2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

Integrated Grasp and Motion Planning

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

***Abstract*— In this work, we present an integrated planner for collision-free single and dual arm grasping motions. The proposed Grasp-RRT planner combines the three main tasks needed for grasping an object: finding a feasible grasp, solving the inverse kinematics and searching a collision-free trajectory that brings the hand to the grasping pose. Therefore, RRT-based algorithms are used to build a tree of reachable and collision-free configurations. During RRT-generation, potential grasping positions are generated and approach movements toward them are computed. The quality of reachable grasping poses is scored with an online grasp quality measurement module which is based on the computation of applied forces in order to diminish the net torque. We also present an extension to a dual arm planner which generates bimanual grasps together with corresponding dual arm grasping motions. The algorithms are evaluated with different setups in simulation and on the humanoid robot ARMAR-III.**

I. INTRODUCTION

Humanoid robots are designed to work in human-centered environments and to assist people in daily work. This means that robots must be able to operate autonomously in non-artificial surroundings in contrast to robots working in factories where the environment is structured to the needs of the robot. One essential ability for working autonomously is to grasp a completely known object for which an internal representation is stored in a database (e.g. information about shape, weight, associated actions or feasible grasps). Furthermore, the robot should be able to grasp objects for which the internal representation is incomplete due to inaccurate perception or uncertainties resulting in an incomplete knowledge base.

For grasping an object several tasks have to be solved in general, like searching a feasible grasping pose, solving the inverse kinematics (IK) or finding a collision-free grasping trajectory. With the algorithms proposed in this paper it is possible to solve all these problems with one probabilistic planning approach based on Rapidly Exploring Random Trees (RRT). The planner is searching a feasible and reachable grasp during the planning process and thus pre-calculated grasping positions are not needed. Searching a feasible grasping position online has the advantage that the search is not limited to a potentially incomplete set of offline generated grasps. Furthermore, the search for a feasible grasp is focused on reachable configurations and thus the computation of grasping poses is only performed for positions that can be reached by the robot.

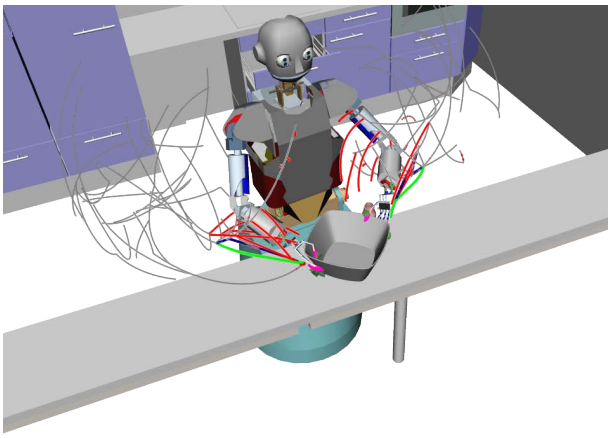

Fig. 1. A bimanual grasping trajectory.

The algorithms can be applied for single and dual arm planning problems and even when just a rough estimation of an unknown object is given, an approximated 3D model can be used to search grasping poses online.

In the next section, related work dealing with planning motions for grasping is presented. The three parts of the Grasp-RRT algorithm (computing grasping poses, generating approach movements and the online grasp quality measurement) are discussed in section III. In section IV the Bimanual Grasp-RRT algorithm, an approach for generating dual arm grasping motions, is presented. Several experiments for planning single arm and bimanual grasping motions in simulation and on the humanoid robot ARMAR-III are discussed in section V.

II. RELATED WORK

Planning collision-free motions for robots with a high number of degrees of freedom (DoF) is a known to be a P-Space hard problem in general [1]. This means that complete algorithms will suffer from low performance mainly caused by the complex task of building a representation of C_{free}, the part of the configuration space (C-Space) whose configurations do not cause work space collisions. Instead of building up a representation of C_{free}, probabilistic algorithms may be used to implicitly cover the free space and thus a time consuming computation of C_{free} can be avoided. RRT-based approaches are widely used in the context of planning grasping and reaching motions for humanoid robots. The general theory for planning collision-free motions with RRT-methods can be found in [2] or [3].

