Proceedings of the 2007 IEEE International Symposium on
Computational Intelligence in Robotics and Automation
Jacksonville, FL, USA, June 20-23, 2007

Programming by Demonstration of Pick-and-Place Tasks for Industrial Manipulators using Task Primitives

Alexander Skoglund* Boyko Iliev* Bourhane Kadmiry* Rainer Palm†
Center for Applied Autonomous Sensor Systems,
Department of Technology, SE-701 82 Örebro University, Sweden

*{*alexander.skoglund,boyko.iliev,bourhane.kadmiry*}*@tech.oru.se*
†*rub.palm@t-online.de*

***Abstract*— This article presents an approach to Programming by Demonstration (PbD) to simplify programming of industrial manipulators. By using a set of task primitives for a known task type, the demonstration is interpreted and a manipulator program is automatically generated. A pick-and-place task is analyzed, based on the velocity profile, and decomposed in task primitives. Task primitives are basic actions of the robot/gripper, which can be executed in a sequence to form a complete a task. For modeling and generation of the demonstrated trajectory, fuzzy time clustering is used, resulting in smooth and accurate motions. To illustrate our approach, we carried out our experiments on a real industrial manipulator.**

I. Introduction and Motivation

Robot programming can be difficult and time consuming depending on the task the robot has to perform in a working process. The main focus of this paper is to make the programming procedure easier and to reduce the programming time by teaching and/or programming of robot tasks by direct human demonstration. The targeted goal is to enable small- and mid-sized enterprises (SME) to benefit from automated production in product handling and assembly. This is also important because even large companies may show an internal structure of small- and mid-sized enterprises. In order to reduce programming time, one paradigm is to teach the robot what to do, either by guiding the robot (lead-through programming), or by so called Programming by Demonstration (PbD) [9].

However, problems and challenges rising from this paradigm are several. In this paper we propose an approach to PbD pick-and-place task and an implementation based on task primitives. More explicitly, starting with assumptions regarding a typical set-up, we show how a simple pick-and-place task can be programmed by human demonstration. By using the human demonstration the robot extracts features such as the pick-point, the place-point, and the path between them.

A. Related Work

On the basis of a PbD procedure Aleotti et al. explored how a virtual environment can be exploited for tasks whose features are known in advance such as pick-and-place operations([1]). The virtual environment was used to analyze and simulate the demonstrated task before its actual reproduction, given that the teacher agrees that the task was correctly interpreted. Once a task was successfully simulated it was transfered to a real 6 DOF Puma560 manipulator.

Recent work by by Erlhagen et. al. [6] used an industrial manipulator in their test setup for learning a goal-directed pick-and-place task including obstacle avoidance. In their work they used two grasping primitives, precision grip and full grip, and two primitives for transportation while avoiding obstacles.

Ekvall and Kragic [5] used a 6 DOF manipulator for task learning from multiple human demonstration, where the task were divided into task primitives (subtasks). Focusing on the world state and the goal of the demonstration, i.e., placing an object at the shown position, the task was reconstructed without the knowledge of the demonstrated path. In our work, we take advantage of the trajectory (path and velocity profile) between the pick- and the place- position to reconstruct a human like motion.

In [4], Dillmann review their PbD system using multiple sensors, were used for capturing demonstrations in a kitchen environment with typical household tasks. In the experiments they conducted, a robot learned pick-and-place operations, including techniques for grasping [3]. The robot used in their experiments had a mobile base and two 7 degrees-of-freedom (DOF) anthropomorphic manipulators mounted on top of it.

An application using task primitives for task learning from observation was presented by Bentivegna and colleagues, who conducted experiments with a humanoid robot that learned to play air hockey [2]. For their experiment a number of primitives (like *Straight-Shot*, *Bank-shot*, *Idle* etc.) were predefined. For learning the task locally weighted learning was used, where the input data were the positions and velocities of the tracked objects (puck, paddles and table edges).

