Project Acronym:	GRASP
Project Type:	IP
Project Title:	Emergence of Cognitive Grasping through Introspection, Emulation and Surprise
Contract Number:	215821
Starting Date:	01-03-2008
Ending Date:	28-02-2012

Deliverable Number:	D17
Deliverable Title :	Requirements on the cognitive architecture for grasping and dexterous manipulation
Type (Internal, Restricted, Public):	PU
Authors	T. Asfour, D. Kragic, A. Morales, V. Kyrki, D. Burschka, A. Argyros, H. Deubel, M. Vincze and R. Dillmann
Contributing Partners	All

Contractual Date of Delivery to the EC:	28-02-2010
Actual Date of Delivery to the EC:	16-03-2010

Contents

Chapter 1

Executive Summary

This Deliverable encapsulates several contributions to the implementation of a cognitive architecture for a robot system with cognitive grasping capabilities. This architecture should facilitates the implementation, demonstration and evaluation of the application scenario on different platforms available in the project.

Starting from the components and tools developed in the first year of the project (see Deliverables D8) and the core architecture of the simulator (see Deliverable D9), we continued our work on the conceptual design as well as the implementation of this architecture.

According to the Technical Annex, Deliverable D17 presents the activities performed in the second year of the project in the context of the tasks 7.1, 7.2, 7.3, 7.4 and 7.5. The objectives of these tasks are:

Task 7.1 Development and implementation of a cognitive control architecture for grasping and dexterous manipulation necessary to bootstrap the system integration and the evaluation of methods and algorithms in different scenarios on different robot platforms as it is the long-term goal of this work package, in particular of task 7.5.

Task 7.2 Software infrastructure that allows for smooth and efficient exchange of modules in the project in order to minimize the overhead in the overall system integration.

Task 7.3 Specification of interfaces for knowledge and control flows between perception, action, learning and reasoning modules.

Task 7.4 Definition of the first year scenario, which is necessary for evaluating the developed algorithms in the first year of the project.

Task 7.5 Evaluation of the developed methods and algorithms in different scenarios on different robot platforms.

Task 7.6 Benchmarking environment for grasping and dexterous manipulation including software, robot control frameworks, data sets of objects, robot and human hand models as well as grasp-related algorithms.

The work in this deliverable is related to **Milestone 5** "Implementation of high-level controllers including a global uncertainty model, integration and evaluation in the simulator and on experimental platforms, grounding grasping primitives" as well as **Milestone 6** "Integration and evaluation of human body and hand tracking on active robot heads, demonstration of a grasping cycle on the experimental platforms".

The document is organized as follows. In Chapter 2, we present briefly general requirements on the cognitive architecture. In Chapter 3, a summary of performed integration activities in the project is given. These include 1) sensor-based scene exploration and grasping 2) human grasping activities 3) integration of the simulator in WP6 and the control architecture in WP3 and 4) integrated grasp and motion planning. Chapter 4 concludes the Deliverable.

Chapter 2

Requirements on the Architecture

2.1 GRASP: the main idea

GRASP develops a framework for reasoning about the world state, causes, and effects by using the simulator as a memory and reasoning agent. In contrast to traditional uses of a simulator, where the simulator is assumed to model perfect knowledge of the world, in GRASP the simulator models the belief of the robotic agent about its best guess of the world. Thus, world knowledge is acquired using imprecise sensors such as vision, and continuously refined during interaction with the world. Update of the world knowledge can be triggered on several levels from simple adaptation and estimation to high-level conceptual surprise. The simulator thus partially solves the problem how to store and represent world knowledge. This encapsulates the "introspection" and "surprise" aspects of GRASP.

The relation of GRASP to developmental robotics and classical execution of pre-programmed skills is also important. GRASP combines high-level concepts with a learned mapping to embodiment specific capabilities. For example, the task-relevant qualities may be defined by a human, however the anchoring is a result of learning. Thus, GRASP addresses the major issue how to combine high-level symbolic representations with low-level sensor processing. The use of machine learning techniques is not unique *per se*, however the use of learning to explicitly link sensors (visual, haptic) to high-level concepts provides a basis for symbolic reasoning.

GRASP rests upon the Predict-Act-Perceive paradigm where the knowledge of grasping in humans can provide the initial model of the grasping process which then has to be grounded through introspection to the specific embodiment. Some of the ideas originate from findings in human brain research where the self-knowledge is retrieved through different emulation principles.

In the Predict-Act-Perceive paradigm, two loops run in parallel (not necessarily synchronously): One in the real world and one in simulation (see Fig. 2.1). The current state of the world is first observed with different real sensors (*real perception*), and used to build the world model in the simulator. In the simulator (*internal world of the robot*), there are two different modes of prediction:

- Predict the values of different sensors, given an exploratory action and potential assumptions. E.g., the backside of objects is unobserved; an assumption is that objects are symmetric; the backside can be explored with haptic sensors; given the assumed symmetry, the haptic sensor values can be predicted. Note that the goal of this mode is to explore the world, to extend or verify the state estimate.
- Predict the outcome of a specific action applied to the world. When interacting with objects, dynamic simulation can provide the estimation of the next state. Given measurements of tactile sensors after a grasp has been applied to an object, predict the outcome of a lifting action. When interacting with objects, dynamic simulation can provide the estimation of the next state that can ease, for example, tracking of objects.

Once the prediction has been made by the simulator, the selected action can be applied in reality and the real sensor values can be compared with the predicted ones. A mismatch will in our case be treated as a surprise that may, for example, trigger an exploratory action.

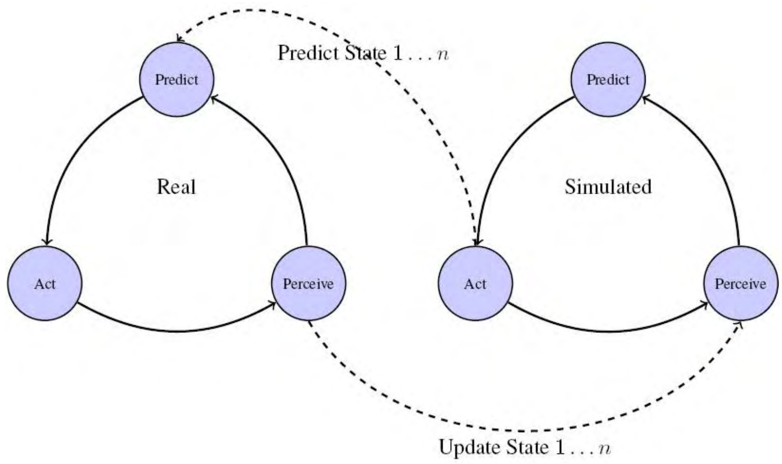


Figure 2.1: Predict-Act-Perceive paradigm.

Thus, the paradigm encompasses on the one hand the evaluation of a currently executed action based on its predicted outcome in simulation, while on the other hand the simulation state is grounded and refined based on the experience in the real world, which consequently leads to an improvement in the prediction and parameterization of future actions.

2.2 Requirements on Grasping Architecture

Research on cognitive architectures plays a central role in the development of artificial cognitive systems as the architecture represents the underlying infrastructure that allows the integration of perception, action, adaptation, reasoning and communication components, which should provide a concrete framework for modeling of cognitive aspects through specifying essential structures, division of modules, relation between modules, and a variety of other aspects. Several cognitive architectures have been proposed in the literature. For a review on several known cognitive architectures, the reader is referred to [VMS07, LLR09] and [Sun04]. Vernon et al. present in [VMS07] a broad survey on the various paradigms of cognition and review several cognitive architectures. They summarize from a developmental approach point of view the key architectural features that systems capable of autonomous development of mental capabilities should exhibit. In [LLR09], important aspects of cognitive architecture related to representation, organization, performance, and learning, as well as evaluating such architectures at the system level are discussed. Sun discusses in [Sun04] behavioral characteristics commonly exhibited in human everyday activities, which one should attempt to capture in cognitive architectures.

Despite of conceptual advances that have been achieved in the past and the practical use of some of the proposed architectures, there remains a considerable amount of unsolved issues, which have to be addressed. Most architectures emphasize a specific domain, where they rarely are confronted with the interactions between a physical embodiment and the environment. Cognitive architectures for human-inspired grasping should take into consideration the following aspects:

- **Learning** of grasping skills and task knowledge from human observation as well as the adaption of such skills to new situations and contexts.
- **Action representations**, which allow the adaption of grasping skills to different situations and contexts as well as the mapping of human grasping activities to robots with different embodiments.
- **Scene representations**, which integrate different sensorial entities and exploration strategies.
- The use of the **simulator** as memory and reasoning engine, which models the belief of the world state for the system, and allows to predict the outcomes of performed actions.
- The connection between high-level symbolic and low-level sensorimotor space to facilitate **planning** of complex tasks based on learned action primitives and explored scenes as well as reasoning about the world.

Chapter 3

Integration Activities

3.1 Sensor Based Scene Exploration and Grasping

The central part of GRASP regarding scene modeling for exploration and context understanding is studied in WP5 and it is closely related to work in WP1 (modeling and tracking human activity) and WP2 (studying sensory-motor representations for object and action modeling). The main novelty in WP5 is to map human attention strategies to artificial systems adapting the view-planning and detection capabilities of the human eye to the capabilities of a given sensor and actuator configuration of a technical vision system. Thus, WP5 makes use of investigation of human strategies of monitoring a specific task or interacting with a given contextual situation to find best sensor data processing and next best view strategies in the context of a manipulation task.

In GRASP, we employ two strategies in this context:

- Active scene exploration: done by robot either in real world or in simulation.
- Spatio-temporal scene understanding from human observation.

3.1.1 Scene Exploration

The knowledge representation in our system consists of geometric properties of the object (shape, texture), its physical properties (density, center of mass, stiffness, grasp points), and action attributes that describe the actions observed during handling of the object and typical locations where such objects can be found or deposited. We extended the mismatch-based surprise detection from Year 1 to actions where the initial mismatch-trigger triggers further verification based on known actions in the scene (Fig. 3.1).

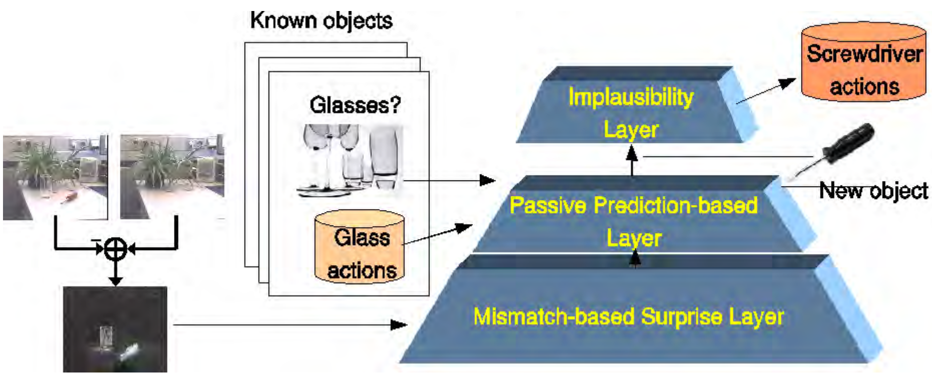


Figure 3.1: Surprise hierarchy proposed in the Technical Annex

In addition to the standard exploration task that reconstructs the 3D information about the environment, the system needs to distinguish between geometric structures that are not mission-relevant and structures

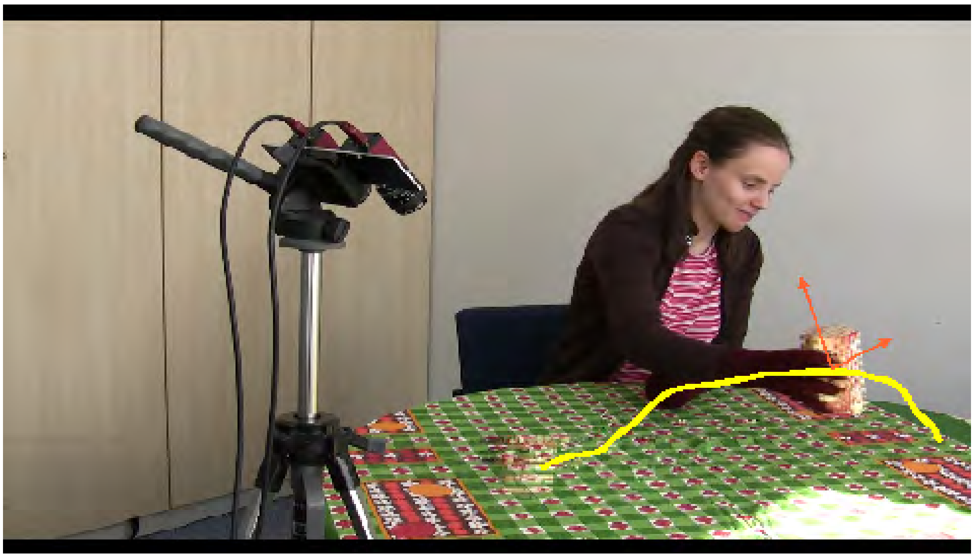


Figure 3.2: System learns handling properties of objects stored in the *Working Memory* in WP5 from human actions.

that need to be manipulated and, therefore, explored in their exact geometric and tactile properties. Since our goal is operation in cluttered complex environments, we implemented a novel registration system that allows object identification under significant occlusion in the scene, where the initial assumption from Year 1 about a planar supporting surface does not hold anymore [PBed]. We implemented a system that tracks the human actions (Fig. 3.2) and derives from the observed trajectories information about allowed object handling properties [PBar].

Neuroscientific work on human grasping strategies in WP1 provides here additional information about expected physical properties of objects depending on grasp point modifications of objects. Known objects are stored in the a-priori *Atlas* representation often with multiple physical state alternatives, e.g. full or empty, that result in changing grasp point selection by the human. The knowledge about human grasp strategies allows here a pre-selection of possible object states that need to be verified in the active scene exploration.

