Int J Soc Robot (2012) 4:437–448
DOI 10.1007/s12369-012-0162-y

Learning of Planning Models for Dexterous Manipulation Based on Human Demonstrations

Rainer Jäkel · Sven R. Schmidt-Rohr · Steffen W. Rühl · Alexander Kasper · Zhixing Xue · Rüdiger Dillmann

Accepted: 13 June 2012 / Published online: 29 June 2012

Abstract In the human environment service robots have to be able to manipulate autonomously a large variety of objects in a workspace restricted by collisions with obstacles, self-collisions and task constraints. Planning enables the robot system to generalize predefined or learned manipulation knowledge to new environments. For dexterous manipulation tasks the manual definition of planning models is time-consuming and error-prone. In this work, planning models for dexterous tasks are learned based on multiple human demonstrations using a general feature space including automatically generated contact constraints, which are automatically relaxed to consider the correspondence problem. In order to execute the learned planning model with different objects, the contact location is transformed to given object geometry using morphing. The initial, overspecialized planning model is generalized using a previously described, parallelized optimization algorithm with the goal to find a maximal subset of task constraints, which admits a solution to a set of test problems. Experiments on two different, dexterous tasks show the applicability of the learning approach to dexterous manipulation tasks.

Keywords Programming by demonstration · Motion planning · Machine learning

R. Jäkel (✉)
Institute for Anthropomatics, Karlsruhe Institute of Technology, Karlsruhe, Germany
e-mail: rainer.jaekel@kit.edu

S.R. Schmidt-Rohr · A. Kasper · R. Dillmann
Karlsruhe Institute of Technology, Karlsruhe, Germany

S.W. Rühl · Z. Xue
FZI Forschungszentrum Informatik, Haid-und-Neu-Str. 10-14, 76131 Karlsruhe, Germany

1 Introduction

In recent years, bimanual, dexterous service robots with multi-finger hands, e.g. Justin (DLR), Rosie (TUM), Adero (FZI) or Armar (KIT), have been developed to solve common manipulation tasks autonomously in the human environment. Since the environment changes constantly and multiple obstacles as well as self-collisions restrict the workspace of the robot, the successful execution of predefined robot trajectories isn't possible in general. More complex representations of low-level manipulation tasks have emerged, which allow to adapt to small, e.g. Gaussian Mixture Models (GMM), Dynamic Movement Primitives (DMP), or large changes in the environment, e.g. planning models (PM). The latter requires a sophisticated task model, which describes all relevant constraints of the task and enables the robot system to decide, if the task can be executed in the given configuration, how to plan a robot trajectory in a goal-directed way to achieve the goal of the task and how to monitor the execution and react to changes in the environment. In the context of dexterous manipulation, the task model consists of constraints restricting the motion of the robot's fingers and hands, the forces applied to an object and the object motion itself. The manual definition of such a task model is demanding and error-prone. One of the key ideas of Programming by Demonstration (PbD) is to exploit the domain knowledge of a human teacher, who is an expert in object manipulation, to choose instructional examples for a given task and to demonstrate different solutions to the robot in order to learn new manipulation tasks automatically. The online demonstration of finger motions including real force interaction with an object is unintuitive and complicated. Instead, the human teacher is observed using cameras, datagloves, motion trackers or tactile sensors while performing the task naturally with his own hands.

The differences between the robot and the human, e.g. in geometry, kinematics and dynamics, have to be considered explicitly, which is known as the *correspondence problem*. Consequently, a direct mapping of the human trajectories to the robot system will fail and the automatic deduction of manipulation goals and the underlying task model becomes most important. Based on the learned definition of the task, search-based AI methods, e.g. constrained motion planning, can be used to generate robot trajectories to accomplish the task, which offers flexibility to large changes in the environment at the cost of higher execution times. Since the goals, preconditions and constraints of the task are learned explicitly, the execution can be monitored, which is fundamental to the execution on an autonomous robot system.

In this work, we learn planning models for dexterous manipulation tasks based on explicit demonstrations by a human teacher. The search space defined by the task constraints is automatically enlarged to consider the correspondence problem. The initial planning model is overspecialized, i.e. it contains a large number of irrelevant task constraints. We use a previously described, semi-automatic optimization algorithm to reduce the number of task constraints based on information about inconsistency of constraints during constrained motion planning and to improve generalization to different environments. In the planning process, physics simulation is used to simulate non-rigid object contacts and to plan object and robot motions in a goal-directed way, e.g. to push a slider on a straight line to the goal position. We demonstrate the validity of the approach on two dexterous manipulation tasks.

2 Related Work

In literature, PbD is applied to tasks of varying complexity, e.g. setting a table [21] or playing pool [23]. In most approaches, a set of human demonstrations is abstracted to a more general trajectory representation, e.g. Dynamic Movement Primitives [12], which describe a motion as a set of differential equations, or Gaussian Mixture Models [5], which represent motions as probability density functions on the feature space. In the examples the feature vectors are defined manually, e.g. in the pool playing task, the feature vector contains the offset of the pushing hand to the bridge, the value of the redundant joint of the pushing arm and the orientation of the cue around the bridge.

For household tasks, manual definition of the learning features is also predominant, e.g. performing a chess move [6] or pouring in [22]. Due to the large variety of objects and object arrangements, generalization of the learned manipulation knowledge to new environments is important. If the reproduction of the learned knowledge is goal-directed [8], a potentially higher generalization capability can be reached since only the effects of the task will be reproduced. The definition of the feature space has a large influence on the generalization since it is the basis of the task description (including goals). In probabilistic approaches, the set of learning features, e.g. the position of the left hand relative to the object, is predefined and each feature is weighted based on the variance in the training set [5]. If a feature refers to an object or the environment, it has to exist in the new environment to generalize, too, which shows the importance and influence of the choice of the feature space In order to learn new tasks on an autonomous service robot it is necessary to deduce a set of relevant features automatically. Mühlig et al. introduce the concept of a task space pool [20], which contains a set of predefined features. In order to choose a set of features, a variance-based, an attention-based and a kinematic selection criterion are defined. The criteria can be used to weight features and, what is stated explicitly, to select a subset of features. The criteria were applied to a single example but the selection of features and the integration into a complete learning system weren't presented. By gazing [4] or pointing [3] at important objects the teacher can also clarify ambiguities. Due to the interaction with the human teacher, a small set of features can only be considered and semantics have to be assigned to each feature.

Mapping human actions to symbolic actions, e.g. [28], leads to a higher generalization capability but the problem of symbol grounding, i.e. how to generate a robot motion based on symbols like *near* or *aligned*, exists and remains unsolved. Kuniyoshi et al. apply classifiers for a small set of predefined actions to a set of human demonstrations [19]. Generalization to different poses is possible due to the use of relative coordinates but ambiguities and inconsistencies are not handled explicitly. Repetitions [29], branches [9] or dependencies [21] can be detected. In addition to motion observation, speech and interaction with the robot can be used to learn more complex tasks, e.g. [24].