Planning grasping motions with pre-defined sets of grasping poses is discussed in [4], [5], [6], [7]. These approaches use offline calculated grasping poses for which the IK-solutions are searched during the planning process. The grasping poses can be calculated automatically in an offline step [8], [9] and the grasping information is stored in a database for use during the online search. [10] presents algorithms to automatically build a database of stable grasps for numerous objects and their application resulting in *The Columbia Grasp Database*. Multi-grasp manipulations are discussed in [11].

Planning dual arm motions is addressed in [7] where collision-free motions for two end effectors are planned with RRT-based algorithms for bimanual grasping or re-grasping actions.

In the work presented in [12], object specific task maps are used to simultaneously plan collision-free reaching and grasping motions. The proposed motion optimization scheme uses analytic gradients to jointly optimize the motion costs and the choice of the grasp on the manifold of valid grasps.

Evaluation of an object grasp by a multi-fingered robot hand has been a major topic in robotics for years. A common approach is based on the computation of the wrench space formed by the contact points between hand and object, also called Grasp Wrench Space (GWS). Based on the GWS, a score is introduced in [13] which approximates the GWS by a convex hull and tries to fit in the largest wrench space sphere. [14] proposes the concept of the Object Wrench Space (OWS) which represents the optimal grasp in wrench space by applying forces on numerous points distributed along the objects surface. The OWS is scaled to fit within the GWS leading to a score in the form of the scaling factor. In [15], which proposes a task-dependent wrench space, the complexity of calculating the OWS is reduced by approximating it by an ellipsoid.

III. Integrated Grasp and Motion Planning

In this section the Grasp-RRT planner and the required components, like the definition of an end effector, the generation of approach movements and the algorithms for measuring the grasp quality, are presented.

A. Grasp-RRT: The Concept

The proposed Grasp-RRT planner combines the search for a collision-free motion with the online search for a feasible grasp. Thus there is no explicit definition of a target configuration, since the target is derived from a feasible grasp which is calculated during the planning process (see Fig. 2). In Alg. 1 the main planning loop is presented. The planner is initialized with the root configuration q_{start} and p_{obj}, the 6D pose of the object that should be grasped. Starting from q_{start} RRT-based extension methods are used to build up a tree of collision-free and reachable configurations. For every new configuration q_i, that is created to extend the tree, the corresponding workspace position p_i of the grasp center point is calculated and stored together with the configuration. Later, these workspace positions are used

Algorithm 1: $GraspRRT(q_{start}, p_{obj})$

```
1 RRT.AddConfiguration(q_start);
2 while (!TimeOut()) do
3     ExtendRandomly(RRT);
4     if (rand() < p_SearchGraspPose) then
5         n_grasp ← ApproachTrajectory(RRT, p_obj);
6         if (ScoreGrasp(n_grasp) > score_min) then
7             return BuildSolution(Grasp);
8     end
9 end
```

to choose a candidate for testing a grasping pose. From time to time a node of the RRT is selected and via the pseudoinverse Jacobian $J^+(q)$ the TCP is moved toward a feasible grasping pose in the *ApproachTrajectory* method (see Alg. 2). The Jacobian matrix $J(q)$ for the participating joints is built in every loop and $J^+(q)$ is derived via single value decomposition.

When Alg. 2 succeeds, the resulting RRT-node defines a potential grasping pose which is scored by the grasp quality measurement module. In case the quality score lies above a threshold, the final grasping trajectory can be built easily since the approach trajectory already defines a collision-free connection to the RRT. Furthermore, no explicit IK-solution has to be computed for the grasping pose, since through the pseudoinverse Jacobian-based movements, the IK-problem is implicitly solved. In order to produce appealing solution trajectories, the result is finally smoothed with path pruning techniques.

