2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

oHMM-PA: A Learning from Demonstration Approach Using Online Hidden Markov Models with Path Planning

Jan Irsperger[†], Diego Fernandez Prado[†2], and Eckehard Steinbach[1]

Abstract— **Learning from Demonstration (LfD) enables humans to teach skills to robots quickly, without the need of knowledge in programming or robotics. Despite many proposed approaches, the field still lacks a stable solution with high task generalization, industrial-level success rates, and minimal demonstration requirements. In this work, a combination of Coarse-to-Fine (CtF) LfD with online Hidden Markov Models (HMMs) and impedance control is proposed to achieve higher task generalization with no prior task knowledge using only a handful of demonstrations. A coarse trajectory is generated to approach the task object using a self-supervised neural network with visual data. This is followed by an object interaction trajectory, employing the HMM approach with haptic data and a non-linear stiffness impedance controller. A new online HMM path algorithm is employed, demonstrating its ability to adapt to changes in the environment during execution. Experiments on a robotic arm show that complex tasks can be learned within 30 minutes and accomplish a 95 % success rate with kinesthetic teaching. The learned skill is stable and evaluated analytically.**

I. Introduction

Robots are used in countless applications, ranging from industry, medicine, space exploration and agriculture, all the way to consumer products like vacuum cleaning robots [1], [2]. The main reason for this is their ability to solve tedious tasks automatically, allowing human workers to be involved in more creative tasks. Additionally, further advantages from automation are increased productivity, higher product quality or non-human interaction in dangerous environments [1], [2].

Traditionally, robot automation involves a human who carefully programs the robot, but this requires the human to possess robotics and programming skills. The complexity of programming robot skills makes re-configuring production lines in factories very time-consuming and limits the speed at which a new product can be finalized. In this context, Learning from Demonstration (LfD), often also referred to as Imitation Learning, can greatly reduce the amount of time required to reprogram robots and it also allows workers with no programming knowledge to teach the task as good as an expert roboticist would.

In order to learn the best skill for a task, the worker must carefully select and record the most relevant data modalities. Learning from purely visual data has proven successful in tasks related to grasping [3], [4] and manipulation [5]. In cases where the applied force plays an important role, like in contact-rich tasks (e.g. peg-in-hole or gear insertion), combining the visual data with haptics has been applied with success [6], [7]. Nevertheless, the differences between these two modalities, such as receiving fields, signal types or frequencies, hinder the application of these methods. Additionally, these methods usually assume that visual and haptic data are available at all time, which is often not true in case of industrial insertions, where haptic interaction is only felt towards the end of the execution when in contact with the objects.

In this paper, we present a new online LfD approach that uses Hidden Markov Model (HMM) with path planning, allowing the robot to quickly adapt to the current uncertain conditions of the environment. Additionally, we use this algorithm in a pipeline that expands the work of Johns [8] and Prado et al. [9], decoupling the visual and haptic data processing by modeling the robot's motion as a two-stage process, in a Coarse-to-Fine (CtF) fashion: a coarse object approaching process followed by a fine interaction step.

The contribution of this work is the combination of a new online path planning algorithm operating on the transition matrix of a HMM learned from demonstrations. Key aspects include that the algorithm tries multiple paths if the robot gets stuck or a pathway is blocked. This is achieved because during every execution step, paths of the HMM tree, learned from the different demonstrations, are combined to find a solution. Furthermore, the fine trajectory, generated from a statistical model, shows that task success is still possible even when there are errors from the visual coarse trajectory. We present our algorithm in detail, show experiments on the real robot and compare success rates to other commonly used algorithms, aspects that other research papers are missing.

The CtF step limits the scope of our work to tasks where a contactless top-down approaching step is followed by an interaction step. Nevertheless, the online approach itself should be able to handle any multi-dimensional motion. Tasks that require forceful interaction, like sliding movements, are possible due to the impedance controller. Additionally, the online approach is able to try multiple paths from the learned demonstrations during execution, e.g., if a pathway is blocked or the robot is stuck. Tasks where the interaction object may be in different states can be recognized and handled by the algorithm if the demonstrations provide enough information. Specifically, the multi-path aspect allows for task completion when the object can have different states, e.g., in a gear insertion task, the gear teeth positions can vary. Tasks with long executions and thus high amount of

[1]The authors are with the Technical University of Munich / School of Computation, Information and Technology / Chair of Media Technology, and Munich Institute of Robotics and Machine Intelligence (MIRMI), Munich, Germany. Email: {jan.irsperger, diego.prado, eckehard.steinbach}@tum.de

[2]Diego Fernandez Prado is also with Agile Robots SE, Munich, Germany.