In [26], the dexterous task of removing a lid from a glass jar was partially learned using bio-inspired approaches and a predefined feature space. Heuristics for approaching and grasping the lid were applied. The screwing motion was generated based on the observation of the human hand with a dataglove. The correspondence problem was considered by letting the teacher adapt the screwing motion until the mapping to a simulated hand was adequate. Since forces weren't considered in the learning process, the radius of the lid was underestimated so it could be rotated successfully.

3 Overview

We represent manipulation knowledge as a planning model for constrained motion planning, which is called a *manipulation strategy*. In Fig. 1, an example of a manipulation

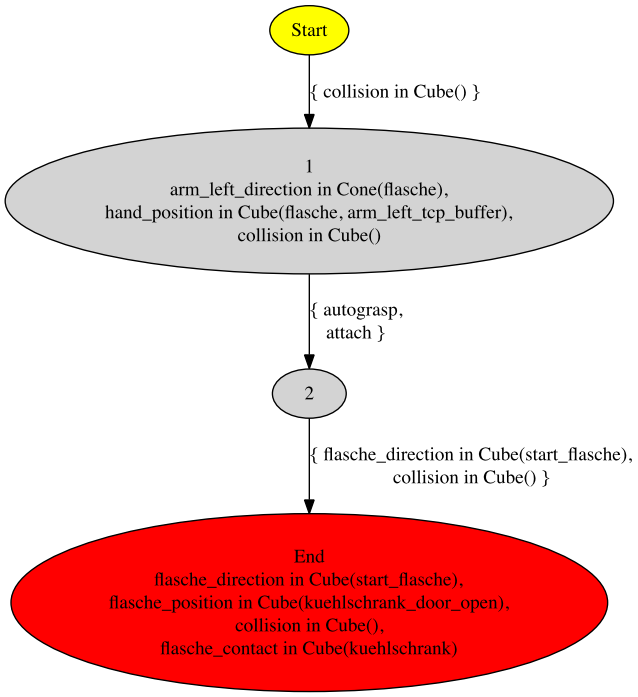


Fig. 1 Planning model to grasp a bottle and place it in a fridge door, [15]

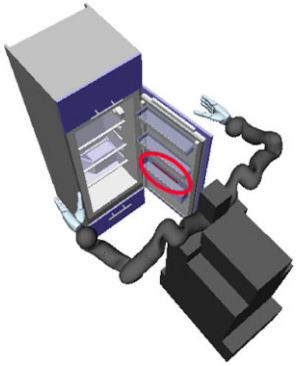


Fig. 2 Visualization of goal position constraint restricting the bottle position relative to the fridge door, [15]

Fig. 3 Sampled goal configurations consistent with the goal position constraint in Fig. 2

strategy is shown. A bottle has to be grasped and placed in a fridge door. The task constraints in the node *End* restrict, how the bottle has to be placed in the fridge door. The position constraint restricts the position of the bottle relative to the fridge door, see Fig. 2. The direction constraints restricts the bottle to stay upright. The contact constraint will be obeyed if the bottle has a contact with the fridge door. If the learned manipulation is executed by the robot, goal configurations, for which all constraints are obeyed, are generated online and automatically in a constrained motion planning process, see Fig. 3. The bottle can be placed at arbitrary door angles in the fridge door, even if obstacles are present in the shelf, resulting in a high flexibility of the robot.

In Sect. 4, manipulation strategies are defined, extending the definitions presented in [15] to include force, contact and object constraints. The extended set of constraints allows to restrict the motion of objects, which are not rigidly attached to the robot. The manipulation strategy is learned automatically based on the observation of a human teacher, see Sect. 5, with focus on hand observation using a precise hand model. In Sect. 6, the learning process in [14] is summarized, in which an automatically generated feature space consisting of a large number of task constraints is used. Additionally, we consider the correspondence problem explicitly by relaxing learned task constraints for the human hand and fingers. In order to generalize the learned manipulation strategy for dexterous manipulation tasks, the generalization algorithm in [15] was used, which is summarized in Sect. 7. In Sect. 8, we describe the mapping of automatically generated contact frames and the RRT-based motion planner. Experiments using two dexterous manipulation tasks are discussed in Sect. 9.

4 Planning Models

In the human environment, service robots have to be able to manipulate objects in a restricted workspace with multiple obstacles and to generalize to new objects and environments. In order to achieve this kind of flexibility a sophisticated planning model is required, which contains a minimal set of task constraints to describe the goals and execution of a task. In this work, a planning model is considered a task model suitable for constrained motion planning.

4.1 Strategy Graph

The planning model can be visualized as a directed graph, which is called *strategy graph*. In the graph, nodes describe (sub-)goals of the task and arcs describe the transition between (sub-)goals. Each node and arc is described by a temporal constraint and a conjunction of task constraints. In the simplest case, the planning model contains only a linear sequence of nodes, each connected by a single arc. In the execution, a constrained motion planning process is started for each arc in the planning model. The arc constraints describe

the search space for motion planning, while the node constraints define the search space for valid goal configurations of the robot system. A strategy graph defines formally a temporal constraint satisfaction problem with domain constraints, see [13], where temporal constraints are described by 1-dim. closed intervals.

4.2 Task Constraints

We define four different types of task constraints: position, orientation, configuration and contact constraints.

Position and orientation constraints are written as $(a, b, \mathcal{F}(a, b))$ and restrict a (coordinate) frame a to stay in a volume $\mathcal{F}(a, b)$ relative to a frame b. Let ${}^bT_a$ describe the pose of frame a relative to frame b. The transformation is described by a 4×4 homogenous transformation matrix using the 3×3 rotation matrix ${}^bR_a$ and the 3×1 vector ${}^bt_a$:

$$ {}^bT_a = \begin{pmatrix} {}^bR_a & {}^bt_a \\ \mathbf{0} & 1 \end{pmatrix} $$

In order to test, if the constraint is obeyed, we transform the restricted frame a with pose ${}^0T_a$ into the reference frame b with pose ${}^0T_b$. For the position constraint in the manipulation strategy in Fig. 1, a is the coordinate frame in the bottom of the bottle and b is the coordinate frame located in the fridge door. The resulting pose ${}^bT_a$ represents the relative transformation. By imposing a constraint on the relative transformation, the motion of a is restricted relative to b, e.g. the goal position of the bottle will automatically adapt to changes in the door angle. We map the relative transformation onto a 3d-vector depending on the type of the constraint. For position constraints, the 3d-vector is ${}^bt_a$. For orientation constraints, the 3d-vector is the scaled-axis representation[1] ${}^br_a$ of ${}^bR_a$. Finally, the constraint will hold, if and only if the 3d-vector is included in the region $\mathcal{F}(a, b)$, e.g. a simple cube or a tube defined by a Gaussian Mixture Model (GMM).

