Robot learning of industrial assembly task via human demonstrations

Maria Kyrarini¹ · Muhammad Abdul Haseeb¹ · Danijela Ristić-Durrant¹ · Axel Gräser¹

Received: 31 December 2016 / Accepted: 20 March 2018 / Published online: 6 April 2018
© The Author(s) 2018

Abstract

Human–robot collaboration in industrial applications is a challenging robotic task. Human working together with the robot at a workplace to complete a task may create unpredicted events for the robot, as humans can act unpredictably. Humans tend to perform a task in a not fully repetitive manner using their expertise and cognitive capabilities. The traditional robot programming cannot cope with these challenges of human–robot collaboration. In this paper, a framework for robot learning by multiple human demonstrations is introduced. Through the demonstrations, the robot learns the sequence of actions for an assembly task (high-level learning) without the need of pre-programming. Additionally, the robot learns every path as needed for object manipulation (low-level learning). Once the robot has the knowledge of the demonstrated task, it can perform the task in collaboration with the human. However, the need for adaptation of the learned knowledge may arise as the human collaborator could introduce changes in the environment, such as placing an object to be manipulated in a position and orientation different from the demonstrated ones. In this paper, a novel real-time adaptation algorithm to cope with these changes in the environment, introduced by the human factor, is proposed. The proposed algorithm is able to identify the sequence of actions needed to be performed in a new environment. A Gaussian Mixture Model-based modification algorithm is able to adapt the learned path in order to enable robot to successfully complete the task without the need of additional training by demonstration. The proposed framework copes with changes in the position and orientation of the objects to be manipulated and also provides obstacle avoidance. Moreover, the framework enables the human collaborator to suggest different sequence of actions for the learned task, which will be performed by the robot. The proposed algorithm was tested on a dual-arm industrial robot in an assembly scenario and the results are presented. Shown results demonstrate a potential of the proposed robot learning framework to enable continuous human–robot collaboration.

Keywords Robot learning · Human–robot collaboration · Modified GMM algorithm · Industrial assembly task · Adaptation to environmental changes

Electronic supplementary material The online version of this article (<https://doi.org/10.1007/s10514-018-9725-6>) contains supplementary material, which is available to authorized users.

✉ Maria Kyrarini
mkyrar@iat.uni-bremen.de

Muhammad Abdul Haseeb
haseeb@iat.uni-bremen.de

Danijela Ristić-Durrant
ristic@iat.uni-bremen.de

Axel Gräser
ag@iat.uni-bremen.de

¹ Institute of Automation (IAT), University of Bremen, Bremen, Germany

1 Introduction

There is a growing interest for human–robot collaboration in manufacturing (Nikolaidis and Shah 2013; Pedersen et al. 2016; ABB 2014; KUKA Robotic 2017). Whereas in the past it was all about replacing the work of humans with that of robots, today the focus is much more on the robot as an assistant or collaborator. The work presented in this paper contributes to the research and development in the human–robot collaboration in industrial applications, by proposing a framework for the manufacturing assembly scenarios with the following goals:

- robot learns the sequence of actions (high-level learning) needed to perform an assembly task from the human collaborators without pre-programming, which includes the

object moving and grasping/releasing actions (low-level learning of trajectories),

- robot adapts the learned paths to environmental changes, which are introduced by the human collaborators, without additional training and
- robot performs the learned sequence of actions in different order of occurrence after suggestion from the human collaborator. If the human collaborator is satisfied with the new sequence of actions, the robot learns it.

Traditionally, the industrial robot manipulators are widely used in production lines as they are precise and swift. However, traditional industrial manipulators are pre-programmed by human programmers to perform a specific task in a well-structured and constant environment. In the case of environmental changes, the human programmer would have to take into account all the possible events in advance, and would need to code the actions of the robot as response to all these different events. This process would involve thorough testing of each different case. However, if new circumstances, which were not considered beforehand, appear, the entire pre-programming process would be repeated for which the robot would have to stay out of service until it is re-programmed. In order to make industrial robot programming easier, several technologies have been used in past years. One of them is a framework that enables human collaborators to simply demonstrate the task to the robot as opposite to time consuming and high technical skills demanding traditional robot programming methods (Zhang et al. 2016).

Additionally, regardless of their high precision, accuracy and speed, nowadays industrial robots are not able to perform every task autonomously. Human workers are still needed in assembly and manufacturing production lines, as they have cognition that enables them to respond and adapt quickly to uncertain environments and unpredicted events in a way that is still not possible for robots. Therefore, currently, research is focused on human–robot interaction in industry and on the human–robot synergetic work to perform a given task (Tellaeche et al. 2015).

However, humans sharing the same workspace and manipulating the same objects with robots may introduce uncertainty in the robot's working environment. Let's take as an example a human and robot collaboration in an assembly of an object consisting of several parts. Firstly, in the training phase, the human demonstrates to the robot the actions that have to be performed in order to complete the task. Actions could be the following: pick object-part 2, move and place it on top of the object-part 1, and then pick object-part 3, move and place it on top of the object-part 2. In the working phase, the robot–human team is jointly assembling the objects. Firstly, the human co-worker places the object's parts on the working table and the robot performs the actions of

manipulating the object's parts, learned during the training phase, so to position them on top of each other. Since the human collaborator is not as precise as the robot, he/she cannot place the object's parts in every repetition of the assembly task exactly at the object's parts positions and orientations demonstrated during the training phase. Furthermore, the robot may cooperate with different human collaborators during a working day, which may introduce uncertainties for the robot's performance as each human collaborator could position and orientate the object's parts differently. Moreover, by mistake, human collaborators may place some objects, such as working tools, on the working table beside the parts of the objects to be assembled introducing so obstacles in robot's working environments. This may cause collision, bad handling or other damages to the robot. In order to perform the task successfully and without interruptions in spite of the obstacles in the workspace, the robot needs to adapt to environmental changes and to avoid the obstacles without any additional training, alike the way a human collaborator would react to an obstacle appearance. In this paper, a novel framework that can enable robot to adapt the learned task to environmental changes without the need of additional training is suggested.

The proposed framework is based on an approach that enables the robot to learn object manipulation tasks involving as many objects as the industrial assembly task requires. This learning can be achieved by multiple human demonstrations (demos), which introduces both the autonomous adaptation of the learned actions to the different objects' poses and the obstacle avoidance as requirements. In the presented framework, robot learns the sequence of actions for the demonstrated task, which are adapted in real-time to environmental changes if they appear. Additionally, the human collaborator is able to change the sequence of actions to perform the assembly task in an alternative way, and to provide feedback to the robot on its performance, so the robot learns the sequence with positive feedback.

This paper is organized as follows. The related work is presented in Sect. 2. The overview of the proposed novel robot learning framework is given in Sect. 3. The off-line learning of sequence of actions is presented in Sect. 4, while the online working phase for the adaptation to environmental changes and obstacle avoidance is described in Sect. 5. The results of the experiments with the real robot are shown in Sect. 6 and the conclusion and future work are discussed in Sect. 7.

2 Related work

Robot learning from demo is a method in which the human “teacher (demonstrator)” shows (demonstrates) his/her performance to the robot learner and the robot tries to imitate the

demonstrated performance. Each process of robot learning from demo is composed of three fundamental steps: *observation*, *representation* and *reproduction* (Dindo and Schillaci 2010).

There are different methods to gather the datasets (observation of the demo) as summarized in Argall et al. (2009). These methods are: *tele-operation* with joystick or different other interfaces (Forbes et al. 2015; Pathirage et al. 2013), *kinesthetic teaching* (Kormushev et al. 2010; Kober and Peters 2009), *wearable sensors placed on the teacher's body* (Krug and Dimitrov 2013; Leitner et al. 2014), and *vision-based external sensors* (Quintero et al. 2014; Zhang et al. 2016).

