Learning from Demonstration in Static Environment and Generalizing to Dynamic Environments

Amir M. Ghalamzan E.

***Abstract*— Robot learning from demonstration has been successfully used, in industrial environments, to increase the speed and reduce the complexity of the programming phase. A major challenge, however, consists in enabling a robot to generalize task demonstrations to a complex dynamic environment. To tackle this problem, an approach has been proposed in [1], combining GMM/GMR and DMPs with optimal control. This approach allows a robot to generalize noisy task demonstrations to different goal points and environments with static obstacles. Considering that in practical applications, a robot must perform a task in dynamic environments, here we extend the approach in such a way that a robot can learn a task model from demonstrations in an environment with stationary obstacles and reproduce the task in an environment with moving obstacles.**

I. Introduction

Robot Learning from Demonstration (robot LfD) has been proposed to reduce the programming time and cost. This allows a user to teach a robot how to perform a task by demonstrating that task and without explicitly hardcoding it, where the robot is able to replicate the demonstrated task in different conditions. The main issue, however, arises where a robot must replicate the task in a different environment. The problem of generalizing a task to a new environment may be considered at different levels. For example, Dynamic Movement Primitives (DMP) [2], analogous with *imitation* in human learning, generalizes an optimal demonstrated trajectory to new start and goal points; Gaussian Mixture Model/Gaussian Mixture Regression (GMM/GMR) [3] can be used to get a smooth nonlinear trajectory from a data set collected from noisy and suboptimal human demonstrations [4], analogous with *mimicking* in human learning.

A task-parametrized probabilistic model is proposed in [5] allowing a robot to scale a trajectory to a new goal point from a set of different demonstrations, combining mimicking with imitation learning.

Assume a person would like to teach by demonstration a pick-and-place task to a learner robot or person. Although human learns the pick-and-place task from demonstrations by imitating a teacher, he/she also learns from demonstrations how to respond to different objects in order to avoid colliding with the objects by *emulating* the teacher. Using the DMP model, the robot can only learn how to pick and place the object from and at different positions.

In this regard, a number of methods have been proposed combining some hardcoded models of obstacle avoidance with DMP [6]–[9]. For example, Park et al. [10] included obstacle avoidance in DMP model by adding to the equations of motion a repellent force, a gradient of a potential field centered around the obstacle. Although these proposed approaches enable a robot to avoid colliding with an obstacle, they do not provide a robot with the ability to learn a user-specific response to an obstacle; thus, a non-expert cannot teach his/her own desired response to an obstacle to the robot by demonstrating it on-the-fly.

Fig. 1. Demonstrating the manipulation example with ABB FRIDA prototype (YuMi). A cylinder is considered as an obstacle on the table during demonstration.

A multi-layered approach of robot LfD [1] has been proposed in analogy with observational learning [11] allowing a user to readily teach a robot both a desired task model and responses to different object classes. This approach only works where the environment is static and obstacles do not move.

Nonetheless, in practical applications, e.g. Human-Robot Collaboration (HRC), a robot must avoid collision with both static and moving obstacles. Here, we focus on a specific use of robot LfD in medium and small size company in which a user needs to teach a task and adaptation to another worker carrying an object in a collaborative environment. For example, assume that a person is teaching an industrial dual arm robot how to place an object in a box by moving its arm to provide a demonstration (Fig. 1), where the robot must perform the task while the task production interferes with an object moved by another worker. Although the robot can learn a task model and a different object avoidance policy for each object using the multi-layered approach, it will not be capable of generalizing the task in an environment with an object moved by a human worker along its path.

In this work, we extend the multi-layered approach of robot LfD enabling a robot to reproduce the learned task in an environment with a moving obstacle. In the context of HRC it is of utmost importance to provide human workers with a methodology to easily teach a robot a desired task and

A. M. Ghalamzan E. is from the Department of Electronic, Information and Bioengineering, Politecnico di Milano, Piazza Leonardo Da Vinci 32, 20133, Milano, Italy (email: {amir.ghalamzan}@polimi.it).

a desired adaptation to an environment with moving obstacle.

Motivating background of this work: Based on the studies of sensorimotor learning in cognitive psychology and neuroscience, many intelligent, context-specific responses of humans to different stimuli during a task execution are consistent with the theoretical framework of optimal control [12]. Here, we adopt the same perspective, using the multi-layered approach of robot LfD to estimate a utility function that represents a set of task demonstrations and can be used to reproduce the task in an environment with a moving obstacle.