Contact constraints, which are also written as $(a, b, \mathcal{F}(a, b))$, restrict the 3d model assigned to a to stay in contact with the 3d model assigned to b. The contact normal, pointing from b to a, has to be included in $\mathcal{F}(a, b)$. In Fig. 1, the contact constraint describes, that the 3d model assigned to the bottle has to stay in contact with the 3d model assigned to the fridge door, i.e. the bottle has to touch the floor in the fridge door.

A configuration constraint is written as $(\theta_1, .., \theta_n, \mathcal{F}(\theta_1, .., \theta_n))$, where θ_i represents a degree of freedom (dof) of the robot and $\mathcal{F}(\theta_1, .., \theta_n)$ is a n-dim. hypercube. It will hold if and only if the n-dim. vector $(\theta_1, .., \theta_n)^T$ is included in $\mathcal{F}(\theta_1, .., \theta_n)$.

[1]By using the quaternion representation, a rotation matrix can be represented as a single rotation around a fixed axis with a fixed angle. The scaled axis representation is the product of this axis and angle.

Fig. 4 Sensory environment front (*left*) and back (*right*): datagloves, motion tracker and 3d visualization. Human demonstrations to lift a spoon

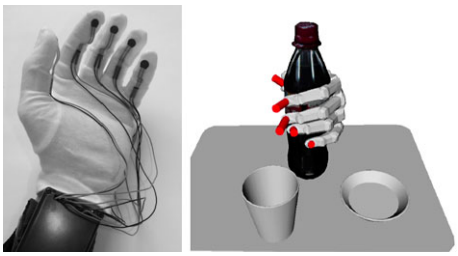


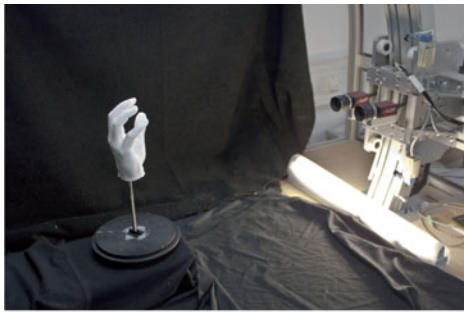
Fig. 5 Tactile glove prototype (*left*) and visualization of force measurements (*right*)

5 Observation

The human demonstration of dexterous manipulation tasks takes place in a dedicated sensory environment, see Fig. 4, to overcome the limitations of state-of-the-art vision based sensor systems onboard a robot, which doesn't allow robust fingertip and contact measurements. We use four different sensors: a stereo camera system with DragonFly II cameras, two Fastrak motion trackers, two Cyberglove II datagloves and a custom-built glove with tactile sensors. The sensor frequency is 25 Hz.

Objects are localized based on SIFT-features using the vision library IVT [1], which generates a 6d-pose for each object. The forces in the fingertips of the human hand are measured using force sensing resistors (FSR), see Fig. 5, resulting in 10 real-valued force values.

We measure the 6d-pose of both human wrists using the Fastrak motion tracker, which is calibrated to a small area above the table. 20 degrees of freedom of the human hand, four for each finger, are measured using a Cyberglove II. In order to observe dexterous manipulation tasks and to compute contacts and force directions, the Cartesian position of the finger parts is necessary. Since a direct measurement is not available, a precise 3d model of the author's hand was generated. First, we generated a plaster cast of the human hand using a Creaform reproduction set. The model was cleaned and mounted on a rotary table. We used a high-accuracy laser scanner, Minolta VI-900, to generate 3d points on the surface of the hand from different views. The points were registered and the resulting mesh cleaned up using the Rapidform library. Finally, we imported the mesh into the 3d modeling tool Blender to manually set up bones and joints. The process is visualized in Fig. 6.



(a) Plaster cast of human hand in laserscanner

(b) 3d mesh of the human hand

(c) Manually added bone structure

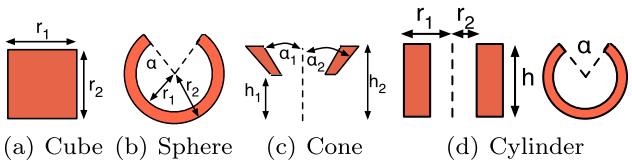
Fig. 6 Generation of 3d human hand model for the observation of dexterous manipulation tasks

The observation result is visualized in a 3d simulation during teaching. Based on the 3d models, the distance between objects and the hands is calculated and contacts will be added as additional measurements if the distance is closer than 5 mm.

6 Learning and Correspondence Problem

In the learning process, we synthesize the initial planning model based on the observation of the human demonstrations. The approach in [14] was adapted to learn force, contact and object constraints. The structure of the planning model, i.e. the number of subgoals and their connections, is generated based on the segmentation of the human demonstrations similar to [10]. In the segmentation algorithm, we generate a new subgoal if the velocities of finger joints, fingertip forces and human wrists is below a threshold using a simple hysteresis model. In order to ensure that all learning examples have the same number of segments, we take the subset, which has the largest number of learning examples with the same number of segments. Since the segmentation concept is easy to communicate only a small number is rejected in practice.

Similar to task space pools [20], in which a general feature space is used, we generate a large set of task constraints automatically for each segment, i.e. each node and



(a) Cube (b) Sphere (c) Cone (d) Cylinder

Fig. 7 Constraint volume types, [14]

arc, based on combinations of coordinate frames. The approach in [14], in which predefined coordinate frames of the finger tips, wrists of the human hands and objects in KIT object database [16] are used, was extended to consider automatically generated *contact frames*, i.e. coordinate frames in contact points on the objects. For each contact between two 3d models with coordinate frames a and b in the center of mass (com), which was measured consistently in 95 % of the segment's points, we generate the contact frames $a \diamond b$ and $b \diamond a$. $a \diamond b$ is relative to b. The position is the point with smallest distance on the 3d mesh assigned to a relative to the 3d mesh assigned to b. The orientation is the same as a. $b \diamond a$ in the same way.

For each segment, the set of coordinate frames consists of the human fingertips, human wrists, predefined coordinate frames of the detected objects and the automatically generated contact frames. We create an extended set with one additional *start* frame for each coordinate frame, which represents the value of the frame at the beginning of the segment.

Finally, the initial set of task constraints contains a position and orientation constraint $(a, b, \mathcal{F}_{ab})$ for each combination with a in the set of coordinate frames, b in the extended set and each type of region $\mathcal{F}_{ab}$. In the experiments, $\mathcal{F}_{ab}$ is a cubic, spherical, cylindrical or cone cut, see Fig. 7. Geometric primitives offer fast learning, fast constraint evaluation and sampling of a configuration, in which the constraint is obeyed. We calculate the parameters of each region using Rosenbrock optimization to determine the smallest region, in which the values of a relative to b are included using all points of all demonstrations assigned to the current segment. The resulting task constraint represents a volumetric approximation of the segment of demonstrated trajectories and serves as one of the atomic elements of the representation of the manipulation task.