[†]Authors contributed equally to this work. Corresponding author: Diego Fernandez Prado (diego.prado@tum.de).

states were not investigated. A high amount of states may lead to a too large computational complexity.

II. Related Work

In this section we present related research in the areas of LfD, and more specifically, of HMM-based LfD approaches.

A. Learning from Demonstration

LfD enables robots to learn and replicate skills by observing the actions and decisions of human operators. This approach leverages the expertise of humans to train robots, enabling them to perform tasks that might be challenging to program explicitly.

In the last few decades, many LfD approaches have been proposed. One class are statistical models that often require only five to ten demonstrations. One such statistical model is the HMM that was used by Calinon et al. [10] to replicate reaching tasks from demonstrations on a humanoid robot. This work was continued by Asfour et al. [11], who used HMMs with key points to find the best trajectory from a few demonstrations. Key points are samples in the motion data that are characteristic to the movement of the given task. Asfour's work determined those samples by detecting angles in the trajectory data, which are conditioned to be larger than a chosen threshold. The key point approach was also used by Kumra and Sahin [12], where the trajectory samples were grouped over all demonstrations using k-means clustering. A statistical model that also considers the time-aspect of the demonstrations are Hidden semi-Markov Models (HSMMs), which were used by Tanwani et al. [13].

A popular robot decision and control approach that extends on Markov chains by using actions and rewards is the Partially Observable Markov Decision Process (POMDP) [14][15]. To determine the current state, POMDP uses a belief state estimated by the previously executed actions, new observations and old belief state. In contrast, our work uses the Gaussian PDF of a state to deduce the current state. For POMDP in a LfD context, a policy is learned from demonstrations using rewards. A trajectory can be executed online by determining the next action from the learned policy and model [14][16]. In contrast, we use demonstrations to train a HMM and during online execution our own path algorithm determines the next state from the HMM. The next state serves as a movement or action for the robot. Thus, we do not require to define a reward, action or policy. The combination of a POMDP with Monte-Carlo Tree Search (MCTS), proposed by Silver and Veness [17], is similar to our path algorithm operating on HMMs. MCTS is used to determine the next action in four steps: selection, expansion, simulation and backpropagation. MCTS builds a tree during the expansion phase, which is different to our work where the tree structure of the HMM is used instead. Another distinction of MCTS is the simulation of actions and the backpropagation of rewards to the model. Instead, we evaluate a limited amount of paths that lead to the final states on the offline trained HMM tree. Only a state history is kept per execution.

A different LfD approach that requires only a single demonstration proposed by Johns [8] uses a Neural Network (NN) to guide the robot always to the same pose relative to the interaction point (bottleneck pose). From the bottleneck pose, it is enough to just replay the same demonstration in order to successfully complete a task. Therefore, an expert must only provide the bottleneck pose and one demonstration. The NN, which guides the robot to the aforementioned pose, is trained in a self-supervised manner by having the robot automatically move around the object from various directions. Overall, the approach showed various success rates from 0% to 100% for several different tasks. The task complexity and the precision variation in reaching the bottleneck pose make the success rate vary.

B. Online vs Offline Learning

The HMM approaches mentioned so far were all used in an offline setting, which means that a trajectory was precomputed for the robot to execute. In contrast, online means that the trajectory is computed at every time step during the execution, allowing the robot to adapt to the actual physical environment. One such online approach is shown in the work of Le et al. [18]. They use a HSMM with a modified Viterbi algorithm to calculate the next step during the execution. In our work, this is achieved with a path planning algorithm operating on the HMM state transition tree. In the work of Le et al. [18], the pose of the interaction object is assumed to be known and supplied to the HSMM. Additionally, their online approach uses the object's pose to correct the trajectory in case the position is off during execution. In contrast, our approach does not require object pose information.

III. Methodology

A. Setup

The experiments in this work were performed on a Franka Emika Panda robot arm. A Realsense camera d435i is mounted on the gripper. The end-effector frame's forces and torques are calculated using the robot model and its seven built-in torque sensors. Position, rotation and force data of the end-effector are sampled with a frequency of 100 Hz.