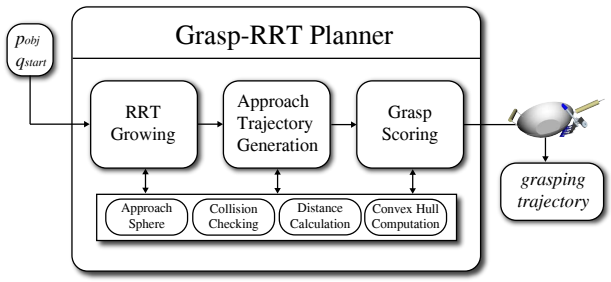

Fig. 2. Overview of the Grasp-RRT planner.

B. End Effector

The proposed planning approach uses a virtual representation of the hand including a grasping point and an approach direction. Based on the work of [16], the grasp center point (GCP) and the approach direction are defined for the hand that should be used for grasping. The definition of the GCP and the approach direction of the anthropomorphic hand that is used in our experiments can be seen in Fig. 3.

C. Online Computation of Potential Grasping Poses

At the beginning of Alg. 2 a node $n_{Approach}$ of the RRT, and thus an associated C-Space configuration $n.q_{Approach}$ together with the workspace pose of the GCP $n.p_{Approach}$, is selected and used for calculating p_{grasp}, a 6D grasping

Algorithm 2: $ApproachTrajectory(RRT, p_{obj})$

$n_{Approach} \leftarrow SelectGraspExtensionNode(RRT)$;
$p_{grasp} \leftarrow ComputeGraspingPose(n_{Approach}, p_{obj})$;
$n \leftarrow n_{Approach}$;
repeat
  $\Delta_p \leftarrow p_{grasp} \cdot (n.p)^{-1}$;
  $\Delta_q \leftarrow J^{+}(n.q) * LimitCartesianStepSize(\Delta_p)$;
  $n'.q \leftarrow n.q + \Delta_q$;
  if *(Collision($n'.q$) || !InJointLimits($n'.q$))* **then**
    if *(NumberOfContacts(CloseHand(n)) ≥ 2)* **then**
      return n;
    else
      return $NULL$;
  end
  $n'.p \leftarrow ForwardKinematics(n'.q)$;
  $RRT.AddNode(n')$;
  $n \leftarrow n'$;
until $(Length(\Delta_p) > Threshold_{Cartesean})$;
return n;

pose. The loop of Alg. 2 moves the TCP toward p_{grasp} and if no self-collisions, no collisions with obstacles and no violations of joint limits occur during the movements, the target grasping pose is returned. In case a collision or a violation of joint limits is noticed during the approach movement, the last valid configuration n is used to check the number of contact points when closing the hand. If n results in more than one contact point between the hand and the grasping object, the RRT-node is returned as a potential grasping pose.

The target grasping pose p_{grasp} is determined by searching the point pt_{obj} on the object's surface which has the shortest distance to the GCP. pt_{obj} defines the translational part of p_{grasp} and the rotational component is derived by rotating the coordinate system of the GCP by α, so that the approach direction points toward pt_{obj} (see Fig. 3).

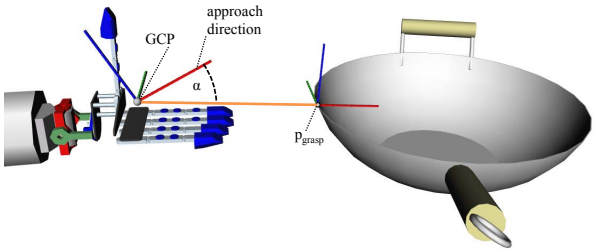

Fig. 3. The computation of the grasping pose p_{grasp}.

D. Representing Approach Directions

The approach direction toward an object is essential for finding a feasible grasp, since in general a stable grasp may only be found for a small amount of all possible approach directions. In our case, where a RRT-node $n_{Approach}$ has to be selected as a starting point for generating an approach movement, a random node selection does not respect this fact since the distribution of configurations of the RRT is independent from the 3D relation between TCP and object. In contrast, if the distribution of the node selection uniformly covers the approach directions, the search for good scored grasps benefits from varying relations between object and TCP.