In our approach we assume that the demonstrator is aware of the functionality of the particular robot, and the task primitives directly reflect specific capabilities of robot arm/gripper. In this way the task primitives serves as common language for the robot and the demonstrator. Furthermore a magnetic tracker is used for the demonstrator to record the demon-

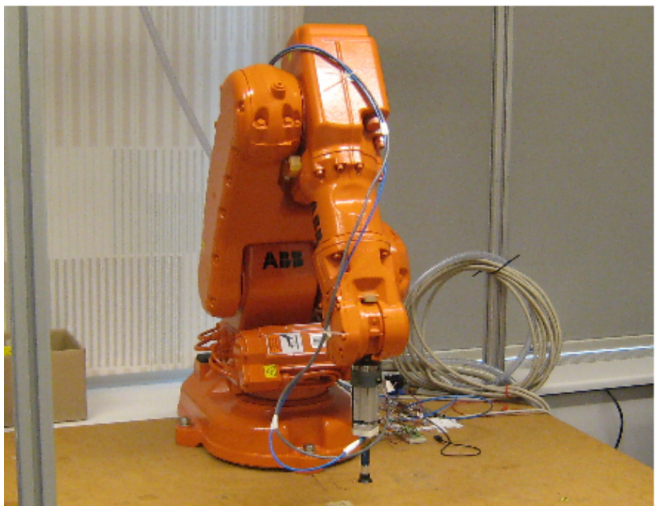

Fig. 1. To the left is the manipulator in our experimental setup, an ABB IRB140 manipulator equipped with a vacuum gripper. A video of the task performed is available at the authors homepage: www.aass.oru.se/∼asd/. To the right the 6D-tracker, mounted on a data glove, is shown that was used to capture the human demonstration.

strated trajectories. Other benefits our approach offers is that the teaching is done in the manipulator's own coordinate system, since the manipulator and the demonstrator shares the same workspace.

The outline of the paper is the following: Section II deals with *programming by demonstration* including *automatic task assembly*, *segmentation of motions* and *task primitives*. Section III deals with the experimental results for a real robot manipulator, and Section IV draws conclusions and refers to some future work.

II. Programming by Demonstration

A. Automatic Task Assembly

The approach we present in this article can extract a set of instructions corresponding to a demonstration, without writing a single line of code, given that the task type is known (that is, a pick-and-place task). One important benefit derived from the PbD method is the human-like appearance of the motion which also increases safety implicitly since the motion is predictable to humans (in contrast to, e.g., time-optimal motions).

The scenario we consider is to teach an industrial robot equipped with a vacuum gripper, shown to the left in Fig. 1, how to execute a pick-and-place task. The demonstration is done under the assumption that the teacher's index finger is associated with the suction cup of the gripper. During demonstration,the fingertip is tracked by a motion capturing device, shown to the right in Fig. 1. Initially, the demonstrator moves from a starting point to the desired pick-point, P_{pick}. Then, he/she moves along a certain path towards the desired place-point, P_{place} and finally, back to the starting position. The collected data consists of position coordinates P, which are used for the following purpose:

- Detect the pick- and place-positions, P_{pick} and P_{place}
- Reconstruct the desired trajectory that the robot should travel from P_{pick} to P_{place} (`FollowTaj`).

In summary, the steps from the demonstration to the compilation of instructions are:

1) A human demonstration is captured and transformed into robot's reference frame.
2) Trajectories are smoothed to reduce noise from the sensors (see Section II-D).
3) Trajectories are segmented to extract the points where the motions start and end (see Section II-B).
4) Extracted motions are decomposed into task primitives as those described in Section II-C.
5) Each task primitive is automatically translated into robot-specific code.
6) The complete task is executed on the real ABB IRB140 manipulator.

For step 4 in the above list it is important to note that the task is known in advance, which makes it possible to describe the task as a sequence of task primitives. These task primitives are designed specifically for the task, and can be executed on most 6 DOF serial manipulators. The primitive controlling the grasp and release are specific to the type of gripper used.

