching motion optimisation is solved by incorporating object-specific *task maps* directly into the movement optimisation process.

These maps represent a functional characterization of objects in terms of their grasp affordances, i. e. a manifold of feasible pregrasp poses. To acquire such maps, we present an exploration scheme based on Rapidly exploring Random Trees, which efficiently finds contiguous regions of valid grasp manifolds.

While typical task goals define a single state in task space, a task map defines an attractor hyper volume in

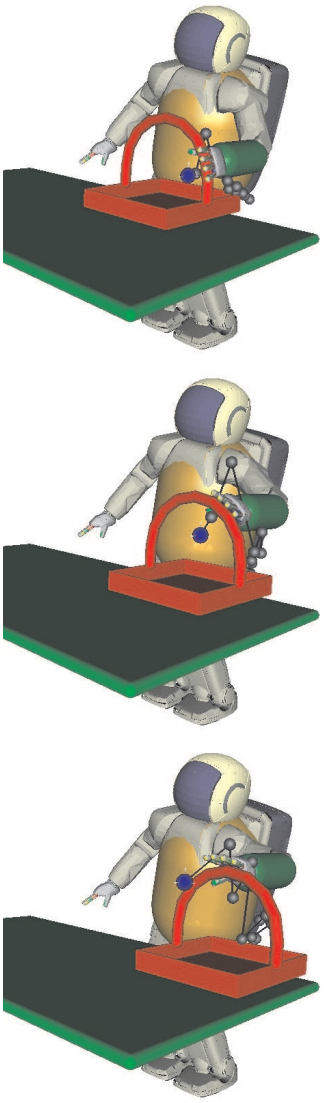


Figure 9: Grasping a basket from a table

which solutions may be found. In this way, the learnt task maps can be incorporated in our whole body motion control and optimisation framework. The overall scheme efficiently couples collision-free reaching with grasping, leading to a preference for grasps which are easy to reach.

A limitation of the currently presented scheme is that it will find only local optimal results that depend on the initialization of the optimisation. In particular the proposed exploration strategy will only find one contiguous cluster of feasible grasps. This can be circumvented by sampling multiple starting points for the exploration. Also, we have so far only investated single-handed grasps. However, the motion optimisation scheme generalizes to two-handed coordinated movement (including the control of relative task spaces, see [6]). When task maps are defined in a relative bimanual task space there is no limitation of our methods to be applied also to bimanual grasping tasks.

Future work will focus on applying the simulation results to real-world problems including vision and tactile feedback. Further, we plan to extend the framework to account for an optimal stance position with respect to the object. Since the framework allows to generate movements within interaction time, we will also focus on extending the scheme to dynamic scenes with moving objects and human interaction.

REFERENCES

[1] Raúl Suárez, Máximo Roa, and Jordi Cornellà, "Grasp quality measures," Tech. Rep. IOC-DT-P 2006-10, Universitat Politècnica de Catalunya, Institut d'Organització i Control de Sistemes Industrials, 2006.

[2] D. Schwammkrug, J. Walter, and H. Ritter, "Rapid learning of robot grasping positions," in *Proceedings of the International Symposion on Intelligent Robotics Systems (SIRS)*, 1999.

[3] R. Haschke, J.J. Steil, I. Steuwer, and H. Ritter, "Task-oriented quality measures for dextrous grasping," in *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, 6 2005, pp. 689 – 694.

[4] A.T. Miller, S. Knoop, H.I. Christensen, and P.K. Allen, "Automatic grasp planning using shape primitives," in *Proceedings of the IEEE International Conference of Robotics and Automation (ICRA)*, 2003, pp. 1824 – 1829.

[5] D. Berenson, R. Diankov, K. Nishiwaki, S. Kagami, and J. Kuffner, "Grasp planning in complex scenes," in *Proceedings of the IEEE-RAS/RSJ International Conference on Humanoid Robots*, 12 2007.

[6] M. Toussaint, M. Gienger, and Ch. Goerick, "Optimization of sequential attractor-based movement for compact movement representation," in *Proceedings of the IEEE-RAS/RSJ International Conference on Humanoid Robots*, 12 2007.

[7] M. Toussaint and Ch. Goerick, "Probabilistic inference for structured planning in robotics," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2007.

[8] Franziska Zacharias, Christoph Borst, and Gerd Hirzinger, "Capturing robot workspace structure: representing robot capabilities," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2007, pp. 3229–3236.

[9] J. Kuffner and S. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *Proceedings of the IEEE International Conference of Robotics and Automation (ICRA)*, 2000.

[10] M. Gienger, H. Janssen, and Ch. Goerick, "Task-oriented whole body motion for humanoid robots," in *Proceedings of the IEEE-RAS/RSJ International Conference on Humanoid Robots*, 12 2005.

[11] C. L. Colby, "Action-oriented spatial reference frames in cortex.," *Neuron*, vol. 20, no. 1, pp. 15–24, January 1998.

[12] M. Gienger, H. Janssen, and Ch. Goerick, "Exploiting task intervals for whole body robot control," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robot and Systems (IROS)*, 12 2006.

[13] A. Liegeois, "Automatic supervisory control of the configuration and behavior of multibody mechanisms," in *IEEE Transactions on Systems, Man, and Cybernetics*, 12 1977, vol. SMC-7 no. 12.

[14] Y. Nakamura, *AdvancedRobotics: Redundancy and Optimization*, Addison-Wesley Publishing Company, 1991.

[15] A. Ijspeert, J. Nakanishi, and S. Schaal, "Movement imitation with nonlinear dynamical systems in humanoid robots," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2002.

[16] M. Riedmiller and H. Braun, "Rprop- a fast adaptive learning algorithm," in *Proceedings of the International Symposium on Computer and Information Science VII (ISCIS)*, 1992.

[17] Sunil Arya, David M. Mount, Nathan S. Netanyahu, Ruth Silverman, and Angela Y. Wu, "An optimal algorithm for approximate nearest neighbor searching fixed dimensions," *Journal of the ACM*, vol. 45, no. 6, pp. 891–923, 1998.