In order to encode different approach directions an *ApproachSphere*, an approximated sphere located at the object's 3D position, is used. Whenever a new RRT-node n_{new} is added during the planning loop, the corresponding triangle t_n of the *ApproachSphere* is determined by projecting the TCP position onto the sphere (see Fig. 4(a)). Then n_{new} is added to a list of associated RRT-nodes of t_n.

When a random RRT-node $n_{Approach}$ for grasp testing is selected, at first one of the available approach directions is randomly chosen and then one of the associated nodes is randomly selected. Hence the distribution of the selection of grasp testing nodes uniformly covers the possible approach directions (within the limits resulting from the approximation of the sphere). The advantage of selecting extension nodes this way can be seen in Fig. 4(b). Here the state of the *ApproachSphere* after building up a RRT is shown. The color intensity of a triangle is proportional to the number of RRT-nodes in direction of the triangle. It can be seen clearly that a random selection of $n_{Approach}$ out of all RRT-nodes will result in a non-uniform distribution of approach directions.

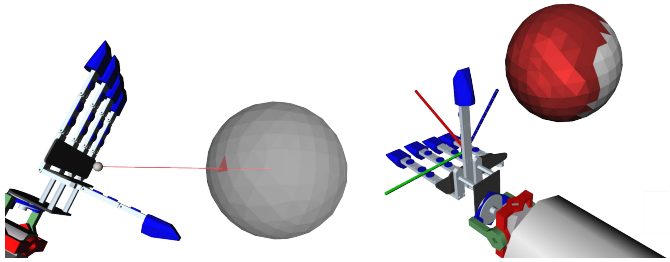

Fig. 4. (a) For each RRT-Node the corresponding triangle of the *ApproachSphere* is determined by projection the TCP position on the sphere's surface. (b) The distribution of approach directions is visualized by setting the color intensity proportional to the number of RRT-nodes in the direction of the triangle.

E. Scoring a Grasp

The quality of a grasp is an important aspect for the selection of the best grasp candidate from the set of grasps resulting from grasp planning. A common approach to evaluate the quality of grasps is the construction of the grasp wrench space (GWS), which describes the set of all wrenches that can be applied on the grasp contact points. A single wrench is defined as the concatenation of the force and the torque vector exerted on a grasp contact point. However, the calculation of the wrench space and even its approximation e.g. by a convex hull is highly complex and time consuming in the context of online planning. Hence, inspired by the works presented in [14], we implemented a grasp quality measure based on forces, which are adapted to the torques exerted on the object.

Analogue to the determination of the object wrench space (OWS), the surface of an object is sampled once to generate

a set of m possible contact points C_o. Initially, unit forces are applied on these points. The direction of a contact force f at each contact point is constrained by a friction cone. To reduce the complexity, a friction cone is approximated by friction pyramid with k sides. Therefore, following equation holds for f:

$$f = \sum_{j=1}^{k} \alpha_j f_j \,, \tag{1}$$

whereas f_j denotes a force on the boundary of the friction pyramid. Furthermore, for all contact forces applied on the object the following condition is imposed:

$$\sum_{i=1}^{m} f_i = f_c. \tag{2}$$

Each applied force leads to a torque vector, which magnitude and direction depends on the geometry of the object and the length of the force vector. A stable grasp is given if the sum of all torques, the net torque, on the contact points is zero, i.e. the exerted forces immobilize the object in the hand. For this purpose, the magnitude of f_i is scaled by a factor b_i, which can be formulated as an optimization problem:

$$min(\sum_{i=1}^{m}(c_i - p_{com}) \times b_i f_i)^2 \,, \tag{3}$$

where c_i denotes the i-th contact point and p_{com} the object center of mass. Using steepest descent method, a solution for the force magnitude scaling is found subject to Eq. 2. Since the steepest descent method tends to get stuck in local minima, an initial solution b_{init} close to the desired one is generated by separating the set of contact points C_o by a plane, which goes through p_{com} and leads to two point sets C_1 and C_2, which maximize the distance between both net torques τ_1 and τ_2. Force magnitudes of the point set with the smaller net torque are gradually increased, while force magnitudes of the other set are decreased until the distance $\parallel \tau_1 - \tau_2 \parallel < \epsilon$. Like in [13], [14], [15], a convex hull is used to approximate the space of forces applied on the object. Hence, for C_o, the convex hull CH_o is obtained. A depiction of CH_o is shown in Fig. 5.