In the scenario, the demonstrator and the robot share the same workspace. Our assumption is that the demonstrator knows the manipulator's structure such as workspace and possible motions which makes a complicated data preprocessing unnecessary.

B. Segmentation of Motions

To distinguish the different segments of a demonstrated motion, the captured data needs to be segmented. This is done by measuring the mean squared velocity of the demonstrator's end effector. A "semi-atomized" segmentation technique is used, equivalent to the one developed by Matarić et. al. [7], but instead of using the mean squared velocity of the joints angles we use the end effector's velocity. The mean squared velocity is given by:

$$MSV(t) = \sum_{i=1}^{3} \left(\frac{dx(i,t)}{dt} \right)^2 \quad (1)$$

where $i = 1...3$ is the number of the spacial coordinate, t is the time, $dx(i,t) \in R^1$ is the position difference $dx(i,t) = x(i,t+dt) - x(i,t)$ between two samples, dt is the time difference between two samples. To define the start time t_r of a motion, a constant threshold v for MSV is set. If $MSV(t) > k*v$, where k is a multiplication factor, then $t = t_r$. The start time t_r is checked for each time instant t after a motion has stopped. A similar procedure is used to find the stop of a motion.

The segmentation algorithm can be further improved by incorporating tactile sensing using a force sensitive resistor mounted on the demonstrator's fingertip.

C. Task Primitives

The decomposition of a Pick-and-Place task into task primitives is illustrated in Fig. 2. `MoveToPosJ` moves the manipulator's end effector to the point where it can "hook on" to the demonstrated trajectory. `MoveToPosL` moves the manipulator's end effector to the desired point linearly in the workspace coordinates. `MoveZ` is a the task primitive for making a search motion when there is uncertainty of how to approach an object. `FollowTraj` takes a sequence of points, with relatively hight granularity, as the input and execute the motion linearly between these points.

These are the basic primitives, which are highly dependent on the task to perform. By using primitives reflecting the commands available in the robot language of the manipulator, we achieve a simple implementation. The basic primitives, which all other higher level tasks (in this work) consists of, are:

- `MoveL` which moves linearly (Cartesian space) to a predefined point,
- `MoveJ` which moves to a predefined point,
- `MoveZ` which moves in an up-down direction,
- `FollowTraj` which follow a demonstrated trajectory,
- `Grasp` which grasps an object,
- `Release` which releases an object,

The two first primitives are usually implemented on standard industrial manipulators, yet needed as primitives for many tasks. In our setup the two primitives controlling grasp and release are very simple due to the design of the vacuum gripper. However, for more complex grippers, or even anthropomorphic hands, a complex set of primitives is needed [11].

When the trajectory is transfered to the manipulator's controller, it is written in a compact META format which represents the local program of the controller processes. By doing so, we assume the lower control levels of the manipulator, such as inverse kinematics, constraints of the workspace, generation of the trajectory with a higher granularity etc., to perform the required task in a proper way.

Two task primitives, `FollowTraj` and `MoveZ`, are described in more detail in the following two Sections.

D. Following Demonstrated Trajectories using Time-Clustering and Takagi-Sugeno Modeling

Between the points detected as pick-and-place, the trajectory chosen by the demonstrator should be followed in order to reproduce the demonstration and to make the motion human-like. The task primitive `FollowTraj`, which implements the trajectory following, uses a so-called time clustering method [11] which is based on Takagi-Sugeno fuzzy modeling [13] and fuzzy clustering [10]. The basic idea is to consider the 3 end effector coordinates of the hand as model outputs and the time instants as model inputs.

The incorporation of the time in the modeling process has the following advantages

- Representation of the dynamic arm motion by a small number of time points with only few parameters.
- Nonlinear filtering of noisy trajectories

The Takagi-Sugeno (TS) fuzzy model is constructed from captured data from the end effector trajectory described by the nonlinear function

$$\mathbf{x}(t) = \mathbf{f}(t) \quad (2)$$

where $\mathbf{x}(t) \in R^3, \mathbf{f} \in R^3$, and $t \in R^+$.

