Learning Haptic Exploration Schemes for Adaptive Task Execution

Thomas Eiband[1], Matteo Saveriano[1], and Dongheui Lee[1,2]

***Abstract*— The recent generation of compliant robots enables kinesthetic teaching of novel skills by human demonstration. This enables strategies to transfer tasks to the robot in a more intuitive way than conventional programming interfaces. Programming physical interactions can be achieved by manually guiding the robot to learn the behavior from the motion and force data. To let the robot react to changes in the environment, force sensing can be used to identify constraints and act accordingly. While autonomous exploration strategies in the whole workspace are time consuming, we propose a way to learn these schemes from human demonstrations in an object targeted manner. The presented teaching strategy and the learning framework allow to generate adaptive robot behaviors relying on the robot's sense of touch in a systematically changing environment. A generated behavior consists of a hierarchical representation of skills, where haptic exploration skills are used to touch the environment with the end effector, and relative manipulation skills, which are parameterized according to previous exploration events. The effectiveness of the approach has been proven in a manipulation task, where the adaptive task structure is able to generalize to unseen object locations. The robot autonomously manipulates objects without relying on visual feedback.**

I. Introduction

In human daily life scenarios, interaction forces are an important means of detecting and exploring the surroundings. Besides visual observations, haptic feedback is often used as an additional source of information when exploring or manipulating an uncertain environment [1]–[4]. This is achieved by relying on the sense of touch in a region of the human's spatial imagery where a physical contact is expected [5]. Especially when a human cannot rely on vision alone, haptic sensing is purposeful to gain feedback about object locations. In contrast to humans, technical systems might not be equipped with a vision system. Moreover, vision is not always applicable due to occlusions, adverse camera angle, weak illumination, or small color gradients. In service robotics or in human-robot shared workspaces, there is a demand to act in unstructured or uncertain environments. Examples are variable pick/place positions, a manually filled stack with mutable structure or a staple which varies in height.

Programming a robot to use exploration motions by conventional interfaces is elaborate and requires expert knowledge. To ease this procedure, learning from demonstration [6] is employed as teaching interface to directly extract the desired behavior. Hereby, the user demonstrates the task consisting of exploration and manipulation motions multiple times by kinesthetic teaching. The locations of the involved objects are changed in a region where they are expected to be positioned during execution. This is an intuitive input method for a novice user, where also pick and place actions can be easily demonstrated by controlling the gripper with a foot pedal or a button.

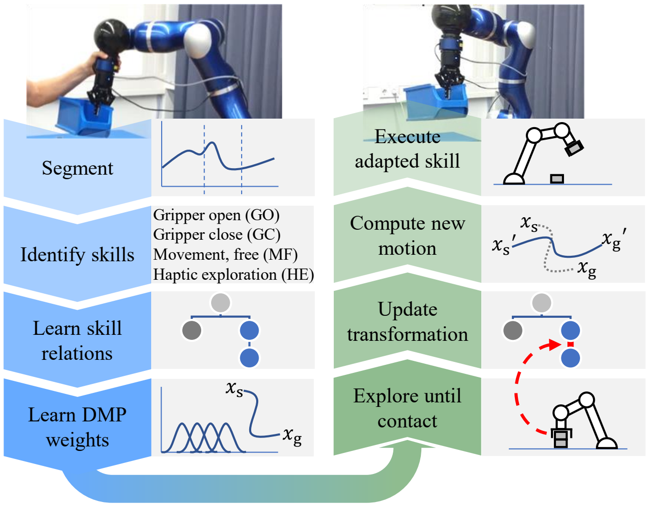


Fig. 1. The skill demonstration, learning and execution framework to execute parameterized motions. Motion start and end points are adapted by previously observed constraints during exploration of the environment. All behaviors are extracted from human demonstrations via kinesthetic teaching.

Having this in mind, we propose a strategy to segment multiple demonstrations of a task into parametrizable robotic skills, considering intended variations of the environment. We developed a method, which extracts position-based relations between skills grounded on the observed variance during the demonstrations. Further, we propose a strategy to learn exploration paths from demonstrations to enable adaptive movements in an uncertain environment.

Our work is outlined as follows: Section II summarizes related work in the domain of haptic exploration and contact-based localization. Section III describes our approach for segmentation and skill identification and Sec. IV describes our method for learning skill relations. The experiments are presented in Sec. V. Finally, conclusions and future work are stated in Sec. VI.

II. Related Work

Learning of force-based robot skills from demonstrations has been shown in a variety of applications like door-opening and ironing [7], pouring from a bottle [8], screw driving [9], grasping [10], engraving [11], wood planing [12] or writing [13]. Further, in collaborative setups such as assembly [14]

[1] German Aerospace Center (DLR), Institute of Robotics and Mechatronics, Wessling, Germany

[2] Technical University of Munich, Chair of Human-centered Assistive Robotics, Munich, Germany

{thomas.eiband, matteo.saveriano, dongheui.lee}@dlr.de

or transportation tasks [15], [16]. However, none of these approaches addressed sensing of object locations in the environment. A force-based learning framework is proposed in [17], where the robot continuously reacts to force variables which have been identified by Mutual Information analysis. The framework does not consider motion segmentation required to reach absolute positions alongside with several robot skills, such as pick and place. Continuous online movement adaption by force perceptions is also presented in [10], [17] and [18] but they are again not suitable in a task requiring both fixed and adaptive positions. Learning force-based tasks combined with motion segmentation in different frames of reference is proposed in [19], which requires vision-tracked object positions, where our approach extracts important points vision-free during demonstration. Another learning scheme pairs vision with tactile sensing [20] and focuses on material and object detection. The system is not bootstrapped from demonstration and the inferred class is not used to adapt the robot's behavior. In the context of segmenting and sequencing, [21] proposes the prediction of manipulation phases and [22] shows learning of hierarchical skills by reinforcement learning, where both approaches require object tracking if the object is not initially in the hand. Learning of complex tasks using kinematic and video data is proposed in [23]. In [24], compliant manipulation is achieved by interaction-based phase transitions in a Hidden Markov Model, where non-linear motions in free space and exact goal points in free space were not addressed. In [25], autonomous exploration of the whole workspace is combined with tactile-based object discrimination. In [26], the search policy is learned from human behavior to act in the whole workspace, where we instead focus on the transfer of specialized skills to rapidly program and execute a novel task. The approach in [27] combines in-hand object localization using a tactile sensor array with tactile based manipulation. During execution, a DMP approach is used similarly to [28], which reproduces both desired motion and tactile trajectory. More elaborate sensing techniques make use of embedded sensors in gripper fingers [29], or tactile sensor arrays for object recognition [30]. The latter authors use sampling-based motion planning [31], which requires a model of the environment or a huge number of real robot executions. Learning from human hand motion observation is presented in [32], with the goal to correct wrongly observed hand postures with the sensed contact information while grasping. The authors of [33] present teaching of stiffness profiles, where a haptic interaction lies between robot and human in order to adapt the compliance to the task requirements. A large review about tactile sensing in [34] shows that a variety of sensors is available but the major challenge lies in development of novel manipulation algorithms.