6.1 Constraint Relaxation

In this work, we are interested in the effects of the hand and finger motions on objects and not the motion itself. The effects are encapsulated in contact constraints and position and orientation constraints restricting the motion of coordinate frames assigned to objects relative to each other. The hand and finger motions are encapsulated in task constraints, which restrict the motion of a coordinate frame

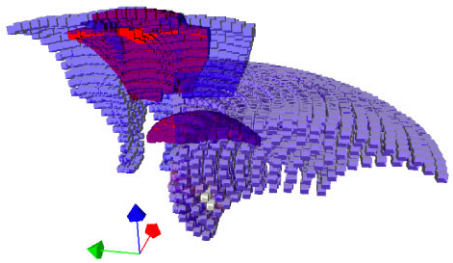


Fig. 8 Comparison of workspace of the SAH (*blue*) with a part of the workspace of the human hand (*red*)

assigned to the hand or fingers. Although the used robot hands are anthropomorphic, differences in geometry, kinematics and dynamics exist, the so called *correspondence problem*. Similar to the workspace analysis in [11], we compared the workspace of the Schunk Anthropomorphic Hand (SAH) with the human hand. The human teacher demonstrated 100 finger motions in different, dexterous manipulation tasks, e.g. screwing, inhand-manipulation, pushing. Based on this data, the minimal and maximal value for each finger joint was determined. We transformed the resulting hypercube into the Cartesian space using forward kinematics. The result is shown in Fig. 8. We applied Rosenbrock optimization to determine a 3d-translation (x, y, z) and 1d-rotation (y), which aligns the transformed hypercube with the SAH workspace. The objective function is the sum of four weighted distance values. For each finger, the distance value is the average distance of the set of points in the hypercube, which are not included in the SAH workspace, to the closest point in the SAH workspace. We assigned the weight 0.5 to the thumb and $0.1\overline{6}$ to the other fingers to reflect the important role of the thumb. The final result was 2.3 mm for the thumb, 7.3 mm for the point, 7.0 mm for the middle and 15.9 mm for the ring finger.

We enlarge each region R of a position constraint restricting the motion of a finger coordinate frame using the determined distances. For the human wrist, the distances were determined experimentally. Position constraints, which restrict the wrist coordinate frame, are enlarged by 40 mm and orientation constraints by 15°.

7 Generalization

The learned planning model is overspecialized due to the automatic generation of task constraints. In Fig. 9 a simple, overspecialized version of the manipulation strategy in Fig. 1 is shown. The position of the bottle is not only constrained relatively to the fridge door but also relative to the shelfs in the fridge by three additional constraints. If the set of human demonstrations contains only a small range of door angles, the learned position constraints won't overlap

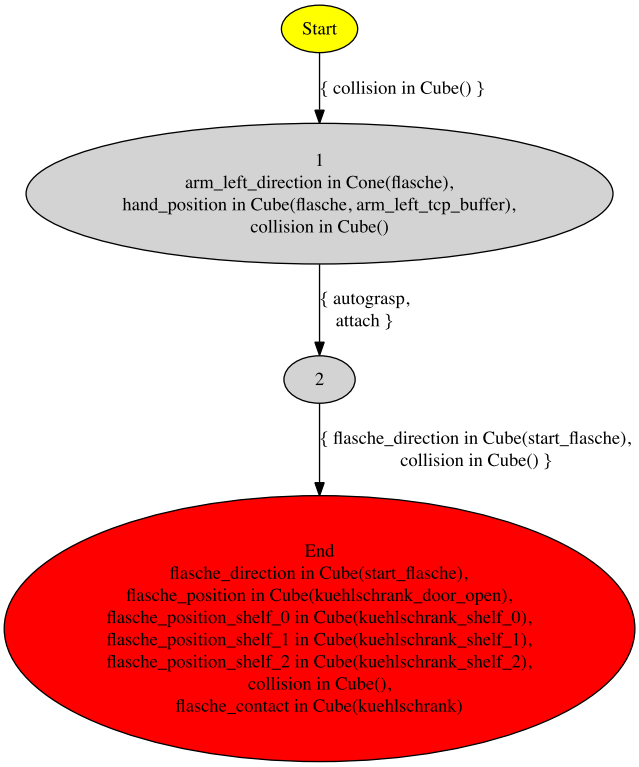
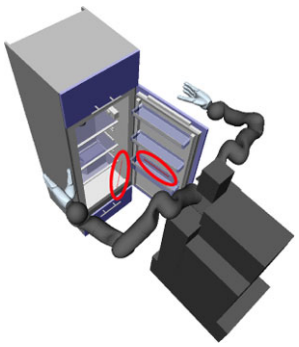


Fig. 9 Planning model to grasp a bottle and place it in a fridge door, [15]

Fig. 10 Non overlapping constraints (*circles*) for a different door angle in the fridge experiment, [15]



at a different door angle, see Fig. 10, and the manipulation strategy can't be planned successfully.

In general, a large number of human demonstrations is necessary to achieve a sufficient generalization, e.g. in this case with multiple door angles and multiple placement positions and orientations for each door angle.

We follow the approach in [15] to generalize learned manipulation strategies by extracting a maximal set of task constraints, which are relevant to the task. The robot has to solve example problems, e.g. picked by a human teacher or encountered during online execution. If the robot is not able to solve the problem, an optimization process is started to determine the planning model with a maximal subset of task constraints, which admit a successful solution to the set of test problems. In Fig. 11, a set of test problems for the

Fig. 11 Test: different door angles, [15]

fridge-task is shown. The robot has to successfully place a grasped bottle in the fridge door at different door angles.

7.1 Robot Tests

In general, the number of human demonstrations is limited and it can't be assumed, that the planning model is generalized sufficiently after the learning step. Since we expect a service robot to operate autonomously in the human environment with a large variety of objects, situations, in which a planning model can't be executed due to violated but irrelevant task constraints, might occur.

We apply the supervised generalization algorithm from [15] to reduce the number of constraints. The algorithm generates multiple hypotheses about which task constraints should be included in the planning model. For each hypothesis a constrained motion planning process is started to generate statistics about inconsistencies of task constraints. A mutation operator adapts iteratively each hypothesis and removes task constraints based on the statistics about inconsistencies. The goal of the optimization algorithm is to find a planning model consisting of a maximal subset of the task constraints, which can be used to solve the robot test.

In the PbD system, the human teacher generates tests by placing a set of objects in the sensory environment and defining a mapping of the localized objects to the objects in the manipulation strategy. The object poses are generated automatically using stereo vision-based object localization.