Equation (2) is linearized at selected time points t_i with

$$\mathbf{x}(t) = \mathbf{x}(t_i) + \frac{\Delta \mathbf{f}(t)}{\Delta t}|_{t_i} \cdot (t - t_i) \quad (3)$$

which results in a linear equation in t.

$$\mathbf{x}(t) = \mathbf{A}_i \cdot t + \mathbf{d}_i \quad (4)$$

where $\mathbf{A}_i = \frac{\Delta \mathbf{f}(t)}{\Delta t}|_{t_i} \in R^3$ and $\mathbf{d}_i = \mathbf{x}(t_i) - \frac{\Delta \mathbf{f}(t)}{\Delta t}|_{t_i} \cdot t_i \in R^3$. Using (4) as a local linear model one can express (2) in terms of an interpolation between several local linear models by applying Takagi-Sugeno fuzzy modeling [13] (see Fig. 3)

$$\mathbf{x}(t) = \sum_{i=1}^{c} w_i(t) \cdot (\mathbf{A}_i \cdot t + \mathbf{d}_i) \quad (5)$$

$w_i(t) \in [0,1]$ is the degree of membership of the time point t to a cluster with the cluster center t_i, c is number of clusters, and $\sum_{i=1}^{c} w_i(t) = 1$.

The degree of membership $w_i(t)$ of an input data point t in an input cluster C_i is determined by

$$w_i(t) = \frac{1}{\sum_{j=1}^{c} \left(\frac{(t-t_i)^T M_{ipro}(t-t_i)}{(t-t_j)^T M_{jpro}(t-t_j)} \right)^{\frac{1}{\tilde{m}_{proj}-1}}} \quad (6)$$

The projected cluster centers t_i and the induced matrices M_{ipro} define the input clusters C_i ($i = 1 \ldots c$). The parameter $\tilde{m}_{pro} > 1$ determines the fuzziness of an individual cluster [8].

E. Approach, Grasp- and Release Primitives

Before performing a grasp or release operation, the approach phase of the gripper towards the object is of high importance. In our current work we only consider approach motions towards the object perpendicular to the surface of the table (this also imply objects with flat top surfaces),

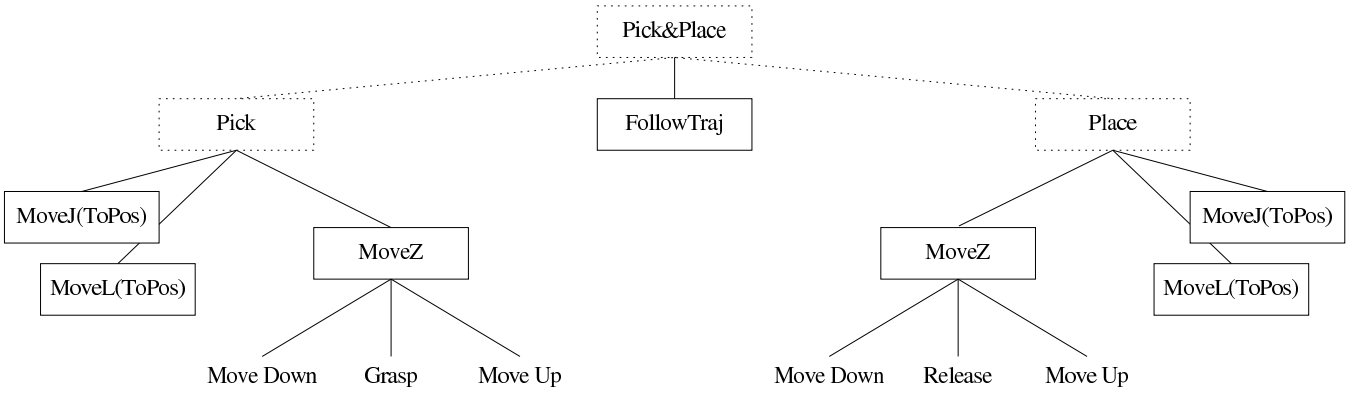

Fig. 2. Decomposition of a task into primitives. The solid boxes are the implemented task primitives.