III. Skill Identification from Human Demonstrations

We propose a learning from demonstration framework, which extracts skills from a demonstrated task. A skill is a predefined robot behavior [35] parameterized by the demonstrations. First, we introduce our *Task Demonstration System* and explain how the user transfers the sequence of skills to the robot (starting in Fig. 1 at the top left). The *Task Segmentation* produces labeled segments of the demonstration data. The next step in our learning framework is the *Skill Identification*, where we construct predefined skills from the previously labeled segments.

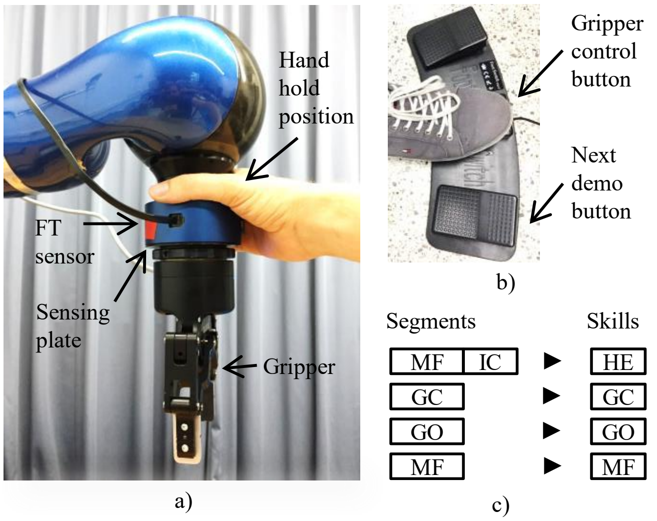


Fig. 2. Kinesthetic teaching by manually guiding the robot (a) and controlling gripper actions and demo recording by a foot pedal (b). Labeled segments and identified skills (c)

A. Task Demonstration System

The system is able to identify a sequence of predefined robot skills, which can be motion in free space (MF), haptic exploration (HE), and motions ending with gripper open (GO) or gripper close (GC). The software framework is implemented in *Python*, communicating via a realtime-capable C++ middleware with *Matlab/Simulink* models connected to the robot. In order to demonstrate a task consisting of a sequence of skills, the robot is guided by kinesthetic teaching. Hereby, the user guides the robot using a hold close to the tip of the robot (see Fig. 2a). Whenever the user wants the robot to explore a part of the environment, the user lets the gripper fingers or other parts of the gripper touch the desired object while the gripper state can be open or closed. The interaction forces are recorded by a force-torque (FT) sensor, mounted between robot tip and gripper. Gripper actions can be triggered by a foot pedal (see Fig. 2b). During teaching, the robot position $\boldsymbol{x}_\mathrm{p} \in \mathbb{R}^3$ and orientation quaternions elements $\boldsymbol{x}_\mathrm{o} \in \mathbb{R}^4$, interaction FT measurement $\boldsymbol{f} \in \mathbb{R}^6$ and gripper state g are recorded. A FT measurement consists of $\boldsymbol{f} = [f_x, f_y, f_z, t_x, t_y, t_z]^T$ and the gripper state is defined by $g = \{1 \textbf{ if } \text{gripper\_open}; -1 \textbf{ if } \text{gripper\_closed}\}$. The whole input vector is denoted as $\boldsymbol{x} = [\boldsymbol{x}_\mathrm{p}, \boldsymbol{x}_\mathrm{o}, \boldsymbol{f}, g]^T$. Each sample of user demonstration i is stacked in a matrix $\boldsymbol{X}^i \in \mathbb{R}^{N_\mathrm{d} \times N^i}$, where N_d is the number of input dimensions and N^i the number of samples in demo i. The set $\boldsymbol{T} = \{\boldsymbol{X}^1, ..., \boldsymbol{X}^I\}$ consists of I demonstrations, where our method requires a minimum number of $I = 2$. Ideally, the user covers significant changes in object locations to extract the variance in the data. In fact, the length of an exploration path is derived from the exposed variance during teaching.

B. Task Segmentation

a) Pre-processing: All demonstrations recorded for one task are preprocessed to allow a robust segmentation into skills. First, the FT measurements are filtered by a 1st order low-pass Butterworth filter with a cutoff frequency of $f_{\text{co}} = 1\,\text{Hz}$. Periods in which the robot movement does not exceed a defined threshold in translation or rotation over time are removed.

Dynamic Time Warping (DTW) [36] is used to align each demo in the set $\boldsymbol{T}$ with a medoid demo. A medoid demo can be found by pairwise computing DTW distances between all demos and selecting the demo with minimum sum of squared distances to all other demos. The output is stored in $\boldsymbol{C} = [\tilde{\boldsymbol{X}}_{\text{p}}, \tilde{\boldsymbol{X}}_{\text{o}}, \tilde{\boldsymbol{F}}, \tilde{\boldsymbol{G}}]^T \in \mathbb{R}^{I \times N_{\text{d}} \times \tilde{N}}$, with forces and torques in $\tilde{\boldsymbol{F}} = [\boldsymbol{F}_{\text{f}}, \boldsymbol{F}_{\text{t}}]^T$ and with $\tilde{N}$ samples.