Four representative industrial tasks were evaluated: gear insertion, power plug insertion, key insertion, and screw fastening. The setups are shown in Figure 1. For the gear insertion and screw fastening tasks, the ground objects were rigidly mounted on the surface of the ground plane. For the other two tasks the ground object was movable.

B. Hidden Markov Models

HMMs are useful to model a system or process that can be segmented into various states s_i, where $i \in [0 \ldots N-1]$ and N is the amount of states. Each state s_i has so called emissions that link to the actual observable data $d_t \in \mathbb{R}^C$ with C being the amount of variables and $t \in \{1,2,\ldots T\}$ where T is the amount of observed data. In this work, one sample d_t has $C = 9$ variables and is given by $d_t = (x,y,z,r_x,r_y,r_z,F_x,F_y,F_z)$, where (x,y,z) are the Cartesian coordinates, (r_x,r_y,r_z) are the Euler rotations with order xyz

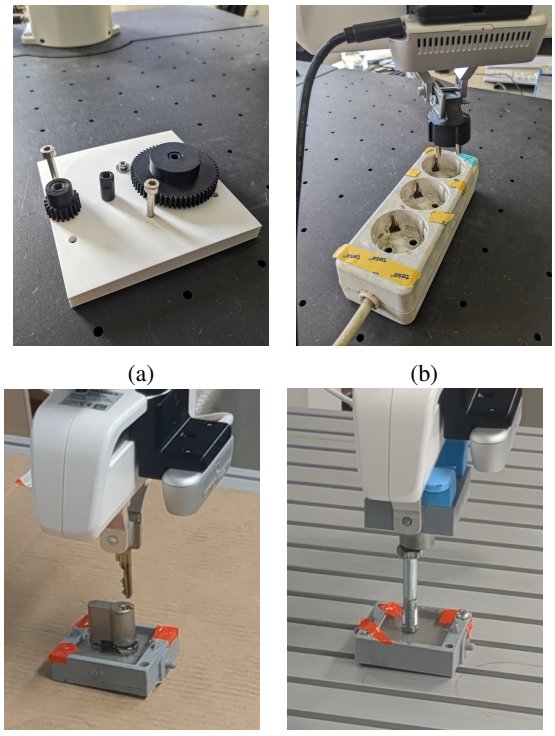

Fig. 1: Image of the (a) gear, (b) power plug, c) key and d) screw tasks.

and (F_x, F_y, F_z) are the forces in Cartesian axis direction of the end-effector in the bottleneck frame. A HMM with Gaussian emissions is employed in this work. Therefore, each state has a mean vector $\mu_i \in \mathbb{R}^C$ and a covariance matrix $\Sigma_i \in \mathbb{R}^{C \times C}$. The probability of a sample d_t belonging to a state s_i is then given by the multivariate Gaussian probability density function (PDF) $p(d_t|\mu_i, \Sigma_i)$. The emissions in each state are only dependent on the current state and not on the previous ones. To reflect the change of states over time within a system, the state transition probabilities a_{ij} give the probability going from one state s_i to another state s_j. State transition probabilities a_{ij} over all states are represented with a matrix $A \in \mathbb{R}^{N \times N}$. The initial state probabilities are given by a vector $\pi \in \mathbb{R}^N$. A continuous HMM λ with Gaussian emissions is thus defined by $\lambda = (A, \mu, \Sigma, \pi)$.

C. Learning from Demonstration

The LfD approach in this work consists of two parts. First, the visual coarse approach of [8] is adopted to move the end-effector to a bottleneck pose. The bottleneck is a fixed pose relative to the object. Second, in contrast to Johns' fine trajectory, a continuous HMM with Gaussian emissions is trained with LfD to perform an online executed trajectory from the bottleneck pose. The online trajectory employs impedance controlled motion to achieve forceful interaction and human like behaviour. The approach benefits from the combination of the visual coarse trajectory and the HMM fine trajectory because the HMM can compensate for inaccuracies of the coarse trajectory. This leads to higher generalization of a task because small changes in the object environment can be caught statistically by the HMM. Furthermore, the HMM is employed with an online algorithm that reacts to the environment with forceful interaction, thus allowing for higher task complexity and task generalization. Moreover, the HMM parameters can be analytically evaluated. All benefits are achieved while maintaining no prior task knowledge and only a few demonstrations.