Passive observation of the environment does not allow to recover complete information about the environment. A static stereo head on a manipulator provides 2.5D information about the environment with no information about the occluded faces of the object. Active exploration of the environment with moving cameras as a possible extension of the manipulation system was proposed in [RBed] in continuation of our previous work on next best view strategies in Year 1.

In terms of the active vision system, different exploration strategies are possible. During the second year, we have developed methods for multimodal scene exploration, based on visual and haptic input. Here, initial object hypotheses formed by active visual segmentation are confirmed and augmented through haptic exploration with a robotic arm. We update the current belief about the state of the map with the detection results and predict yet unknown parts of the map with a Gaussian Process. We show that through the integration of different sensor modalities, we achieve a more complete scene model. We also show that the prediction of the scene structure leads to a valid scene representation even if the map is not fully traversed. Furthermore, we propose different exploration strategies and evaluate them both in simulation and on the KTH robotic platform. Fig. 3.3 displays the overall process.

3.1.2 Grasping Known and Unknown Objects

Integration progressed along lines of known objects and towards unknown objects. Methods for known objects are related to the methods mentioned above for scene exploration and interaction [PBar].

Familiar objects are recognized based on shape context presented in [BBK09]. We have worked further on generating grasping actions for familiar and novel objects based on visual input from a stereo camera. The work integrated two methods advantageous either in predicting how to grasp an object or where to

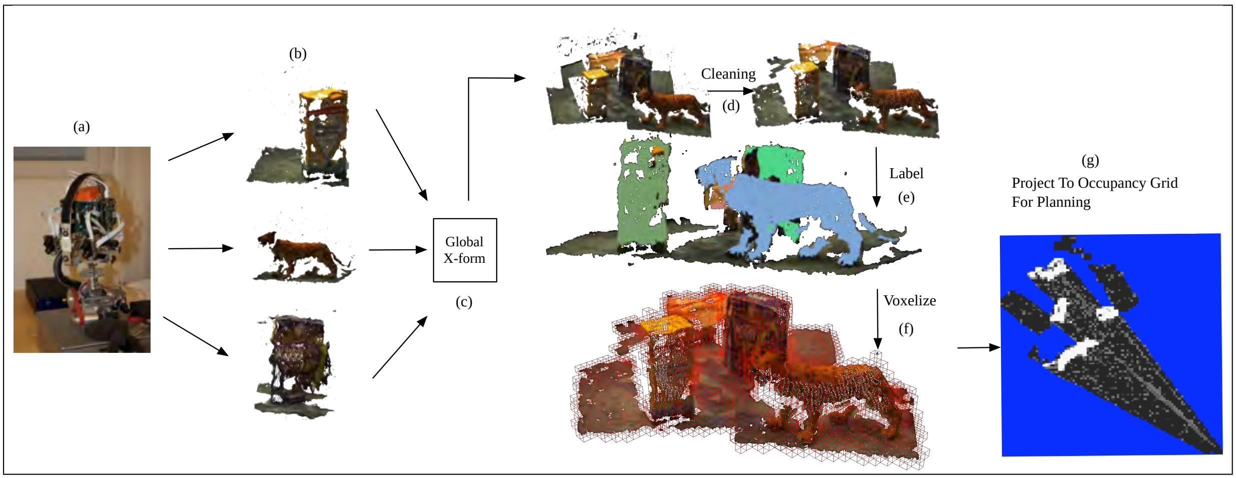


Figure 3.3: The pipeline for the generation of an occupancy grid from individual views for scene exploration. (a) displays the robotic head from which several individual views (b) are gathered. (c) These views are projected into a common reference frame. and (d) cleaned to remove noise. The points are labeled according to the 3D object segmentation. In (e) and in (f) these points are voxelized. The voxels for those points labeled as objects are projected down into the map which appears in (g) with blue being computed as unseen and gray levels corresponding to occupancy probability.

apply a grasp. The first one reconstructs a wire frame object model through curve matching. Elementary grasping actions can be associated to parts of this model. The second method predicts grasping points in a 2D contour image of an object. By integrating the information from the two approaches, we can generate a sparse set of full grasp configurations that are of good quality. We demonstrate our approach integrated in a vision system for complex-shaped objects as well as in cluttered scenes. An example of the whole system is shown in Figure 3.4.

An important step to work towards the final project goal is to move towards grasping unknown objects and to integrate this work into the overall system. To this end the work presented in [RV09] has been

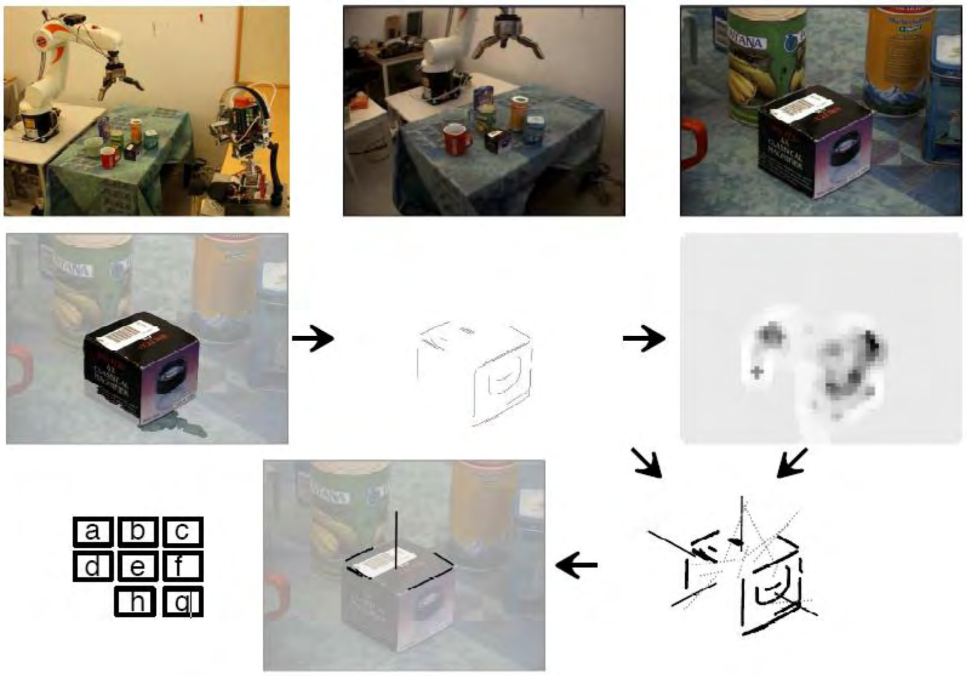


Figure 3.4: (a): System setup with 6 DoF KUKA arm, a 7 DoF SCHUNK hand and the 7 Dof ARMAR-III stereo head. (b,c): Left peripheral and foveated views. (d-h): The steps of the grasping system.

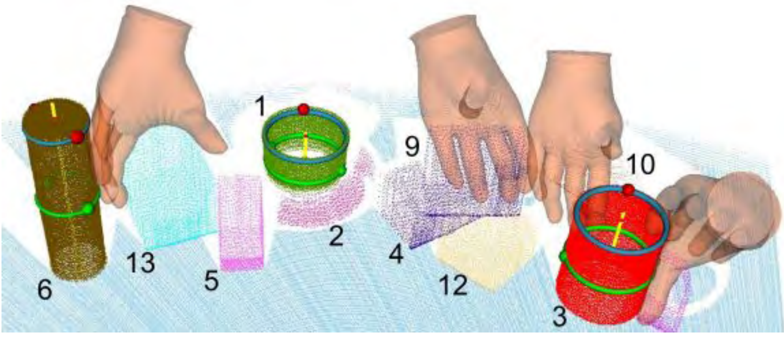


Figure 3.5: Grasping unknown objects.

made available to other partners. In this work stereo or laser data is used to obtain the shape of parts in the scene. An example is given in Figure 3.5. The approach is to split work in two groups of objects, rotationally symmetric objects and all other objects, which are estimated with planar patches starting from the top surfaces.

Figure 3.5 shows an example of a scene with objects, the detection of grasping points and hand poses. The green points display the computed grasping points for rotationally symmetric objects. The red points show an alternative grasp along the top rim. The illustrated hand poses with the Otto Bock hand prosthesis on the TUW robot arm shows a possible grasp for the remaining graspable objects. The numbers refer to the objects in the database for purposes of evaluation in [RV09].

To achieve the integration of detecting these grasp points, the partners implemented similar stereo systems as TUW or adapted their stereo systems to the needs. For example, for the Karlsruhe head, that is also used at KTH, special calibration of the cameras has been developed to achieve optimal results of stereo matching as the first step in the above approach of grasping point detection. As indicated, the approach does not know which objects are presented, but starts from the detected geometrical primitives. Tests in the next weeks will give feedback on the present state of the method. With this integration the partners are able to exploit the visual grasp point detection. This is needed as the first step for progressing with the work on integrating with tactile grounding, haptic exploration and estimating grasping stability.

3.1.3 Haptic Exploration and Stability Learning

Related to haptic exploration, Deliverable D13 presents how machine learning can be used to link tactile sensor measurements to the abstract concept of "stable grasp". The approach allows the use of abstract representation of manipulation actions developed in WP3 such that embodiment-specific mappings can be learned and embodiment-independent plans can be used. Integration of the system to the overall GRASP system is work in progress. The approach (see Fig. 3.6) has been studied on two GRASP platforms at KTH and LUT.

3.1.4 Integration of Vision and Haptics

Integration of visual pose estimation with contact force control was demonstrated at LUT. The task demonstrates grasping of known objects using vision to determine the objects' initial poses. A stereo camera is first used to detect the location of an object (a DVD case is shown in the example) on a table. Then a robot hand is used to grasp the object and transfer it to a box in preset location, while controlling contact forces using a tactile sensor. The overall setup is shown in Fig. 3.7(a) and few frames from a video taken during a test run in Fig. 3.7(b). The setup integrates software components from LUT, TUW and TUM as well as object models from UniKarl. A general diagram of the integrated software components and the flow of information between them is presented in Fig. 3.8.

The process is started by acquiring a 3D point cloud of the scene. This was performed by taking a stereo image using a Bumblebee2 stereo camera from Point Grey Research and then using proprietary software by the camera manufacturer to generate a point cloud. At this step the coordinate transform between

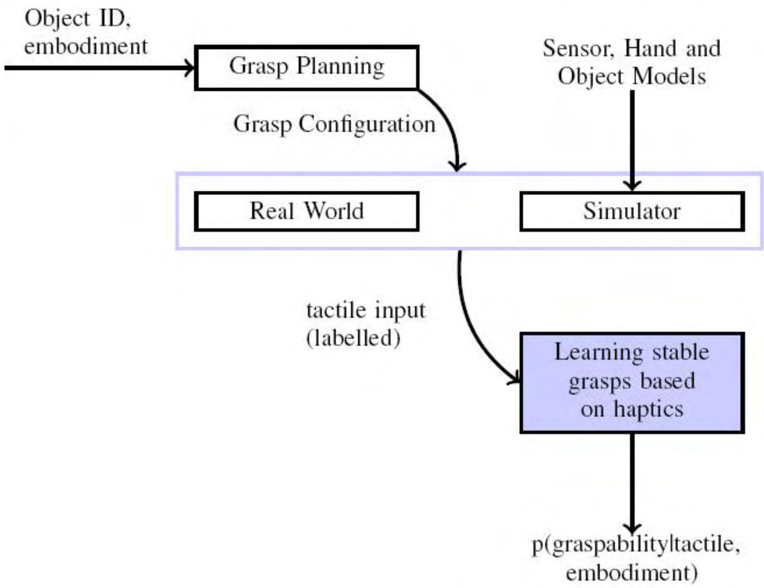
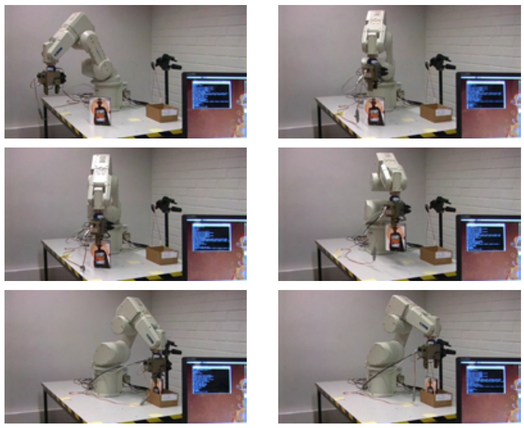


Figure 3.6: Learning grasp stability.

(a)



(b)

Figure 3.7: (a) The setup of the demonstration task: the stereo camera, the robotic hand and the DVD case; (b) A series of images showing how the DVD case is moved to box on the right.

the camera coordinates and the robot coordinates was also performed, by using markers positioned at known coordinates. The corresponding calibration methods have been developed at LUT.

After the point cloud was acquired, a dominant plane was removed and the rest of the point cloud segmented by using a software component from TUW. The next step was to recognize the object in the scene and determine its pose, that is, registration of the object. This requires both 3D models of the used objects and a method for matching a 3D model to an acquired point cloud. The models were provided by UniKarl and the registration software component by TUM. Additional models of simple objects (boxes and cylinders) were constructed algorithmically. The registration component only handled finding a pose in which the model and a segmented point cloud matched the best. Therefore, additional functionality was implemented at LUT for determining which point cloud segment represented the correct object.