The average of $\boldsymbol{F}_{\text{f}}$ and $\boldsymbol{F}_{\text{t}}$ over demonstrations is calculated as $\boldsymbol{F}_{\text{m,f}} = \frac{1}{\tilde{N}} \sum_{i=1}^{I} \boldsymbol{F}_{\text{f}}$ and $\boldsymbol{F}_{\text{m,t}} = \frac{1}{\tilde{N}} \sum_{i=1}^{I} \boldsymbol{F}_{\text{t}}$, which reduces artifacts when segmenting the contact state. Hereby, the number and quality of demonstrations influence the robustness. Possible artifacts are multiple bumps when the user touches a rather stiff environment, accidentally touching the tool below the FT sensor with a part of the hand, or unintended force interactions while picking and placing an object. The goal of the subsequent segmentation is to separate contact states and free movement as well as to split segments by the binary gripper state.

b) Segmentation of contact states: The in-contact segmentation relies on predefined force and torque thresholds δ_{f} and δ_{t} compared with the Euclidean norm of force and torque measurements respectively. Algorithm 1 detects rising and falling edges in the force and torque domain. Whenever a rising edge is detected in one domain, the subsequent segment is labeled as IC. For every falling edge, the subsequent segment is labeled as MF. The independent comparison of force and torque increases segmentation robustness as these signals can occur independently when in contact with the environment. Reaching the threshold in one modality is sufficient to trigger an IC state while both modalities need to be below their thresholds to reset it, leading to the start of an MF segment. For instance, force and torque can alternate while the tool is continuously in contact, which leads to only one IC segment.

c) Segmentation by gripper state: The MF segments coming from the previous segmentation are further split when a gripper action occurs in the task. All indexes for gripper open (GO) n^i_{GO} and gripper close (GC) n^i_{GC} are extracted for each demo i and then averaged over demos, resulting in the averaged indexes $\bar{n}_{\text{GO}} = \frac{1}{I} \sum_{i=1}^{I} n^i_{\text{GO}}$ and $\bar{n}_{\text{GC}} = \frac{1}{I} \sum_{i=1}^{I} n^i_{\text{GC}}$ respectively. Segments before these indexes are labeled as gripper open (GO) or gripper close (GC) accordingly.

A segment s is stored in the matrix $\boldsymbol{S}_s = [\boldsymbol{X}_{\text{p},s}, \boldsymbol{X}_{\text{o},s}, \boldsymbol{F}_s]^T \in \mathbb{R}^{I \times 13 \times N_s}$, with N_s samples from I demos. It contains position $\boldsymbol{X}^i_{\text{p},s}$, orientation $\boldsymbol{X}^i_{\text{o},s}$, and force and torque $\boldsymbol{F}^i_s$ for each demonstration i. The result of the segmentation is the set of all segments, given by $\boldsymbol{S} = \{\boldsymbol{S}_1, ..., \boldsymbol{S}_{N_{\text{s}}}\}$.

Algorithm 1 IC segmentation

```
Input: $\boldsymbol{F}_m$
Initialization :
$c \leftarrow 1$
$IC \leftarrow$ false
for $n = 1 \ldots N_n$ do
    Detect rising edge:
    if $(\|\boldsymbol{F}_{\text{m,f}}(n-1)\| < \delta_f$ and $\|\boldsymbol{F}_{\text{m,f}}(n)\| > \delta_f)$ or
$(\|\boldsymbol{F}_{\text{m,t}}(n-1)\| < \delta_t$ and $\|\boldsymbol{F}_{\text{m,t}}(n)\| > \delta_t)$ and not
$IC$ then
        $\boldsymbol{IC}_{\text{s}}(c) \leftarrow n$          ▷ add segmentation index to list
        $IC \leftarrow$ true                            ▷ set in-contact flag
        $c \leftarrow c + 1$
    end if
    Detect falling edge:
    if $(\|\boldsymbol{F}_{\text{m,f}}(n)\| < \delta_f$ and $\|\boldsymbol{F}_{\text{m,t}}(n)\| < \delta_t)$ and
$IC$ then
        $\boldsymbol{IC}_{\text{e}}(c) \leftarrow n$          ▷ add segmentation index to list
        $IC \leftarrow$ false                         ▷ reset in-contact flag
    end if
end for
return $\boldsymbol{IC}_{\text{c}}, \boldsymbol{IC}_{\text{e}}$
```

C. Skill Identification

In our framework, each contact with the tool is considered as haptic exploration action intended by the user. Furthermore, for object manipulation, skills for robot motion, picking, and placing are required. To identify a haptic exploration (HE) skill, an IC segment preceded by a MF segment needs to be found. Figure 2c) shows the mapping between segments and skills. If a MF segment occurs before an IC segment, a HE skill r is generated, provided with the stacked data $\boldsymbol{R}_r = [\boldsymbol{S}_{\text{MF}}, \boldsymbol{S}_{\text{IC}}]$ of both segments. The remaining segments, such as MF, GO and GC are transformed into their according skills. This leads to a number of skills N_{r}, less than or equal to the original number of segments N_{s}. The identified skills are stored in a set $\boldsymbol{R} = \{\boldsymbol{R}_1, ..., \boldsymbol{R}_{N_{\text{r}}}\}$.

IV. Learning Skill Relations

A. Relation Learning and Representation

After the *Skill Identification*, our goal is to infer whether a skill motion shall be executed relative to another skill or relative to the origin (*root*). Therefore, a metric is developed which pairwise compares all skills to identify how strong they are related to each other. First, the goal position $\boldsymbol{x}^i_{\text{g,r}} \in \mathbb{R}^3$ of a skill r is extracted for each demonstration i. This leads to I goal points for skill r, stored in a key-points matrix $\boldsymbol{K}_r = [\boldsymbol{x}^1_{\text{g},r}, \ldots, \boldsymbol{x}^I_{\text{g},r}]^T \in \mathbb{R}^{I \times 3}$. All key-points are added to a set of available key-points given by $\boldsymbol{L} = \{\boldsymbol{K}_1, ..., \boldsymbol{K}_{N_{\text{r}}}\}$.