A new approach is proposed in this work to train the HMM and execute an online trajectory. The overall procedure is shown in Figure 2. First, a HMM λ with Gaussian emissions is trained by using key points as state means. Key points are extracted from the demonstrations with angle detection. The method of detecting key points through angles is similar to the work in [11] with some modifications. This method allows that no hyper parameters need to be adjusted between tasks. The HMM is trained with the Baum-Welch algorithm without updating the mean μ_i. Second, a new online algorithm, called Online HMM Path Algorithm (oHMM-PA), executes a trajectory with impedance controlled motion.

The proposed oHMM-PA is compared against two state of the art statistical LfD methods. The first comparison is to a Gaussian Mixture Model (GMM) by Calinon [19]. The second comparison is to an HSMM by Calinon and Lee [20]. Offline trajectories are calculated from both methods and used as fine interaction trajectories.

Recording the demonstrations to obtain training data for the models is done in two steps. First, the bottleneck pose is demonstrated by a user. Second, the coarse trajectory is executed to reach the bottleneck pose. A total of L fine demonstrations are demonstrated from the bottleneck pose. Between the fine demonstrations, the coarse trajectory is rerun. The recording and training procedure is summarized in Algorithm 1. Recording the demonstrations in this way enables the statistical HMM to capture the variation of the visual coarse approach.

Algorithm 1 Recording Demonstrations and Training Procedure

```
Demonstrate bottleneck pose
Capture coarse dataset self-supervised
Train CNN with coarse dataset
L ← Amount of fine demonstrations
l ← 0
while l ≤ L do
    Execute coarse trajectory to bottleneck pose
    Demonstrate fine trajectory
    l ← l + 1
end while
Train HMM with fine demonstrations
```

D. Key Points and Angle Detection

The LfD approach shown in Figure 2 uses key points as state means of the HMM λ. A key point is defined if the angle of a point in the trajectory across all variables is greater than a certain threshold θ. Furthermore, the start point of the

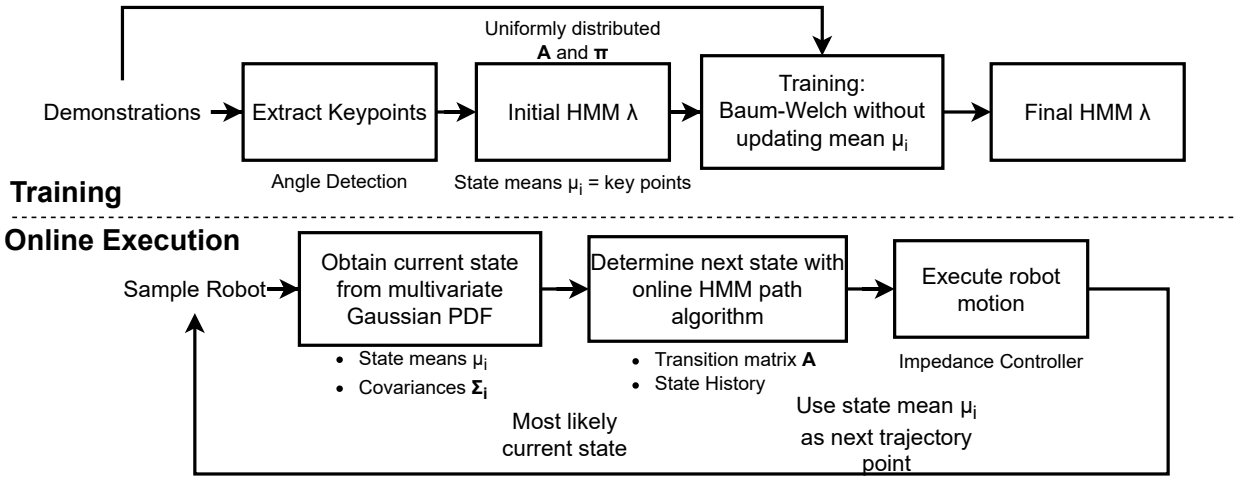

Fig. 2: An overview of the proposed LFD approach.