After the pose of the object was known, pre-planned grasps were used for deciding where to grasp the object. At this point the pose of the robot hand which can be used for grasping the object is known. Grasping was performed by the manipulation architecture developed at LUT, described in Deliverable D13. The grasping action is based on a state machine which defines a set of controllers for the robot platform which are used to control the robot. The structure of this state machine that was used to move and place the object was static, however the variables such as the object pose were changed according to

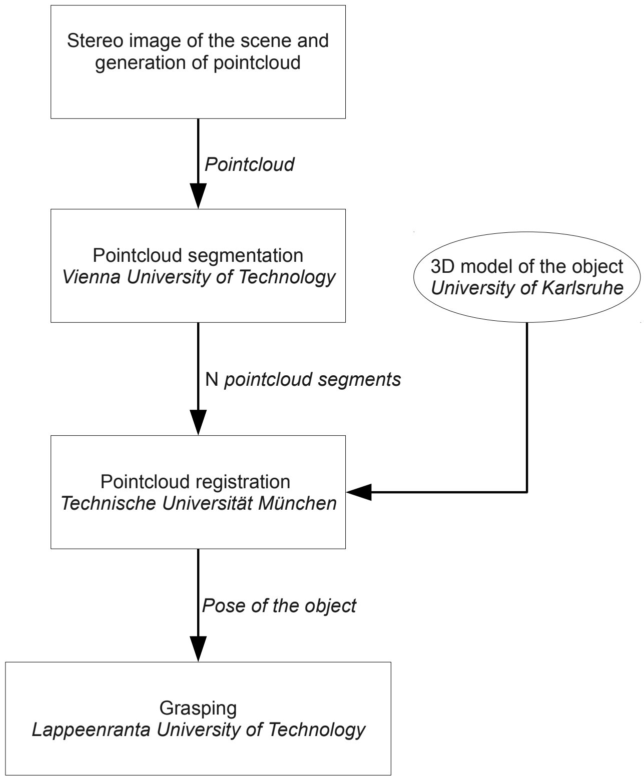


Figure 3.8: Software components and flow of information.

the output of the registration system, so that the object can be grasped from any pose within reachable distance from the robot. Contact force was controlled using tactile sensors.

3.2 Human Grasping Activities

3.2.1 Human Studies

Since the seminal studies by Jeannerod (1981) on primate grasping, many investigations have analyzed, in considerable detail and for a variety of different conditions, properties and control of reach-to-grasp movements. A particular focus of many of these studies was on kinematic parameters such as transport velocity, time and size of maximal grip aperture, and selected posture. However, normally these actions do not occur in isolation, but are part of a large action sequence by which the actor aims at reaching one or several goals. Indeed, there has been surprisingly little research on how actors move and shape their hands depending on the type of action they intend to perform with the goal object, on whether other objects in the field of view also need consideration, and on whether the other hand is also somehow involved in the action plan.

Obviously, robotic benchmark tasks such as emptying a dishwasher are characterized by just these complex conditions: objects are grasped in the presence of obstacles, they are moved to other locations, and new objects then have to be picked up. In order to provide GRASP with data on human strategies and behaviour in such tasks, we studied in the second work period various aspects of human grasping in more complex, prototypical actions, in several lines of experiments:

- Effects of obstacles and intermediate goals on reach-to-grasp kinematics
- Kinematics of grasping when attention resources have to be shared with a secondary action
- Planning of sequential pick-and-place actions

- Relation of covert and overt attention in combined eye and hand movements
- Gaze direction in pinch grasp preparation
- Grasping irregular shapes and natural objects with 2, 3, and 4 fingers
- Understanding manifolds of grasping actions

Together, our findings are not only exciting and novel for the neuroscience of human grasping, but we are convinced that they are also of considerable importance for the development of artificial systems that should be able to produce cognitive grasping. The studies have revealed several important principles that we deem to be of basic significance for the production of fluent, human-like grasping movements in a complex environment.

Manifolds of Grasping Actions

Natural human hand motion is highly non-linear and of high dimensionality. For some specific activities such as handling and grasping of objects, the observed hand motions lie on a lower-dimensional non-linear manifold in hand posture space. This notion has been commonly used in the area of robotics for the design of grasp taxonomies. The goal of the work was to, differently from all the existing grasp taxonomies, model the spatial dimensionality and temporal context of hand actions. Instead of studying how different objects are grasped, we study how different grasps are performed. Apart from the important insights of human hand motion, the developed technique has also been used to evaluate the state-of-the-art taxonomies. We have shown how the technique can be used to embed high-dimensional grasping actions in a lower-dimensional space suitable for modeling, recognition and mapping. Considering the whole grasping sequence instead of just a single grasp posture facilitates the spatial and temporal reconstruction of a grasping action. The method is evaluated on both synthetic and real data. The resulting latent space for synthetic data is shown in Figure 3.9 and it has a very distinct star shape. This is due to the special nature of the data set, with a common starting posture and linear interpolation to the different end postures. In the middle of the star is the resting position of the hand. If one moves outside along a branch, a specific grasp type will be formed.

An immediate application of the extracted latent space is a non-parametric dynamic model of grasping actions for tracking and classification. We do not model dynamics explicitly but include back-constraints that indirectly enforce temporal continuity in the latent space. This avoids the unimodal nature of the GPDM dynamics. The created GPLVM model allows the generation of concatenated grasping actions with natural transitions.

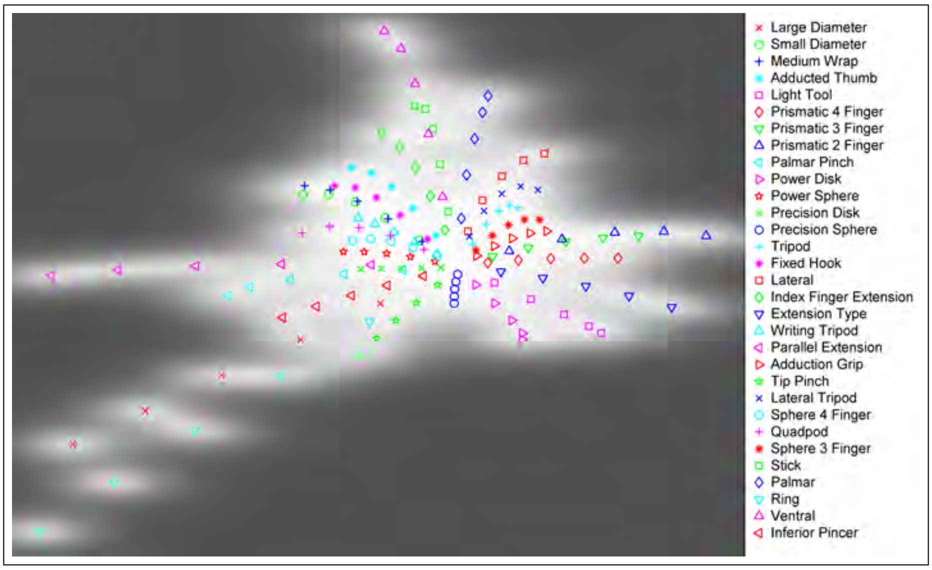


Figure 3.9: Grasp space spanned by synthetic data of 31 grasp actions.

3.2.2 Human Tracking

In [DRK+09] (*Attachment 1* to this Deliverable) we present a system for vision-based grasp recognition, mapping and execution on the humanoid robot ARMAR-IIIb. The system comprises three components: a real-time and markerless human upper body motion capture system which provides the approaching direction towards an object, hand pose estimation and grasp recognition system, which provides the grasp type performed by the human as well as a grasp mapping and execution system for grasp reproduction on a humanoid robot with five-fingered hands. Once an object is reached, the hand posture is estimated, including hand orientation and grasp type. For the execution on a robot, hand posture and approach movement are mapped and optimized according to the kinematic limitations of the robot. Experiments are performed on the humanoid robot ARMAR-IIIb.

Further progress on the tracking of human hands is reported in Deliverable D11.

3.2.3 Goal-Directed Imitation of Grasping Activities

We continued the work on learning motor skills from human demonstration and the realization of a goal-directed grasping imitation framework (see Fig 3.10).

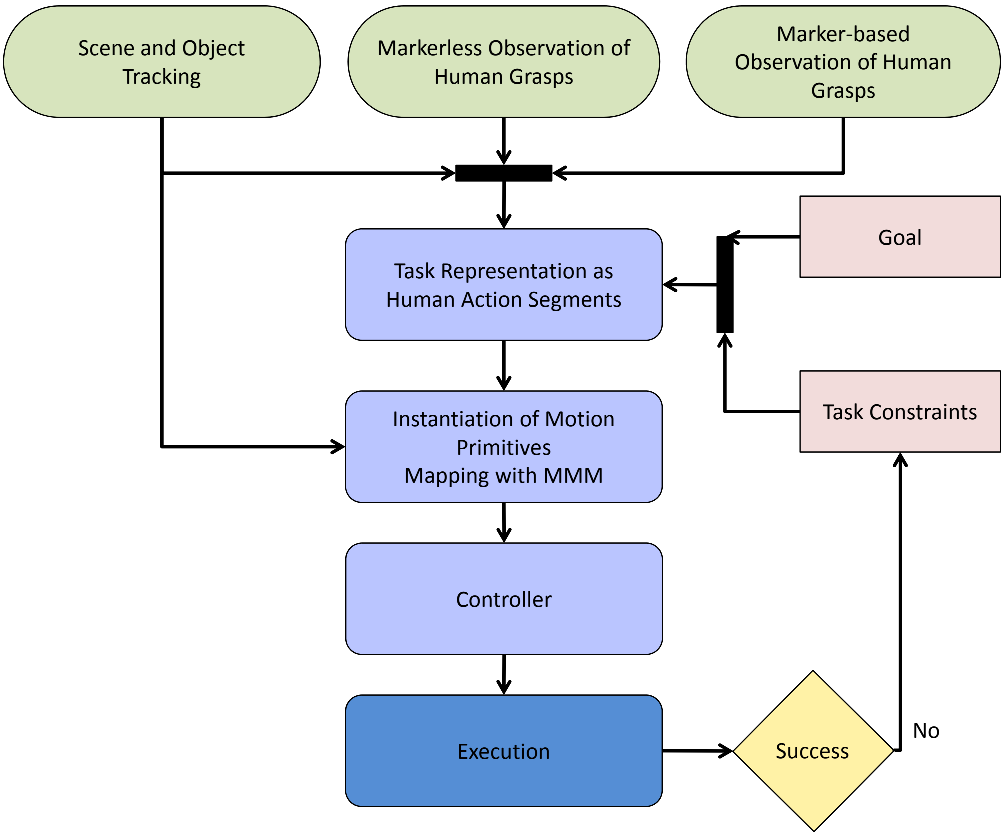


Figure 3.10: Framework for goal-directed imitation of grasping activities

Starting from observation of human actions, a library of motion primitives is built. The dynamic movement primitive (DMP) formulation proposed in [INS02] and [SIB03] is used for the representation of demonstrated movements with a set of differential equations. Representing a movement by a differential equation has the advantage that a perturbance can be automatically corrected for by the dynamics of the system (robustness against perturbation). Furthermore, the equations are formulated in a way that adaptation to a new goal is achieved by simply changing a goal parameter. This characteristic allows generalization. Based on this representation, we build a library of movements by labeling each recorded movement according to task and context (e.g., grasping, placing, and releasing). For further details the reader is referred to [PHAS09].

In WP2 we extend the framework of dynamic movement primitives to action sequences that allow object manipulation. We added semantic information to movement primitives, such that they can code object-oriented actions. We demonstrated the feasibility of the approach in an imitation learning setting, where a humanoid robot learned a pick-and-place task from a human demonstration, and could generalize this task to novel situations.

Ongoing work in cooperation between UniKarl and KTH is the learning of task constraints in grasping and the incorporation of such constraints in the imitation framework. These constraints describe the relationship between task, objects and actions and are thus of great importance for the adaptation of grasping actions to different objects and situations.

More details are given in Deliverable D12.

3.3 Integrated Grasp and Motion Planning on Armar-III

For grasping an object several tasks have to be solved in general, as searching a feasible grasping pose, solving the inverse kinematics (IK) or finding a collision-free grasping trajectory. We developed an algorithm to solve such problems based on a probabilistic planning approach using Rapidly Exploring Random Trees (RRT) (see [VDAD10], *Attachment 2* to this Deliverable). The so-called RRT-Grasp planner searches a feasible and reachable grasp during the planning process and thus pre-calculated grasping positions are not needed. The developed approach 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. This means that 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. 3.11). Searching a feasible grasping position online has the advantage that the

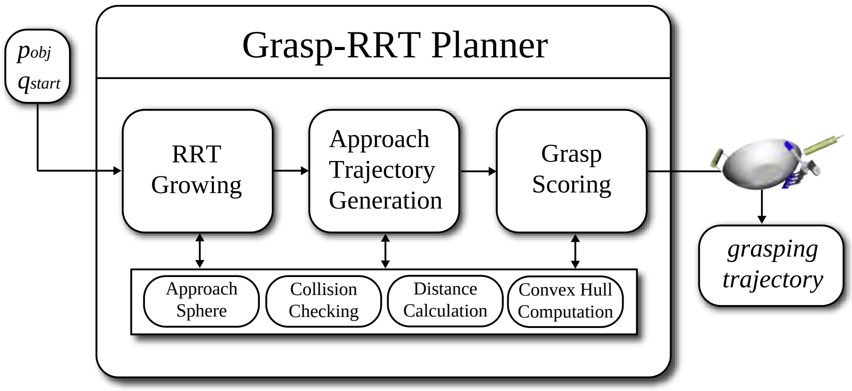


Figure 3.11: Overview of the Grasp-RRT Planner.

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. The algorithms can even be used when just a rough estimation of an unknown object is given, since an approximated 3D model can be used to search grasping poses online.

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 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

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

$RRT.AddConfiguration(q_{start})$;
while *(!TimeOut())* **do**
    ExtendRandomly(RRT);
    if $(rand() < p_{SearchGraspPose})$ **then**
        $n_{grasp} \leftarrow ApproachTrajectory(RRT, p_{obj})$;
        if $(ScoreGrasp(n_{grasp}) > score_{min})$ **then**
            return $BuildSolution(Grasp)$;
    end
end

trajectory can be built easily since the approach trajectory already defines a collision-free connection to the RRT.