Next, a dimension-wise metric for the relation distance between skills m and n with their key-points $\boldsymbol{K}_m$ and $\boldsymbol{K}_n$ in dimension d over all demonstrations is given by

$$v^d_{m,n} = \text{Var}\left(\begin{bmatrix} \boldsymbol{K}_m(1,d) - \boldsymbol{K}_n(1,d) \\ \vdots \\ \boldsymbol{K}_m(I,d) - \boldsymbol{K}_n(I,d) \end{bmatrix}\right) \text{ for } m \neq n. \quad (1)$$

Similarly, the dimension-wise relation distance between a single skill m with key-point $\boldsymbol{K}_m$ and the coordinate origin

Algorithm 2 Constructing the skill relation tree

```
Input: $\boldsymbol{R}, \boldsymbol{R}_{\text{rel}}$
for each child_parent_relation $c, p$ in $\boldsymbol{R}_{\text{rel}}$ do
    $\boldsymbol{T}_{K_p}^{\mu_p} \leftarrow [\boldsymbol{\mu}_p, \dots, \boldsymbol{\mu}_p]^T - \boldsymbol{K}_p$    ▷ parent key-point translations
    $\boldsymbol{K}_c \leftarrow \boldsymbol{K}_c + \boldsymbol{T}_{K_p}^{\mu_p}$    ▷ transform child keypoints
    $\boldsymbol{R}_{\text{p},c} \leftarrow \boldsymbol{R}_{\text{p},c} + \left[\boldsymbol{T}_{K_p}^{\mu_p}, \dots, \boldsymbol{T}_{K_p}^{\mu_p}\right]$    ▷ transform skill data
    $\boldsymbol{t}_p^c \leftarrow \text{mean}(\boldsymbol{K}_c - \boldsymbol{K}_p)$    ▷ parent child translation
    connect_nodes_with_trafo$(c, p, \boldsymbol{t}_p^c)$
end for
return $\boldsymbol{R}$
```

O is

$$v_{m,n}^d = \text{Var}\left(\begin{bmatrix} \boldsymbol{K}_m(1,d) \\ \vdots \\ \boldsymbol{K}_m(I,d) \end{bmatrix}\right) \quad \text{for } m = n. \tag{2}$$

The relation distance over all dimensions for all m, n is computed using the root mean squared error

$$\boldsymbol{V}(m,n) = \sqrt{\frac{1}{3}\sum_{d=1}^{3} {v_{m,n}^d}^2} \quad \text{for } \forall\, m, n. \tag{3}$$

For each skill m, in row m of the relation distance matrix $\boldsymbol{V} \in \mathbb{R}^{N_{\text{r}} \times N_{\text{r}}}$, the skill which has minimum relation distance to skill m is found by

$$k = \underset{k \in \{1,\dots,m\}}{\text{argmin}} \{\boldsymbol{V}(m,k)\} \quad \text{with } k \leq m, \tag{4}$$

considering all skills in the demonstrated sequence until skill m to exclude non-causal relations to future skills. The skill k found with minimum distance is stored in a relation matrix $\boldsymbol{R}_{\text{rel}} \in \mathbb{R}^{N_{\text{r}} \times N_{\text{r}}}$ by $\boldsymbol{R}_{\text{rel}}(m,k) = 1$. In the case $m = k$, the skill is relative to the coordinate origin (*root*) by setting $\boldsymbol{R}_{\text{rel}}(m,m) = 1$, as it does not depend on any other skills. All other entries in $\boldsymbol{R}_{\text{rel}}$ are set to 0.

A hierarchical structure of skills in form of a relation tree is constructed from the relation matrix $\boldsymbol{R}_{\text{rel}}$ to represent translations between skill goal points (Fig. 3), where each node is a skill. The translations are directed from a parent skill goal point to a child skill goal point and are updated during the execution by the previously unknown contact points of the HE skills. Skills which are directly attached to the *root* node (acting as the coordinate origin O) are executed with absolute motions, such as HE0 and GO3. The first skill of a task is always attached to *root* as it cannot be influenced by any preceding skill. Skills attached in deeper layers are executed with relative motions to their parent skills. For instance, the goal point of GC2 is defined relative to the reached contact point of HE1. As a consequence, a single HE skill is able to affect multiple child skills. In addition, a child skills can be executed directly after its parent skill but also at any later time.

The relation tree is constructed by Algorithm 2, which uses a mean key-point $\boldsymbol{\mu}_p = \frac{1}{I}\sum_{i=1}^{I} \boldsymbol{K}_p^i$ of a parent skill p to transform the position data $\boldsymbol{R}_{\text{p},c}$ and key-points $\boldsymbol{K}_c$ of the child skill c.

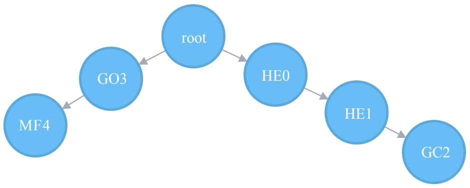


Fig. 3. Relative skill tree from the box experiment. The goal point transformation between two skills is a directed relationship from parent to child skill.

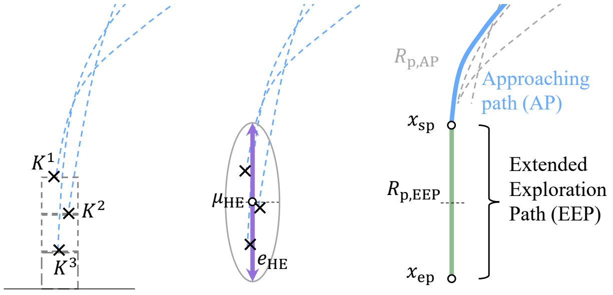
B. Skill Learning

The skills to execute free motions only, such as MF, GO, and GC are directly learned from the data of multiple demonstrations. In contrast, a HE skill needs to handle uncertain environments where the contact point is unknown before execution. Therefore, a collision aware motion and a modification of the skill data are necessary

To allow the robot to start exploring outside of the demonstrated region, the motion learning considers an approaching path (AP) to reach the exploration area and an extended exploration path (EEP) inside the exploration area (Fig.4). In the following, the generation of the full HE motion is explained, where the subscript *HE* is used for referring to any HE skill.