Regarding a multi-fingered hand grasping an object, the contact point set C_g consists of n points. After adjusting the force magnitudes (see Eq. 3), the grasp is represented by the convex hull CH_g as depicted in Fig. 5. The quality of a grasp $q_g \in [0, 1]$ is determined by the factor, which scales CH_o to optimally fit in CH_g as described in [14]. Unlike grasp quality measures in wrench space, the method described above is computationally efficient, since the force space can be easily approximated by a convex hull consisting of only a few facets.

IV. DUAL ARM GRASP PLANNING

When large objects like the wok in Fig. 8 should be grasped by a humanoid robot, both hands are needed for applying a stable grasp. On basis of the Grasp-RRT planner, introduced in the last section, we propose the Bimanual-Grasp-RRT planner which combines the search for a bimanual feasible grasp with the search for a collision-free grasping motion for both arms.

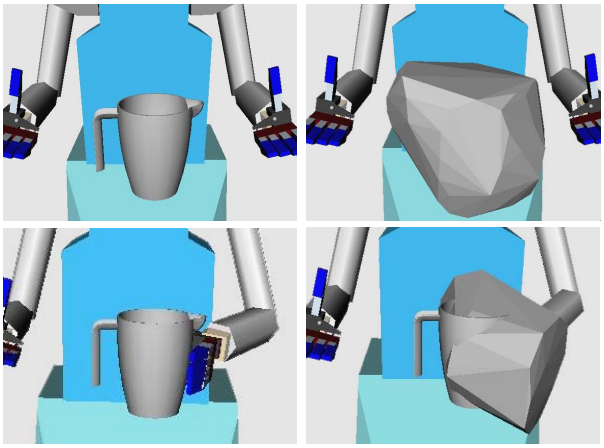

Fig. 5. Top Row: The object with a visualization of CH_o. Bottom Row: The grasp specific CH_g is used to compute the grasp quality score. For the measuring cup a grasp quality score of $q_g = 0.46$ is determined.

A. Bimanual Grasp-RRT

Fig. 6 depicts an overview of the Bimanual Grasp-RRT planner. The planner instantiates two Grasp-RRT planners, one for each hand. These instances are started in parallel, so that the search for feasible grasps is done simultaneously for the left and the right hand. Furthermore they are configured to search and store grasps until the main thread terminates. The main thread collects the grasps and the corresponding grasping trajectories for the left and the right hand and tries to find a feasible bimanual solution by calculating quality scores of the bimanual grasping combinations. Every time a planner for one hand reports that a new grasping trajectory was found, all possible bimanual combinations of this grasp together with the already stored grasps of the other hand are built and scored as described in section IV-B. If the resulting bimanual score is above a certain threshold the self-collision status of the two pruned grasping trajectories is checked. If no collision was determined the combined solution for both arms together with the resulting grasping information is returned (see Alg. 3).

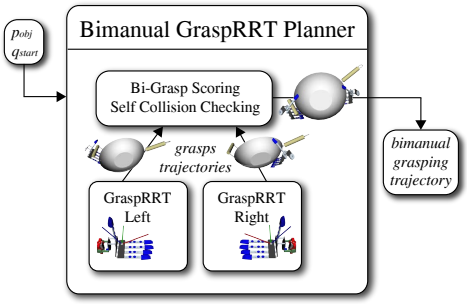

Fig. 6. Overview of the Bimanual Grasp-RRT Planner.