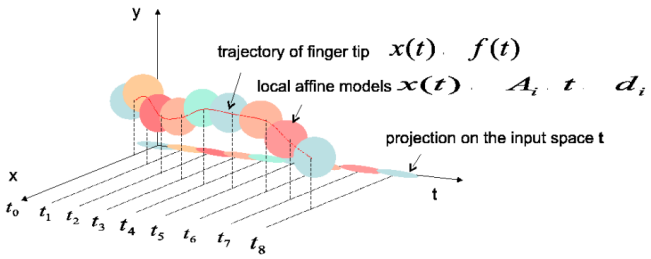

Fig. 3. Time-clustering principle.

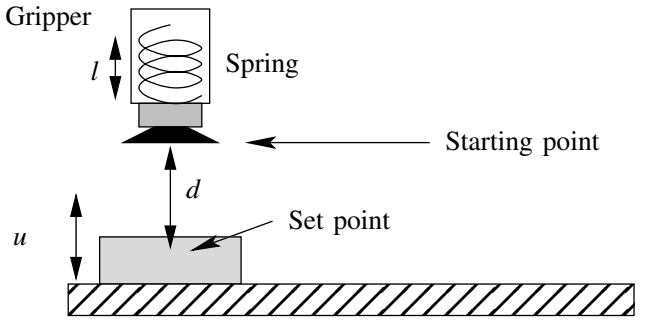

Fig. 4. The gripper and the spring switch. d, the distance target set point and the starting point, is dependent of the sensor inaccuracy, y. The length which the spring is compressed, l, is determined when the switch turns on, indicating a resistance.

which is a simplification derived from the design of the 1 DOF vacuum gripper. The position and orientation of the table are known since the manipulator is mounted on the table (see Fig. 1). However, the height of the object to be picked is unknown due to the uncertainty of the sensor, thus the approach primitive has to deal with the uncertainty. Therefore the manipulator's gripper is equipped with a switch that detects when a spring is compressed to a certain extent (illustrated in Fig. 4). When the switch has detected a contact, the motion downwards is immediately stopped and the appropriate action (grasp or release) is performed.

A grasp operation is distinguished from a release operation by two discrete states, internally represented in the `MoveZ` primitive. When performing a grasp or release operation the manipulator is given a set coordinate to move towards the object and search for contact with it. When the switch detects a certain resistance (that is, the spring is compressed at a length, l) the motion stops. The starting point is determined by the distance d, which is derived from the inaccuracy of the sensor that performs the motion capturing, typically factor of $1.1 \sim 2.0$. The approach task primitive is implemented with the `SearchL` command in RAPID (see the ABB specific programming languages, facilitating instructions for manipulator motions).

III. Experimental Results for a Robot Manipulator

The experimental setup consists of a 6D tracker, Polhemus Fastrak, for the demonstrator and a 6 DOF industrial manipulator, the IRB 140 from ABB Robotics. A vacuum gripper is mounted on the manipulator together with a magnetic switch and a spring. We have selected a pick-and-place task for our work since it is a common task for manipulators. The demonstrator is placed in front of the manipulator by means of which both share the same workspace. The demonstration consists of pointing at two different locations in the workspace and returning to the initial position. The first detected position is the pick position, denoted by p_{pick}, and the second detected position is the place position, denoted by p_{place} (see Fig. 6). The manipulator's initial and final positions are denoted by L_0 and $L_w(t)$ respectively in Fig. 6, and the demonstrator's initial and final positions are denoted by P_0 and $P_w(t)$. t indicates the time. The different positions at the start and end of the trajectory are due to the fact that demonstrator and manipulator are not in the same start and final position, but they share the same workspace and therefore the pick and place points and the trajectory in between are the same.