Scoring a grasping configuration is realized by 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, which is approximated by a friction pyramid to reduce the complexity. Each applied force leads to a torque vector, whose magnitude and direction depend 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:

$$min(\sum_{i=1}^{m}(c_i - p_{com}) \times b_i f_i)^2 \,, \tag{3.1}$$

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. 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 (depicted in Fig. 3.12).

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. 3.1), the grasp is represented by the convex hull CH_g as depicted in Fig. 3.12. 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. 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.

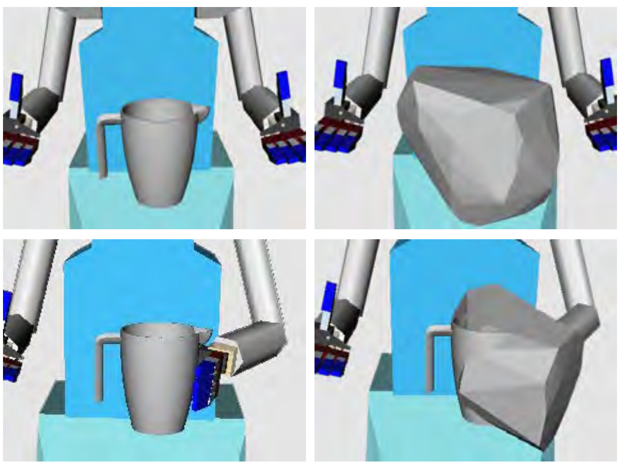


Figure 3.12: Top Row: The object with a visualization of CH_o. Bottom Row: The grasp specific CH_g is used to compute the grasp quality score. For the measuring cup a grasp quality score of $q_g = 0.46$ is determined.

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;

When large objects like the wok in Fig. 3.14 should be grasped by a humanoid robot, both hands are needed for applying a stable grasp. Based on 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. 3.13 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 algorithms have been evaluated with different setups in simulation and on the humanoid robot Armar-III (see [VDAD10], (*Attachment 2* to this Deliverable)).

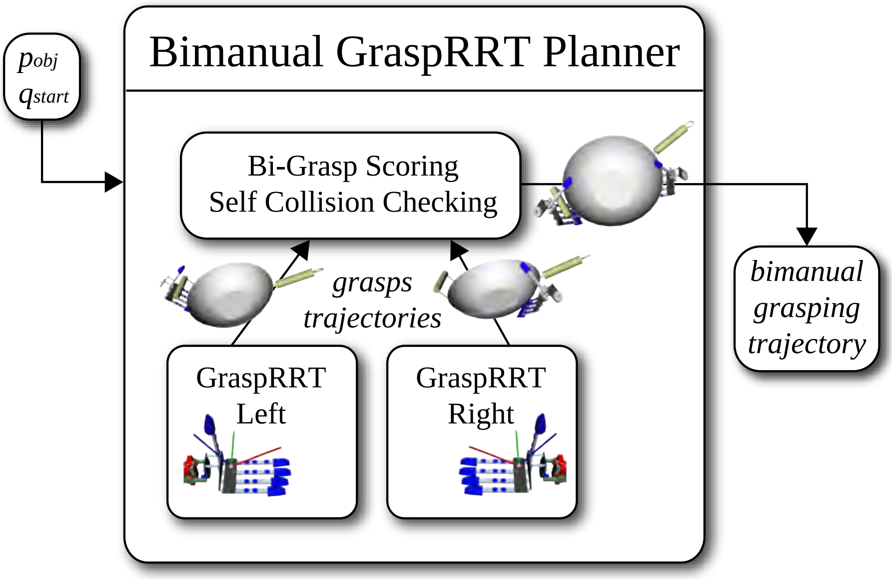


Figure 3.13: Overview of the Bimanual Grasp-RRT Planner.

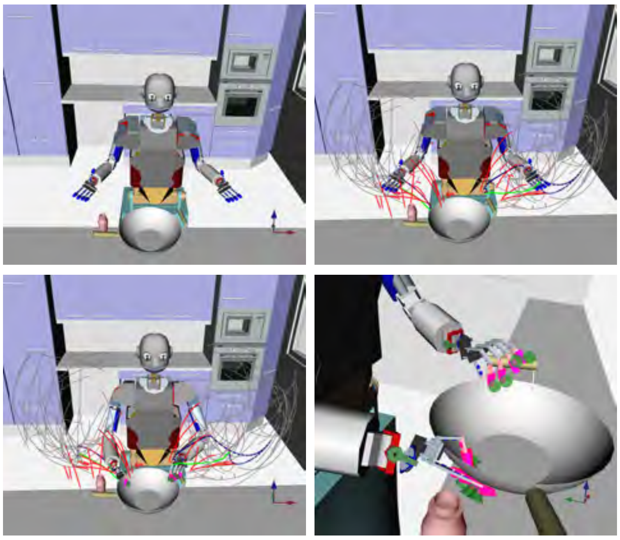


Figure 3.14: The Bimanual Grasp-RRT planner is used to search a collision-free grasping trajectory for 14 DoF of both arms of Armar-III.

3.4 Integration of Simulator and Control Architecture

The control architecture described in D13 is currently being used to control different robots platforms, focusing on manipulation, especially grasping by a robotic arm and hand. In GRASP, it is necessary that the simulated robot performs identically to a real one, including the control logic and sensors. In the following we describe work aiming towards this goal.

In summary, for creating a simulated model of a real robot, for which the controllers already exist, the following are needed:

1. Implement the interfaces for the simulated hand and arm, addressing the commands to the simulated actuators.
2. Implement the simulated sensors and attach them to the robot
3. Initialize the High-level controller with the specific simulated robot

The rest of the implementation should remain the same for both cases.

The manipulation actions are described by abstract state machines which are defined in eXtensible Markup Language (XML). The state machine contains the definition of states and the transitions between the states. It also describes the conditions that determine whether the described action is a success or a failure, which is especially useful in simulation as the system can learn the reasons of the failures before actual action takes place in the real world. As the abstract state machine is hardware independent, it does not need to be changed in the simulated case. This hardware independent description of the action is translated to a hardware- or embodiment-specific state machine, which allows the abstract state machine to be adapted to the hardware platform. The embodiment-specific state machine contains the control logic in a hybrid discrete-continuous automaton. This hybrid structure allows real-time control of the robot while having a discrete set of states, which are dictated by the abstract state machine.

The high-level controller consists of the embodiment-specific state machine, interfaces to the hardware manipulator and the control arbitrator. These interfaces should be implemented for the simulated arm and hand and passed to the controller at initialization.

Each state of the embodiment-specific state machine contains the control logic in the form of primitive controllers. These controllers output the control signals to the hardware actuator. The signal should be the same as in the real robot but the control arbitrator will send it to the simulated robot instead. The simulated manipulator will then send the given command to the specific simulated actuator.

The transitions and primitive controllers have access to all the sensors in the system. When defining the specific robot, the simulated sensors are attached and can be accessed as any real sensor.

The first example of a simulated robot using this abstraction architecture was implemented using the platform of the Lappeenranta University of Technology, consisting of a Melfa RV-3SB robot arm with a Schunk PG70 parallel jaw gripper. The arm was not considered in order to concentrate only on the gripper fixed to a static position. Each finger of the gripper has a Weiss tactile sensor (DSA 9205) attached.

The control architecture design for this specific gripper is shown in Figure 3.15 and the implementation of its components is described in the following sections.

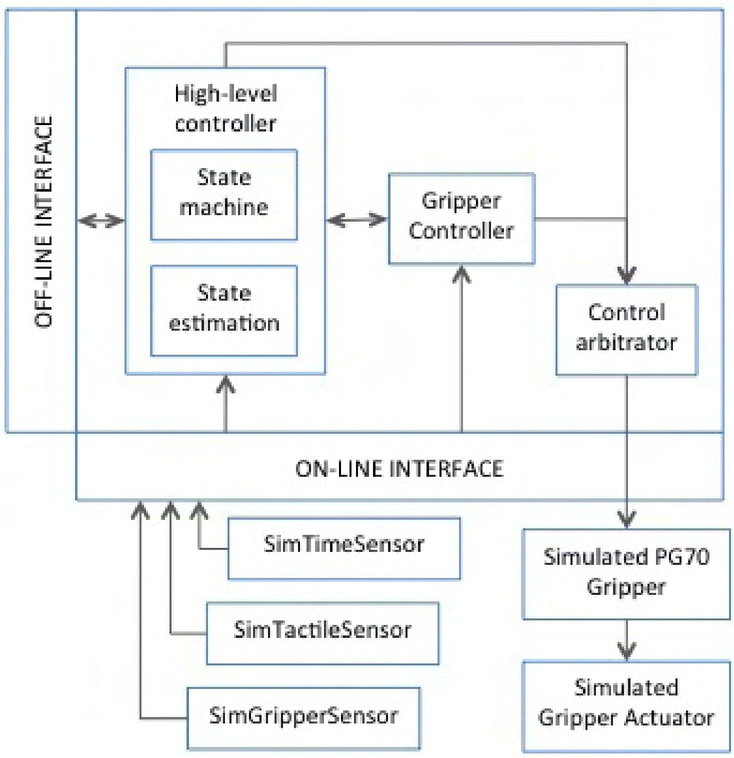


Figure 3.15: Control architecture design for the simulated gripper

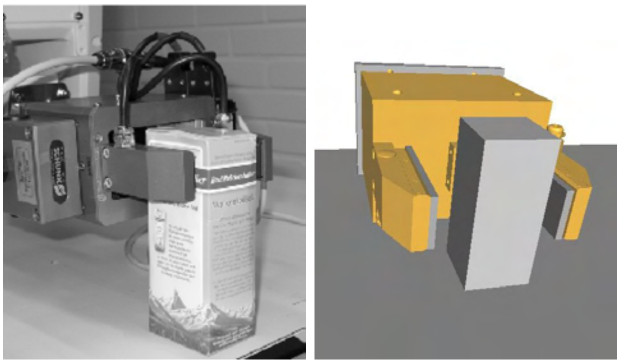
3.4.1 Implementation

The implementation of the different components of the system is based on OpenRAVE [DK08], a planning architecture developed at the Carnegie Mellon University Robotics Institute. It has been designed to be an open architecture targeting a simple integration of simulation, visualization, planning, scripting, and control of robot systems. We have extended its functionality developing our own custom plugins for controllers, sensors, actuators and physics engines.

The real Schunk PG70 parallel jaw gripper was modeled using three Inventor files, one for the base and one for each finger. The environment consists of the robot, a table and a box laying on it. The description of the robot and the environment was stored in XML files.

The real robot and its model in OpenRAVE are shown in Figure 3.16.

In addition to the robot geometry, several sensors had to be implemented to model the PG70 gripper: a) A time sensor which queries the physics engine defined in the environment about the simulated time is used by the high-level controller to synchronize events. b) The PG70 gripper has a sensor which returns three values: the position of the gripper fingers, which is the distance between them; the velocity of the



(a) Schunck PG70 gripper (b) OpenRAVE model

Figure 3.16: Schunck PG70 gripper, real and simulated.

fingers and finally the measured current. c) The Weiss tactile sensor attached to each finger was modeled using soft contacts. The detailed implementation is described in [SMM10].

All these sensors were implemented in a sensor plugin loaded to OpenRAVE at runtime. Each of them has a specific type defined, which should be specified when attaching the sensor to a robot and will differentiate them from the real ones such as *SimTimeSensor* or *SimTactileSensor*.

The possibility to create actuators did not exist in OpenRAVE. The architecture originally was meant for planning purposes so it does not perform dynamic simulations. For example, a robot can be moved by changing the values of its joint angles, but not applying a torque to them. In order to dynamically move a robot, actuators that represent the real motors should be modeled and added to the specific joints. In order to allow this functionality in OpenRAVE a new interface called *ActuatorBase* was defined, allowing plugins of this type to be created and attached to a robot.

For our example, a new plugin modeling the Gripper Actuator was implemented. The slider joint name of each gripper finger should be specified in the XML file. It uses the function *SendCmd* to set the velocity to each joint in order to open, close or stop the gripper fingers.

The diagram in Figure 3.17 shows in a simplified way, the sequence of actions performed by OpenRAVE to control the simulated robot.

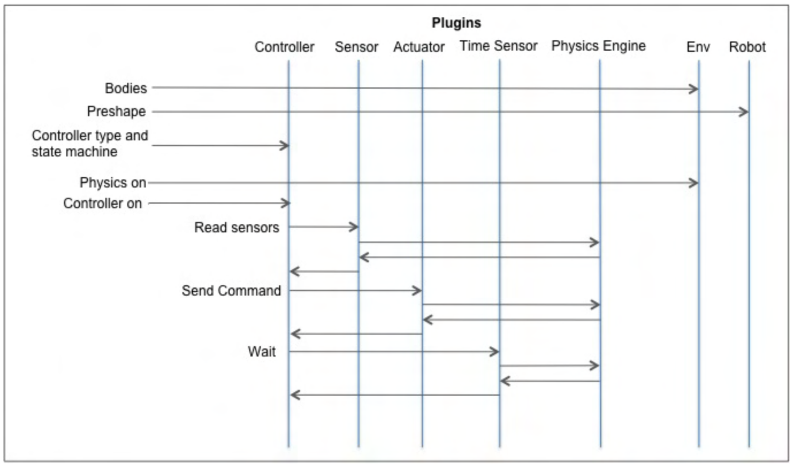


Figure 3.17: OpenRAVE control sequence diagram.

The example used to test the integration was the following:

- The robot hand starts static, in a predefined position, with the fingers open.
- An object is placed between the fingers.