first demonstration's trajectory and all end points are also defined as a key point. To avoid tuning the later defined minimum distance hyperparameter between different robotic tasks, a normalization is performed on the samples $\mathbf{d}_t$ of the demonstrations. The normalization factor $\mathbf{l}_{norm}$ is defined as

$$\begin{aligned} l_{norm}^{(u)} &= \max_{d_t^{(u)}} \ |\mathscr{D}|, \quad u \in [1,\dots 9], \forall t, \forall l, \\ \mathbf{d}_t &= (d_t^{(1)}, d_t^{(2)}, \dots d_t^{(9)}), \\ \mathbf{l}_{norm} &= (l_{norm}^{(1)}, l_{norm}^{(2)}, \dots l_{norm}^{(9)}) \end{aligned} \tag{1}$$

which is the maximum value per variable u over all samples of all demonstrations l in the dataset $\mathscr{D}$. Thus, all samples of the dataset $\mathscr{D}$ are normalized by

$$(\mathbf{d}_t)_{norm} = \left(\frac{d_t^{(1)}}{l_{norm}^{(1)}}, \frac{d_t^{(2)}}{l_{norm}^{(2)}}, \dots \frac{d_t^{(9)}}{l_{norm}^{(9)}}\right) \quad \forall t, \forall l, \tag{2}$$

before calculating the angles in the trajectory. The resulting samples of the demonstration are denoted as $(\mathbf{d}_t)_{norm}$. For a sample $\mathbf{d}_t$ in the demonstrations to be a key point k_i, any of the three following conditions must apply:

$$\begin{aligned} &1)\ \ angle((\mathbf{d}_t^{(u)})_{norm}, (\mathbf{d}_t^{(v)})_{norm}) > \theta, \text{ with } 1 \le t \le T, \forall l \text{ and} \\ &\quad u, v \in [1\dots 9],\ u \neq v \\ &2)\ \ t = 1 \ \textit{and}\ \ l = 1 \\ &3)\ \ t = T,\ \forall l, \end{aligned} \tag{3}$$

where $angle()$ calculates the angle of the lines given by the variables $(u), (v)$ of the samples at $t-1$, t and $t+1$ in the trajectory, and T is the amount of samples per demonstration.

Additionally to the above angle criteria, a minimum distance between key points is defined. This reduces the amount of key points because some key points are perhaps in a close proximity given that the sample rate is 100 Hz. Moreover, close states might not provide any additional information to the HMM in order to model the process. The Euclidian distance is used to define a criteria

$$||(k_i)_{norm}, (k_j)_{norm}||_2 > \Delta_{min}, \quad \forall i, \forall j, j \neq i \tag{4}$$

which enforces a minimum Euclidian distance Δ_{min} between all key points.

E. Online HMM Path Algorithm

The online algorithm proposed in this work allows the robot arm to react to the environment by influencing the trajectory based on continuously sampled observations at each iteration. Decisions are made depending on the observations and the offline trained statistical HMM λ. A detailed view on the online algorithm is shown in Figure 3. The algorithm assumes that the robot is first moved to the bottleneck pose. Subsequently, repeated steps are performed starting with sampling a new observation. The first observation $\mathbf{d}_1$ defines the bottleneck frame. Every sample is transformed to the bottleneck frame resulting in a sample $\mathbf{d}_t$. Afterwards, the current state s_c is determined using the multivariate Gaussian PDF. For every state, the probability of the sample $\mathbf{d}_t$ is calculated using the PDF. An assumption is made that the state with the highest probability equals the current state, thus, s_c is determined by

$$s_c = \underset{s_i}{argmax}\ p(\mathbf{d}_t | \mu_i, \Sigma_i) \tag{5}$$

where $p(\mathbf{d}_t | \mu_i, \Sigma_i)$ is the Gaussian PDF.

The next state s_n, to which the robot should move, is determined with oHMM-PA. The idea is to find a path through the state transition matrix A from the current state s_c to any of the demonstrations' final states. Thus, the first criteria of the online HMM path algorithm is defined as:

Criteria 1: Find a path to any of the final states.