The top row of Fig. 5 shows three example trajectories of the same pick-and-place positions. The second row of Fig. 5 shows one sample trajectory where the segmentation has detected three starts and the respective ends of three movements. The MSV threshold was set to 40 and the multiplication factor to 1.5 for this specific example.

Through this illustrations, we can see that the models cover correctly the task trajectory, as performed by the human. The pick-and-place points as reflected by both the model and the raw data show a discrepancy of 3.62 mm, which is tolerable for the kind of tasks to be performed by the robot arm.

The trajectories are segmented and from the pick to

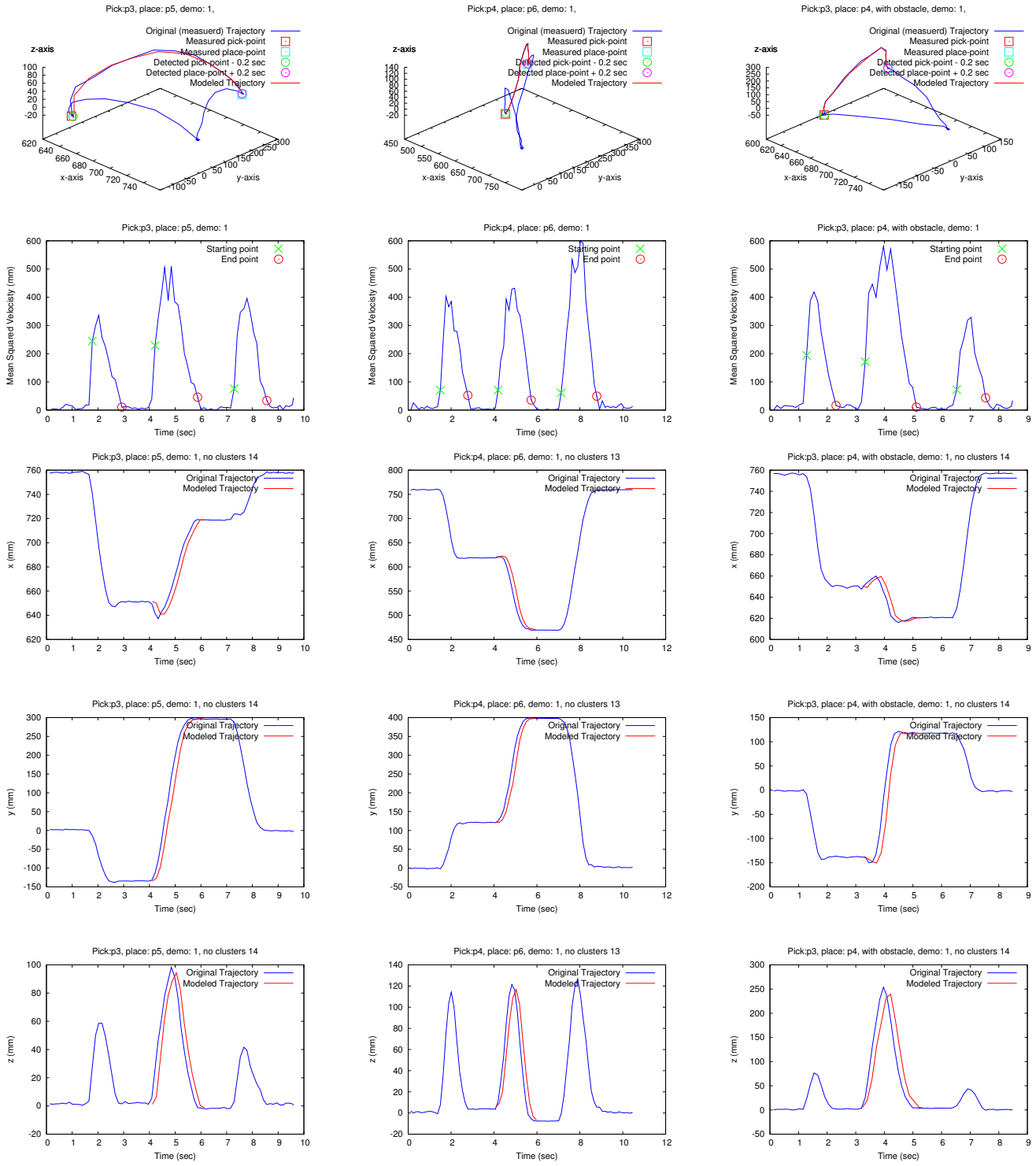

Fig. 5. Each column shows demonstrations of different pick-and-place tasks. The first row shows three measured demonstration and the second row the how velocity profile segment the demonstration. The third to fifth row illustrates trajectory profiles obtained throughout measurement and task-model along the x-axis (3:rd row), along the y-axis (4:th row) and along the z-axis (bottom row).

the place position a time-cluster modeling is performed. The result shows a good modeling quality, with a slight displacement in time.

The different illustrations in Fig. 5 show a series of three sets of measurement for the same task (pick-and-place), between two demonstrated points in the workspace. We can first establish that the data obtained by repetition of the task shows similar profiles, and the models extracted from these sets of data show both similarities to their respective sets of data, and a good premiss to introducing “signatures” to tasks

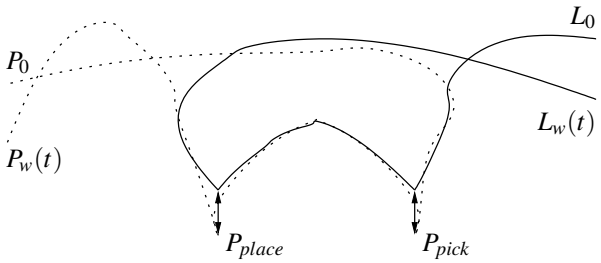

Fig. 6. An illustration of the trajectories, where the dotted line is the demonstration, and the solid line the manipulators end effector path. The manipulator's initial and final positions are denoted L_0 and $L_w(t)$ respectively, and the demonstrator's P_0 and $P_w(t)$. t.

through the design of simple task-models. These signatures can be used either to communicate in a high-level mean the task to be performed by the robot arm, or at low level to recognize the kind of task performed: This will be developed more specifically in a future work.