a) Modeling the exploration motion: First, the distribution of HE key-points is modeled as normal distribution $\boldsymbol{K}_{\text{HE}} \sim \mathcal{N}(\boldsymbol{\mu}_{\text{HE}}, \boldsymbol{\Sigma}_{\text{HE}})$ with mean $\boldsymbol{\mu}_{\text{HE}} = \frac{1}{I}\sum_{i=1}^{I} \boldsymbol{K}_{\text{HE}}^i$ and covariance $\boldsymbol{\Sigma}_{\text{HE}} = \text{Cov}(\boldsymbol{K}_{\text{HE}})$. Note that at least two demonstrations need to be provided to exploit the variance in the data, however more demonstrations might cover a wider range of key-points. It is further assumed that exploration takes place in a defined region of this distribution and along a linear exploration path to find an object. To identify this path, a principal component analysis is applied, using the covariance matrix $\boldsymbol{\Sigma}_{\text{HE}}$. Its principal eigenvector $\boldsymbol{e}_{\text{HE}}$ is termed as the main exploration vector (Fig. 4 middle). However, its sign might be inverse to the desired linear exploration direction. We use the maximum force vector of the skill while in contact $\hat{\boldsymbol{F}}_{\text{f,IC}}^i$ of each demo i (see Fig. 7 for an example) to get the average force vector $\bar{\boldsymbol{F}}_{\text{f,IC}} = \frac{1}{I}\sum_{i=1}^{I} \hat{\boldsymbol{F}}_{\text{f,IC}}^i$. The alignment with the main exploration vector $\boldsymbol{e}_{\text{HE}}$ is ensured using the sign of the dot product $\bar{\boldsymbol{F}}_{\text{f,IC}} \cdot \boldsymbol{e}_{\text{HE}}$. If positive, the main exploration vector $\boldsymbol{e}_{\text{HE}}$ needs to be inverted to be aligned with the approaching path. The position trajectory $\boldsymbol{R}_{\text{p,EEP}}$ of the EEP is constructed by connecting the exploration start point $\boldsymbol{x}_{\text{sp}} = \boldsymbol{\mu}_{\text{HE}} - k_{\text{e}}\boldsymbol{e}_{\text{HE}}$ and the exploration end point $\boldsymbol{x}_{\text{ep}} = \boldsymbol{\mu}_{\text{HE}} + k_e \boldsymbol{e}_{\text{HE}}$ with linearly spaced points, where k_e is a parameter to scale the exploration region (Fig. 4 right). In contrast to the position, the orientation is kept fixed during the whole EEP, taken from the quaternions average [37] at the contact points.

The motion of the HE skill consists of two phases. In the first phase, the robot follows the AP until the exploration region is reached. The data $\boldsymbol{R}_{\text{p,AP}}$ for AP is constructed from samples from the beginning of the corresponding segment until the index where a sample falls below a distance to the exploration start point $\boldsymbol{x}_{\text{sp}}$ (Fig. 4 right). In the second phase, we want the robot to follow the EEP where $\boldsymbol{x}_{\text{sp}}$ is a desired via-point before the robot reaches the exploration end-point



Demonstrated trajectories (blue) and objects (grey)

Distribution of contact key-points K^i (×) and principal eigenvector e_{HE}

Learned exploration trajectory

Fig. 4. Schema of multiple demonstrations in a changing environment (left), the distribution over extracted key-points with principal component eigenvector (middle) and learned exploration trajectory (right).

$\boldsymbol{x}_{\mathrm{ep}}$. The final definition of the input data of a HE skill is then $\boldsymbol{R}_{\mathrm{p,HE}} = [\boldsymbol{R}_{\mathrm{p,AP}}, \boldsymbol{R}_{\mathrm{p,EEP}}]$ with an example given in Fig. 7.

b) Motion Encoding: For any skill r, a Gaussian Mixture Model [38] is used to encode position $\boldsymbol{R}_{\mathrm{p},r}$, orientation $\boldsymbol{R}_{\mathrm{o},r}$, and a time vector $\boldsymbol{t} \in \mathbb{R}^{N_{\mathrm{t}}}$ from multiple demonstrations in a model M_{GMM}. Expectation Maximization is used to fit a number of Gaussian distributions on the data from all stacked demonstrations in

$$\boldsymbol{R}_r^{\mathrm{in}} = \begin{bmatrix} \boldsymbol{R}_{\mathrm{p},r}^1 & \dots & \boldsymbol{R}_{\mathrm{p},r}^I \\ \boldsymbol{R}_{\mathrm{o},r}^1 & \dots & \boldsymbol{R}_{\mathrm{o},r}^I \\ \boldsymbol{t} & \dots & \boldsymbol{t} \end{bmatrix} ; \quad M_{\mathrm{GMM}} = \mathrm{GMM}(\boldsymbol{R}_r^{\mathrm{in}}). \quad (5)$$

Gaussian Mixture Regression is applied to obtain a generalized trajectory $\boldsymbol{R}_r^{\mathrm{out}} \in \mathbb{R}^{7\times N_{\mathrm{t}}}$ conditioned on the time vector $\boldsymbol{t}$ and denoted as

$$\boldsymbol{R}_r^{\mathrm{out}} = \begin{bmatrix} \boldsymbol{R}_{\mathrm{p},r}^{\mathrm{out}} \\ \boldsymbol{R}_{\mathrm{o},r}^{\mathrm{out}} \end{bmatrix} = \mathrm{GMR}(M_{\mathrm{GMM}}, \boldsymbol{t}) = P(M_{\mathrm{GMM}}|\boldsymbol{t}). \quad (6)$$

The quaternions in $\boldsymbol{R}_{\mathrm{o},r}^{\mathrm{out}}$ are normalized to unit quaternions after GMR. The computations of the GMM and GMR are carried out using *pbdlib* [39]. We decided to encode the motion first in a probabilistic model before learning a dynamical system for execution. Even so, it is possible to encode multiple demonstrations directly in a Dynamic Movement Primitive (DMP). DMPs are employed to learn a stable motion which is guaranteed to converge to a desired goal point [40]. In this work, the approach from [41] has been adopted to learn DMPs for position and quaternion based DMPs by using the quaternion logarithmic and exponential map. The DMPs enable adaption of starting point and goal given the transformations in the relation tree. The DMP weights for a skill r are learned from its motion data $\boldsymbol{R}_r^{\mathrm{out}}$ generated by the GMR.