Given a dataset of the skills or task executions that have been acquired using one or combination of the methods mentioned above, the robot learner must be able to learn a skill or a task from the dataset. In some studies, instead of using one dataset, datasets of multiple demos (Calinon et al. 2010b), (Ekvall and Kragic 2006) are used. In the work presented in this paper, a combination of multiple datasets from demos via kinesthetic teaching with vision-based external sensor is used.

There are different approaches to represent (abstract) and to learn the skill or the task from the datasets of demos. These approaches are grouped by Calinon et al. (2010b) and Billard et al. (2007) into two categories: *Skill learning at trajectory level* (low-level learning) and *Symbolic task learning* (high-level learning). One of the goals of the work presented in this paper has been to create a bridge between those two categories by combining high-level and low-level learning.

2.1 Skill learning at the trajectory level (low-level learning)

The main goal of this learning approach is to enable robot to learn the basic movements or gestures (motor skills). However, the approach does not allow reproducing of more complicated high-level tasks.

One popular method from this category is *Dynamic Movement Primitives (DMP)* (Schaal et al. 2005; Pastor et al. 2009), which allows robot to learn a non-linear differential equation based on the movement observed by one demo. In order to achieve generalization, in the *modified DMP method* (Park et al. 2008), the learned differential equation can be adapted for different start and goal desired positions of the movement or for the obstacle avoidance. Moreover, since the DMP algorithm works with one demo only, it was combined with *Gaussian Mixture Model (GMM)* to learn from multiple demos (Yin and Chen 2014). DMP can also be combined with *Reinforcement learning* to initialize the set of primitives for the demonstrated skill in order to improve and adapt the encoded skill by learning optimal policy parameters (Kormushev et al. 2010; Kober and Peters 2009). Another approach

to model the robot motion with dynamic systems is *Stable Estimator of Dynamic Systems (SEDS)* (Khansari-Zadeh and Billard 2011). SEDS method is able to model the robot skill from multiple demos with different starting positions and same target position.

Hidden Markov model (HMM) based approaches have been also used for learning at trajectory level. In Calinon et al. (2011), the time and space constraints are learned by the HMM based framework. Additionally, a comparison between HMM with *time dependent Gaussian Mixture Regression (GMR)* and DMP is presented in the work by Calinon et al. (2010a), where it was shown how each framework can learn a trajectory based on multiple demos. Few demos are firstly taken having the same initial points but different goal point and after that few demos with the same goal point and different initial points are considered. The shown reproduction satisfied the constraints of the demonstrated task.

Another approach for the skill learning at the trajectory level is the estimation of a distribution, which is determined by *Gaussian Mixture Models (GMM)* (Calinon et al. 2007; Sabbaghi et al. 2014), where *Gaussian Mixture Regression (GMR)* is used to generalize the trajectories. GMM can also be combined with *virtual spring-damper system (DS-GMR)* (Calinon et al. 2014) or with *linear attractor system* combined with GMR (Mühlig et al. 2012) so to enable the robot to perform the learned skill with new start and target positions. Moreover, GMM/GMR is combined with DMP for generalizing the learned trajectory to new goal point and obstacle avoidance (Ghalamzan et al. 2015).

Furthermore, an algorithm based on discrete Laplace–Beltrami operator (Nierhoff et al. 2016) enables online adaptation of a learned trajectory to dynamic environmental changes while keeping the shape of the trajectory similar to the shape of the original trajectory.

In the approach presented in this paper, GMM is used for the low-level trajectory learning because it is a method which enables automatic extraction of trajectory constraints (Calinon et al. 2007). Also, a novel modified GMM/GMR method is used, which enables the robot to adapt the trajectory to environmental changes and to provide obstacle avoidance. The GMR is a real-time and analytic solution to produce smooth trajectory from the GMM/modified GMM. The produced trajectory can be used directly for the efficient robot control. The task is demonstrated offline multiple times in the unchanged environmental conditions and the adaptation module modifies the trajectories to meet the environmental changes appeared in the online robot functioning (working) phase. Table 1 shows a comparison between the related work and the presented novel approach for the skill learning at trajectory level with respect to number of human demos (single or multiple), method for reproduction, and possibilities for the adaptation to environmental changes.

Table 1 Comparison of related work for the low-level learning

Method for learning	Demonstrations from human	Method for reproduction	Adaptation to environmental changes			References
			Different start pose	Different goal pose	Obstacle avoidance	
DMP	Single	Modified DMP	✓	✓	✓	Park et al. (2008)
SEDS	Multiple	SEDS	✓			Khansari-Zadeh and Billard (2011)
HMM	Multiple	HMM	✓	✓		Calinon et al. (2010a)
GMM	Single/multiple	GMR				Calinon et al. (2007)
GMM	Single/multiple	DS-GMR	✓	✓		Calinon et al. (2014)
GMM	Multiple	GMR/DMP		✓	✓	Ghahraman et al. (2015)
Any method	Not specified	Laplace–Beltrami	✓	✓	✓	Nierthoff et al. (2016)
GMM	Multiple	Modified GMM/GMR	✓	✓	✓	Presented novel approach

2.2 Symbolic task learning (high-level learning)

In symbolic learning, the task is encoded according to the sequences of predefined motion elements. This approach allows the robot to learn the sequence of actions, so the robot can learn high-level task (Ekvall and Kragic 2006). A disadvantage of symbolic learning is that it relies on a priori knowledge to be able to abstract the important key-points of the demonstrated task.

For the abstraction and recognition of high-level skills, *Hidden Markov Models (HMMs)* have been widely used. HMM-based frameworks are used to generalize a task demonstrated multiple times to a robot (Kruger et al. 2010; Akgun and Thomaz 2016). The redundancies across all demos are identified and used for the reproduction of the task by the robot.

Another approach for the high-level task learning is the *Growing Hierarchical Dynamic Bayesian Network (GHDBN)* which is used for the representation and reproduction of complex actions from data (Dindo and Schillaci 2010). The GHDBN is a two-level Hierarchical Dynamic Bayesian Network (HDBN) where one level describes the high level representation of the task and the other describes the low level behavior of the robot.

A probabilistic approach for the representation and learning of complex manipulation based on multi-level *Hierarchical Hidden Markov Model (HHMM)* is presented in Patel et al. (2014), where the complex manipulation tasks are decomposed into multiple levels of abstraction to represent the actions in simpler way called action primitives.

A segmentation GMM-based framework provides task learning from a single demo (Lee et al. 2015). The GMM is combined with *Principal component analysis (PCA)* for dimensionality reduction and selection of the number of segments.

In the approach presented in this paper, the authors use a segmentation algorithm to split the demonstrated task to individual actions, and the Gaussians from the GMM model are grouped based on those actions.

2.3 Human–robot collaboration in industrial applications

As the robots are increasingly being used on factory floors, researchers are looking for the ways to help humans to work safe and more efficient with their electromechanical counterparts. In Weistroffer et al. (2014) the focus is on the acceptability of human–robot collaboration in industrial environments by designing a use case in which a human and a robot work side-by-side on the automotive assembly lines in physical and virtual environments. The results showed that the human collaborators preferred the virtual environment as the direct physical contact of the robot and the human collab-

Fig. 1 Overview of the suggested robot learning framework for assembly task (Color figure online)

operator is avoided. In some other works, the implementation of methods that enabled acceptance of direct human–robot object manipulation was investigated. For example, in Maeda et al. (2017) an imitation learning method based on probabilistic movement primitives is proposed and it is tested in the hand-over of objects between the robot and the human.

In this paper, the authors present a framework which enables human–robot cooperation in an industrial assembly application combined with the virtual environment-based situation awareness, which enables testing of robot's movements before their execution.