To do so, we recognize a learning phase (bottom block in Fig. 2) and a reproduction phase (top block in Fig. 2) in the multi-layered approach of robot LfD. The learning phase is identical to the original approach, while the reproduction phase is proposed to account for a moving obstacle according to the studies of sensorimotor learning in cognitive psychology and neuroscience [12], [13].

In the reproduction phase, the images acquired from an off-the-shelf RGB camera are used by an object tracking block which provides us with measurement of an object position at each time. Furthermore, a Kalman filter is employed to filter the noisy measurements and to provide predictions of the object positions in a time horizon. To include the uncertainty of the estimated object position within the model we modify the utility function used in the multi-layered approach of robot LfD allowing for robustly reproducing a task learned from demonstrations in the presence of moving obstacles. The approach of the multi-layered robot LfD, herein proposed, can be easily extended to a scenario with multiple obstacles as well.

II. Utility Function Formulation

As per multi-layered approach of robot LfD, we consider the problem of producing a task to be formulated as an optimal control problem. A set of trajectories $\zeta_d \in D \,\forall\, d = 1,...,N_{dem}$, are available through demonstrations in the corresponding environments, E_d, corrupting with noise. A single obstacle is also fixed at O_d in the environment during every demonstration.

The optimal control problem is defined by a state space $\mathscr{S} \subset \mathbb{R}^n$, an action space $\mathscr{A} \subset \mathbb{R}^m$, a state transition function $T(s \in \mathscr{S}, a \in \mathscr{A}) : \mathbb{R}^{n+m} \to \mathbb{R}^n$, and a utility function $R(s \in \mathscr{S}) : \mathbb{R}^n \to \mathbb{R}$. We assume the utility function is a function of the state $x \in \mathbb{R}^p$ and of the features of the environment $f \in \mathbb{R}^q$, so that $R(s = (x, f))$ and $n = p + q$. Our goal is to learn a utility function that allows us to compute the optimal action $a \in \mathscr{A}$ at each state $s \in \mathscr{S}$ for a new unobserved environment. A robot can use this utility function, $R(x_t, f_t) = R_I(x_t) + R_E(f_t)$, obtained from demonstrations, to compute a sequence of optimal actions that maximizes the expected return $\rho^\pi = \sum_{t=1}^{T_e-1} R(s_{t+1})$, resulting in an optimal policy π^*,

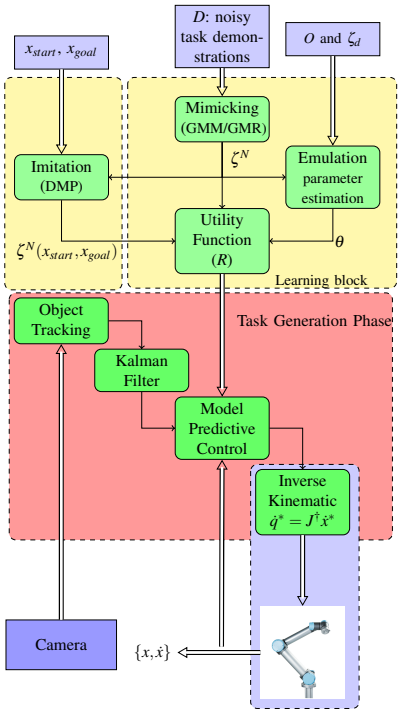


Fig. 2. Incremental robot learning from demonstration scheme (top) and the extended scheme to generate a task in an environment with a moving obstacle (bottom).

as per eq. (1).

$$\begin{aligned} \pi^* = \underset{\pi}{\operatorname{argmax}} \sum_{t=1}^{T_e-1} \left(R_I\left(x_{t+1}\right) + R_E\left(f_{t+1}\right)\right) \\ \text{subj. to} \quad s_{t+1} = T(s_t, a_t), \\ s_t \in \bar{\mathscr{S}}, \\ a_t \in \mathscr{U}, \end{aligned} \tag{1}$$

where

$$\begin{aligned} R_I\left(x_t : \mathbf{Q}\right) = -\left(x_t - x_t^{\mathrm{N}}\right)^T \mathbf{Q} \left(x_t - x_t^{\mathrm{N}}\right) \\ x_t^N \in \zeta^N(x_{start}, x_{goal}) \;,\; t = 1,...,T_e \end{aligned} \tag{2}$$

and

$$R_E\left(x_t, \zeta^N, O : \mathbf{R}\right) = -e^{\left(-(x_t - O_t)^T \, \mathbf{R}^{-1} \, (x_t - O_t)\right)} \tag{3}$$