C. Skill Sequencing and Execution

The adaptive execution scheme is implemented in Algorithm 3, generating motions with parameterized start $\boldsymbol{x}_{\mathrm{s}}$ and goal point $\boldsymbol{x}_{\mathrm{g}}$ by the DMPs learned beforehand. The coordinate origin is termed as O. During execution of a HE skill, a contact is detected when the exploration force or torque exceeds a defined threshold. Whenever a contact is registered, the controller stops the current motion. The contact point $\boldsymbol{x}_c$ is extracted and the goal point translation between this skill and its parent skill is updated in the relation tree (Alg. 3, line 13).

Algorithm 3 Adaptive task execution

```
 1: for $r = 1 \dots N_{\mathrm{r}}$ do
 2:     $p \leftarrow$ get_parent_skill($r$)
 3:     if skill $r$ is HE skill then
 4:         $\boldsymbol{x}_{\mathrm{g}} \leftarrow \text{get\_trafo}(O, p) + \boldsymbol{R}_{\mathrm{p},r}^{\mathrm{out}}(N_{\mathrm{n},r}) - \boldsymbol{R}_{\mathrm{p},r}^{\mathrm{out}}(1)$
 5:     else
 6:         $\boldsymbol{x}_{\mathrm{g}} \leftarrow \text{get\_trafo}(O, r)$
 7:     end if
 8:     $\boldsymbol{x}_{\mathrm{s}} \leftarrow$ get_robot_pos()
 9:     execute_skill($r, \boldsymbol{x}_{\mathrm{s}}, \boldsymbol{x}_{\mathrm{g}}$)
10:     if skill $r$ is HE skill then
11:         $\boldsymbol{x}_{\mathrm{c}} \leftarrow$ get_contact_pos()
12:         $\boldsymbol{t}_p^c \leftarrow \boldsymbol{x}_{\mathrm{c}} - \text{get\_trafo}(O, p)$
13:         update_trafo($r, p, \boldsymbol{t}_p^c$)
14:     end if
15: end for
```

If required, object positions can be explored in multiple dimension, e.g. an arrangement of staples of objects or a single object in a plane. As relative skill's motions are transformed by their parent skills, multiple HE skills can be chained to allow more complex behaviors, which is shown in the following *Experiments* section.

V. Experiments

The robot task is to pick boxes from an uncertain location and place them on a demonstrated position. We use a DLR LWR4 [42], mounted on a linear axis, equipped with a two-finger Robotiq 85 gripper and a JR3 force-torque sensor mounted between robot tip and gripper. The FT measurements caused by the gripper mass are compensated. The robot is operated in Cartesian impedance control with constant stiffness.

As shown in Fig. 8 (right), we use a 2-dimensional arrangement of boxes with 4 possible box locations. The number of boxes can vary from 1 to 4 and the boxes can be stapled in different configurations. The experimenter gives two demonstrations to keep the teaching effort low, where a subset of 2 box configurations has been used. In the first demo, boxes are set at locations p11, p21, p22, while in the second demo, one box is located at p11. As shown in Fig. 5, during each demonstration one box is touched in horizontal and vertical direction in order to identify its position. The demonstrator starts exploring horizontally at the bottom row because it is guaranteed to find a box when one is available instead of starting at the top row. Starting in vertical direction to explore a column would not succeed whenever this column contains no boxes.

After gathering two demonstrations, the task is segmented into contact states, free movements, and gripper states. The result of the segmentation with according labels can be found in Fig. 6. From the labeled segments, the skills HE0, HE1, GC2, GO3, MF4 are identified, where the trailing number denotes the sequential order. Skill relations are learned based on the relation distance matrix $\boldsymbol{V}$ (Fig. 8 left) and represented in the tree structure in Fig. 3. The position data $\boldsymbol{R}_{\mathrm{p},r}$ and

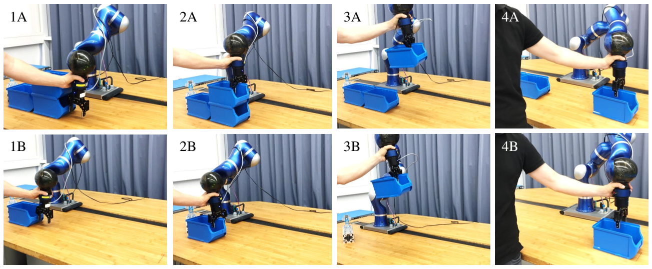


Fig. 5. Demonstration of the adaptive task. In demonstration A, the demonstrator touches the bottom row of boxes in horizontal (1A) and the upper box of the first staple in vertical direction (2A). The box is picked (3A) and placed (4A) on a fixed position on the table. After rearranging boxes for demonstration B, the demonstrator touches the residual box in horizontal (1B) and vertical (2B) direction, picks it (3B) and places it (4B) on the same position on the table.

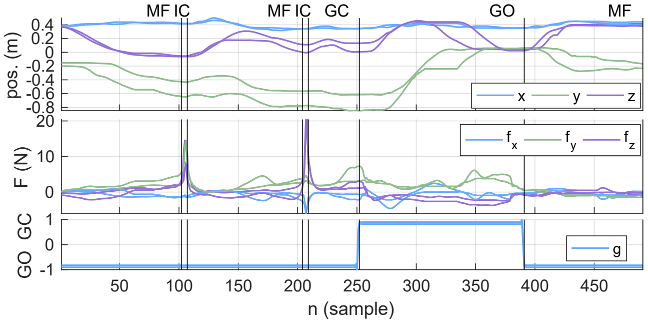


Fig. 6. Segmented task with labeled segments.

learned motion of both HE skills are shown in Fig. 7, where the motion and according key-points of HE1 are already transformed by Algorithm 2. The system is able to generalize to unseen box arrangements, showing the motion and contact forces in Fig. 9 for a new box configuration {p11, p12, p21}. See the accompanying video for the task demonstration and generalization capability of the execution for all seen and unseen box configurations.

The experiments show that no prior knowledge about the initial configuration is needed, as long as the same objects[1] are consistently used in demonstration and execution. Although our approach works only for a bounded region in the workspace and does not explore object orientations, it is tailored to the task requirements of the demonstrator and able to parameterize subsequent motions.