3 Overview of the suggested robot learning framework

The overall structure of the proposed framework is shown in Fig. 1. The dual-arm industrial *pi4 Workerbot 3* (<http://www.pi4.de/english/systems/workerbot.html>) is used as the framework robotic platform. It consists of two 6-degrees of freedom UR10 robotic arms controlled by the gravity-compensation controllers, which enables the use of the

kinesthetic teaching for the human demo of the task to the robot. In the kinesthetic teaching, the human teacher demonstrates the task by grasping the robot's end-effector and moving it along the appropriate trajectories throughout the task demo. In the used two-arm robotic platform, the end-effector of each robotic arm is an industrial vacuum gripper with two actuation status, "On" denoting the activated object manipulation (gripping) and "Off" denoting the not-activated gripping status. A *Kinect for Xbox One* camera (<http://www.xbox.com/en-US/xbox-one/accessories/kinect>) is mounted onto the head of the Workerbot and it is used as the vision-sensor which provides information about the working place, which is the working table located in front of the Workerbot and about the objects placed on the working table. The Kinect sensor features an RGB camera and a Depth sensor, and the Point Cloud Library (PCL—<http://pointclouds.org/>) is used to generate the point cloud of the scene from the Kinect sensor data and to process it.

As illustrated in Fig. 1, the proposed framework is organized in two main modules, the *offline learning phase* and the *online working phase*. A sub-module which is common for both main modules is the *Environmental Perception*

Fig. 2 **a** The demonstrated scenario of the assembly of the robot gripper. **b–d** Initial scene of the gripper assembly scenario for three different cases (Color figure online)

module. This module processes the Point Cloud Data from the vision-sensor. Firstly, plane detection is performed to locate the working table with the objects placed on it. Secondly, a 3D-point cloud based processing object detection and recognition is performed. The output of this module is the information on position, orientation and dimension of every object on the working table with respect to the world coordinate system (Fang 2016).

During the learning (teaching) phase, all the actions needed to complete the assembly task are demonstrated via kinesthetic teaching several times by one or more human teachers (demonstrators). The working hypothesis of multiple demos is that the learning from a single demo has limitations as the human teacher may make mistakes during the demo so that the robot could be vulnerable to those mistakes. Also, the human teacher has a low precision compared to the robot and may perform unnecessary movements in attempts to be very precise in positioning the robot's end-effector. Furthermore, as the different human demos lead to differently demonstrated skills, an optimally learned skill could outcome from a combination of different demos.

Because of this, in the presented work, learning from multiple demos is suggested.

For the evaluation of the presented framework, an assembly of a robot gripper is selected as an appropriate industrial scenario. The assembly of the robot gripper consisting of 5 parts is illustrated in Fig. 2a. As shown, there are 4 objects, 4 gripper's parts, which shall be assembled with the 5th, base part, to complete the assembly task. The manipulation of each gripper's part, so called subtask, is done through the sequence of actions: "start arm moving", "object grasping", "arm moving while carrying the grasped object", "object releasing" and "arm moving away from the working table". During the demos, both robot arms can be used, but never at the same time because of safety. Different human collaborators are asked to demonstrate the complete task multiple times in the following order: manipulate the top part with the left arm (subtask 1), manipulate the black part with the right arm (subtask 2), manipulate the left side part with the left arm (subtask 3) and manipulate the right side part (subtask 4). The scenes after every manipulation are shown in Fig. 2a.

3.1 Offline learning phase

The offline learning module consists of the following 2 sub-modules:

- *Data acquisition module* This module records and stores in the robot database the joint angles, the end-effector pose of both robotic arms with respect to the world coordinate system as well as the actuation status of the grippers during the human demos. Each demo presents the complete assembly task. Additionally, this module stores in the database all the data provided by the Environmental Perception Module.
- *Learning of sequence of actions module* This module enables the robot to learn the sequence of actions needed to complete the demonstrated object manipulation task without the need for pre-programming by the human collaborator. The learning module is organized into two layers:
 - *High-level task learning* comprising *Symbolic Task Learning* and consisting of the following steps: automatic task segmentation into actions (Sect. 4.1.1), which splits the task into subtasks and each subtask into actions, and labeling of the involved objects with specific IDs (Sect. 4.1.2).
 - *Low-level skill Learning* comprising the *Skill Learning at the trajectory level*, which consists of two steps resulting in the learned end-effector path. These steps are: selection of similar demos (Sect. 4.2.1) and Gaussian mixture model (GMM)/Gaussian mixture regression (GMR) (Sect. 4.2.2).

The outcomes of the Learning of Sequence of Actions module are stored in the database and they are: the task segmented into actions; the position, orientation, dimensions and assigned ID of every involved object (as shown in Fig. 2b), the learned Gaussian mixture model (GMM) of the demonstrated end-effector paths and the learned end-effector paths generated by the GMR.

3.2 Online working phase

The online working phase is the phase in which the robot performs the learned task in real-time and the human collaborator is working together with the robot to complete the assembly task. In the presented scenario, the robot manipulates the gripper parts so to place them in appropriate places next to each other, while the human collaborator is supposed to screw the parts together in order to successfully complete the assembly task. Additionally, human collaborator also provides inputs to robot's execution of the object manipulation tasks. Namely, before the task execution the learned sequence

of actions will be displayed via Graphical User Interface (GUI) to the human collaborator. The human collaborator has than opportunity to either confirm the learned sequence of actions or to suggest new sequence of actions needed for completion of the task. Also, at the end of the robot's task execution, the human collaborator provides a feedback (positive/negative) on the robot performance of particular sequence of actions. If the feedback is positive, the executed sequence of actions is stored so that only the sequences with the positive feedbacks are displayed to the human collaborator for the confirmation/adaptation before the next task execution.

The online phase consists of the following:

- *Real-time adaptation module* This module is organized into two layers:
 - *High-level adaptation module* firstly performs object identification by identifying the objects on the working table based on their dimensions and pose. For the sake of explanation simplicity, the objects involved in the task during demo, which are stored in the database, are referred as *original objects* and the objects present in the scene in the working phase are *current objects*. The current objects are labeled based on their role in the task. In order to identify which current object matches which original object, the dimensions of the objects are compared. If there are more original objects with the same dimensions as the current object, the distance between the pose of each such original object and the pose of the current object is calculated. A current object is labeled with the ID of the original object of the same dimensions, which is the closest to the pose of the current object. The current objects, whose dimensions are not matched to any of the original objects, are treated as obstacles. An example of object identification and ID assignment is shown in Fig. 2c.
 - Once the objects identification is completed, the learned sequence of actions is displayed via GUI to the human collaborator and he/she can confirm the sequence of actions or select a different one. For example, in the presented assembly scenario shown in Fig. 2a, the human collaborator can change the originally demonstrated sequence of actions by changing the order of the object manipulation. The originally learned sequence of actions and the newly suggested sequence of actions for the assembly of the same parts are shown in Fig. 2b, d respectively.
 - *Low-level adaptation module* is a novel GMM-based method for the real-time adaptation of the learned low-level skills (trajectories) to the new environmental conditions. The new environmental conditions

Fig. 3 Task segmentation into actions for the demonstrated gripper assembly scenario. **a** Task segmentation to actions for left arm, **b** task segmentation to actions for right arm (Color figure online)

could arise due to changes in position and orientation of the objects as well as due to obstacles. The GMM-based adaptation is followed by the GMR method to generate the adapted path to be followed by robot's end-effector so that the object manipulation task can be completed. Further details of this module are given in Sect. 5.

- *Virtual environment-based situation awareness module*
A virtual environment has been developed using the ROS-based tool rviz (<http://wiki.ros.org/rviz>) to illustrate the robot's environmental awareness and the robot's execution of the learned task. For safety reasons in the suggested framework, the human collaborator firstly observes what the robot intends to do in the virtual environment and then, if safety criteria are satisfied, confirms that the real robot can perform the visualised sequence of actions.

4 Robot learning of sequence of actions for assembly task (offline learning phase)

4.1 High-level learning

This Module is responsible to learn the sequence of actions for the demonstrated assembly task. For the sake of explanation clarity, the following notations (Table 2) are introduced for this section.