$\pi = \{a_1, ..., a_{T_e-1}\}$ is a sequence of optimal actions resulting in a sequence of optimal states $\bar{\zeta} = \{s_1, \bar{s}_2, ..., \bar{s}_{T_e}\}$, s_1 is a given initial condition, $\bar{\mathscr{S}}$ and $\mathscr{U}$ are the polyhedral feasible subset of the set of all states and actions respectively. $\mathbf{Q}$ is the imitation parameter and $x_t^{\mathrm{N}} \in \mathbb{R}^n$ is a point on the nominal path $\zeta^N(x_{start}, x_{goal})$. The nominal path may be computed from a set of noisy demonstrations by using GMM/GMR in learning phase, where in the reproduction phase with a new target point the nominal path is computed using a DMP trained by the path computed in the learning phase (Fig. 2). $f_t = x_t - O_t$ is a vector of the environmental features at x_t, captured during the d^{th} demonstration.

Here, execution of a robotic task is assumed to be formulated as an episodic optimal control problem with fixed time horizon T_e, deterministic world, with discrete time and a continuous state-space and action-space.

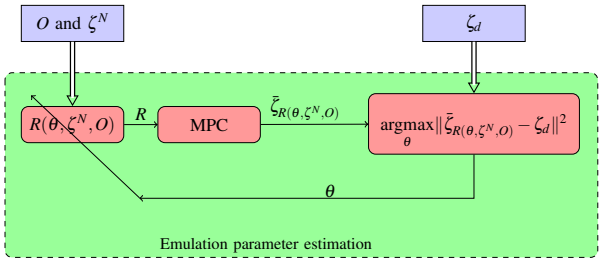


Fig. 3. Parameters of the utility function are computed by minimizing the distance between a demonstration and a reproduction.

Emulation parameter estimation

According to principal of sensorimotor learning [13], humans use the concept of predictive control to regenerate a task in a dynamic environment. We use the same concept; hence, in the learning phase we assume the obstacle position is fixed where in the task generation phase the concept of predictive control is used to generate the task in a dynamic environment. The free parameters of the utility function define a trade-off between imitation and emulation, being close to the nominal trajectory, imitation, and being far from an obstacle, emulation. The value of $\mathbf{Q}$ is fixed to be an identity matrix while the value of the matrix $\mathbf{R}$ is computed so that the cumulative distances between the optimal solution $\bar{\zeta}_{\mathbf{R}(\theta,\zeta^N,O)}$ and the demonstrations is minimized as depicted in figure 3:

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \sum_{d=1}^{D} \sum_{t}^{T_e} \left\| \bar{\zeta}_{R(\theta,\zeta^N,O)}(t) - \zeta_d(t) \right\|^2 \tag{4}$$

where $\zeta_d(t)$ and $\bar{\zeta}_{R(\theta,\zeta^N,O)}(t)$ are the corresponding points on the demonstration and the solution to the estimated utility function. The free parameters θ of the utility are iteratively computed by minimizing eq. (4), using a quasi-Newton strategy and limited-memory BFGS updates [14][1].

III. Generalizing the Task to an Environment with a Moving Obstacle

To account for dynamic environments characterized by the presence of moving obstacles, the multi-layered approach of robot LfD is here extended. Although, for the sake of simplicity, we present the formulation for a single moving obstacle, it can be easily extended to a scenario with arbitrary number of obstacles.