The sampling rate of the tracker with which the data is collected is approximately 12 Hz, for position and orientation data, in reference to a fixed point. When incorporating a data glove for capturing the demonstrator's hand motions the tracker will be mounted on the wrist, and the data glove provides the appropriate transformation to the end effector. With a fixed coordinate system on both the manipulator and the demonstrator it is straight forward to transform the local coordinate system of the demonstrator into robot coordinates. One significant drawback of the magnetic tracker is that it is relatively accurate in office environments but is less suitable if there are any large metal parts in the neighborhood of the probe. In our experiments, the position error could be over 40 mm for demonstrations near the robot. When we replicated the workspace of the robot in another area with no surrounding metal parts, the position error was reduced to under 4 mm. The reason for the degrading performance in the robot cell was the amount of metal in manipulator and frames of the cell. A passive motion capturing device, such as an optical measuring device used by [12], would remedy this effect. A visual system could be another option, but most vision systems have lower accuracy and problems with occlusions, in comparison with the 6D-tracker.

IV. Conclusions and Future Work

We have presented an approach to PbD of pick-and-place tasks for industrial robots, which rapidly minimizes the programming effort. The method is based on task primitives, which link high-level human instructions to particular robot/gripper functionalities. In our setup, the demonstrator is aware of the function of the task primitives. In this way, the risk for miss interpretations and infeasible instructions to the robot is reduced. The pick-and-place task can be mapped to a sequence of task primitives that are scalable w.r.t. different sizes of the workspace of the manipulator. The demonstrator motions are captured by a wearable input device, which records the coordinates of pick and place positions and the path in between. We use the mean squared velocity to segment the recorded motions and extract the pick- and place positions. This solution can be further improved if a tactile sensor is used to detect the true contact point. To approximate the desired path in space we apply a fuzzy time-clustering method. The method shows excellent modeling performance and also preserves dynamical (human-like) properties of the motion.