4.1.1 Task segmentation into actions

The inputs to the high-level task learning are the datasets $\{L_m, R_m\}$. As example, the z dimension (blue line) of one demo of the gripper assembly scenario is shown in Fig. 3a, b for the left and right robot arm, respectively. As first, the task segmentation algorithm splits the task into subtasks. In Fig. 3a, b, the notations "Left Subtask n_L ", "Right Subtask n_R " mean that the n_L th, n_R th subtask, was performed by the

Table 2 Notations for Sect. 4

<i>Data acquisition</i>	
D	Total number of demonstrations
qx, qy, qz, qw	The four quaternions
L_m, R_m	Matrix of the left and right end-effector pose in Cartesian space $\{x, y, z, qx, qy, qz, qw\}$ and $m = 1, \dots, D$
$ee.on, ee.off$	Index of the actuated and not-actuated end-effector state
<i>Task segmentation to actions</i>	
P_L, P_R	Total number of times the left and right arm, respectively, was used to perform a subtask
n_L, n_R	the number of subtasks for left and right robot arm, respectively
a	Sub-path between “start moving” and “grasp” actions
b	Sub-path between “grasp” and “release” actions
c	Sub-path between “release” and “stop moving” actions

left, right robot arm, respectively. After this, the algorithm segments the subtasks to the following sequence of actions: “start arm moving”, “object grasp”, “object release” and “stop arm moving” with the sub-paths, b and c that should be followed by the end-effector between those actions (Table 2). Segmentation of the subtasks into the sequence of actions is illustrated in Fig. 3a, b for the z-coordinate (dimension) of the end-effector. The segmented sub-paths a, b, c are used as the input to the low-level learning (Sect. 4.2).

4.1.2 Object labeling

For the high-level learning of the task, that is for the learning of the sequence of task actions, the manipulated objects during the demonstration are labeled with specific IDs that denote the robot arm (left or right), which was used for the object manipulation and the role of the object in the subtask. For example, the ID “*left_pick_n_L*” means that the identified object was grasped (picked up) by the left robot arm during n_L th subtask. Since the proposed framework is focused on the assembly scenario, the “*left_pick_n_L*” object will be assembled with another object, which will then be labelled as “*left_place_n_L*”. In this ID “place” could mean that the object “*left_pick_n_L*” shall be placed onto the “*left_place_n_L*”. or shall be placed next to, that is aligned to, the “*left_place_n_L*”, depending on the demonstrated task. This labeling of the objects to be manipulated is a generic method where no matter how many objects are involved in the manipulation task the unique ID is assigned accordingly. Figure 2b shows the labeling of objects for the presented assembly scenario. One object can have more than one IDs, depending on its role in the task. For example the Base Part in Fig. 2b–d is denoted as “*right/left_place_1/2*” meaning that all gripper parts, those

picked up by the left and right robot arms, shall be finally assembled with the base part.

4.2 Low-level learning

4.2.1 Selection of similar demonstrations

During human demos via kinesthetic teaching, there are two main problems: the different duration of each demo and big variety in demos. In Fig. 4a, 9 different demos of the right subtask 1 are shown as an example. A widely used solution for the first problem is Dynamic Time Warping (DTW) (Akgun et al. 2012; Sabbaghi et al. 2014). Using of DTW also was proposed by the authors as possible solution of the second problem (Kyrarini et al. 2016, 2017). However, one disadvantage of DTW is the complexity of the algorithm which is time consuming (Movchan and Zymbler 2015).

In the presented work, a novel method for selection of similar demos without the use of DTW is proposed. More precisely, the proposed algorithm is able to select similar sub-paths. The sub-paths consists of data-points in $7 + 1$ dimensions $\{x, y, z, qx, qy, qz, qw, t\}$, where t is temporal variable, for the left and right arm are represented by $L_{m,n_L,l}, R_{m,n_R,l}$, respectively, where $n_L = \{1, \dots, P_L\}$, $n_R = \{1, \dots, P_R\}$ and $l = \{a, b, c\}$. The proposed algorithm is implemented in the following 3 steps:

- *Step 1: Sub-paths of the same length (duration)*

Since the sub-paths have different lengths (different duration of the end-effector movements between the actions of a sub-task) due to different speeds of performing the tasks during the demos, a variation of the Ramer–Douglas–Peucker (RDP) algorithm (Ramer 1972; Douglas and Peucker 1973) is used to provide sub-paths of the same length. The input to this algorithm are the desired length and the sub-paths $L_{m,n_L,l}, R_{m,n_R,l}$, and the output is the RDP sub-paths $L_RDP_{m,n_L,l}, R_RDP_{m,n_R,l}$. The desired length of the specified sub-path l is defined as the shortest length among the sub-paths of every group, where a group of the sub-paths consists of all the demos of the n_L th subtask for the left arm, that is of all the demos of the n_R th subtask for the right arm. In Fig. 4b, the output of this step is shown for nine different demos of the right subtask 1.

- *Step 2: Calculation of similarity between the sub-paths of the same group*

As mentioned above, the sub-paths are organized into groups. For every group, the dimensions $\{x, y, z\}$ are normalized in the range of $[-1, 1]$ and a similarity matrix *dist* is computed as:

Fig. 4 Example of learned GMM and modified GMM for the x -dimension of the end effector's pose from the demonstrated Right Subtask 1 of the gripper assembly scenario. The *Pick_Right_1* object is moved in positive x -dimension by 0.02 m in comparison to the demos while the *Pick_Rlace_1* is in same place as in demos **a** 9 different demos of the Right Subtask 1, **b** right subtask is segmented into a, b, c sub-paths and RDP algorithm is applied, **c** selected similar demons, learned

GMM and identification of Gaussians from the modification of Gaussian Means algorithm (step 1, 2, 4), **d** generated path via GMR from the learned GMM **e** modification of pick and place Gaussians based on the position of current objects—Gaussian Means algorithm (step 3), **f** modification of Gaussian Means algorithm (step 5), **g** modification of Gaussian Means algorithm (step 6), **h** generated path via GMR from the modified GMM (Color figure online)

$$\begin{aligned} dist_n(i, j) = & w1 * distPath(A_RDP_{i,n_A,l}, A_RDP_{j,n_A,l}) \\ & + w2 * distIP(A_RDP_{i,n_A,l}, A_RDP_{j,n_A,l}) \end{aligned} \quad (1)$$

where: $i, j = \{1, \dots, D\}$, $A = \{L, R\}$, $n_A = \{n_L, n_R\}$. $distPath$ is the 7 dimensional $\{x, y, z, qx, qy, qz, qw\}$ Manhattan point-to-point distance between the points of two sub-paths, $distIP$ is the 7 dimensional Manhattan distance between the important points (IP) of two sub-paths where important points in assembly task are the grasping and releasing points, $w1$ and $w2$ are the weights which satisfy the following equations: $0 \leq w1 \leq 1$, $w2 = 1 - w1$.

The *similarity* vector is calculated as follows:

$$similarity(i) = \sum_{j=1}^D dist_n(i, j), \forall j \in \{1, \dots, D\} \quad (2)$$

- *Step 3: Selection of demonstrations for every sub-path group*

The presented algorithm gives the option to the human demonstrator to select a number of desired demos $D_s < D$. For every group of sub-paths, the demo with the smallest value in the vector *similarity* is selected as the “reference” demo r . Then, the algorithm selects the reference demo and $D_s - 1$ demos that have minimal distance $dist_n(r, j), \forall j \in \{1, \dots, D\}, j \neq r$.

The selected demos for every sub-path group are joined together to create a complete path for each subtask as follows:

$$A_RDP_{k,n} = A_RDP_{k,n,a} \cup A_RDP_{k,n,b} \cup A_RDP_{k,n,c} \quad (3)$$

where $k = \{1, \dots, D_s\}$.