B. Scoring Bimanual Grasps

The grasp score presented in this work can be easily applied on bimanual grasping. Considering a robot with two hands, one obtains the two contact point sets C_g^l and C_g^r, for the left and the right hand. The united set $C_g' = C_g^l \cup C_g^r$ is used to adjust the contact forces and to build the convex hull CH_g' analogously to the single-handed case. The increase of

Algorithm 3: $BimanualGraspRRT(q^{left}_{start}, q^{right}_{start}, p_{obj})$

```
1  GraspRRT_left ← GraspRRTInstance(q^left_start, p_obj);
2  GraspRRT_right ← GraspRRTInstance(q^right_start, p_obj);
3  GraspRRT_left.start();
4  GraspRRT_right.start();
5  while (!TimeOut()) do
6      /* process new results of GraspRRT_left */
7      s_l ← GraspRRT_left.GetNewSolution();
8      if (s_l) then
9          Results_left.add(s_l);
10         foreach (s_r ∈ Results_right) do
11             if (BiGraspScore(s_l, s_r) > score_min &&
               !SelfCollision(s_l, s_r)) then
12                 GraspRRT_left.stop();
13                 GraspRRT_right.stop();
14                 return BuildSolution(s_l, s_r);
15             end
16         end
17     end
18     /* process new results of GraspRRT_right */
19     ...
20 end
```

the number of contact points leads to wider force space which results in an higher grasp score, whereas the position of the contact points, respectively the pose of the hands, plays a more crucial role.

V. EXPERIMENTS

A. A Measuring Cup in a Drawer

In this experiment the humanoid robot ARMAR-III should grasp a measuring cup located in a drawer of a kitchen. The robot should use three hip and seven arm joints and thus the C-Space used for planning is 10-dimensional. The setup depicted in Fig. 7 limits the possibility of applying a feasible grasp in a collision-free way, since the measuring cup is located near the side walls of the drawer. Nevertheless, the Grasp-RRT algorithm is able to find a suitable grasping pose together with a collision-free trajectory in 3.7 seconds on average (measured over 30 test runs).

B. A Wok in the Kitchen: Evaluating the Bimanual Grasp-RRT Planner

In this simulation experiment, the Bimanual Grasp-RRT planner is queried to find a grasping trajectory for a wok located at the sideboard of the kitchen. The use of both arms of ARMAR-III results in a 14 DoF planning problem which is solved in 1.7 seconds on average. Due to the parallelized search for a left and a right trajectory, the planner performs well in this experiment (see table I). A resulting grasping configuration together with the collision-free trajectories for the left and the right arm are shown in Fig. 8.

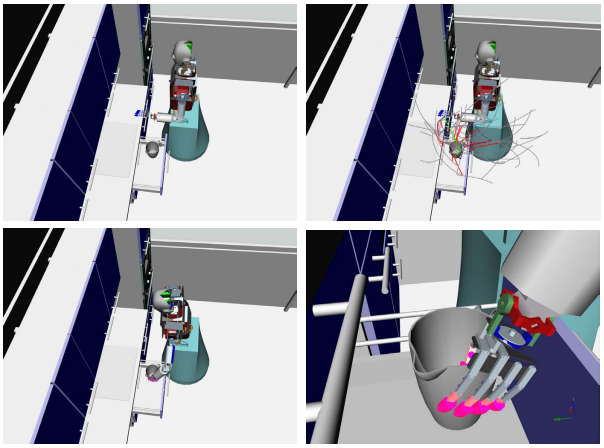

Fig. 7. The Grasp-RRT planner is used to search a feasible grasp together with a collision-free grasping trajectory for 10 DoF of ARMAR-III.

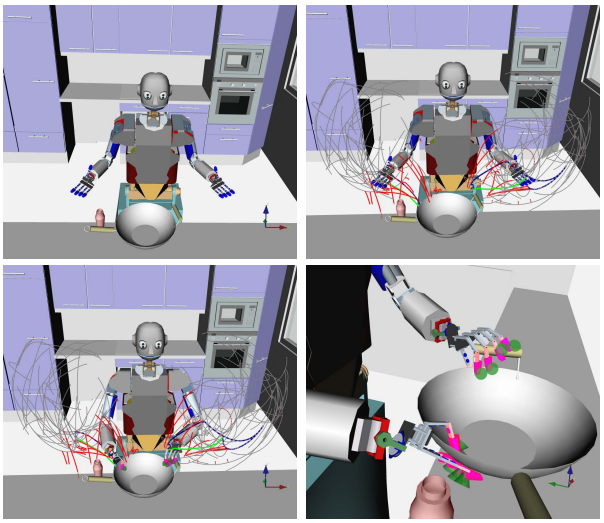

Fig. 8. The Bimanual Grasp-RRT planner is used to search a collision-free grasping trajectory for 14 DoF of both arms of ARMAR-III.