To account for uncertainty in the predicted obstacle position we sum the emulation component of the utility function over a set of points sampled from a uniform distribution, whose expected value is equivalent to the predicted position of the obstacle, as follows:

$$R_g(\mathbf{s}_t) = \sum_{h=1}^{N_{sample}} -\left(x_t - x_t^N\right)^T \mathbf{Q} \left(x_t - x_t^N\right) - e^{-\left(f_{t,h}^T \mathbf{R}^{-1} f_{t,h}\right)} \tag{5}$$

where N_{sample} is the number of sampled data points, $f_{t,h} = x_t - \hat{O}_{t,h}$, and $\hat{O}_{t,h}$ is a sample of the estimated position of the obstacle at time t. Based on the Model Predictive Control (MPC) formulation used in multi-layered approach of robot LfD, one needs to get a prediction of the obstacle position in a time horizon T_p to find an optimal solution at each computational time step. In order to get the obstacle position information, an object tracking is first performed by applying a color filtering algorithm (which is part of the OpenCV library [15]) to the images acquired from a calibrated RGB camera. The resulting object silhouette is further processed to extract its center of gravity in image plane coordinates. The intrinsic and the extrinsic calibration parameters of the camera is computed using Camera Calibration Toolbox for Matlab[2]. Since we consider an example in which the object is moving on the plane of the table, it is possible to reconstruct the Cartesian position of the tracked point and use it as the obstacle position. To filter noisy measurements of the object position and to estimate the object velocity, a linear *Kalman filter* is used. Furthermore, this linear model is used to compute the prediction of the object positions in the prediction time horizon. We derive the equation of motion for an object moving in 2-D with a constant velocity as follows:

$$\mathbf{O}_{i+1} = \mathbf{O}_i + \mathbf{V}_i \Delta t \tag{6}$$

where $\mathbf{O}_i$ is the two-dimensional position of the object at time step i, while $\mathbf{V}_i$ is the two-dimensional velocity, once again at time step i. The state space representation of eq. (6) is

$$\mathbf{y}_{i+1} = \mathbf{F}_i \mathbf{y}_i + \mathbf{w}_i, \qquad \mathbf{z}_i = \mathbf{H}_i \mathbf{y}_i + \nu_i \tag{7}$$

where

$$\mathbf{y}_i = \begin{bmatrix} \mathbf{O}_i & \mathbf{V}_i \end{bmatrix}^T,$$
$$\mathbf{F}_i = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{H}_i = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{8}$$

The designed Kalman filter is also used to compute a set of predicted obstacle position, $\hat{\mathbf{O}}_t = \{\mathbf{z}_{i+1}, ..., \mathbf{z}_{i+T_P}\}$, in the prediction time horizon T_P, where $\mathbf{z}_{i+1} = [O_{1,i+1}, O_{2,i+1}]^T$.

In order to robustly find a point maximizing the utility function, we consider an uncertainty bound on the predicted values. The uncertainty bound is a circle centered at the predicted value. The larger the prediction time, the larger the uncertainty bound on the predicted values. This implies that we consider less confidence in the predicted values with larger prediction time. In order to account for uncertainty bound in the utility function we generate some points uniformly sampled within the considered uncertainty bounds (eq. 9); then, we find the optimal solution over all these sample points (Fig. 4).

$$\hat{O}_{t+1,h} = F(\mathbf{z}_{i+1}, \sigma_{i+1}) \tag{9}$$

The larger the prediction time, the larger the considered radios of the uncertainty bound σ.

$$\sigma_{i+1} = i\, \sigma_0$$

[1]For more details about the multi-layered approach of robot LfD see [1].

[2]http://www.vision.caltech.edu/bouguetj/calib_doc/

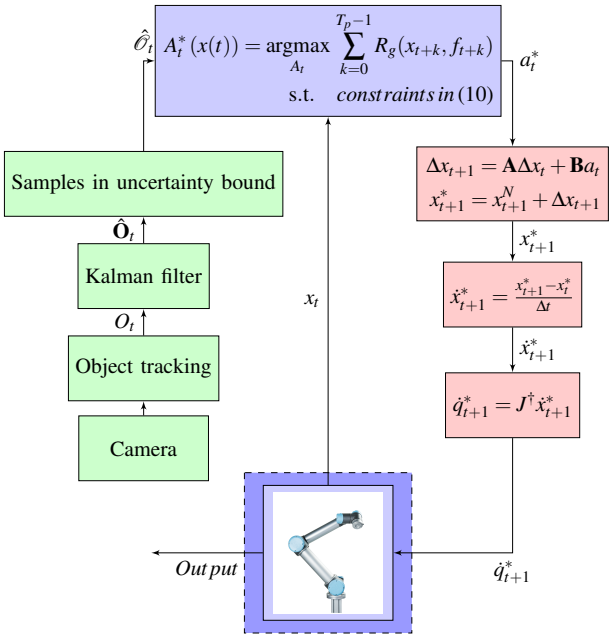


Fig. 4. Scheme of model predictive control architecture used to compute an optimal solution at each time t in the presence of a moving obstacle.

where σ_0 is a tunable parameter. Finally, the set of future obstacle positions, which is used to compute the optimal action, is $\hat{\mathcal{O}}_t = \{\hat{O}_{t,1}, ..., \hat{O}_{t,N_{Sample}}, ..., \hat{O}_{t+T_P,1}, ..., \hat{O}_{t+T_P,N_{Sample}}\}$.