- The fingers start closing while the gripper sensor is being read.
- When the readings indicate that the fingers have closed around the object, they stop closing.
- The fingers are then opened to the start position.

3.5 New Features in the Simulator

See Deliverable D16.

Chapter 4

Conclusions and Future Work

This Deliverable presented the work in the second year in WP7 towards the implementation of a cognitive control architecture for grasping and manipulation. Major integration efforts related to scene exploration, learning from human observation, grasp planning as well as the use of the simulator are described.

The integration efforts will be continued in the next period of the project. The focus will be on further development of platform independent software components to allow on the one hand the smooth transferability of the developed methods to all robot platforms in the project as well as on the other hand to serve as a basis for a framework for benchmarking in the context of object grasping.

References

[BBK09] Niklas Bergström, Jeannette Bohg, and Danica Kragic. Integration of visual cues for robotic grasping. In *ICVS '09: Proceedings of the 7th International Conference on Computer Vision Systems*, pages 245–254, Berlin, Heidelberg, 2009. Springer-Verlag.

[DK08] Rosen Diankov and James Kuffner. Openrave: A planning architecture for autonomous robotics. Technical Report CMU-RI-TR-08-34, Robotics Institute, Pittsburgh, PA, July 2008.

[DRK+09] M. Do, J. Romero, H. Kjellström, P. Azad, T. Asfour, D. Kragic, and R. Dillmann. Grasp Recognition and Mapping on Humanoid Robots. In *IEEE/RAS International Conference on Humanoid Robots*, Paris, France, December 2009.

[INS02] Auke Jan Ijspeert, Jun Nakanishi, and Stefan Schaal. Movement imitation with nonlinear dynamical systems in humanoid robots. In *In IEEE International Conference on Robotics and Automation (ICRA2002*, pages 1398–1403, 2002.

[LLR09] Pat Langley, John E. Laird, and Seth Rogers. Cognitive architectures: Research issues and challenges. *Cognitive Systems Research*, 10(2):141–160, June 2009.

[PBar] S. Petsch and D. Burschka. Estimation of Spatio-Temporal Object Properties for Manipulation Tasks from Observation of Humans. In *Proc. of IEEE ICRA 2010*, Anchorage, USA, May 2010 (to appear).

[PBed] Chavdar Papazov and Darius Burschka. Sampling in Constant Time for 3D Object Detection in Noisy and Cluttered Scenes. In *ECCV2010*, submitted.

[PHAS09] Peter Pastor, Heiko Hoffmann, T. Asfour, and Stefan Schaal. Learning and generalization of motor skills by learning from demonstration. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Kobe, Japan, 2009.

[RBed] Oliver Ruepp and Darius Burschka. Towards On-Line Intensity-Based Surface Recovery from Monocular Images. submitted.

[RV09] Mario Richtsfeld and Markus Vincze. Robotic grasping from a single view. In *18th International Workshop on Robotics in Alpe-Adria-Danube Region (RAAD 09)*, 2009.

[SIB03] S. Schaal, A.J. Ijspeert, and A. Billard. Computational approaches to motor learning by imitation. *Philosophical transaction of the Royal Society of London, series B*, 358(1431):537–547, 2003.

[SMM10] P. Korkealaakso S. Moisio, B. Leon and A. Morales. Model of tactile sensors using soft contacts and its application to simulated robot grasping. Technical report, Lappeenranta University of Technology and Universitat Jaume I, February 2010.

[Sun04] Ron Sun. Desiderata for cognitive architectures. *Philosophical Psychology*, 17:341–373, 2004.

[VDAD10] N. Vahrenkamp, M. Do, T. Asfour, and R. Dillmann. Integrated grasp and motion planning. In *IEEE International Conference on Robotics and Automation*, Anchorage, USA, Mai 2010.

[VMS07] D. Vernon, G. Metta, and G. Sandini. A survey of artificial cognitive systems: Implications for the autonomous development of mental capabilities in computational agents. *Evolutionary Computation, IEEE Transactions on*, 11(2):151–180, April 2007.

Appendix A

Attached Papers

1 **Grasp Recognition and Mapping on Humanoid Robots**. Martin Do, Javier Romero and Hedvig Kjellström and Pedram Azad, Tamim Asfour, Danica Kragic and Rüdiger Dillmann. In *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, Paris, France, December, 2009

2 **Integrated Grasp and Motion Planning**. Nikloaus Vahrenkamp, Martin Do, Tamim Asfour and Rüdiger Dillmann. In *IEEE International Conference on Robotics and Automation (ICRA)*, Anchorage, USA, May, 2010

Grasp Recognition and Mapping on Humanoid Robots

Martin Do, Javier Romero, Hedvig Kjellström, Pedram Azad,
Tamim Asfour, Danica Kragic, Rüdiger Dillmann

Abstract— **In this paper, we present a system for vision-based grasp recognition, mapping and execution on a humanoid robot to provide an intuitive and natural communication channel between humans and humanoids. This channel enables a human user to teach a robot how to grasp an object. The system comprises three components: human upper body motion capture system which provides the approaching direction towards an object, hand pose estimation and grasp recognition system, which provides the grasp type performed by the human as well as a grasp mapping and execution system for grasp reproduction on a humanoid robot with five-fingered hands. All three components are real-time and markerless. Once an object is reached, the hand posture is estimated, including hand orientation and grasp type. For the execution on a robot, hand posture and approach movement are mapped and optimized according to the kinematic limitations of the robot. Experimental results are performed on the humanoid robot ARMAR-IIIb.**

I. Introduction

A humanoid robot's capability of autonomously adapting and acting in new and unstructured environments is very limited. In the majority of cases, a skilled and experienced user is needed for the programming in order to adapt an existing action to a new situation. To enable teaching of a robot by non-expert users, a natural intuitive interface is needed. Since imitation presents an obvious solution for tackling this problem, this field has received great interest in humanoid robotics. The benefit of exploiting demonstration is clearly revealed in [1], where an anthropomorphic arm is capable of balancing a pole in the first trial after observing a human.

A challenging problem where a robot could greatly benefit from a human demonstration is an object grasping task. Such a task involves the control of several degrees of freedom, visual servoing, tactile feedback, etc., turning it to a highly complex task. About the grasp action, a grasp can be divided in two stages: an approach stage and final grasp stage. Due to high object variety concerning shape, size, and mass, determining an adequate approach movement and selecting a suitable grasp type increase the chances that an object is successfully grasped. Instead of telling the robot explicitly which approach movement and which grasp type shall be used, it is desirable to have a system which enables the robot to observe a human during grasp execution and to imitate the demonstration. For the implementation of such a system, various problems have to be tackled, like observation of the human performing the grasp, the mapping of the grasp, and the final execution on the robot.

An important part of the grasp imitation system is the block in charge of getting information about the arm and hand movements. In order to provide this information, the approach movement of the arm as well as the hand pose have to be recognized. Aiming towards ease of use, markerless systems seem to be the most obvious solution for the observation of human grasps since, besides vision sensors, additional equipment is avoided and the preparation effort is kept to a minimum. However, markerless 3D motion capturing and reconstruction of hand pose based on image data are extremely difficult problems due to unstructured environments, the large self-occlusion, high dimensionality and non-linear motion of the arm and the fingers.

Besides the perception modules, another crucial part of an imitation system consists of the mapping and the execution of an observed human grasp on a humanoid robot. Due to severe constraints of mechanical systems and differences between the human and the robot's embodiment, a large number of requirements arise, which are difficult to be satisfied at once. Towards enabling a humanoid to imitate a human grasp, our system integrates several subsystems and methods. First, using a stereo camera setup human observation is initiated by capturing upper body motion and scanning the scene for known objects to attain information on the approach stage. Subsequently, grasp classification and hand orientation are provided through the estimation of the full hand pose in a non-parametric fashion. Finally, the motion data is gathered and mapped onto the robot for execution. The mapping is accomplished via a standardized interface and the ensuing execution is achieved by means of non-linear optimization.

II. Related Work

Several approaches have been made to create a markerless human motion capture system for humanoid robots. Especially, image-based approaches have been a major focus of this field. These approaches are either search-based ([2], [3]), utilize an optimization approach based on 2D-3D correspondences [4], [5], or are based on particle filtering. In [6], it was shown that human motion can be successfully tracked with particle filtering, using three cameras positioned around the scene of interest.

J. Romero, H. Kjellström and D. Kragic are with KTH - Royal Institute of Technology, Stockholm, Sweden, as members of the Computer Vision & Active Perception Lab., Centre for Autonomous Systems, e-mail: jrgn,hedvig,dani@kth.se

M. Do, P. Azad, T. Asfour and R. Dillmann are with the University of Karlsruhe (TH), Karlsruhe, Germany, as members of the Institute for Anthropomatics, e-mail: do,azad,asfour,dillmann@ira.uka.de

Towards imitation of human motion by a robot, the mapping and execution of motion capture data are issues whereas possible solutions pursue strategies which either make use of artificial markers and landmarks or which are based on the transfer and post-processing of joint angles. Marker-based approaches are presented in [7] and [8] where methods based on minimization of the mismatch between robot and human markers are introduced. However, in [9] and [10], joint angles of a demonstrators posture are determined and transferred to the robot for execution. Due to joint and velocity constraints, a scaling and transformation process must be performed in order to obtain a feasible joint angle configuration for the robot.

Analysis of human hand pose for the purpose of learning by demonstration (LbD), see [11] has been thoroughly investigated, almost exclusively with the help of markers and/or 3D sensors attached to the human hand. In the work by Oztop [12] motion capture, color segmentation with artificially colored hands, and active-marker capture systems were compared. Magnetic gloves have also been used extensively because of their accuracy [13]. Another input source for LbD systems is the passive joint measurements of the robot itself [14]. However, the methods shown above all use invasive devices. We envision a LbD scenario where the teaching process can be initiated without calibration and where the robot-user interaction is as natural as possible. For this reason, we want to reconstruct the hand posture in a visual markerless fashion.

Methods for hand pose estimation that are not constrained to a limited set of poses can largely be classified into two groups [15]: I) model based tracking and II) single frame pose estimation. Methods of type I) usually employ generative articulated models [16], [17], [18], [19]. Since the state space of a human hand is extremely high-dimensional, they are generally very computationally demanding, which currently makes this approach intractable for a robotics application. Methods of type II) are usually non-parametric [20], [21]. They are less computationally demanding and more suited for a real-time system, but also more brittle and sensitive to image noise, since there is no averaging over time. The method presented here falls into the second approach. However, it takes temporal continuity into account and it can be used for online real-time reconstruction.

III. Grasp Observation

As mentioned before, we assume that a grasp consists of an approaching stage and a final grasp stage. The observation of the whole grasping process involves recognition of the grasp type, estimation of the approach arm movement and object detection. Following the target of having an intuitive and natural programming interface for robots, we use a markerless human motion capture system for the observation of human motion using the stereo vision system of the robot's head [22]. The head has two eyes and each eye is equipped with two cameras, one with a wide-angle lens for peripheral vision and one with a narrow-angle lens for foveal vision.

First, the robot recognizes known objects in the scene and starts capturing human motion. The hand pose estimation system is triggered as soon as the human hand is in the vicinity of the object. To obtain a close-up of the hand, the foveal cameras are used. The grasp observation is finished with the classification of the observed human grasp.

A. Hand Pose Estimation

The input to the method is a sequence $[\mathbf{I}_t], t = 1, \ldots, n$ of monocular images of the human hand [21].

In each frame $\mathbf{I}_t$, the hand is segmented using skin color segmentation based on color thresholding in HSV space. The result is a segmented hand image $\mathbf{H}_t$.

The shape information contained in $\mathbf{H}_t$ is represented with a Histogram of Oriented Gradients (HOG). This feature has been frequently used for representation of human and hand shape [23], [24], [25]. It has the advantage of being robust to small differences in spatial location and proportions of the depicted hand, while capturing the shape information effectively.

1) Non-parametric Pose Reconstruction: In this section, we omit the time index and regard the problem of reconstructing a *single* pose $\mathbf{p}$ from a *single* HOG $\mathbf{x}$.

Our goal is to obtain the grasp class and orientation of the human hand. We can infer this information from the pose $\mathbf{p}$ of the hand, since all this information is stored for each entry of the database. Therefore, we want to find the mapping $\hat{\mathbf{p}} = \mathcal{M}(\mathbf{x})$, where $\hat{\mathbf{p}}$ is the estimated 31D hand pose in terms of global orientation (lower arm yaw, pitch, roll) and joint angles (3 wrist joint angles, 5 joint angles per finger) , and $\mathbf{x}$ is the observed 512D HOG representation of the hand view, described in Section III-A.

The mapping function $\mathcal{M}$ can be expected to be highly non-linear in the HOG space, with large discontinuities. Following [21], $\mathcal{M}$ is therefore represented non-parametrically, i.e., as a database of example tuples $\{\langle \mathbf{x}_i, \mathbf{p}_i \rangle\}, i \in [1, N]$. Due to the high dimensionality of both the HOG space (512D) and the state space (hereafter denoted JOINT space, 31D), the database needs to be of a considerable size to cover all hand poses to be expected; in our current implementation, $N = 90000$. This has two implications for our mapping method, as outlined in the subsections below.

2) Generation of Database Examples: Generating a database of 10^5 examples from real images is intractable.

(a) HOG $\mathbf{x}$, JOINT $\mathbf{p}$ (b) HOG $\mathbf{x}_1$, JOINT $\mathbf{p}_1$ (c) HOG $\mathbf{x}_2$, JOINT $\mathbf{p}_2$

Fig. 1. Ambiguity in mapping from HOG space to JOINT space. Even though it is visually apparent that $\|\mathbf{p} - \mathbf{p}_2\| \ll \|\mathbf{p} - \mathbf{p}_1\|$ in JOINT space, database instance 1 will be regarded as the nearest neighbor as $\|\mathbf{x} - \mathbf{x}_1\| < \|\mathbf{x} - \mathbf{x}_2\|$. *Note that the object in the hand just contributes with occlusion of the hand in HOG extraction, as it is then colored uniformly with background color.*

Instead, we used the graphics software Poser 7 to generate synthetic views $\mathbf{H}_i^{\text{synth}}$ of different poses. The database examples are chosen as frames from short sequences of different grasp types from different view points, different grasped objects, and different illuminations.