With L demonstrations there are L possible final states as defined by the angle key point criteria in Equation 3. Finding a path is achieved by following the states with the highest probabilities until a final state is reached. At each state, M

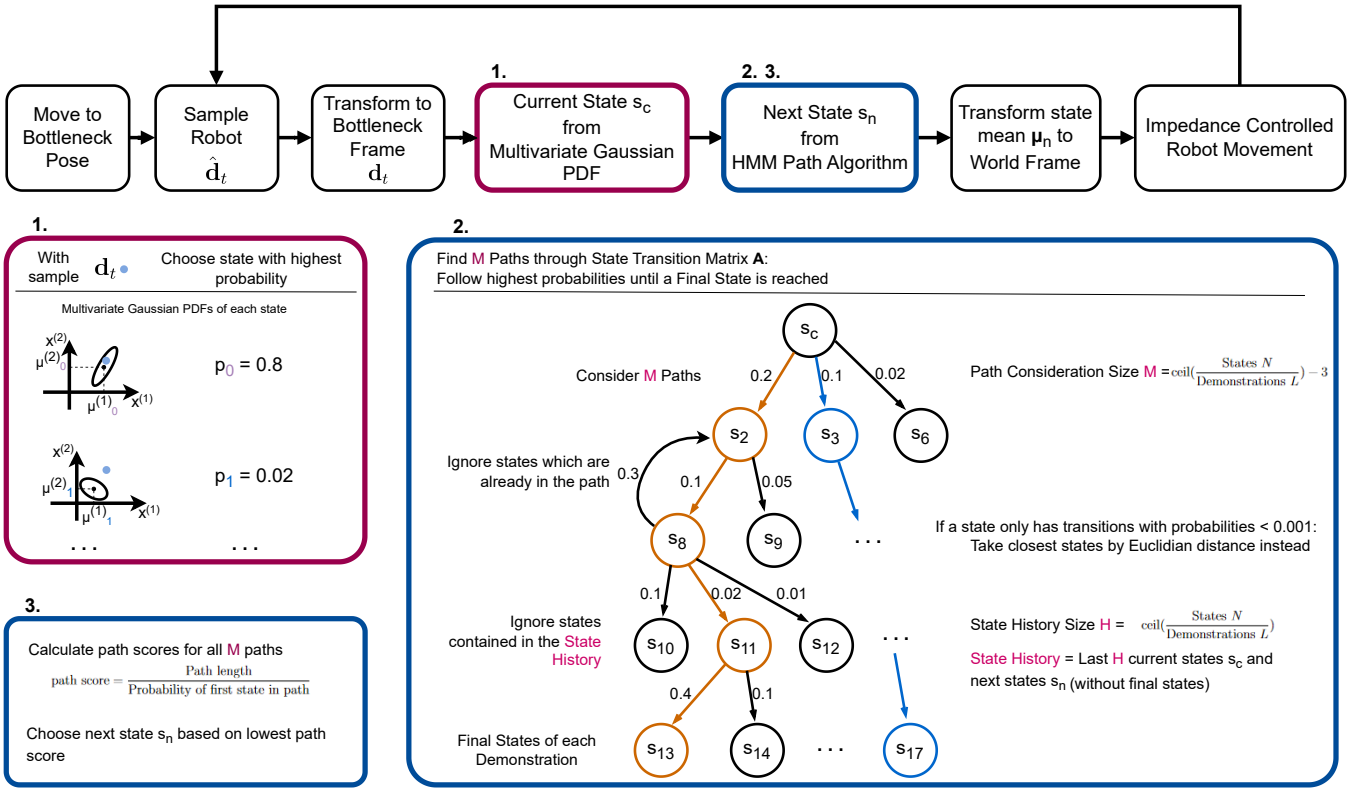

Fig. 3: The proposed Online HMM Path Algorithm.

next states are considered, ordered from highest to lowest probability. This condition is written as:

Criteria 2: Follow states by highest probabilities. Consider M states at each branch.

M is called the path consideration size and determined by

$$\mathrm{M} = \mathrm{ceil}\left(\frac{\#\text{States } N}{\#\text{Demonstrations } L}\right) - 3. \tag{6}$$

This equation was chosen similar to Equation 7, but with one difference. The path consideration size can be lowered to reduce computation time, in this case -3 was chosen empirically. Increasing M leads the algorithm to check more path possibilities. To prevent state loops during the path search, states that are already in the current path are not considered. Accordingly, a criteria is defined as:

Criteria 3: Ignore states that are already in the current path.

Furthermore, to prevent state loops during the execution, a state history is defined which consists of the last H current states s_c and next states s_n. The state history size H is calculated by

$$\mathrm{H} = \mathrm{ceil}\left(\frac{\#\text{States } N}{\#\text{Demonstrations } L}\right). \tag{7}$$

States in the state history are not considered during the path search. Thus, another rule is given by:

Criteria 4: Ignore states contained in the state history.