Task Generation Phase: To compute an optimal policy using the obtained utility function, the expected return in eq. (1) is maximized, resulting in a sequence of optimal actions $a_t^* \; \forall t = 1, ..., T_e - 1$. We consider that the task is episodic with a constant episode time interval $\Delta t = \frac{H}{Te}$, i.e. $x_i = x(i\Delta t)$, assuming the robot completes the task in H seconds. To find an optimal solution to the utility function with continuous state space, the following MPC problem with a prediction time horizon T_p is solved.

$$
\begin{aligned}
A_t^*(x(t)) = \underset{A_t}{\text{argmax}} \sum_{k=0}^{T_p-1} & R_g(x_{t+k}, f_{t+k}) \\
\text{s.t.} \quad x_t = x(t),\ \mathbf{O}_t = \mathbf{O}(t) & \qquad \textit{Measurement} \\
\hat{\mathcal{O}}_t = \{\hat{O}_{t+1,1}, ..., \hat{O}_{t+T_P,N_{sample}}\} & \qquad \textit{Prediction} \\
\Delta x_{t+k+1} = \mathbf{A}\Delta x_{t+k} + \mathbf{B}a_{t+k} & \qquad \textit{System model} \\
\Delta x_{t+k} = x_{t+k} - x_{t+k}^N & \\
x_{t+k} \in \mathcal{X} & \qquad \textit{State constraints} \\
a_{t+k} \in \mathcal{U} & \qquad \textit{Action constraints} \\
A_t = \{a_{t+k}, ..., a_{t-1+T_p}\} & \qquad \textit{Optimization variables}
\end{aligned}
\tag{10}
$$

where $\mathcal{X}$ is the polyhedral feasible sets of robot states, $\mathbf{A}$ and $\mathbf{B}$ are chosen such that the corresponding equations form a stable dynamical system. At time t, a sequence of optimal actions, $A_t^*(x(t)) = \{a_t^*, ..., a_{t+T_p}^*\}$, is computed in the prediction time horizon T_p. However, only the first action, a_t^*, at time t is used to move the robot to the next optimal position, resulting in a sequence of optimal actions, $\pi^* = \{a_1^*, ..., a_{T_e-1}^*\}$. Hence, the corresponding sequence of optimal states is $\bar{\zeta}_{R(\theta,\zeta^N,O)} = \{x_1, \bar{x}_2, ..., \bar{x}_{T_e}\}$ with initial value x_1. As per [1], the action a^* is computed such that the vectors $\overrightarrow{x_t^N x_{t+1}^N}$ and $\overrightarrow{x_i x_i^N}$ are always orthogonal (for further information see [16]).

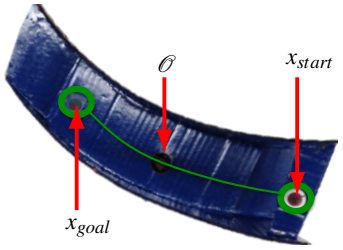


Fig. 5. The task model used for data collection with da Vinci surgical robot with a single obstacle (marker). The green line is the nominal path that expert follows in the absence of obstacles;

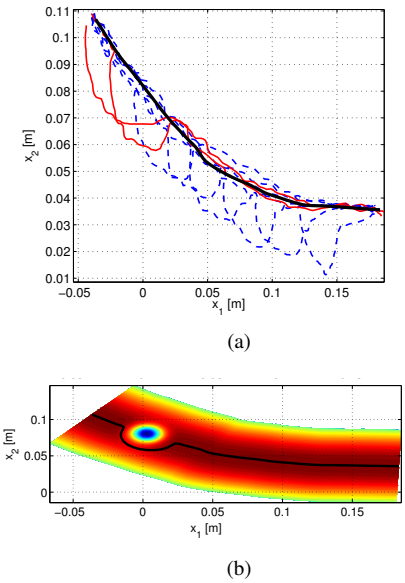


Fig. 6. (a) The data set collected with da Vinci robot for different positions of the marker (green shaded circles), training set (red solid line) and test set (blue dashed line), the average path computed by GMM/GMR (black thick line); (b) The contour of the learned utility function for the da Vinci experiment, and the computed optimal solution to the utility function. Areas with hotter colors represent positions with higher associated utilities.