In order to generalize the planning model, an evolutionary algorithm [7] using the evolving objects library [17] is used to determine a maximal subset of task constraints, which can be included in the planning model to be able to solve the robot test. The manipulation strategy is mapped to a binary vector, where each bit corresponds to a task constraint. For each state, a task constraint will be considered in the planning process if and only if the corresponding bit is set.

In Fig. 12(a), the state representing the planning model in Fig. 9 is visualized. The first bit represents the position constraint *flasche_position* in the node *End*. The last bit corresponds to the direction constraint on arc $2 \mapsto End$. The

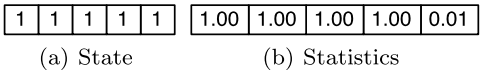


Fig. 12 5-bit state (**a**) and statistics about inconsistency of the constraints (**b**), here position constraints are inconsistent

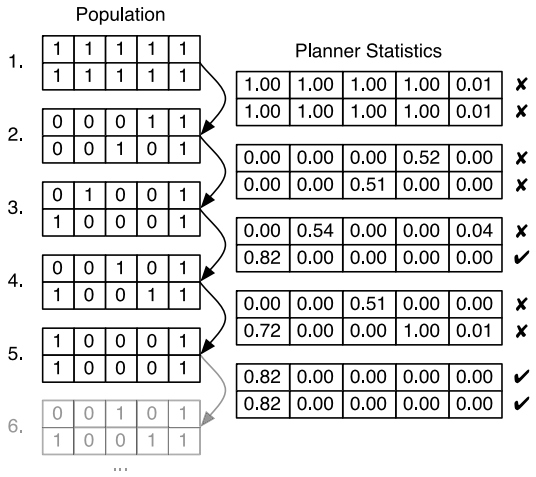


Fig. 13 Example of the evolutionary algorithm applied to the example in Fig. 9 using 2 states, [15]

remaining bits correspond to the *flasche_position_shelf_** constraints in the node *End*, which restrict the bottle relative to the shelves.

For a given state, statistics about inconsistency of constraints are gathered. In the planning process, projection is used to generate goal configurations for which all constraints are obeyed, and to project a random configuration to the nearest configuration on the constraint manifold. The percentage of failed projections is calculated for each constraint, see Fig. 12(b).

The goal is to find a maximal subset of task constraints, which can be included in the planning model to be able to solve the robot test. The objective function for the optimization process is the sum of a factor proportional to the number of 1-bits, the number of subgoals, which could be planned successfully, and the overall result of the planning process, i.e. 0 or 1. States or subsets of task constraints will be rated high if the progress in the planning process was high, a solution was found and the number of constraints is high.

Based on the statistics obtained in the planning process of a given state the state is mutated. If the planning process failed, a constraint with bit set to 1 will be deactivated randomly proportional to the percentage of failed projections. If the planning process was successful, a random bit will be set to 1. The algorithm will be stopped if the best state doesn't change in 20 iterations or the time threshold of 4 h is reached.

In Fig. 13, an example using the planning model in Fig. 9 and 2 individuals is shown. Based on the planner statistics,

the individuals are mutated until in the third step, a valid solution is found. In the fourth step, a constraint is added to the best state in order to maximize the number of constraints. Planning fails and the constraint is removed in the fifth step.

8 Frame Mapping and Constrained Motion Planning

We execute the learned, generalized planning model on the robot using a bidirectional RRT planner with Randomized Gradient Descenet (RGD) based on CBiRRT [2]. The goal is to generate a robot motion, in which all constraints are obeyed and all (sub-)goals are reached. Goal configurations are generated by sampling the task constraints of a given node and using inverse kinematics to generate a configuration. The resulting configuration is projected on the constraint manifold defined by the goal task constraints. More information can be found in [15].

8.1 Contact Projection

Given a set of contact constraints, e.g. between two fingers and an object, a configuration has to be found, in which all contacts are established. Due to the additional constraints induced by the articulated model of the robot, RGD is insufficient.

In proximity query packages, e.g. C2A [27], contacts can be enforced for articulated bodies starting in a collision-free configuration, e.g. closing the fingers until contact, using Continuous Collision Detection (CCD). The work of Min Tan using penetration depth (PD) calculations has been integrated, which allows to find a configuration with minimal distance to the start configuration, in which all collisions were removed. Contact projection is applied after RGD. First, the PD algorithm is applied to remove all collisions. Second, force directions are sampled from the force constraints and CCD is applied to move the object or articulated body in the direction of the force until contact. Finally, it is checked if all task constraints are obeyed.

8.2 Mapping of Coordinate Frames

For objects in the KIT ObjectModels Database, coordinate frames are predefined and a label was assigned, e.g. *opening*, *center of mass* or *bottom*. If the learned manipulation strategy has to be executed with a different object, the coordinate frames are mapped to the coordinate frame of the new object with the same label. For automatically generated contact frames, an automatic procedure is necessary.

We map contact frames to a new object based on a surface mapping of the original 3d model to the 3d model of the new object. Based on the surface mapping, the original 3d model can be morphed into the new 3d model. The surface mapping is calculated using Blended Intrinsic Maps [18]. In Fig. 14, the mapping of a small spoon to a large spoon of the same set of silverware is shown, where a simple scaling was not sufficient. The contact frame is mapped by searching the face of the first 3d model, on which the origin is placed. The position on the face is calculated using barycentric coordinates. The rotation matrix to rotate the frame into the local coordinate frame of the face is calculated. Based on the surface mapping, the corresponding face on the new 3d model is calculated. By using the barycentric coordinates, the position of the mapped contact frame is calculated. The orientation is calculated by applying the rotation matrix to the coordinate frame of the face.

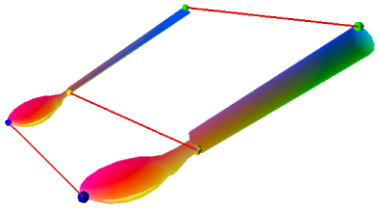


Fig. 14 Surface mapping of small spoon and large spoon of the same set of silverware. A simple scaling is not sufficient

8.3 Physics Simulation

Object constraints restrict the motion of an object, which is not rigidly attached to the robot. Object constraints play a crucial role to learn the effects of a motion of the fingers or robot hand, e.g. that a bottle cap will rotate if forces are applied at certain positions on its surface. In order to plan manipulation motions in a goal-directed way, we use physics simulation to generate the object motion. In the current system, physics simulation is applied to the generated, smoothed planning result to validate all object constraints. The planner will be restarted if an object constraint was violated. Since the learned task constraints restrict the search space in a very efficient way, valid solutions can be found efficiently in these simple tasks. In future, the physics simulation will be integrated fully into the RRT planner.