VI. Conclusions and Future Work

We introduced a framework consisting of a demonstration system, a segmentation method and a skill learning approach to execute adaptive tasks relying on haptic exploration skills. We showed that learning of an adaptive task structure is possible within only two demonstrations. Segmentation is applied on averaged demonstration data with the aim to increase robustness. The haptic exploration skill learning considers environmental constraints such that the exploration path starts at a point outside of the demonstrated contact points to avoid unexpected collisions. We validate that the HE behaviors can be extended to more than one dimension and to adapt skills online in a systematically changing environment. The proposed system does not require computer vision and purely relies on the robots sense of touch. Future research addresses the system's capability in situations with uncertain object orientations and previously unseen objects.

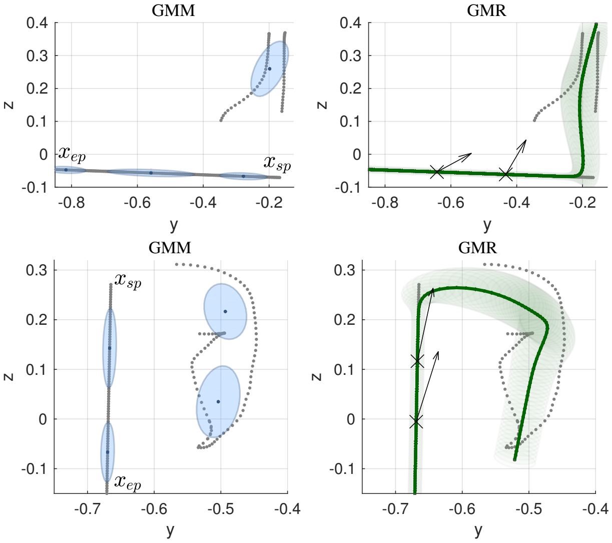


Fig. 7. Top: HE0, with skill's position samples $\boldsymbol{R}_{\mathrm{p},r}$ (grey dots), GMM (blue), key-points (×) and learned GMR motion (green). Black arrows show magnitude and direction of $\hat{\boldsymbol{F}}^{i}_{\mathrm{f,IC}}$. Exploration takes place in approximately horizontal direction. Bottom: HE1, which is executed relative to HE0, exploring in approximately vertical direction.

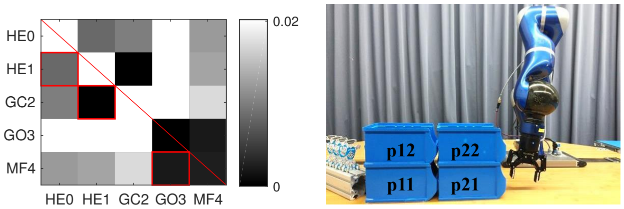


Fig. 8. Left: Relation distance matrix V with extracted relations, where the red boxes mark which skill (in each row) is relative to a skill in the column. Right: Arrangement of boxes at locations p11,p12,p21,p22 and approaching robot at first skill HE0.

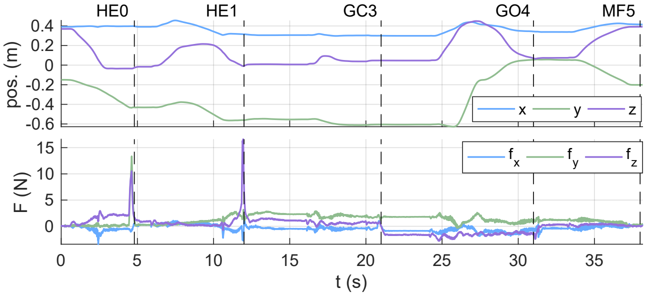


Fig. 9. Measured position and forces for an unseen box configuration {p11, p12, p21}. Dashed lines show where each skill terminated. Detected contacts by the HE skills stop the robot movement and adapt the pick position at GC3. Place position at GO4 is reached with a fixed goal.

Acknowledgments

The first author would like to thank Roman Weitschat for the advice in the application and experiments. This work has been partially funded by the Helmholtz Association.

[1] In theory, each object might be manipulated which is touchable and graspable at the same points as the corresponding object used in demonstration.

REFERENCES

[1] S. J. Lederman and R. L. Klatzky, "Haptic perception: A tutorial," *Attention, Perception, & Psychophysics*, vol. 71, no. 7, pp. 1439–1459, 2009.

[2] G. Robles-De-La-Torre, "The importance of the sense of touch in virtual and real environments," *IEE Multimedia*, vol. 13, no. 3, pp. 24–30, 2006.

[3] M. O. Ernst and M. S. Banks, "Humans integrate visual and haptic information in a statistically optimal fashion," *Nature*, vol. 415, no. 6870, p. 429, 2002.

[4] G. Metta and P. Fitzpatrick, "Better vision through manipulation," *Adaptive Behavior*, vol. 11, no. 2, pp. 109–128, 2003.

[5] Y. Hatwell, A. Streri, and E. Gentaz, *Touching for knowing: cognitive psychology of haptic manual perception*. John Benjamins Publishing, 2003, vol. 53.

[6] S. Calinon and D. Lee, "Learning control," in *Humanoid Robotics: a Reference*, P. Vadakkepat and A. Goswami, Eds. Springer, 2018.

[7] P. Kormushev, S. Calinon, and D. G. Caldwell, "Imitation learning of positional and force skills demonstrated via kinesthetic teaching and haptic input," *Advanced Robotics*, vol. 25, no. 5, pp. 581–603, 2011.

[8] L. Rozo, P. Jiménez, and C. Torras, "Force-based robot learning of pouring skills using parametric hidden markov models," in *9th Workshop on Robot Motion and Control (RoMoCo)*. IEEE, 2013, pp. 227–232.

[9] L. Peternel, T. Petrič, and J. Babič, "Robotic assembly solution by human-in-the-loop teaching method based on real-time stiffness modulation," *Autonomous Robots*, pp. 1–17, 2017.

[10] A. M. Schmidts, D. Lee, and A. Peer, "Imitation learning of human grasping skills from motion and force data," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011, pp. 1002–1007.