The above demos are the output of the selection of similar demos. In Fig. 4c, the selected demos of the right subtask 1 of the gripper assembly scenario is shown as example. Additionally, the gripper actuated and not-actuated state is mapped to the $A_RDP_{k,n}$. The notation of the matrix index $A_RDP_{k,n}$ is $ee.on = size(A_RDP_{k,n,a})$ for the actuated state and $ee.off = size(A_RDP_{k,n,a} \cup A_RDP_{k,n,b})$ for the not-actuated state, where *size* is a function to find the number of data-points.

4.2.2 Gaussian mixture model (GMM)/Gaussian mixture regression (GMR)

The input to GMM is the above described result on the selection of similar demos. The GMM is used to extract constraints of the aligned trajectories (Calinon et al. 2007) and GMR is used to produce the learned end-effector path which can be used to control the robot efficiently (Calinon 2009).

The set of selected demos are fed to the learning system that trains the GMM in order to build the probabilistic model of the data. Every demo consists of data-points $\beta_\gamma = \{\beta_s, \beta_t\}$, where $\beta_s \in R^7$, s is spatial variables, $\beta_t \in R$. In the presented approach the dimensionality of the data-points is equal to 8; 7 dimensions $\{x, y, z, qx, qy, qz, qw\}$ and the temporal dimension. In the learning phase, the model is created with a predefined number N of Gaussians. Each Gaussian consists of the following parameters: mean vector, covariance matrix and the prior probability. Each Gaussian has a dimensionality 8. The probability density function $p(\beta_\gamma)$ for a mixture of N Gaussians is calculated based on the following equation (Calinon 2009)

$$p(\beta_\gamma) = \sum_{n=1}^N \pi_n \frac{1}{\sqrt{(2\pi)^8 |\Sigma_n|}} e^{-\frac{1}{2}[(\beta_\gamma - \mu_n)^T \Sigma_n^{-1} (\beta_\gamma - \mu_n)]} \quad (4)$$

where: π_n are the prior probabilities, $\mu_n = \{\mu_{n,t}, \mu_{n,s}\}$ are the mean vectors and $\Sigma_n = \begin{pmatrix} \Sigma_{n,t} & \Sigma_{n,ts} \\ \Sigma_{n,st} & \Sigma_{n,s} \end{pmatrix}$ are the covariance matrices of the GMM. The parameters (prior, mean and covariance) of the GMM are estimated by the expectation-maximization (EM) algorithm (Dempster et al. 1977). In Figs. 4c and 6a, the GMM for one subtask of the gripper assembly scenario is shown.

The learned GMM parameters for the task are given as input to the GMR in order to generalise a path. The GMR has the advantage that generates a fast and optimal output from the GMM (Calinon 2009). The output path $\hat{\beta}$ of the GMR is calculated as:

$$\hat{\beta} = \left\{ \beta_t, \sum_{n=1}^N a_n \hat{\beta}_{n,s} \right\} \quad (5)$$

where: $a_n = \frac{p(\beta_t|n)}{\sum_{n=1}^N p(\beta_t|n)}$ and

$$\hat{\beta}_{n,s} = \mu_{n,s} + \Sigma_{n,st} (\Sigma_{n,t})^{-1} (\beta_t - \mu_{n,t}), \forall n = \{1, \dots, N\}.$$

In Figs. 4d and 6b, the generated GMR path from the learned GMM is shown.

5 Low-level adaptation of the learned task (online working phase)

An algorithm based on GMM is used for the adaptation of a learned task (end-effector path) to different poses of the involved objects and to obstacle avoidance. This algorithm starts from the real-time object identification so that the robot knows which of the identified objects shall be grasped (picked

Table 3 Notations for Sect. 5

<i>Offline Learning Phase: Output</i>	
N	Total number of Gaussians in the GMM model
i	$i = \{1, \dots, 7\}$ represents the dimensions $\{x, y, z, qx, qy, qz, qw\}$ respectively
j	$j = \{1, \dots, N\}$ the number of the Gaussian respectively
k	$k = \{1, \dots, 7\}$ represents the position, orientation and dimensions for the objects $\{x, y, z, \text{yaw angle}, \text{length}, \text{width}, \text{height}\}$, respectively
$GM_{i,j}$	Mean values of the learned GMM for 7 dimensions of the end effector of each robotic arm
$pick$	The Gaussian that corresponds to the grasping point $ee.on.index$ in the learned path $\hat{\beta}$
$place$	The Gaussian that corresponds to the releasing point $ee.off.index$ in the learned path $\hat{\beta}$
$Obj.pick_k$	Original object position and orientation for an object to be picked up (grasped)
$Obj.place_k$	Original object position and orientation for an object to be placed (released)
$ee.on_i, ee.off_i$	Position and Orientation (quaternion) of the end effector when gripper actuation status is on, off in the learned path $\hat{\beta}$
<i>Online working phase: environmental perception</i>	
$Obj.pick'_k$	Current object position and orientation for a pick object
$Obj.place'_k$	Current object position and orientation for a place object
$Obj.obst'_k$	Current object position and orientation for an obstacle
<i>Online working phase: output of the modification of Gaussian means of the learned GMM</i>	
$GM'_{i,j}$	Mean values of the modified GMM for 7 dimensions of the end effector of each robotic arm

up), with which robot arm and where those objects shall be released (placed). Secondly, the algorithm uses the human collaborator input on the sequence of actions, either confirmation of the learned sequence or suggestion of a different sequence (as explained in Sect. 3). Then, a novel robust algorithm is introduced that modifies the means of the learned Gaussian Model for the task in order to adapt to the new pose of the objects and to avoid collision with obstacles in real time. Once the new GMM has been created, the GMR is used to generate the new path (low-level) which is mapped to the learned actions (high-level) of the task. The execution of the task is performed by the real robot. The notations that are used in this section can be found in Table 3.

5.1 Modification of Gaussian means of the learned Gaussian mixture model

In this section, the proposed algorithm for modification of the means of learned Gaussians is explained in order to successfully adapt the actions of the task to the position (dimension x, y) and orientation of the current objects. The dimension z is also modified only when there are obstacles in order to avoid collision.

The inputs of the proposed algorithm are the outputs of the offline learning phase: $GM, pick, place, N, Obj.pick, Obj.place, ee.on, ee.off$, and the output of the environmental perception module for the current identified objects: $Obj.pick', Obj.place'$ and $Obj.obst'$

The Algorithm 1 explains how the modification of Gaussian Means is performed. Algorithms 2–4 explain functions that are used in Algorithm 1. The Algorithm 1 is generic and it does not depend on the number of Gaussians (as long as the Gaussians are more than 4).

The algorithm 1 consists of the following steps in order to achieve the modification of the Gaussians for the specific environment:

– *Step 1: New mean values for first and last Gaussians (Algorithm 1, line 1–4)*

The new mean values for the first and last Gaussians are equal to the mean values of the first and last learned Gaussians, as shown in Fig. 4c. The first and last Gaussian represents a position in which the arm is far away from the working table.

– *Step 2: Identification of the pick and place Gaussians*

The *pick* and *place* Gaussians are identified by mapping the position of the end-effector in which the gripper should grasp (pick) and release (place) the object to the corresponding Gaussians, as shown in Fig. 4c.

– *Step 3: Modification of mean value for pick and place Gaussians (Algorithm 1, line 5)*

The new mean values of the *pick* and *place* Gaussians are modified based on the new position and orientation of the current objects, by calling the function for modification of *pick* and *place* Gaussian means (Algorithm 2). In Algorithm 2 the mean values for the *pick* and *place* Gaussians are calculated in lines 2–5 for the dimensions x , y . An example of modification of the *pick* Gaussian can be seen in Fig. 4e. The dimension z is not needed since the current and the original objects are the same (so they have same height). To calculate the mean values for the quaternions, there is the need to know firstly the difference (in quaternion) between the yaw angle of the current object and of the original object. The angle difference in quaternion for the *pick* and *place* object is calculated in the lines 6 and 7 of Algorithm 2 with the help of the function “*findRot*” which is described in Algorithm 3. After the Algorithm 3 calculates the change of the object angle in quaternions, the product of two quaternions is calculated. The first quaternion is calculated from the Algorithm 3 and the second quaternion is the orientation of end effector when the gripper grasped or released an object in the learned path (Algorithm 2, lines 8 and 9). Finally, the modified mean values for the *pick* and *place* Gaussians for the quaternion are calculated in lines 10–13 of the Algorithm 2.