This estimation of H origins from the assumption that each demonstrations has roughly the same amount of key points to reach a final state, a number that is estimated to be the total number of states divided by the number of demonstrations. Thus, if the robot is following a path, then the state history should contain all states from the start of the path to the current state. As a result, when calculating a path to the final states, states which are already in the state history are not considered. Additionally to preventing state loops, this criteria leads the algorithm to explore new path possibilities if a path was previously executed by the robot but a state in the path was not reachable. Note, that final states are not included in the state history. States may have lead away transition probabilities that are close or equal to zero, meaning that states can be a dead end as they have no transitions leading further. Therefore, during the path search, transitions leading away from a state s_i with probabilities lower than $a_{ij} = 0.001$ are ignored. Instead, in order to consider a total of M paths, the remaining transitions are replaced by the closest states with respect to the Euclidean distance to s_i. This defines the last criteria:

Criteria 5: Replace transitions with probabilities close to zero by states determined through the minimum Euclidean distance.

Assuming K transitions are replaced, the probabilities of the new states are calculated by

$$a_{ij} = 0.001 \cdot (M - k) \tag{8}$$

where $k \in \{0, 1, ..., K-1\}$ is the k-th replaced state ordered from closest to farthest. Thus, closer states are considered with a higher probability. The probability threshold of 0.001

was determined empirically by observing that transitions around this value were converging to zero with longer Baum-Welch training. In summary, the Euclidean distance ensures that the trajectory continues from a nearby state if there is a dead end. This may particularly happen when a robot movement to a previous next state s_n was not successful and no other valid transitions are available.

Finally, to obtain the next state s_n, a path score is calculated for the M paths of the current state s_c. The path score is given by

$$\text{path score} = \frac{\text{Path length}}{\text{Probability of first transition in path}} \tag{9}$$

meaning that larger path lengths result in a higher path score. A high probability in the first transition of the path reflects a lower path score. Therefore, the next state s_n is chosen from the path with the lowest path score.

The mean μ_n of the next state s_n is transformed to the world frame to serve as a next trajectory point d_n. The robot is commanded in an impedance controlled way to perform a movement to $\mathbf{d}_n$. Upon reaching the desired pose or if no movement of the robotic arm is detected anymore, the online algorithm starts again from the sample step.

F. Motion Control

An impedance controller is used to replicate the human demonstrations. Human motions are not stiff. The parameters of an impedance controller can be adjusted so that a robot motion is similar to human motions. Another advantage is controlled forceful interaction.

An impedance controller models the motion as a mass-damper-spring system given by

$$M(\ddot{x} - \ddot{x}_d) + D(\dot{x}_d - \dot{x}) + K(x_d - x) = F_{ext} \tag{10}$$

where M is the inertia matrix, D the damping matrix, K the stiffness matrix, x the pose vector of the end-effector, x_d the desired pose vector of the end-effector and F_{ext} is the Cartesian contact force vector of the end-effector [21]. The resulting total force can be transformed into torques with the transposed Jacobian matrix J^T.

The implementation for the impedance controller is adopted from Mayr and Salt-Ducaju [22]. The controller uses the pose and velocity error as direct feedback with the stiffness K and damping D to calculate the forces and torques. The aforementioned controller showed a deviation of 1 *mm* to 15 *mm* between the target position and end-effector position. As a simple solution to this problem, a non-linear impedance controller stiffness is proposed in this work. The non-linearity aims at ensuring high enough forces to move the robot even when the positional error $\Delta x = (x - x_d)$ is small. The non-linearity is only applied below 10 *mm* errors such that the robot arm still moves towards the target pose. As a result, the stiffness force relation of Equation 10 was changed to

$$F = \begin{cases} sgn(\Delta x) \cdot (-0.00875783\ \Delta x^4 + 0.215627\ |\Delta x|^3 \\ -1.8442\ \Delta x^2 + 6.63735\ |\Delta x|), & \text{if } |x| \leq 10mm \\ K \cdot \Delta x, & \text{otherwise} \end{cases}$$

which was empirically found and keeps the linear behavior for errors above 10 *mm* and uses a polynomial function of fourth degree otherwise. In our experiments, positional errors below 1 *mm* were achieved with this method.

	oHMM-PA (Ours)	HSMM10	GMM10	GMM49
Gear ins.	**95%**	25%	20%	50%
Plug ins.	**95%**	-	-	-
Key ins.	**90%**	70%	50%	30%
Screw fast.	**95%**	65%	60%	60%

TABLE I: Results from the measurement series. The interaction object was placed with 20 different object poses and for each imitation learning approach a complete CtF execution was performed. The success rate for each method is displayed.