One problem of the method is that it relies on the accuracy of the data capturing device. Therefore, in our future work we plan to replace the currently used 3D magnetic tracker with a more robust system, e.g. 6-DOF coordinate-measuring arm. To allow programming of more complex grippers we plan to include a data glove to capture the demonstrator's hand pose.

Another direction for future work is to develop a method for human demonstration of the free workspace around the robot. In this context we are currently investigating the design of a 3D-occupancy grid and its integration in the robot simulator. This will enable collision tests of the automatically generated program code prior to execution on the robot.

References

[1] J. Aleotti, S. Caselli, and M. Reggiani, "Leveraging on a virtual environment for robot programming by demonstration," *Robotics and Autonomous Systems*, vol. 47, no. 2–3, pp. 153–161, 2004, doi:10.1016/j.robot.2004.03.009.

[2] D. C. Bentivegna, C. G. Atkeson, and G. Cheng, "Learning tasks from observation and practice," *Robotics and Autonomous Systems*, vol. 47, no. 2–3, pp. 163–169, 2004.

[3] R. Dillmann, O. Rogalla, M. Ehrenmann, R. Zöllner, and M. Bordegoni, "Learning robot behavior and skills based on human demonstration and advice: the machine learning paradigm," in *9th International Symposium of Robotics Research (ISRR)*, Oct 1999, pp. 229–238.

[4] R. Dillmann, "Teaching and learning of robot tasks via observation of human performance," *Robotics and Autonomous Systems*, vol. 47, no. 2–3, pp. 109–116, 2004.

[5] S. Ekvall and D. Kragic, "Learning task models from multiple human demonstrations," in *IEEE International Symposium on Robot and Human Interactive Communication*. Seattle, USA: IEEE, 2006.

[6] W. Erlhagen, A. Mukovskiy, E. Bicho, G. Panin, C. Kiss, A. Knoll, H. von Schie, and H. Bekkering, "Goal-directed imitation for robots: A bio-inspired approach to action understanding and skill learning," *Robotics and Autonomous Systems*, vol. 54, pp. 353–360, 2006, doi:10.1016/j.robot.2006.01.004.

[7] A. Fod, M. J. Matatić, and O. C. Jenkins, "Automated derivation of primitives for movment classification," *Autonomous Robort*, vol. 12, no. 1, pp. 39–54, 2002.

[8] D. Gustafson and W. Kessel, "Fuzzy clustering with a fuzzy covariance matrix." *Proceedings of the 1979 IEEE CDC*, pp. 761–766, 1979.

[9] Y. Kuniyoshi, M. Inaba, and H. Inoue, "Learning by watching: Extracting reusable task knowledge from visual observation of human performance," *IEEE Transactions on Robotics and Automation*, vol. 10, no. 6, pp. 799–822, 1994.

[10] R. Palm and C. Stutz, "Generation of control sequences for a fuzzy gain scheduler," *International Journal of Fuzzy Systems*, vol. Vol. 5, No. 1, pp. 1–10, March 2003.

[11] R. Palm and B. Iliev, "Learning of grasp behaviors for an artificial hand by time clustering and takagi-sugeno modeling," in *Proceedings of the IEEE International Conference on Fuzzy Systems*. Vancouver, BC, Canada: IEEE, July 16-21 2006, pp. 291–298, doi:10.1109/fuzzy.2006.1681728.

[12] A. Skoglund, T. Duckett, B. Iliev, A. Lilienthal, and R. Palm, "Programming by demonstration of robotic manipulators in non-stationary environments," in *Proceedings of 2006 IEEE International Conference on Robotics and Automation*, 2006.

[13] T. Takagi and M. Sugeno, "Identification of systems and its applications to modeling and control." *IEEE Trans. on Syst., Man, and Cyb.*, vol. Vol. SMC-15. No.1, pp. 116–132, January/February 1985.