The grasp types are selected according to the taxonomy developed in the GRASP project[1], which integrates the Cutkosky [26], Kamakura [27], and Kang [28] taxonomies. The whole database is also available at the same place.

From each example view $\mathbf{H}_i^{\text{synth}}$, the tuple $\langle \mathbf{x}_i, \mathbf{p}_i \rangle$ is extracted, where $\mathbf{x}_i$ is generated from $\mathbf{H}_i^{\text{synth}}$ as described in Section III-A, and $\mathbf{p}_i$ is the pose used to generate the view $\mathbf{H}_i^{\text{synth}}$ in Poser 7.

3) Approximate Nearest Neighbor Extraction: Given an observed HOG $\mathbf{x}$, the goal is to find an estimated pose $\hat{\mathbf{p}} = \mathcal{M}(\mathbf{x})$. With the non-parametric mapping approach, the mapping task $\hat{\mathbf{p}} = \mathcal{M}(\mathbf{x})$ is one of searching the database for examples $\langle \mathbf{x}_i, \mathbf{p}_i \rangle$ such that $\mathbf{x}_i \approx \mathbf{x}$. More formally, X_k, the set of k nearest neighbors to $\mathbf{x}$ in terms of Euclidean distance in HOG space, $d_i = \|\mathbf{x} - \mathbf{x}_i\|$ are retrieved.

As an exact kNN search would put serious limitations on the size of the database, an approximate kNN search method, Locality Sensitive Hashing (LSH) [29] is employed. LSH is a method for efficient ϵ-nearest neighbor (ϵNN) search, i.e. the problem of finding a neighbor $\mathbf{x}_{\epsilon\text{NN}}$ for a query $\mathbf{x}$ such that

$$\|\mathbf{x} - \mathbf{x}_{\epsilon\text{NN}}\| \leq (1+\epsilon)\|\mathbf{x} - \mathbf{x}_{\text{NN}}\| \tag{1}$$

where $\mathbf{x}_{\text{NN}}$ is the true nearest neighbor of $\mathbf{x}$. The computational complexity of ϵNN retrieval with LSH [29] is $\mathcal{O}(DN^{\frac{1}{1+\epsilon}})$ which gives sublinear performance for any $\epsilon > 0$.

4) The Mapping $\mathcal{M}$ is Ambiguous: The database retrieval described above constitutes an approximation to the true mapping $\hat{\mathbf{p}} = \mathcal{M}(\mathbf{x})$, robust to singularities and discontinuities in the mapping function $\mathcal{M}$.

However, it can be shown empirically that $\mathcal{M}$ is inherently ambiguous (one-to-many); substantially different poses $\mathbf{p}$ can give rise to the similar HOGs $\mathbf{x}$ [23]. An example of this is shown in Figure 1.

Thus, the true pose $\mathbf{p}$ can not be fully estimated from a single HOG $\mathbf{x}$ (using any regression or mapping method); additional information is needed. In the next section, we describe how temporal continuity assumptions can be employed to disambiguate the mapping from HOG to hand pose.

5) Time Continuity Enforcement in JOINT Space: We now describe how temporal smoothness in hand motion can be exploited to disambiguate the mapping $\mathcal{M}$.

Consider a sequence of hand poses $[\mathbf{p}_t], t = 1, \ldots, n$, that have given rise to a sequence of views, represented as HOGs $[\mathbf{x}_t], t = 1, \ldots, n$. Since the mapping $\mathcal{M}$ is ambiguous, the k nearest neighbors to $\mathbf{x}_t$ in the database, i.e. the members of the set X_k, are all similar to $\mathbf{x}_t$ but not necessarily corresponding to hand poses similar to p_t. An important implication of this is that a sequence of hand poses $[\mathbf{p}_t], t = 1, \ldots, n$ does not necessarily give rise to a sequence of HOGs $[\mathbf{x}_t], t = 1, \ldots, n$ continuous in the HOG space.

[1] www.grasp-project.eu.

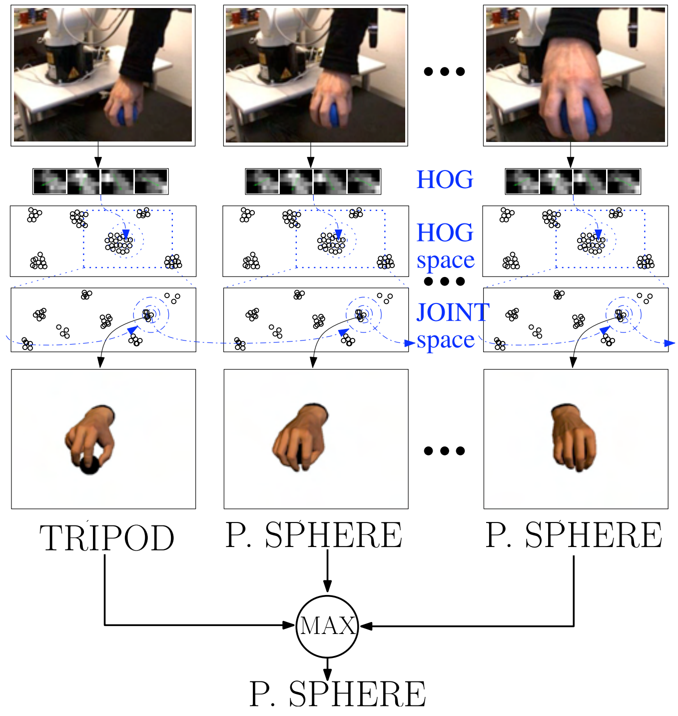


Fig. 2. Grasp Classification with continuity enforcement in JOINT space

However, due to the physics of the human body, the speed of the hand articulation change is limited. Thus, the sequence of hand poses $[\mathbf{p}_t], t = 1, \ldots, n$, i.e. the *hidden variables*, display a certain continuity in the JOINT space. This is illustrated in Figure 2.

The hand pose recognition for a certain frame t is therefore divided into two stages; I) retrieval of a set of k nearest neighbors X_k using single frame non-parametric mapping, as described in Section III-A.1; II) weighting of the members of X_k according to their time continuity in the JOINT space.

Let P_k be the set of poses corresponding to the kNN set X_k found in stage I). Moreover, let $\hat{\mathbf{p}}_{t-1}$ be the estimated pose in the previous time step. In stage II), the members $\mathbf{p}_j, j \in [1, k]$ of P_k are weighted as

$$\omega_j = e^{-\frac{\|\mathbf{p}_j - \hat{\mathbf{p}}_{t-1}\|}{2\sigma^2}} \ . \tag{2}$$

where σ^2 is the variance of the distance from each entry pose $\mathbf{p}_j$ to the previous estimated pose p_{t-1}.

The pose estimate at time t is computed as the weighted mean of P_k:

$$\hat{\mathbf{p}}_t = (\sum_{j=1}^{k} \omega_j \mathbf{p}_j)/(\sum_{j=1}^{k} \omega_j) \ . \tag{3}$$

The grasp class estimation G_t is obtained through a majority voting process within the N_p poses with the highest weight ω_j (for our experiments $N_p = 15$). G_t is then smoothed temporally taking the majority vote in a temporal window of N_f frames ($N_f = 10$ in our experiments). This can be seen in Figure 2. The whole system runs at 10 Hz on a 1.8 GHz single core CPU.

B. Object Recognition

For the robust recognition and accurate 6D pose estimation of single-colored objects, in our previous work, we have developed a model-based approach based on a combination of stereo triangulation, matching of global object views and online projection of a 3D model of the object [30]. The requirement for the approach is global segmentation of the objects, which is accomplished by color segmentation. For training, a 3D model of the object is used to generate views with different object orientations in simulation. Each view is stored along with its corresponding orientation. For recognition, each region candidate obtained by the segmentation routine is matched against the database. An initial orientation estimate is given by the stored orientation information with the matched view. An initial position estimate is given by the stereo triangulation result of the segmented regions in the left and right camera image. The triangulation result of the centroids depends on the view of the object and thus cannot serve as a constant reference point. In order to solve these problems, a pose correction algorithm is applied, which make use of online projection of the 3D model. This pose correction algorithm is an iterative procedure, which in each iteration corrects the position vector by computing the triangulation error in simulation and correcting the orientation estimate on the basis of the updated position estimate.

C. Markerless Motion Capture

In the following, our real-time stereo-based human motion capture system presented in [31] will be summarized briefly. The input to the system is a stereo color image sequence, captured with the built-in wide-angle stereo pair of the humanoid robot ARMAR-IIIb, which can be seen in Figure 5. The input images are preprocessed, generating output for an edge cue and a so-called distance cue, as introduced in [32]. The image processing pipeline for this purpose is illustrated in Figure 3. Based on the output of the image processing pipeline, a particle filter is used for tracking the movements in joint angle space. For tracking the movements, a 3D upper body model with 14 DoF (6 DoF for the base transformation, $2 \cdot 3$ for the shoulders, and $2 \cdot 1$ for the elbows) consisting of rigid body parts is used, which provides a simplified description of the kinematic structure of the human upper body. The model configuration is determined by the body properties like the limbs length of the observed human subject. The core of the particle filter is the likelihood function that evaluates how well a given model configuration matches the current observations, i.e. stereo image pair. For this purpose, an edge cue compares the projected model contours to the edges in the image. On the basis of an additional 3D hand/head tracker, the distance cue evaluates the distance between the measured positions and the corresponding positions inferred by the forward kinematics of the model. Various extensions are necessary for robust real-time application such as a prioritized fusion

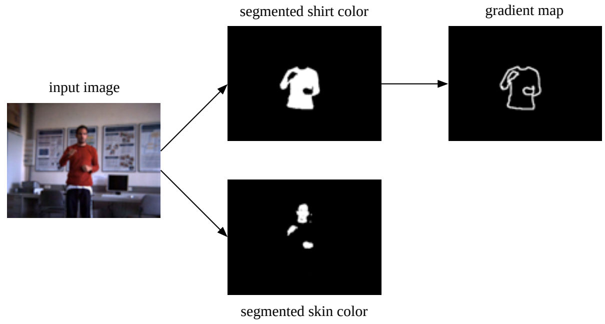


Fig. 3. Illustration of the image processing pipeline.

method, adaptive shoulder positions, and the incorporation of the solutions of the redundant arm kinematics. The system is capable of online tracking of upper body movements with a frame rate of 15 Hz on a 3 GHz single core CPU. Details are given in [31].

IV. Grasp Mapping

Before the execution on the robot, the approach movement in the form of joint angle configurations and the recognized grasp type are mapped onto the robot. In order to map motion onto the robot, we proposed in our previous work (see [33]) the Master Motor Map (MMM), a standardized interface which features a high level of flexibility and compatibility, since it allows mapping from various motion capture systems to different robot embodiments. The MMM provides a reference kinematic model of the human body by defining the maximum number of DoF, currently 58, that can be used by a human motion capture module and a robot. Trajectories in the MMM file format can be represented in joint angle space as well as in Cartesian space. Concerning movements in Cartesian space, in order to enable grasping and manipulation tasks, the MMM provides mapping of the desired 6D pose and the grasp type on the robot's end effector. A proper connection via the MMM of a motion capture module to a robot requires the implementation of a conversion module which transforms module specific data into the MMM file format and vice versa for overcoming different Euler conventions, active joint sets and orders of the joint angle values between the modules. As depicted in Figure 4, in the current system one conversion module has been implemented for each human motion capture system, converting the motion capture data to the MMM format. A third conversion module is implemented for mapping the MMM data to the kinematics of ARMAR-IIIb.

Along with the approach movement in the form of joint angle values the grasp type and the estimated hand orientation are passed from the hand pose estimation system to the robot through the MMM interface. According this data, from a set of preimplemented grasp the corresponding one is selected to be executed. To complete the grasp mapping, the grasp type to be performed is adjusted regarding the extent of the object shape. For this purpose, a rudimentary grasp type adjustment is implemented, which projects the object

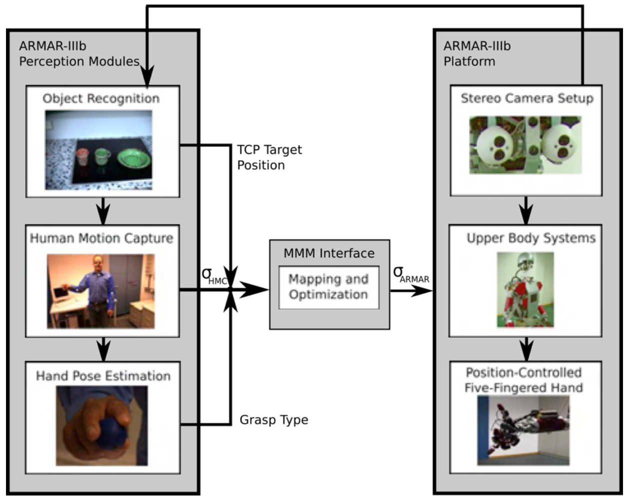


Fig. 4. Structure of the entire framework.

shape onto the thumbs position such that the thumbs tip lies on the shapes margin. The aperture of the fingers is scaled in a way that the positions of the remaining finger tips also approximately meet the margin of the shape. This method works on objects with simple shape properties.