9 Experiments

We evaluated the developed PbD framework on two manipulation tasks on the bimanual robot Adero with 40 dofs: 7 for each KUKA Lightweight Arm and 13 for each Schunk Anthropomorphic Hand. All 3d object models can be found in the KIT ObjectModels Web Database [16].

9.1 Spoon Lifting and Grasping

The full PbD system was evaluated on this task, i.e. learning of contact constraints, generalization using robot tests,

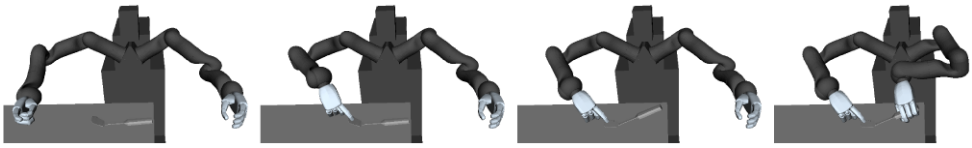


Fig. 15 Lifting a spoon: planning result for learned manipulation strategy for lifting a spoon using physics simulation

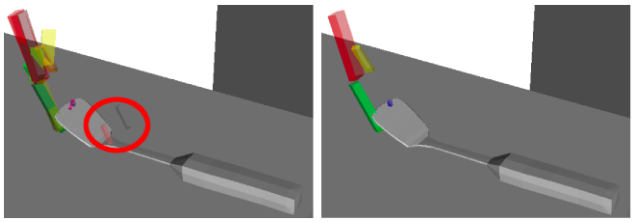


Fig. 16 Lifting a spoon: learned task constraints before (*left*) and after (*right*) generalization. Highlighted is a set of inconsistent constraints, which was removed

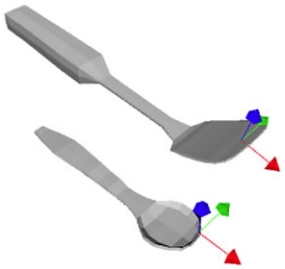


Fig. 17 Lifting a spoon: frame mapping

Table 1 Lifting a spoon/Opening a bottle: planning results

Manipulation strategy	Success (%)	Planning time (s)
Tipping on the spoon	85	11.2
Grasping the spoon	79	3.4
Tipping and grasping	67	14.5
Opening a bottle	84	35.6

mapping of frames using Blended Intrinsic Maps, planning using physics simulation and execution on the real Adero robot system. In this task, the human teacher pushed on the front part of the spoon to lift the handle and grasp it.

The initial manipulation strategy contained 140 and the generalized manipulation strategy 55 task constraints. Cubic regions were used. In Fig. 16 the set of task constraints is visualized. The highlighted constraints restrict the motion of the spoon, when the handle moves up. Due to the different geometry, both constraints were inconsistent and removed. The automatically extracted goal of the task can be described as: the contact frame on the spoon is slightly rotated, a contact between the spoon and the table and a contact between the spoon and the finger exists.

Table 2 Lifting a spoon: generalization results

Path	Obstacles	Success (%)	Planning time (s)
1	no	88	4.73
2	no	84	0.855
3	no	80	3.71
4	no	87	2.26
1	yes	64	5.05
2	yes	55	1.07
3	yes	62	4.08
4	yes	63	2.42

Two contact frames with origins $(110.65, 10.81, 7.16)$ and $(111.25, 9.38, 7.23)$ relative to the center of mass were generated. In the execution environment, a large kitchen tool was used instead of a spoon. The contact frames were mapped using Blended Intrinsic Maps, see Fig. 17.

We planned the learned manipulation strategy 100 times with the spoon at a fixed position on the table, see Table 1. Since the validation is part of the learning process, only valid trajectories are executed on the real robot system. Valid trajectories are stored relative to the object coordinate frame and executed using Closed Loop Inverse Kinematics (CLIK) [25]. In order to test the generalization we executed 4 random trajectories each 100 times with random spoon and obstacle poses, see Table 2. Objects are placed randomly in the range $(600, -300, -0.4)$ and $(1200, 300, 0.4)$ on the table using x, y and orientation around z, see Fig. 18. In Fig. 19, the results for the last trajectory are shown. If the spoon is near the robot and the handle points away from it, it can't be lifted due to collisions of the hand with the base or the workspace is left. In more than 1 m distance to the robot, the workspace is left. In a large area on the table surface, the spoon can be lifted successfully. The results show, that the observation setup was sufficient to capture relevant contacts between the human finger and the spoon as well as between the spoon and the table. The initial manipulation strategy could be generalized efficiently and a (sufficient) representation of the goals and relevant task constraints could be deduced automatically. Automatically generated contact frames could be mapped successfully to a different object with similar geometry. Planning with validation in physics simulation, see Fig. 15, is still time-demanding but the learned set of task constraints restricts the search space efficiently, so that in 66.9 % a trajectory was generated, which reproduces the learned effect, i.e.

that the spoon handle is lifted and reachable for grasping. The execution on Adero is shown in Fig. 20.

9.2 Bottle Opening

In the second experiment, a bottle had to be opened by using two fingertips only. The human teacher kept the wrist at the same position and used only the thumb and index finger for the rotation. A human demonstration is shown in Fig. 21. In the sensory environment, the motion of the cap couldn't be observed. We added a position constraint to the node and arc of the resulting planning model, which restricts the cap to stay in the same position, and an orientation constraint to restrict, that the cap is rotated at least 30°. Forces couldn't be observed consistently with the FSR sensors, if the finger was moved during contact, which results in non-representative force constraints. Consistent force measurements were added manually. In the physics simulation, we replaced the bottle and cap with a simplified model to predict the motion of the cap when forces are applied through the fingertips, see Fig. 22.

The finger motion can only be executed in a small section of the workspace of the fingers. Due to the relaxation of constraints for the wrist and fingers a lot of backtracking occurred in the motion planner until a wrist pose could be found, in which the rotation motion could be planned successfully. In Fig. 22, the planning result is visualized. The average of 100 planning attempts of the complete sequence was 35.6 s, see Table 1. We tested the generalization to environments, where the bottle is held in front of the robot with 100 random poses in the range $(750, -500, 700, -0.4, -0.4, 0)$ to $(1250, 0, 900, 0.4, 0.4, 0)$ with position x, y, z and Roll-Pitch-Yaw angles, see Table 3. In Fig. 23, the execution of the task to open a bottle on Adero is shown.