A scheme of the extended controller is shown in Fig. 4, integrating MPC, object tracking; Kalman filter and inverse kinematic.

IV. Experiment

We present an example of pick-and-place task during surgical training while a fixed marker exists in the environment. This is a common task during surgeon training. To simulate the small available space within a patient's body, a specifically designed structure, shown in Fig. 5, has been used in [1] to collect a data set composed of eight pick-and-place trajectories going from x_{start} to x_{goal} (Fig. 6). The operator performed the pick-and-place task while avoiding collision with both walls and a marker fixed within the structure at different positions.

At first, data set of demonstrations has been collected:

$$\zeta_d = \{x_{d,1}, \ldots, x_{d,T_e}\},\ d = 1, \ldots, N_{demo}$$

TABLE I

ERROR AND ACCURACY OF PICKING AND PLACING TASK WITH DA VINCI.

MSE_R	: 0.0911	$var(MSE_R)$	: 0.0064
$Pr_{training}$	: 84.0%	Pr_{test}	: 79.5%

where $N_{demo} = 8$ was the number of demonstrations (Fig. 6(a)).

The utility function representing the demonstrations has been computed using eq. (4). The corresponding nominal path ζ^N and a generated path using the obtained utility function are shown in Fig. 6.

The data set is divided into a training set (blue dashed line in Fig. 6(a)) and a test set (red line in Fig. 6(a)). To evaluate the obtained task model, the mean square error (*MSE*) of the paths generated by the obtained model is computed for both test set and training set, as follows:

$$MSE_R = \frac{1}{N_{(dem,.)}T_e} \sum_{d=1}^{N_{(dem,.)}} \sum_{i=1}^{T_e} \|x_{d,i} - x_t^R\|$$

where $T_e = 100$ is the number of sample points of the demonstrated path and $N_{(dem,.)}$ is the number of paths within the training set ($N_{(dem,tr)} = 2$) or test set ($N_{(dem,tst)} = 6$).

Furthermore, an accuracy measure for every generated path is computed (Table I), as follows:

$$Pr = \frac{MSE_{av} - MSE_R}{MSE_{av}} \times 100$$

where

$$MSE_{av} = \frac{1}{N_{(dem,.)}T_e} \sum_{d=1}^{D} \sum_{i=1}^{T_e} \|x_{d,i} - x_t^N\|$$

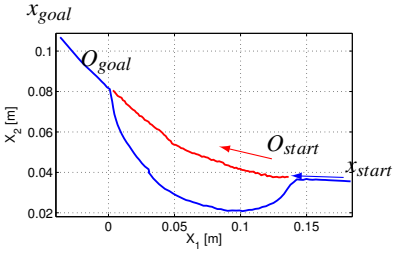
and x_t^N is a point of the average path computed by Gaussian Mixture Model/Gaussian Mixture Regression (GMM/GMR).

The computed Pr for the training set and the test set in Table I show that the obtained model with the training set generates the paths corresponding to the ones in the test set with good accuracy.

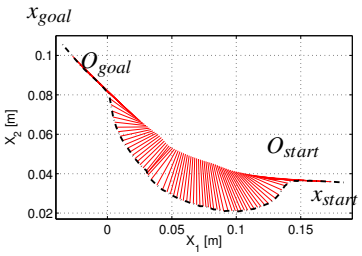
Simulation of the task generation with a moving bstacle: We further simulate the task reproduction with a moving marker. The marker is considered to move along the average path with constant velocity V_o while the operator moves the da Vinci tool with a constant velocity $V_r > V_o$. Hence, at time $t = 0$ both the tool and the marker start moving from O_{start} and x_{start} with constant velocity V_o and V_r, respectively.

As shown in Fig. 7, the learned model and the extended reproduction phase is capable to robustly reproduce the task in the presence of a moving object. This simulation illustrates the effectiveness of the proposed approach to deal with moving obstacles. In the future work, we apply the extended multi-layered robot LfD to the manipulation task with YuMi robot.