A. Grasp Execution

The grasp reproduction of ARMAR-IIIb is performed in three different stages. The first stage describes the approach movement of the end effector towards the object based on the observed movement, while in the second stage the end effector is placed at the final grasp pose. The reproduction concludes with the execution of the recognized grasp type. Regarding the approach stage, by mapping these joint angle movements onto the robot, through forward kinematics one obtains a trajectory of the TCP in Cartesian space. The resulting trajectory is not sufficient for a goal-directed reproduction due to differences in the kinematic structure between the embodiments of the robot and a human e.g. mechanical joint constraints, differing joints and limb measurements. Therefore, the TCP trajectory for movements such as grasping is stretched and directed towards the object position to be reached. In order to attain a goal-directed reproduction, which additionally should feature a high similarity to the demonstrated human movement, in each frame, joint angles as well as desired TCP position of the modified trajectory have to be considered during execution. In [34], we developed an approach, which supports reproduction of observed human motion on the robot using non-linear optimization methods. In order to formulate an optimization problem which comprises displacements in Cartesian space regarding the TCP position as well as in joint angle space, a similarity measure is defined as follows:

$$S(\boldsymbol{\sigma}) = 2 - \frac{\frac{1}{n}\sum_{i=1}^{n}\left(\hat{\sigma_i}^t - \sigma_i\right)^2}{\pi^2} - \frac{\frac{1}{3}\sum_{k=1}^{3}\left(\hat{p_k}^t - p_k\right)^2}{\left(2 \cdot l_{arm}\right)^2} \quad (4)$$

with n representing the number of joints, $\sigma_i, \hat{\sigma_i}^t \in [0, \pi]$ and $p_k, \hat{p_k}^t \in [-l_{arm}, l_{arm}]$, whereas l_{arm} describes the robot's arm length. The reference joint angle configuration is denoted by $\hat{\boldsymbol{\sigma}} \in \mathbb{R}^n$, while $\hat{\mathbf{p}} \in \mathbb{R}^3$ stands for the desired TCP position. The current TCP position $\mathbf{p}$ can be determined by applying the forward kinematics of the robot to the joint angle configuration $\boldsymbol{\sigma}$. Based on Equation 4 and the joint constraints $\{(C_{min}, C_{max})\}$ of a robot with n joints, one obtains following constrained optimization problem:

$$\min S'(\boldsymbol{\sigma}) = 2 - S(\boldsymbol{\sigma}) \quad (5)$$

$$\text{subject to} \quad C_{i_{min}} \leq \hat{\sigma}_i \leq C_{i_{max}} \quad (6)$$

For solving Equation 5, we apply the Levenberg-Marquardt algorithm, since it features numerical stability and more robust convergence compared to other optimization algorithms such as the Gauss-Newton and the steepest descent method. Following this optimization approach a trade-off is attained, which on the one hand results in an accurate TCP positioning with small displacement error while it provides on the other hand a feasible robot joint angle configuration resembling the observed human configuration. This way goal-directed imitation of the approach movement is achieved. For further details, the reader is referred to [34]. For the execution of the final grasp phase, due to errors and inaccuracies originating from the object localization and the robot's mechanical elements, a displacement error arises between the TCP and the object that has to be diminished. To achieve exact alignment of the end effector and the robot, we make use of visual servoing methods as presented in [35]. Within this approach the hand and object are tracked. The resulting distance between both is reduced and the hand orientation is controlled. The hand orientation estimate coming from the grasp recognition module is used to determine if the grasp should be executed from the top or from the side. Therefore, the hand is placed over the object if the palm orientation was similar to the table plane, or next to the object otherwise.

V. Experiments

A. Experimental Setup

The humanoid platform ARMAR-IIIb, a copy the humanoid robot ARMAR-IIIa [36], serves as the experimental platform in this work. From the kinematics point of view, the robot consists of seven subsystems: head, left arm, right arm, left hand, right hand, torso, and a mobile platform. The head has seven DoF and is equipped with two eyes, which have a common tilt and independent pan. Each eye is equipped with two digital color cameras, one with a wide-angle lens for peripheral vision and one with a narrow-angle lens for foveal vision. The upper body of the robot provides 33 DoF: 2·7 DoF for the arms and three DoF for the torso. The arms are designed in an anthropomorphic way: three DoF for each shoulder, two DoF in each elbow and two DoF in each wrist. Each arm is equipped with a five-fingered hand with eight DoF. The locomotion of the robot is realized using a wheel-based holonomic platform.

The proposed approach was integrated on the humanoid platform ARMAR-IIIb and was successfully applied. For

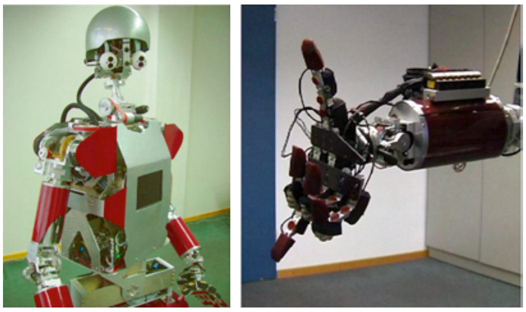


Fig. 5. Left: The humanoid robot ARMAR-IIIb. Right: Position-controlled right hand with 8 DoF.

the experiments, objects were used which can be easily identified such as single-colored cups. The experimental setup stipulates that demonstration of the grasp is performed in front of the robot. Observation is initiated by scanning the scene for known objects. Once an object is found, tracking of the human upper body is triggered leading to the capturing process of movements in the approach stage. This process is finished once the hand is positioned within a tolerated distance to a specific object. At this point, observation is switched to the hand pose estimation whereby its classification and the outcoming orientation complete the motion data of the grasp. As described in Section IV, the data is mapped onto robot, optimized to its embodiment and executed. In the execution phase, the robot searches for the same object which was grasped in the demonstration and approaches it. Based on the classification of the grasp type, an adequate instance is selected from the set of implemented grasp on the robot which is modified to the objects appearance. The hand pose recognition system was running on an external computer, while the rest of the system was running on ARMAR-IIIb. The communication between the two systems was performed through UDP sockets. It is possible to run the whole system on the robot, but this setup was more preferable for debugging purposes. Two sets of experiments were performed: in the first one, the whole system (grasp observation, mapping and execution) was tested with a reduced set of grasps: power grasp from top, power grasp from side, and pinch grasp(see Figure 6). In the second one, the set of grasps was extended to five of them (power sphere, prismatic wrap, parallel extension, tripod, and pinch). However, the execution of the grasp was reduced to the hand pose, keeping the arm still (see Figure 7).

B. Experimental Results

As depicted in Figures 6 and 7 the robot successfully imitated the demonstrated grasp including approach and grasp type. Since a non-linear optimization method is applied during approaching, we attained a trade-off between the similarity of the reproduced movement concerning the demonstration and accuracy in terms of positioning of the end effector regarding goal-directed tasks. Furthermore, the applied method provided a unique solution in terms of joint angles, which standard inverse kinematics methods fail to do due to singularities and redundancies. Nevertheless, in the approach phase, we experienced a displacement error of max $65mm$ caused by kinematic inaccuracies which varies depending on the cups distance regarding the end effector. The displacement could be recovered by using visual servoing. In order to test the grasp classification module, each grasp was executed 20 times for the Experiment 2. The results are shown in Table I. An overall classification accuracy of 72% was achieved, clearly over the human baseline for grasp recognition with similar grasps [21], with four out of five grasp types with accuracies over 80%. The differences between human model and synthetic had a stronger effect in the parallel extension grasp, lowering the accuracy for that particular grasp. Results of the grasp recognition, mapping and execution on the humanoids robot ARMAR-IIIb are shown in the accompanying video submission, which is also available under `wwwiaim.ira.uka.de/users/do/GraspRecognitionDivx.avi`.

Grasp Type	Illustration	Correct Classification Rate
Power Sphere		80 %
Prismatic Wrap		95 %
Parallel extension		50 %
Tripod		85 %
Pinch		80 %

TABLE I

GRASP TYPE CLASSIFICATION RESULTS.

VI. CONCLUSIONS

In this paper, we presented a system for grasp recognition, mapping and execution on a humanoid robot. Human grasping activities are captured using markerless motion capture system and mapped to the humanoid robot ARMAR-IIIb. Human upper body tracking, object tracking and hand pose estimation techniques are applied to perceive human object grasping movements. The recognized grasps are mapped and executed on a humanoid robot with a five-fingered hand.

VII. ACKNOWLEDGMENTS

The work described in this paper was partially conducted within the EU Cognitive Systems projects GRASP (FP7-215821) and PACO-PLUS (FP6-027657) funded by the European Commission.

REFERENCES

[1] C. G. Atkeson and S. Schaal, “Learning tasks from a single demonstration,” in *IEEE International Conference on Robotics and Automation (ICRA97)*, 1997, pp. 1706–1712.

[2] D. Gavrila and L. Davis, “3-D Model-based tracking of humans in action: a multi-view approach,” in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, USA, 1996, pp. 73–80.

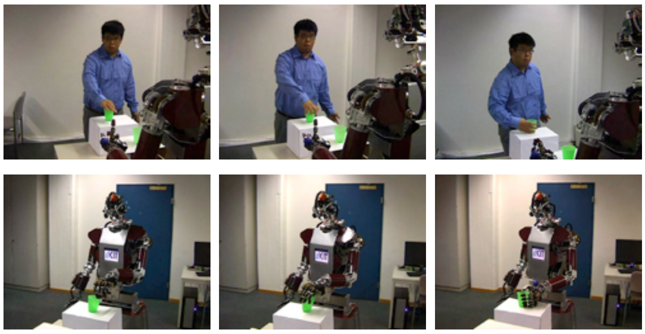


Fig. 6. Image Samples of Experiment 1. Top: human performing pinch grasp, top power grasp, and side power (left to right). Bottom: reproduction of the approach movement and the grasp type to grasp an object.

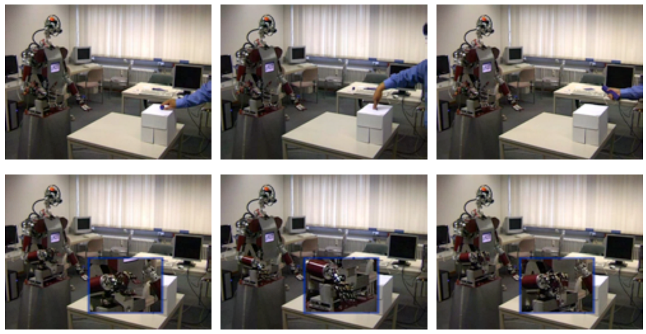


Fig. 7. Image Samples of Experiment 2. Top: human performing tripod grasp, parallel extended finger grasp, and prismatic wrap (left to right). Bottom: reproduction of the grasp type.

[3] K. Rohr, "Human Movement Analysis based on Explicit Motion Models," in *Motion-Based Recognition*, 1997, pp. 171–198.

[4] C. Bregler and J. Malik, "Tracking People with Twists and Exponential Maps," in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, Santa Barbara, USA, 1998, pp. 8–15.

[5] D. Grest, D. Herzog, and R. Koch, "Monocular Body Pose Estimation by Color Histograms and Point Tracking," in *DAGM-Symposium*, Berlin, Germany, 2006, pp. 576–586.

[6] J. Deutscher, A. Blake, and I. Reid, "Articulated Body Motion Capture by Annealed Particle Filtering," in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, Hilton Head, USA, 2000, pp. 2126–2133.

[7] C. Kim, D. Kim, and Y. Oh, "Adaption of Human Motion Capture Data to Humanoid Robots for Motion Imitation using Optimization," *Integrated Computer-Aided Engineering*, vol. 13, no. 4, pp. 377–389, 2006.

[8] A. Ude, C. Atkeson, and M. Riley, "Programming Full-Body Movements for Humanoid Robots by Observation," *Robotics and Autonomous Systems*, vol. 47, no. 2-3, pp. 93–108, 2004.

[9] X. Zhao, Q. Huang, Z. Peng, and K. Li, "Humanoid Kinematics Mapping and Similiarity Evaluation based on Human Motion Capture," in *IEEE International Conference on Information Acqusition*, Hefei, China, June 2004, pp. 426–431.

[10] N. Pollard, J. Hodgins, M. Riley, and C. Atkeson, "Adapting Human Motion for the Control of a Humanoid Robot," in *IEEE International Conference on Robotics and Automation*, Washington, DC, USA, May 2002, pp. 1390–1397.

[11] M. J. Matarić, "Sensory-motor primitives as a basis for imitation: Linking perception to action and biology to robotics," in *Imitation in Animals and Artifacts*, K. Dautenhahn and C. Nehaniv, Eds., 2000.

[12] E. Oztop, J. Babic, J. G. Hale, G. Cheng, and M. Kawato, "From biologically realistic imitation to robot teaching via human motor learning," in *ICONIP (2)*, vol. 4985, 2007, pp. 214–221.

[13] M. C. Lopes and J. S. Victor, "Motor representations for hand gesture recognition and imitation," in *IROS Workshop on Robotic Programming by Demonstration*, 2003.

[14] S. Calinon, A. Billard, and F. Guenter, "Discriminative and adaptative imitation in uni-manual and bi-manual tasks," in *Robotics and Autonomous Systems*, vol. 54, no. 5, 2005, pp. 370–384.

[15] A. Erol, G. N. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly, "A review on vision-based full DOF hand motion estimation," in *Vision for Human-Computer Interaction*, 2005, pp. III: 75–75.

[16] Q. Delamarre and O. D. Faugeras, "3D articulated models and multiview tracking with physical forces," *Computer Vision and Image Understanding*, vol. 81, no. 3, pp. 328–357, 2001.

[17] B. D. R. Stenger, A. Thayananthan, P. H. S. Torr, and R. Cipolla, "Filtering using a tree-based estimator," in *IEEE International Conference on Computer Vision*, 2003.