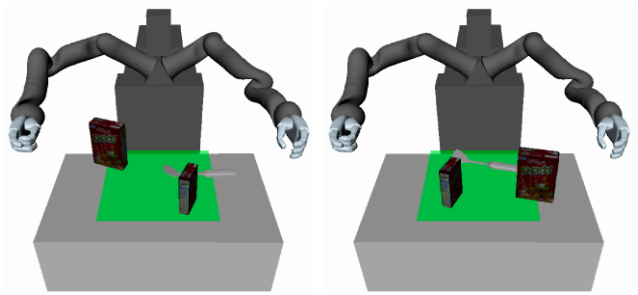


Fig. 18 Lifting a spoon: random object poses

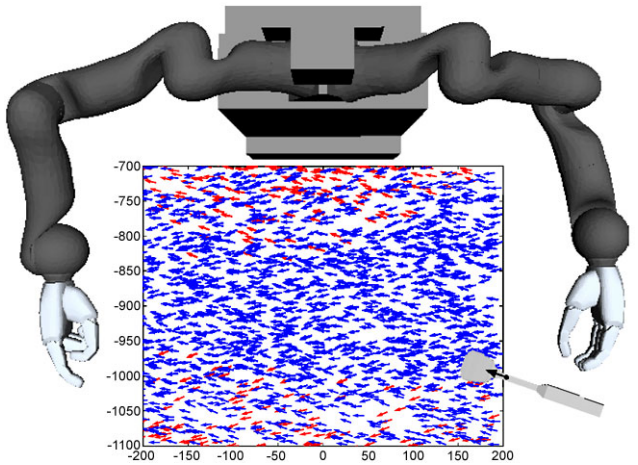


Fig. 19 Lifting a spoon: generalization results

10 Conclusion

The results indicate that dexterous manipulation tasks can be mapped automatically in a goal-directed way from human demonstrations the robot system, if models of the ob-

Fig. 21 Human demonstration of bottle opening task in sensory environment and simulation environment

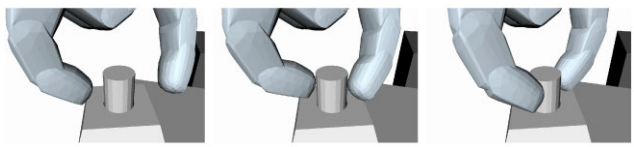


Fig. 22 Planning result for opening a bottle using physics simulation

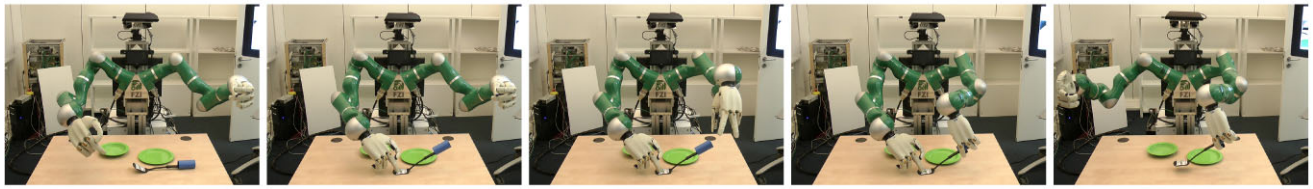


Fig. 20 Lifting a spoon: execution on Adero using online object detection to localize the spoon handle

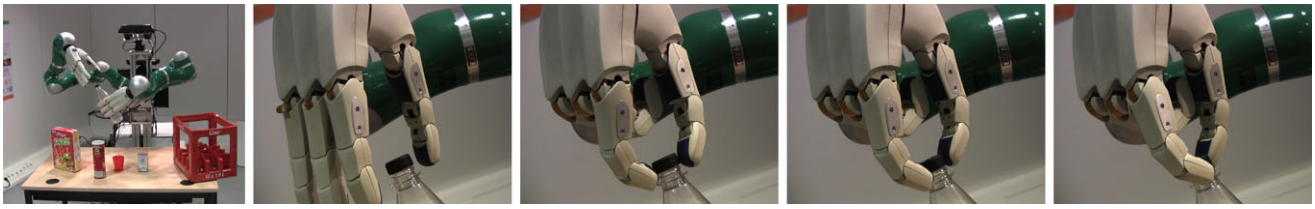


Fig. 23 Opening a bottle: execution of planning result for opening a bottle on Adero

Table 3 Opening a bottle: generalization results

Path	Success (%)	Planning time (s)
1	68.2	1.50
2	64.3	2.02
3	53.6	1.43
4	68.3	1.62
5	61.4	1.53

jects and robots are available and suitable for physics simulation. Task constraints are learned based on the observation of objects, forces, human hands and fingers. Force and contact measurements have to be accurate and consistent since the planner reproduces the contacts using special projection techniques, which don't support constraints covering inconsistent contact or force measurements. In general, constraints restricting hand, finger and object motion can be relaxed to consider the noise in the sensor setup but might lead to violation of task constraints, e.g. placing a bottle above a cup to pour in. A relaxation is possible if the demonstrations don't cover the border of the (real) constraint manifold, e.g. the human teacher tries to place the bottle opening above the cup center and not near the rim.

For dexterous tasks planning times with physics simulation are too high for online execution but offline generated planning results can be executed directly or used to speed up the execution by specializing the learned constraints, which was not in the scope of this paper. In previous work planning results on tasks like moving a chess knight, pour-in or closing a bottle with a firm grasp have proven to be in the range from 1 to 8 seconds, which is appropriate for online execution.

11 Summary

In this work, *strategy graphs*, i.e. planning models for dexterous manipulation tasks, are learned based on multiple human demonstrations. The initial, learned planning model contains a large number of automatically generated task constraints including novel contact constraints for objects and fingers, which implicitly define the feature space for learning. We considered the correspondence problem explicitly by relaxing constraints for the fingers and human hands. Since the effects of the manipulation task, e.g. the object motion, are also learned, a qualitatively equivalent motion with the robot hands and fingers is planned automatically, which produces the same effect.

Due to the small number of human demonstrations the generalization capabilities of the initial planning model are limited. We generalized the learned manipulation strategy by using the parallelized generalization algorithm in [15], which generates a maximal subset of task constraints, which admits a successful solution to a set of test problems. In order to generalize to novel objects, we mapped automatically generated contact coordinate frames using Blended Intrinsic Maps, which define a surface mapping between two 3d models. The PbD system was evaluated on a real robot using two dexterous manipulation tasks.

12 Outlook

Generalization of learned task constraints to novel objects is demanding. In this work, we used objects with simple 3d geometry, e.g. spoons of different sizes, which allow an efficient mapping of object surfaces. In general, we can't assume, that a single mapping algorithm will produce the desired result for complex objects. Different algorithms will be implemented to generate multiple hypotheses about the correct mapped coordinate frame. We intend to solve the resulting ambiguity by extending the generalization algorithm in [15].

Acknowledgements This work has been partially conducted within the German SFB 588 "Humanoid Robots" granted by DFG and within the ECs Integrated Project DEXMART under grant agreement no. 126239 (FP7/2007-2013).