In our experiment, $T_P = 5$, $N_{sample} = 3$ and $\sigma = 0.01$.
In future work, we are going to use the proposed approach to teach a YuMi bimanual robot a pick-and-place task where the robot replicate the task with different placing position while avoiding collision with a moving obstacle (Fig. 1).



(a)



(b)

Fig. 7. (a) Task reproduction in the presence of a moving obstacle (blue line), which starts from x_{start} and ends at x_{goal}. During task reproduction, the obstacle moves along the red line from O_{start} to O_{goal}; (b) the lines depict the correspondence sample points of the path followed by obstacle and robotic tool at each time step.

V. CONCLUSION

Robot LfD has been proposed to increase the speed and reduce the complexity of programming an industrial manipulator. A major challenge, however, is the generalization of task demonstrations to complex dynamic environments. Although some methods has been used to generalize a set of demonstrations to a new goal point and a new environment with stationary obstacles, they are not capable of generalizing demonstrations to an environment with moving obstacles.

In this paper, we extended the approach proposed in [1] to enable a robot to reproduce a task in a dynamic environment. The extended approach allows a robot to learn a task model from demonstrations in an environment with stationary obstacles and reproduce the task in an environment with moving obstacles. This study implies that human may not necessarily need to learn different models from demonstrations for dynamic and static environments. In future work, we apply the extended multi-layered robot LfD to the manipulation task with YuMi robot.

REFERENCES

[1] Ghalamzan, A., C. Paxton, G. Hager, and L. Bascetta, "An incremental approach to learning generalizable robot tasks from human demonstration," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 5616–5621.

[2] A. J. Ijspeert, J. Nakanishi, and S. Schaal, "Movement imitation with nonlinear dynamical systems in humanoid robots," in *International Conference on Robotics and Automation*, vol. 2. IEEE, 2002, pp. 1398–1403.

[3] S. Calinon, *Robot Programming by Demonstration*. EPFL Press, 2009.

[4] B. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and autonomous systems*, vol. 57, no. 5, pp. 469–483, 2009.

[5] S. Calinon, D. Bruno, and D. G. Caldwell, "A task-parameterized probabilistic model with minimal intervention control," in *International Conference on Robotics and Automation*. IEEE, 2014, pp. 3339–3344.

[6] S. Calinon, I. Sardellitti, and D. G. Caldwell, "Learning-based control strategy for safe human-robot interaction exploiting task and robot redundancies," in *International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 249–254.

[7] F. Guenter, M. Hersch, S. Calinon, and A. Billard, "Reinforcement learning for imitating constrained reaching movements," *Advanced Robotics*, vol. 21, no. 13, pp. 1521–1544, 2007.

[8] P. Kormushev, S. Calinon, and D. G. Caldwell, "Robot motor skill coordination with em-based reinforcement learning," in *International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 3232–3237.

[9] H. Hoffmann, P. Pastor, D. Park, and S. Schaal, "Biologically-inspired dynamical systems for movement generation: automatic real-time goal adaptation and obstacle avoidance," in *International Conference on Robotics and Automation*. IEEE, 2009, pp. 2587–2592.

[10] D. Park, H. Hoffmann, P. Pastor, and S. Schaal, "Movement reproduction and obstacle avoidance with dynamic movement primitives and potential fields," in *International Conference on Humanoid Robots*. IEEE, 2008, pp. 91–98.

[11] A. Whiten, N. McGuigan, S. Marshall-Pescini, and L. M. Hopper, "Emulation, imitation, over-imitation and the scope of culture for child and chimpanzee," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 364, no. 1528, pp. 2417 – 2428, 2009.

[12] E. Todorov, "Optimality principles in sensorimotor control," *Nature neuroscience*, vol. 7, no. 9, pp. 907–915, 2004.

[13] D. M. Wolpert, J. Diedrichsen, and J. R. Flanagan, "Principles of sensorimotor learning," *Nature Reviews Neuroscience*, vol. 12, no. 12, pp. 739–751, 2011.

[14] M. Schmidt, "Graphical model structure learning with l1-regularization," Ph.D. dissertation, University of British Columbia, 2010.

[15] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[16] Ghalamzan, A., L. Bascetta, M. Restelli, and P. Rocco, "Estimating a mean-path from a set of 2-d curves," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 2048–2053.