– *Step 4: Find close Gaussians to pick and place Gaussians (Algorithm 1, line 6)*

The Gaussians that are close to the *pick* and *place* are found by calling the function “*findCloseGaus*”. This function compares the difference between each Gaussian mean and the *ee.on*, *ee.off* separately for each dimension $\{x, y, z, qx, qy, qz, qw\}$ and, if this difference is smaller than a defined threshold value, then this Gaussian is selected as close. The output of this function is *closeGaus_{i,m}*, where $m = \{1, \dots, 4\}$ represents the close Gaussians before pick, after pick, before place, after place, respectively. The reason the close Gaussians needs to be determined is to perform the task more accurately since the most important Gaussians are those near the pick and place. An example of close Gaussians can be seen in Fig. 4c.

– *Step 5: Modification of mean values of the close Gaussians (Algorithm 1, line 7)*

The mean values of the close Gaussians are modified by calling the function “*modCloseGM*”. At first, this function calculates the difference $diff_p = GM'_{i,p} - GM_{i,p}$, for $p = \{pick, place\}$. Next, the close Gaussians are also modified by the same difference $diff_p$. An example of the modification of close Gaussians is shown in Fig. 4f.

– *Step 6: Modification of mean values of the remaining Gaussians (Algorithm 1, line 8–14)*

The mean values of the remaining Gaussians are also modified by calling the function “*modMidGM*” (Algorithm 4) for the intermediate Gaussians to provide a smoother path.

An example of the modification of intermediate (remaining) Gaussians is shown in Fig. 4g.

– *Step 7: Find Gaussians which are in collision with obstacles (Algorithm 1, line 15)*

The function “*findObsGM*” determines which Gaussians are in collision with the identified obstacles. Every edge $\{x, y, z\}$ of a bounding box for every obstacle is compared with the GM' . If a mean value of the GM' is inside the obstacle’s bounding box, then this Gaussian is flagged as “Gaussian in obstacle”. Furthermore, for the Gaussians between the *pick* and *place* Gaussians, the dimensions of the picked object are also taken into account.

– *Step 8: Modification of mean values Gaussians which are in collision with obstacles (Algorithm 1, line 16)*

The last step is to provide the modification of the Gaussians in order to avoid collision with obstacles. The mean values of Gaussians in obstacles are modified by adding the obstacle’s height to the z dimension of the GM' or, if the Gaussians in obstacles are in between *pick* and *place* Gaussians, by adding the obstacle’s height plus the picked current object’s height to the z dimension of the GM' plus a safety distance. The obstacle avoidance happens in z dimension as shown in Fig. 6c. The assumption is that the position of obstacles does not prevent the completion of the task, i.e. obstacle is not on the target position.

Algorithm 1: Modification of Gaussian Means

Input: $GM, pick, place, N, Obj.pick, Obj.place, ee.pose.on, quat, ee.pose.off, quat, Obj.pick', Obj.place', Obj.obst'$

Output: GM'

```

1. for ( $i = 1; i \leq 7; i++$ ) do
2.    $GM'_{i,1} \leftarrow GM_{i,1}$ 
3.    $GM'_{i,N} \leftarrow GM_{i,N}$ 
4. end for
5.  $GM' \leftarrow modPickPlaceGM(GM, GM', pick, place, Obj.pick,$ 
    $Obj.place, Obj.pick', Obj.place')$ 
6.  $closeGaus \leftarrow findCloseGaus(GM, GM', pick, place, ee.on_i, ee.off_i)$ 
7.  $GM' \leftarrow modCloseGM(GM, GM', pick, place, closeGaus)$ 
8. for ( $i = 1; i \leq 7; i++$ ) do
9.   if  $i \neq 3$ 
10.     $GM'_i \leftarrow modMidGM(N, place + closeGaus_{i,4}, GM'_i, GM_i)$ 
11.     $GM'_i \leftarrow modMidGM(place - closeGaus_{i,3},$ 
        $pick + closeGaus_{i,2}, GM'_i, GM_i)$ 
12.     $GM'_i \leftarrow modMidGM(pick - closeGaus_{i,1}, 1, GM'_i, GM_i)$ 
13.   end if
14. end for
15.  $obsGaus \leftarrow findObsGaus(GM', pick, place, Obj.pick', Obj.obst')$ 
16.  $GM' \leftarrow modObsGM(GM', pick, place, N,$ 
    $Obj.pick', Obj.obst', obsGaus)$ 
17. return  $GM'$ 
```

Algorithm 2: Function for Modification of Pick and Place Gaussian**Means**

```

1. function modPickPlaceGM ( $GM, GM', pick, place, Obj.pick, Obj.place,$ 
    $Obj.pick', Obj.place', ee.on, ee.off$ )
2. for ( $i = 1; i < 3; i++$ ) do
3.    $GM'_{i,pick} \leftarrow Obj.pick'_i + GM_{i,pick} - Obj.pick_i$ 
4.    $GM'_{i,place} \leftarrow Obj.place'_i + GM_{i,place} - Obj.place_i$ 
5. end for
6.  $qrotation.pick \leftarrow findRot(0,0,1, Obj.pick'_4 - Obj.pick_4)$ 
7.  $qrotation.place \leftarrow findRot(0,0,1, Obj.place'_4 - Obj.place_4)$ 
8.  $qpick.target \leftarrow qrotation.pick * [ee.on_4, ee.on_5, ee.on_6, ee.on_7]$ 
9.  $qplace.target \leftarrow$ 
    $qrotation.place * [ee.off_4, ee.off_5, ee.off_6, ee.off_7]$ 
10. for ( $i = 4; i < 7; i++$ ) do
11.    $GM'_{i,pick} \leftarrow qpick.target_{i-3} + GM_{i,pick} - ee.on_i$ 
12.    $GM'_{i,place} \leftarrow qplace.target_{i-3} + GM_{i,place} - ee.off_i$ 
13. end for
14. return  $GM'$ 
15. end function

```

Algorithm 3: Function for calculating the change in the rotation of an object. Result of the function in quaternion.

```

1. function findRot( $xx, yy, zz, angle$ )
2.  $factor \leftarrow \sin(angle/2)$ 
3.  $qx \leftarrow xx \cdot factor$ 
4.  $qy \leftarrow yy \cdot factor$ 
5.  $qz \leftarrow zz \cdot factor$ 
6.  $qw \leftarrow \cos(angle/2)$ 
7. return  $quat \leftarrow [qx, qy, qz, qw]$ 
8. end function

```

Algorithm 4: Function for Modification of remaining Gaussian Means

```

1. function modMidGM( $a, b, GM', GM_i$ )
2.  $diff \leftarrow [(GM'_{i,a} - GM_{i,a}) + (GM'_{i,b} - GM_{i,b})]/2$ 
3. for ( $j = a - 1$  to  $b + 1$  step  $-1$ ) do
4.    $GM'_{i,j} \leftarrow GM_{i,j} + diff$ 
5. end
6. return  $GM'$ 
7. end function

```

5.2 Gaussian mixture regression (GMR)

The modified mean vector (GM' or $\mu'_{n,s}$), which is the output of the presented algorithm in Sect. 5.1 will be used to generate the adapted to the changed environment path $\hat{\beta}'$ via GMR. The Eq. (5), presented in Sect. 5.2, will be modified to the following equation:

$$\hat{\beta}' = \left\{ \beta_t, \sum_{n=1}^N a_n \hat{\beta}'_{n,s} \right\} \quad (6)$$

where $\hat{\beta}'_{n,s} = \mu'_{n,s} + \Sigma_{n,st} (\Sigma_{n,t})^{-1} (\beta_t - \mu_{n,t})$, $\forall n = \{1, \dots, N\}$.