[11] V. Koropouli, D. Lee, and S. Hirche, "Learning interaction control policies by demonstration," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011, pp. 344–349.

[12] A. Montebelli, F. Steinmetz, and V. Kyrki, "On handing down our tools to robots: Single-phase kinesthetic teaching for dynamic in-contact tasks," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 5628–5634.

[13] F. Steinmetz, A. Montebelli, and V. Kyrki, "Simultaneous kinesthetic teaching of positional and force requirements for sequential in-contact tasks," in *IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, 2015, pp. 202–209.

[14] L. Rozo, S. Calinon, D. Caldwell, P. Jiménez, and C. Torras, "Learning collaborative impedance-based robot behaviors," *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*, vol. 1, no. 1, p. 1, 2013.

[15] J. R. Medina, M. Lawitzky, A. Mörtl, D. Lee, and S. Hirche, "An experience-driven robotic assistant acquiring human knowledge to improve haptic cooperation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011, pp. 2416–2422.

[16] L. Rozo, S. Calinon, D. G. Caldwell, P. Jimenez, and C. Torras, "Learning physical collaborative robot behaviors from human demonstrations," *IEEE Transactions on Robotics*, vol. 32, no. 3, pp. 513–527, 2016.

[17] L. Rozo, P. Jiménez, and C. Torras, "A robot learning from demonstration framework to perform force-based manipulation tasks," *Intelligent service robotics*, vol. 6, no. 1, pp. 33–51, 2013.

[18] P. Pastor, L. Righetti, M. Kalakrishnan, and S. Schaal, "Online movement adaptation based on previous sensor experiences," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011, pp. 365–371.

[19] A. L. Pais, K. Umezawa, Y. Nakamura, and A. Billard, "Task parameterization using continuous constraints extracted from human demonstrations," *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1458–1471, 2015.

[20] O. Kroemer, C. H. Lampert, and J. Peters, "Learning dynamic tactile sensing with robust vision-based training," *IEEE Transactions on Robotics*, vol. 27, no. 3, pp. 545–557, 2011.

[21] O. Kroemer, H. Van Hoof, G. Neumann, and J. Peters, "Learning to predict phases of manipulation tasks as hidden states," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 4009–4014.

[22] O. Kroemer, C. Daniel, G. Neumann, H. Van Hoof, and J. Peters, "Towards learning hierarchical skills for multi-phase manipulation tasks," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 1503–1510.

[23] S. Niekum, S. Osentoski, G. Konidaris, and A. G. Barto, "Learning and generalization of complex tasks from unstructured demonstrations," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 5239–5246.

[24] T. M. Hagos, M. Suomalainen, and V. Kyrki, "Segmenting and sequencing of compliant motions," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 1–9.

[25] M. Kaboli, K. Yao, D. Feng, and G. Cheng, "Tactile-based active object discrimination and target object search in an unknown workspace," *Autonomous Robots*, pp. 1–30, 2018.

[26] G. De Chambrier and A. Billard, "Learning search polices from humans in a partially observable context," *Robotics and Biomimetics*, vol. 1, no. 1, p. 8, 2014.

[27] Y. Chebotar, O. Kroemer, and J. Peters, "Learning robot tactile sensing for object manipulation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2014, pp. 3368–3375.

[28] P. Pastor, M. Kalakrishnan, L. Righetti, and S. Schaal, "Towards associative skill memories," in *12th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. IEEE, 2012, pp. 309–315.

[29] P. A. Schmidt, E. Maël, and R. P. Würtz, "A sensor for dynamic tactile information with applications in human–robot interaction and object exploration," *Robotics and Autonomous Systems*, vol. 54, no. 12, pp. 1005–1014, 2006.

[30] Z. Pezzementi, E. Plaku, C. Reyda, and G. D. Hager, "Tactile-object recognition from appearance information," *IEEE Transactions on Robotics*, vol. 27, no. 3, pp. 473–487, 2011.

[31] H. M. Choset, S. Hutchinson, K. M. Lynch, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of robot motion: theory, algorithms, and implementation*. MIT press, 2005.

[32] P. Falco, R. Jäkel, C. Natale, and R. Dillmann, "Improvement of human hand motion observation by exploiting contact force measurements." in *Humanoids*, 2011, pp. 141–146.

[33] K. Kronander and A. Billard, "Learning compliant manipulation through kinesthetic and tactile human-robot interaction," *IEEE Transactions on Haptics*, vol. 7, no. 3, pp. 367–380, 2014.

[34] J. Tegin and J. Wikander, "Tactile sensing in intelligent robotic manipulation–a review," *Industrial Robot: An International Journal*, vol. 32, no. 1, pp. 64–70, 2005.

[35] M. R. Pedersen, L. Nalpantidis, R. S. Andersen, C. Schou, S. Bøgh, V. Krüger, and O. Madsen, "Robot skills for manufacturing: From concept to industrial deployment," *Robotics and Computer-Integrated Manufacturing*, vol. 37, pp. 282–291, 2016.

[36] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE transactions on acoustics, speech, and signal processing*, vol. 26, no. 1, pp. 43–49, 1978.

[37] F. L. Markley, Y. Cheng, J. L. Crassidis, and Y. Oshman, "Averaging quaternions," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 4, pp. 1193–1197, 2007.

[38] S. Calinon, F. Guenter, and A. Billard, "On learning, representing, and generalizing a task in a humanoid robot," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 37, no. 2, pp. 286–298, 2007.

[39] S. Calinon, "A tutorial on task-parameterized movement learning and retrieval," *Intelligent Service Robotics*, vol. 9, no. 1, pp. 1–29, 2016.

[40] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: learning attractor models for motor behaviors," *Neural Computation*, vol. 25, no. 2, pp. 328–373, 2013.

[41] A. Ude, B. Nemec, T. Petrić, and J. Morimoto, "Orientation in cartesian space dynamic movement primitives," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 2997–3004.

[42] A. Albu-Schäffer, S. Haddadin, C. Ott, A. Stemmer, T. Wimböck, and G. Hirzinger, "The dlr lightweight robot: design and control concepts for robots in human environments," *Industrial Robot: an international journal*, vol. 34, no. 5, pp. 376–385, 2007.