C. Experiment on the Humanoid Robot ARMAR-III

This experiment is performed online on the humanoid robot ARMAR-III. The Bimanual Grasp-RRT is used to search a collision free trajectory for grasping a bowl on the sideboard with both hands. The ketchup bottle, located near the target object, is limiting the number of feasible grasps for the left hand. Fig. 9 shows the results of the planner and the execution of the planned trajectories on the humanoid robot ARMAR-III.

D. Results

The performance of the proposed Grasp-RRT planner in single and dual arm planning setups is presented in Fig. 10 and table I. The runtime analysis has been carried out on an Intel DualCore CPU with 2.0 GHz by averaging 30 test runs. The time spent for the three main parts of the algorithm are distinguished, pointing out that the parameter setup was well balanced since approximatively the same amount of time is spent for building up the RRT, computing the approach directions and for scoring the grasping poses.

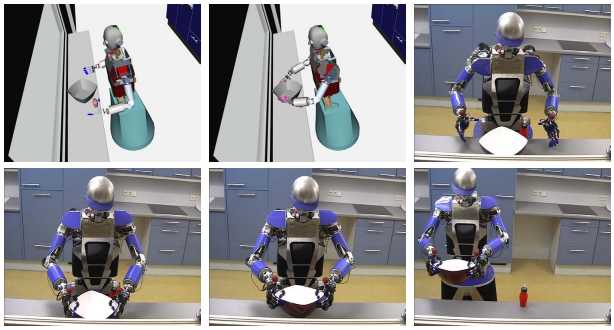

Fig. 9. The Bimanual Grasp-RRT enables the humanoid robot ARMAR-III to grasp a bowl in the kitchen.

The last two columns of table I show the number of approach trajectories which have been generated and the number of grasp measurements which were calculated during the planning process. These values differ, since not all approach trajectories result in a suitable grasping configuration.

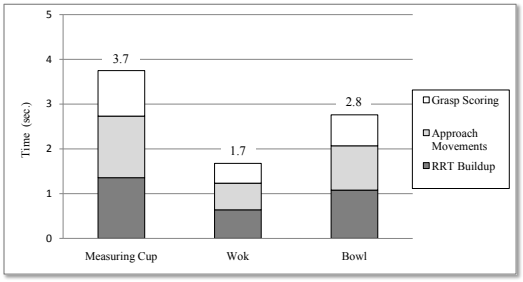

Fig. 10. Overview of the average performance measurements.

TABLE I

PERFORMANCE EVALUATION.

	Planning Time (seconds)				# Appr. Traj.	# Gr. Scores
	Total	RRT	Approach	Score		
Measuring Cup	3.7	1.3	1.4	1.0	26.8	18.9
Wok	1.7	0.6	0.6	0.4	21.3	9.8
Bowl	2.8	1.1	1.0	0.7	35.0	16.5

VI. CONCLUSIONS AND FUTURE WORKS

In this work, a planning approach for computing grasping trajectories was presented. Compared to existing state-of-the art planners, the proposed Grasp-RRT planner does not rely on any precomputed grasping positions, since suitable grasping poses are determined during the planning process. The algorithm integrates the search for solutions of the three main tasks needed for grasping an object: Finding a feasible grasp, solving the inverse kinematics and computing a collision-free trajectory. As shown in the experiments in section V, the setup of the planner is well balanced, since on average for each task (building the RRT, computing the approach trajectories and determining the grasp quality measures) approximately the same part of the planning time is spent.