The adapted path $\hat{\beta}'$ has all the essential features (i.e. pick and place) of the demos but, additionally, adapts to the new environmental conditions, as shown in Figs. 4h and 6d.

6 Real robot experimental results in an industrial assembly scenario

In the presented work, the assembly of a robot gripper is selected as an industrial assembly scenario to evaluate the suggested robot learning framework, as explained in Sect. 3.

Figure 5a shows a real-time working environment, in which the human collaborator set up the gripper parts in poses different from the demonstrated ones. Additionally, the human collaborator has selected a sequence of actions different from the demonstrated one. The robot performs the suggested sequence together with the human collaborator, as shown in Fig. 5b. After each subtask is completed by the robot, which is after two gripper parts are placed next to each other, the human screws the parts together. After that, for safety reasons, human collaborator confirms in Virtual Environment-based Situation Awareness module that the robot can continue with the next subtask. At the end of the sub-task, the human collaborator provides a feedback if the sub-task sequence of actions was performed well or not by the robot.

Figure 6e shows a real-time execution of the right subtask 2 (sequence based on Fig. 2d), in which an additional object is added as an obstacle. The robot was able to avoid the obstacle but the picked object (black part) was not positioned properly. The reason is that the black part is heavy (1 kg) and it oscillated during the manipulation by the vacuum gripper which caused the object misplacement.

7 Conclusion and future work

In this paper, a robot learning framework for industrial assembly applications is presented. The task is demonstrated in multiple human demos via kinesthetic teaching. The robot learns the sequence of actions (high-level) and the end-effector paths (low-level) needed to complete the assembly task. The learning is done in the offline learning phase while the reproducing of the learned actions and paths is done in the online working phase. This reproduction is possible even in the case of the changes in environment, such as different poses of the objects to be manipulated and present obstacles.

Fig. 5 Example of human–robot collaboration for the gripper assembly task. **a** Initial scene during demo (left) and during a real-time working environment (right), **b** the robot executes the learned by demo task with

a different sequence of actions suggested by the human collaborator and different pose of gripper parts (Color figure online)

Fig. 6 Example of the right subtask 2 (suggested sequence from human collaborator as shown in Fig. 2d) with obstacle avoidance for the z -dimensions **a** learned GMM from the demos, **b** generated path via GMR

from the learned GMM, **c** modified GMM (step 8), **d** generated path via GMR from the modified GMM, **e** the robot executes the subtask (Color figure online)

Additionally, the framework offers to the human collaborator the possibility to change the order of the sequence of actions without additionally training.

A novel method for selection of similar demos is suggested and implemented and GMM/GMR is used to reproduce the learned path based on the selected demos. This method also copes with the human demonstrator imprecision in placing the objects during demos.

In the online working phase, a novel modification of the GMM algorithm for the adaptation to environmental changes is suggested and implemented. The algorithm is tested in a real-world assembly scenario with a dual-arm industrial robot collaborating with a human and the experimental results are presented.

Acknowledgements The research was supported by the German Federal Ministry of Education and Research (BMBF) as part of the project MeRoSy (Human Robot Synergy). The authors thank MeRoSy industrial project partner pi4 robotics GmbH for the technical support in assembly scenario.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

- ABB. (2014). Retrieved from ABB unveils the future of human-robot collaboration: YuMi: <http://www.abb.com/cawp/seitp202/6fd1c7e9eb82896bc1257d4b003854fb.aspx>.
- Akgun, B., Cakmak, M., Jiang, K., & Thomaz, A. (2012). Keyframe-based learning from demonstration. *International Journal of Social Robotics*, 4(4), 345–355.
- Akgun, B., & Thomaz, A. (2016). Simultaneously learning actions and goals from demonstration. *Autonomous Robots*, 40(2), 211–227.
- Argall, B. D., Chernova, S., Veloso, M., & Browning, B. (2009). A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5), 469–483.
- Billard, A., Calinon, S., Dillmann, R., & Schaal, S. (2007). Handbook of robotics chapter 59: Robot programming by demonstration. In *Robotics* (Vol. 48, pp.1371–1394).
- Calinon, S. (2009). *Robot programming by demonstration: A probabilistic approach*. EPFL Press.
- Calinon, S., Bruno, D., & Caldwell, D. G. (2014). A task-parameterized probabilistic model with minimal intervention control. In *IEEE International conference on robotics and automation (ICRA)* (pp. 3339–3344).
- Calinon, S., D’halluin, F., Sauser, E., Caldwell, D., & Billard, A. (2010a). Learning and reproduction of gestures by imitation. *IEEE Robotics & Automation Magazine*, 17(2), 44–54.
- Calinon, S., Guenter, F., & Billard, A. (2007). On Learning, representing and generalizing a task in a humanoid robot. *IEEE Transactions on Systems, Man and Cybernetics, Part B, Special Issue on Robot Learning by Observation, Demonstration and Imitation*, 37(2), 286–298.
- Calinon, S., Pistillo, A., & Caldwell, D. (2011). Encoding the time and space constraints of a task in explicit-duration hidden Markov model. In *IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 3413–3418).
- Calinon, S., Sauser, E., Billard, A., & Caldwell, D. (2010b). Evaluation of a probabilistic approach to learn and reproduce gestures by imitation. In *IEEE international conference on robotics and automation (ICRA)* (pp. 2671–2676).
- Dempster, A., Laird, N., & Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1), 1–38.
- Dindo, H., & Schillaci, G. (2010). An adaptive probabilistic approach to goal-level imitation learning. In *IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 4452–4457).
- Douglas, D., & Peucker, T. (1973). Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica the International Journal for Geographic Information and Geovisualization*, 10(2), 112–122.
- Ekvall, S., & Kragic, D. (2006). Learning task models from multiple human demonstrations. In *15th IEEE international symposium robot and human interactive communication (ROMAN)* (pp. 358–363).
- Fang, Q. (2016). *Reliable 3D point cloud based object recognition*, M.Sc. thesis. University of Bremen.
- Forbes, M., Rao, R. P., Zettlemoyer, L., & Cakmak, M. (2015). Robot programming by demonstration with situated spatial language understanding. In *IEEE international conference on robotics and automation (ICRA)* (pp. 2014–2020).
- Ghalamzan E., A. M., Paxton, C., Hager, G. D., & Bascetta, L. (2015). An incremental approach to learning generalizable robot tasks from human demonstration. In *IEEE international conference on robotics and automation (ICRA)* (pp. 5616–5621). <https://doi.org/10.1109/ICRA.2015.7139985>.
- Khansari-Zadeh, S. M., & Billard, A. (2011). Learning stable non-linear dynamical systems with Gaussian mixture models. *IEEE Transactions on Robotics*, 27(5), 943–957.
- Kober, J., & Peters, J. (2009). Learning motor primitives for robotics. In *IEEE international conference on robotics and automation (ICRA)* (pp. 2112–2118).
- Kormushev, P., Calinon, S., & Caldwell, D. (2010). Robot motor skill coordination with EM-based reinforcement learning. In *IEEE/RSJ International conference on intelligent robots and systems (IROS)* (pp. 3232–3237).
- Krug, R., & Dimitrov, D. (2013). Representing movement primitives as implicit dynamical systems learned from multiple demonstrations. In *16th international conference on advanced robotics (ICAR)* (pp. 1–8).
- Kruger, V., Herzog, D., Baby, S., Ude, A., & Kragic, D. (2010). Learning actions from observations. *IEEE Robotics & Automation Magazine*, 17(2), 30–43.
- KUKA Robotics. (2017). Retrieved May 9, 2017, from Human-robot collaboration. <https://www.kuka.com/en-us/technologies/human-robot-collaboration>.
- Kyrarini, M., Haseeb, M., Ristic-Durrant, D., Gräser, A., Jackowski, A., Gebhard, M., et al. (2017). Robot learning of object manipulation task actions from human demonstrations. *Facta Universitatis Series, Mechanical Engineering*, 15(2), 217–229. <https://doi.org/10.22190/FUME170515010K>.
- Kyrarini, M., Leu, A., Ristic-Durrant, D., Gräser, A., Jackowski, A., Gebhard, M., et al. (2016). Human-robot synergy for cooperative robots. *Facta Universitatis, Series: Automatic Control and Robotics*, 15(3), 187–204. <https://doi.org/10.22190/FUACR1603187K>.
- Lee, S., Suh, I., Calinon, S., & Johansson, R. (2015). Autonomous framework for segmenting robot trajectories. *Autonomous Robots*, 38(2), 107–141.
- Leitner, J., Luciw, M., Forster, A., & Schmidhuber, J. (2014). Teleoperation of a 7 DOF humanoid robot arm using human arm accelerations and EMG signals. In *International symposium on artificial intelligence, robotics and automation in space (i-SAIRAS)*.
- Maeda, G. J., Neumann, G., Ewerton, M., Lioutikov, R., Kroemer, O., & Peters, J. (2017). Probabilistic movement primitives for coordination of multiple human-robot collaborative tasks. *Autonomous Robots*, 41(3), 593–612.
- Movchan, A., & Zymbler, M. (2015). Time series subsequence similarity search under dynamic time warping distance on the intel many-core accelerators. In *International conference on similarity search and applications* (pp. 295–306). Berlin: Springer.
- Mühlig, M., Gienger, M., & Steil, J. J. (2012). Interactive imitation learning of object movement skills. *Autonomous Robots*, 32(2), 97–114.
- Nierhoff, T., Hirche, S., & Nakamura, Y. (2016). Spatial adaption of robot trajectories based on Laplacian trajectory editing. *Autonomous Robots*, 40(1), 159–173.
- Nikolaidis, S., & Shah, J. (2013). Human-robot cross-training: Computational formulation, modeling and evaluation of a human team