[18] E. Sudderth, M. I. Mandel, W. T. Freeman, and A. S. Willsky, "Visual hand tracking using non-parametric belief propagation," in *IEEE Workshop on Generative Model Based Vision*, 2004.

[19] M. de la Gorce, N. Paragios, and D. J. Fleet, "Model-based hand tracking with texture, shading and self-occlusions," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.

[20] R. Rosales, V. Athitsos, L. Sigal, and S. Sclaroff, "3D hand pose reconstruction using specialized mappings," in *IEEE International Conference on Computer Vision*, 2001, pp. I: 378–385.

[21] H. Kjellström, J. Romero, and D. Kragić, "Visual recognition of grasps for human-to-robot mapping," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008.

[22] T. Asfour, K. Welke, P. Azad, A. Ude, and R. Dillmann, "The karlsruhe humanoid head," in *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, Daejeon, Korea, December 2008.

[23] W. T. Freeman and M. Roth, "Orientational histograms for hand gesture recognition," in *IEEE International Conference on Automatic Face and Gesture Recognition*, 1995.

[24] G. Shakhnarovich, P. Viola, and T. Darrell, "Fast pose estimation with parameter sensitive hashing," in *IEEE International Conference on Computer Vision*, vol. 2, 2003, pp. 750–757.

[25] H. N. Zhou, D. J. Lin, and T. S. Huang, "Static hand gesture recognition based on local orientation histogram feature distribution model," in *Vision for Human-Computer Interaction*, 2004, p. 161.

[26] M. Cutkosky, "On grasp choice, grasp models and the design of hands for manufacturing tasks," *IEEE Transactions on Robotics and Automation*, vol. 5, no. 3, pp. 269–279, 1989.

[27] N. Kamakura, M. Matsuo, H. Ishi, F. Mitsuboshi, and Y. Miura, "Patterns of static prehension in normal hands," *Am J Occup Ther*, vol. 7, no. 34, pp. 437–45, 1980.

[28] S. B. Kang and K. Ikeuchi, "Grasp recognition using the contact web," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1992, pp. 194–201.

[29] A. Andoni and P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," *Communications of the ACM*, vol. 51, 2008.

[30] P. Azad, T. Asfour, and R. Dillmann, "Stereo-based 6d object localization for grasping with humanoid robot systems," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, 29 2007-Nov. 2 2007, pp. 919–924.

[31] P. Azad, T. Asfour, and R.Dillmann, "Robust real-time stereo-based markerless human motion capture," in *IEEE International Conference on Humanoid Robots*, Daejeon, Korea, December 2006, pp. 700–707.

[32] P. Azad, A. Ude, T. Asfour, and R. Dillmann, "Stereo-based Markerless Human Motion Capture for Humanoid Robot Systems," in *IEEE International Conference on Robotics and Automation (ICRA)*, Roma, Italy, 2007, pp. 3951–3956.

[33] P. Azad, T. Asfour, and R. Dillmann, "Toward an Unified Representation for Imitation of Human Motion on Humanoids," in *IEEE International Conference on Robotics and Automation*, Rome, Italy, April 2007.

[34] M. Do, P. Azad, T. Asfour, and R. Dillmann, "Imitation of Human Motion on a Humanoid Robot using Non-Linear Optimization," in *IEEE International Conference on Humanoid Robots*, Daejeon, Korea, December 2008, pp. 545–552.

[35] N. Vahrenkamp, S. Wieland, P. Azad, D. Gonzalez, T. Asfour, and R. Dillmann, "Visual servoing for humanoid grasping and manipulation tasks," in *Humanoid Robots, 2008. Humanoids 2008. 8th IEEE-RAS International Conference on*, Dec. 2008, pp. 406–412.

[36] T. Asfour, K. Regenstein, P. Azad, J. Schröder, N. Vahrenkamp, and R. Dillmann, "ARMAR-III: An Integrated Humanoid Platform for Sensory-Motor Control," in *IEEE/RAS International Conference on Humanoid Robots*, 2006.

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.

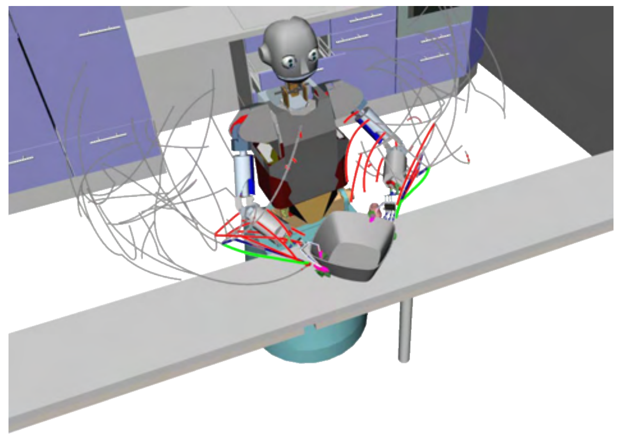


Fig. 1. A bimanual grasping trajectory.

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 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.

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

$RRT.AddConfiguration(q_{start})$;
while *(!TimeOut())* **do**
  ExtendRandomly(RRT);
  if $(rand() < p_{SearchGraspPose})$ **then**
    $n_{grasp} \leftarrow ApproachTrajectory(RRT, p_{obj})$;
    if $(ScoreGrasp(n_{grasp}) > score_{min})$ **then**
      return $BuildSolution(Grasp)$;
  end
end

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.

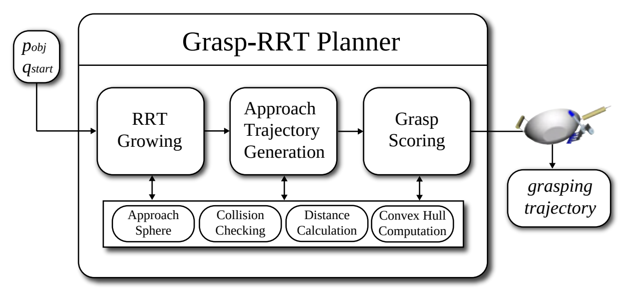


Fig. 2. Overview of the Grasp-RRT planner.

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).

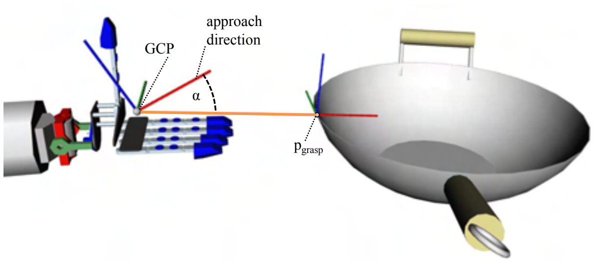


Fig. 3. The computation of the grasping pose p_{grasp}.

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.

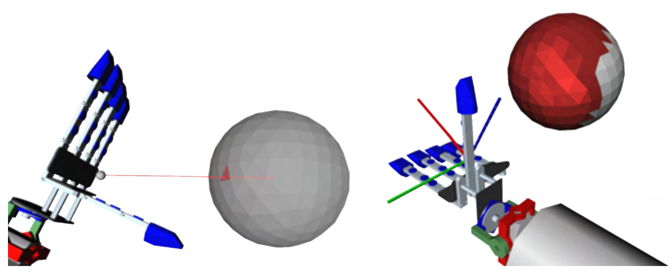


Fig. 4. (a) For each RRT-Node the corresponding triangle of the *ApproachSphere* is determined by projection the TCP position on the sphere's surface. (b) The distribution of approach directions is visualized by setting the color intensity proportional to the number of RRT-nodes in the direction of the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

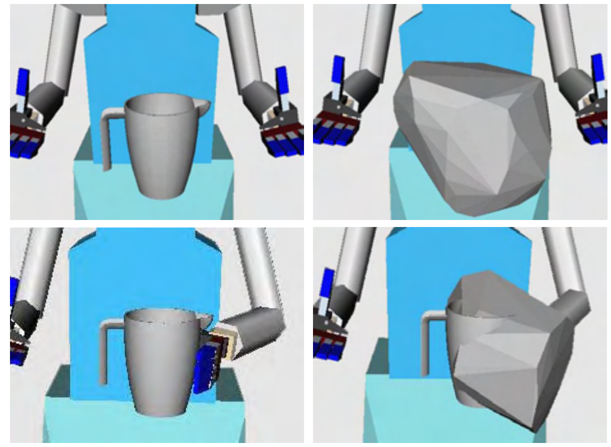


Fig. 5. Top Row: The object with a visualization of CH_o. Bottom Row: The grasp specific CH_g is used to compute the grasp quality score. For the measuring cup a grasp quality score of $q_g = 0.46$ is determined.

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).

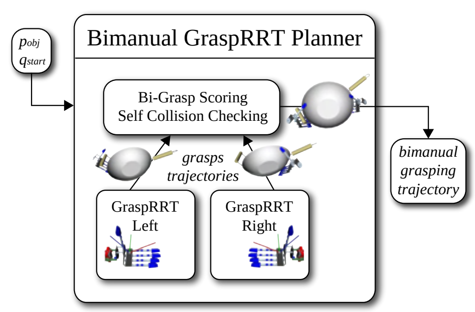


Fig. 6. Overview of the Bimanual Grasp-RRT Planner.

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

$GraspRRT_{left} \leftarrow GraspRRTInstance(q^{left}_{start}, p_{obj});$
$GraspRRT_{right} \leftarrow GraspRRTInstance(q^{right}_{start}, p_{obj});$
$GraspRRT_{left}.start();$
$GraspRRT_{right}.start();$
while *(!TimeOut())* **do**
    /* process new results of $GraspRRT_{left}$*/
    $s_l \leftarrow GraspRRT_{left}.GetNewSolution();$
    if (s_l) **then**
        $Results_{left}.add(s_l);$
        foreach $(s_r \in Results_{right})$ **do**
            if $(BiGraspScore(s_l, s_r) > score_{min}$ && $!SelfCollision(s_l, s_r))$ **then**
                $GraspRRT_{left}.stop();$
                $GraspRRT_{right}.stop();$
                return $BuildSolution(s_l, s_r);$
            end
        end
    end
    /* process new results of $GraspRRT_{right}$*/
    ...
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.

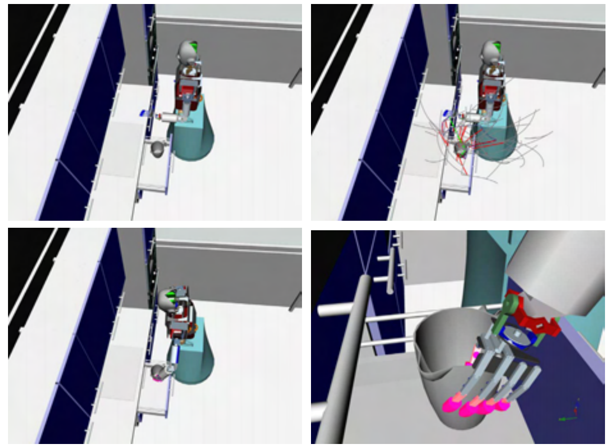


Fig. 7. The Grasp-RRT planner is used to search a feasible grasp together with a collision-free grasping trajectory for 10 DoF of ARMAR-III.

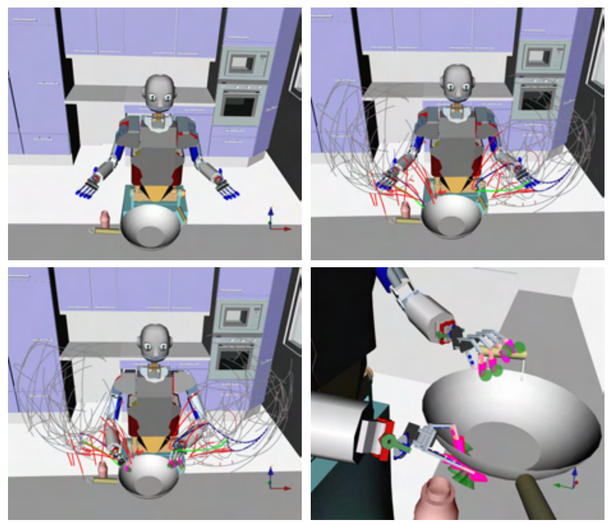


Fig. 8. The Bimanual Grasp-RRT planner is used to search a collision-free grasping trajectory for 14 DoF of both arms of ARMAR-III.

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.

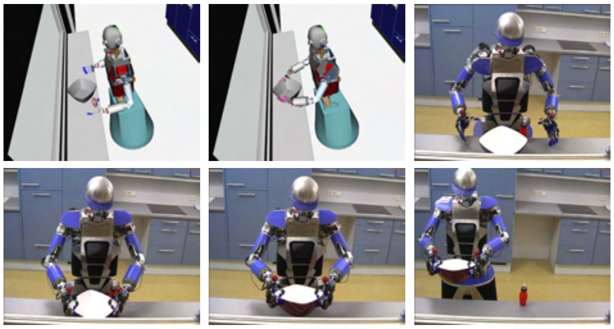


Fig. 9. The Bimanual Grasp-RRT enables the humanoid robot ARMAR-III to grasp a bowl in the kitchen.

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.

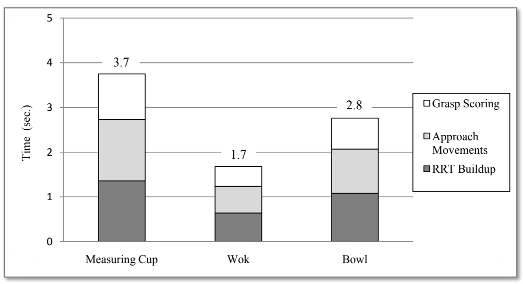


Fig. 10. Overview of the average performance measurements.

TABLE I
PERFORMANCE EVALUATION.

	Planning Time (seconds)				# Appr.	# Gr.
	Total	RRT	Approach	Score	Traj.	Scores
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.