Further improvements may be achieved by adding constraints to the grasp quality scoring algorithms, e.g. if a post-grasping action implies such constraints. Furthermore, a local optimization of the calculated grasping trajectory could be applied to locally maximize the pose for grasping. In case of grasping non-convex objects, better results could be achieved by a hierarchical decomposition in multiple superquadrics, which can be used to generate a more comprehensive set of approach directions as introduced in [17].

VII. ACKNOWLEDGMENTS

The work described in this paper was partially conducted within the German Humanoid Research project SFB588 funded by the German Research Foundation (DFG: Deutsche Forschungsgemeinschaft) and the EU Cognitive Systems project GRASP (IST-FP7-IP-215821) funded by the European Commission.

REFERENCES

[1] J. H. Reif, "Complexity of the mover's problem and generalizations," in *SFCS '79: Proceedings of the 20th Annual Symposium on Foundations of Computer Science (sfcs 1979)*. Washington, DC, USA: IEEE Computer Society, 1979, pp. 421–427.

[2] J. Kuffner and S. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *IEEE Int'l Conf. on Robotics and Automation (ICRA'2000), San Francisco, CA*, 2000, pp. 995–1001.

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

[4] E. Drumwright and V. Ng-Thow-Hing, "Toward interactive reaching in static environments for humanoid robots," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, Oct. 2006, pp. 846–851.

[5] D. Berenson, S. Srinivasa, D. Ferguson, A. Collet, and J. Kuffner, "Manipulation planning with workspace goal regions," in *IEEE Int'l Conf. on Robotics and Automation (ICRA'2009),Kobe,Japan*, 2009.

[6] M. V. Weghe, D. Ferguson, and S. Srinivasa, "Randomized path planning for redundant manipulators without inverse kinematics," in *IEEE-RAS International Conference on Humanoid Robots*, November 2007.

[7] N. Vahrenkamp, D. Berenson, T. Asfour, J. Kuffner, and R. Dillmann, "Humanoid motion planning for dual-arm manipulation and re-grasping tasks," in *Intelligent Robots and Systems, IROS*, October 2009.

[8] A. T. Miller, "Graspit!: a versatile simulator for robotic grasping." Ph.D. dissertation, Department of Computer Science, Columbia University, 2001.

[9] D. Berenson and S. Srinivasa, "Grasp synthesis in cluttered environments for dexterous hands," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids08)*, 2008.

[10] C. Goldfeder, M. Ciocarlie, H. Dang, and P. K. Allen, "The columbia grasp database," in *2009 IEEE International Conference on Robotics and Automation*, May 2009.

[11] D. Williams and O. Khatib, "Experiments in multi-grasp manipulation," in *ISER*, 1993, pp. 14–28.

[12] M. Gienger, M. Toussaint, and C. Goerick, "Task maps in humanoid robot manipulation," in *IEEE International Conference on Intelligent Robot and Systems (IROS 2008)*. IEEE, 2008.

[13] D. Kirkpatrick, B. Mishra, and C. Yap, "Quantitative steinitz's theorems with applications to multifingered grasping," in *Proc.of the 20th ACM Symp. on Theory of Computing*, 1990, pp. 341–351.

[14] N. Pollard, "Parallel methods for synthesizing whole-hand grasps from generalized prototypes," *Technical Report AI-TR 1464, MIT, Artificial Intelligence Laboratory*, 1994.

[15] C. Borst, M. Fischer, and G. Hirzinger, "Grasp planning: How to choose a suitable task wrench space," in *IEEE International Conference on Robotics and Automation*, 2004, pp. 319–325.

[16] T. Asfour, P. Azad, N. Vahrenkamp, K. Regenstein, A. Bierbaum, K. Welke, J. Schroeder, and R. Dillmann, "Toward humanoid manipulation in human-centred environments," *Robotics and Autonomous Systems*, vol. 56, no. 1, pp. 54 – 65, 2008.

[17] C. Goldfeder, P. K. Allen, C. Lackner, and R. Pelossof, "Grasp planning via decomposition trees," in *ICRA*, 2007, pp. 4679–4684.