- training strategy. In *8th ACM/IEEE international conference on human–robot interaction (HRI)* (pp. 33–40).
- Park, D.-H., Hoffmann, H., Pastor, P., & Schaal, S. (2008). Movement reproduction and obstacle avoidance with dynamic movement primitives and potential fields. In *8th IEEE-RAS international conference on humanoid robots* (pp. 91–98).
- 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 (ICRA)* (pp. 763–768).
- Patel, M., Valls, J. M., Kragic, D., Henrik Ek, C., & Dissanayake, G. (2014). Learning object, grasping and manipulation activities using hierarchical HMMs. *Autonomous Robots*, 37(3), 317–331.
- Pathirage, I., Khokar, K., Klay, E., Alqasemi, R., & Dubey, R. (2013). A vision based P300 brain computer interface for grasping using a wheelchair-mounted robotic arm. In *IEEE/ASME international conference on advanced intelligent mechatronics (AIM)* (pp. 188–193).
- Pedersen, M., Nalpantidis, L., Andersen, R., Schou, C., Bøgh, S., Krüger, V., et al. (2016). Robot skills for manufacturing: From concept to industrial deployment. *Robotics and Computer-Integrated Manufacturing*, 37, 282–291.
- Quintero, C., Fomena, R., Shademan, A., Ramirez, O., & Jagersand, M. (2014). Interactive teleoperation interface for semi-autonomous control of robot arms. In *Canadian conference on computer and robot vision* (pp. 357–363).
- Ramer, U. (1972). An iterative procedure for the polygonal approximation of plane curves. *Computer Graphics and Image Processing*, 1(3), 244–256.
- Sabbaghi, E., Bahrami, M., & Ghidary, S. (2014). Learning of gestures by imitation using a monocular vision system on a humanoid robot. In *Second RSI/ISM international conference on robotics and mechatronics (ICRoM)* (pp. 588–594).
- Schaal, S., Peters, J., Nakanishi, J., & Ijspeert, A. (2005). Learning movement primitives. In P. Dario, & R. Chatila (Eds.), *Robotics research. The eleventh international symposium: With 303 figures* (pp. 561–572). https://doi.org/10.1007/11008941_60.
- Tellaiche, A., Maurtua, I., & Ibarguren, A. (2015). Human–robot interaction in industrial robotics. Examples from research centers to industry. In *IEEE 20th conference on emerging technologies & factory automation (ETFA)*. <https://doi.org/10.1109/ETFA.2015.7301650>.
- Weistroffer, V., Paljic, A., Fuchs, P., Hugues, O., Chodacki, J.-P., Ligot, P., & Morais, A. (2014). Assessing the acceptability of human–robot co-presence on assembly lines: A comparison between actual situations and their virtual reality counterparts. In *23rd IEEE international symposium on robot and human interactive communication* (pp. 377–384).
- Yin, X., & Chen, Q. (2014). Learning nonlinear dynamical system for movement primitives. In *IEEE international conference on systems, man and cybernetics (SMC)* (pp. 3761–3766).
- Zhang, J., Wang, Y., & Xiong, R. (2016). Industrial robot programming by demonstration. In *International conference on advanced robotics and mechatronics (ICARM)* (pp. 300–305).

of Prof. Dr. Axel Gräser.

Maria Kyrarini received her 5-years diploma degree in Electrical and Computer Engineering, in 2012, and the master's degree in Automation Systems, in 2014, from National Technical University of Athens. She is currently working as research associate on the project MeRoSy—Human Robot Synergy at the Institute of Automation, University of Bremen, Germany and she is pursuing her Ph.D degree in the topic of machine learning for human robot cooperation under the supervision

of Prof. Dr. Axel Gräser.

Muhammad Abdul Haseeb received his Bachelor of Engineering degree (B.E.) in Industrial Electronics from the NED University of Engineering and Technology in Karachi, Pakistan and his Master of Science degree (M.Sc.) in Information & Automation Engineering from University of Bremen, Germany. He is currently working as a researcher on the EU project Smart Automation of Rail Transport (SMART) and BMBF project MeRoSy (*Human Robot Synergy*) at the Institute of Automation, University of Bremen, Germany and he is pursuing his Ph.D. degree in the topic of machine learning techniques for autonomous multi-sensor environmental perception system under the supervision

of Prof. Dr. Axel Gräser.

Danijela Ristić-Durrant received the M.Sc. degree in automatic control and robotics from the Faculty of Mechanical Engineering, University of Niš, Serbia in 1998 and her Ph.D. in electrical engineering from the University of Bremen in 2007. Since her Ph.D., she has continued at the University of Bremen as a postdoctoral researcher and lecturer. Since July 2016 she has also held the title of Associated Professor for the field of automatic control and robotics in the Faculty of Mechanical Engineering at the University of Niš, Serbia. She was researcher and project manager of the FP7 EU-funded project CORBYS-Cognitive Control Framework for Robotic Systems, and the coordinator of several bilateral projects in the field of rehabilitation robotics and robot vision. Her current research interests include rehabilitation and assistive robotics and sensor-based control of robots. She has co-authored one university book, one book chapter and numerous journals and conference articles.

Axel Gräser received the diploma in electrical engineering from the University of Karlsruhe, Germany, in 1976 and the Ph.D. degree in control theory from the TH Darmstadt, Germany, in 1982. He was the Head of the Control and Software Department at Lippke GmbH, Germany, from 1982 to 1990. From 1990 to 1994 he was Professor of Control Systems, Process Automation and Real-time Systems at the University of Applied Sciences, Koblenz. Since 1994, he has been the Director

of the IAT. He is also a foreign advisory professor at the Korean Advanced Institute of Science and Technology (KAIST). He has been coordinating more than 20 national and international projects in process automation, control, and rehabilitation, among them four EU grants: BrainRobot, BRAIN, IntARWeld and CORBYS. He is a member of IEEE, a member of the executive board of the DFMRS, and a member of the proposal evaluation board AIF.