References

1. Azad P, Gockel T, Dillmann R (2007) Computer vision—das praxisbuch. Elektor, Aachen
2. Berenson D, Srinivasa S, Ferguson D, Kuffner J (2009) Manipulation planning on constraint manifolds. In: IEEE int conf on robotics and automation
3. Breazeal C, Berlin M, Brooks AG, Gray J, Thomaz AL (2006) Using perspective taking to learn from ambiguous demonstrations. Robot Auton Syst 54(5):385–393

4. Calinon S, Billard A (2006) Teaching a humanoid robot to recognize and reproduce social cues. In: Proceedings of the IEEE international symposium on robot and human interactive communication (RO-MAN), pp 346–351
5. Calinon S, Guenter F, Billard A (2005) Goal-directed imitation in a humanoid robot. In: Proceedings of the IEEE international conference on robotics and automation
6. Calinon S, Guenter F, Billard A (2007) On learning, representing, and generalizing a task in a humanoid robot. IEEE Trans Syst Man Cybern, Part B, Cybern 37(2):286–298
7. Eiben A, Smith J (2003) Introduction to evolutionary computing. Springer, Berlin
8. Erlhagen W, Mukovskiy A, Bicho E, Panin G, Kiss C, Knoll A, van Schie HT, Bekkering H (2006) Goal-directed imitation for robots: a bio-inspired approach to action understanding and skill learning. Robot Auton Syst 54(5):353–360
9. Friedrich H, Dillmann R (1995) Robot programming based on a single demonstration and user intentions. In: 3rd European workshop on learning robots at ECML'95
10. Friedrich H, Dillmann R, Rogalla O (1999) Interactive robot programming based on human demonstration and advice. In: Selected papers from the international workshop on sensor based intelligent robots. Springer, London, pp 96–119
11. Hu H, Gao X, Li J, Wang J, Liu H (2004) Calibrating human hand for teleoperating the hit/dlr hand. In: IEEE international conference on robotics and automation, pp 4571–4576
12. Ijspeert AJ, Nakanishi J, Schaal S (2002) Movement imitation with nonlinear dynamical systems in humanoid robots. In: IEEE int conf on robotics and automation. IEEE Press, New York, pp 1398–1403
13. Jäkel R, Schmidt-Rohr SR, Loesch M, Dillmann R (2010) Representation and constrained planning of manipulation strategies in the context of programming by demonstration. In: IEEE international conference on robotics and automation (ICRA'10)
14. Jäkel R, Schmidt-Rohr SR, Loesch M, Kasper A, Dillmann R (2010) Learning of generalized manipulation strategies in the context of programming by demonstration. In: 10th IEEE-RAS international conference on humanoid robots (Humanoids)
15. Jäkel R, Meißner P, Schmidt-Rohr SR, Dillmann R (2011) Distributed generalization of learned planning models in robot programming by demonstration. In: IEEE/RSJ int conf on intelligent robots and systems
16. Kasper A, Xue Z, Dillmann R (2012) A object model database for object recognition, localisation and manipulation in service robotics. Int J Robot Res. doi:10.1177/0278364912445831
17. Keijzer M, Merelo JJ, Romero G, Schoenauer M (2002) Evolving objects: a general purpose evolutionary computation library. Artif Evol 2310:829–888
18. Kim V, Lipman Y, Funkhouser T (2011) Blended intrinsic maps. ACM Trans Graph 30:4. doi:10.1145/2010324.1964974
19. Kuniyoshi Y, Inaba M, Inoue H (1994) Learning by watching: extracting reusable task knowledge from visual observation of human performance. IEEE Trans Robot Autom 10:799–822
20. Muhlig M, Gienger M, Steil J, Goerick C (2009) Automatic selection of task spaces for imitation learning. In: IEEE/RSJ international conference on intelligent robots and systems, IROS 2009, pp 4996–5002
21. Pardowitz M, Glaser B, Dillmann R (2007) Learning repetitive robot programs from demonstrations using version space algebra. In: Proceedings of the 13th IASTED international conference on robotics and applications. ACTA Press, Anaheim, pp 394–399
22. Pastor P, Hoffmann H, Asfour T, Schaal S (2009) Learning and generalization of motor skills by learning from demonstration. In: IEEE international conference on robotics and automation
23. Pastor P, Kalakrishnan M, Chitta S, Theodorou E, Schaal S (2011) Skill learning and task outcome prediction for manipulation. In: IEEE international conference on robotics and automation
24. Rybski P, Yoon K, Stolarz J, Veloso M (2007) Interactive robot task training through dialog and demonstration. In: 2nd ACM/IEEE international conference on human-robot interaction
25. Siciliano B, Villani L, Oriolo G, Sciavicco L (2008) Robotics. Springer, New York
26. Steffen J, Elbrechter C, Haschke R, Ritter H (2010) Bio-inspired motion strategies for a bimanual manipulation task. In: 10th IEEE-RAS international conference on humanoid robots (Humanoids), pp 625–630
27. Tang M, Kim Y, Manocha D (2009) C2a: controlled conservative advancement for continuous collision detection of polygonal models. In: IEEE international conference on robotics and automation (ICRA'09). IEEE Press, New York, pp 849–854
28. van Lent M, Laird JE (2001) Learning procedural knowledge through observation. In: Proceedings of the 1st international conference on knowledge capture, K-CAP '01. ACM Press, New York, pp 179–186
29. Veeraraghavan H (2008) Teaching sequential tasks with repetition through demonstration (short paper). In: Proceedings of the international conference on autonomous agents and multiagent systems, AAMAS'08

Rainer Jäkel is a research scientist at Karlsruhe Institute of Technology, Institute for Anthropomatics, Karlsruhe, Germany. His research interests are robot programming by demonstration and constrained motion planning.

Sven R. Schmidt-Rohr graduated with a Diploma in Computer Science from KIT. He is working as a research assistant at the Humanoids and Intelligence Systems Lab at KIT. His research focus is on Programming by Demonstration of abstract, probabilistic decision making for autonomous service robots.

Steffen W. Rühl received his degree in Computer Science from the University of Karlsruhe in 2007. His main research area is the planning and execution of manipulation tasks for bimanual robots in a variable environment.

Alexander Kasper graduated with a Diploma in Computer Science from KIT. He is currently working as a research assistant at the Humanoids and Intelligence Systems Lab at KIT under Prof. Dillmann. His research interests include knowledge representation and generation for service robots.

Zhixing Xue received his Computer Science degree from the University of Karlsruhe in 2005. He is with FZI since 2006. His main research is directed towards grasp and manipulation planning for multi-fingered robotic hands and the software development of two-armed manipulation.

Rüdiger Dillmann is full Professor at the Karlsruhe Institute of Technology, Institute for Anthropomatics, Karlsruhe, Germany. His research interests include robot programming by demonstration, machine learning, robot vision, humanoid robotics, cognitive cars, and medical applications of informatics.