IV. Results

With the LfD approach of this work, the total time required to teach the robot a new task takes about 20 to 30 minutes. This includes demonstrating the bottleneck pose, capturing a self-supervised dataset for the coarse trajectory, training the neural networks, demonstrating the fine trajectory for the HMM and training the HMM. A total of five interaction trajectories were demonstrated per task with kinesthetic teaching.

To assess the success rate of our LfD approach, a real-world measurement series was conducted. An angle threshold of $\theta = 45°$ was used for the key point criteria defined in Equation 3 and the normalized minimal Euclidian distance between key points was empirically set to 0.3, resulting in 48, 39, 82 and 35 states for the gear, power plug, key and screw model respectively. Pre-processing and training hyperparameters were identical between tasks to emphasize the generalization of the approach. For the power plug and key task, an external wrench in z-direction was commanded to reach the high force requirements of the insertion.

To compare our approach, offline trajectories trained on the same demonstrations were generated from a HSMM with 10 states, a GMM with 10 states and a GMM with 49 states, further denoted as HSMM10, GMM10 and GMM49 respectively. The trajectories were executed as fine trajectories in the CtF pipeline using the same impedance controller as our approach. The gear, key and screw task was performed with all LfD methods. The power plug task did not succeed with any of the offline trajectories because the high force requirement prevented a full power plug insertion. Therefore, the power plug task was only evaluated with our approach. The measurement series were performed with 20 different object poses for each LfD method and task. The results are shown in Table I. Our approach shows a substantially higher success rate than the offline trajectories. On both tasks our approach failed exactly one time. This highlights the generalization and reliability of our approach over multiple tasks. During the experiments, the trajectories from our approach showed that oHMM-PA is reacting to the environment and attempts multiple paths to complete a task, thus reaching

higher success rates. A trade-off for this advantage is that our approach may not always take the most direct path and thus can be slower than the offline trajectories to finish a task.

V. Conclusion

We developed an LfD approach based on an online HMM that is combined with a CtF pipeline and showed that complex tasks like a gear insertion can be imitated while maintaining task generalization. Several contributions like an online HMM path algorithm were presented. The online HMM path algorithm was a key part in achieving increased task complexity. Angle detection in the demonstration trajectories was used to determine the states of the HMM, a method that eradicates the need for manually changing hyper parameters like other imitation learning approaches require. The online HMM path algorithm is able to react to the environment and experiments showed that the algorithm was able to retry failed attempts and choose new paths when a pathway was blocked. All executions showed a stable behaviour. Finally, measurement series were conducted in which our approach showed success rates of up to 95%, a result that is clearly outperforming other comparable methods and close to satisfying industrial requirements.

There are several topics for future work. First, the online HMM path algorithm could be improved by testing more sophisticated path planning algorithms. Second, the robot control logic can be optimized to achieve smoother and faster trajectories. Next, a sophisticated approach on when and how to command external forces during execution is needed. In this work, this was only addressed with a simple external wrench in the power plug insertion task. Extending on that, a dynamic stiffness model like in [18] can provide more task generalized forceful interaction. Moreover, offline trajectories from models with higher amount of states should be further researched as they seem promising to achieve high success rates in an offline manor. Finally, the influence of teleoperation on the imitation learning approach should be investigated. In particular, the effect of communication delays on the demonstrations and the use of virtual fixtures [23] to ease the recording of demonstrations by the operator.

References

[1] P. Corke, *Robotics, Vision and Control - Fundamental Algorithms in MATLAB*, ser. Springer Tracts in Advanced Robotics. Springer, 2011, vol. 73. [Online]. Available: http://dblp.uni-trier.de/db/series/star/index.html#Corke11

[2] A. Loganathan and N. S. Ahmad, "A systematic review on recent advances in autonomous mobile robot navigation," *Engineering Science and Technology, an International Journal*, vol. 40, p. 101343, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2215098623000204

[3] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg, "Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics," *arXiv preprint arXiv:1703.09312*, 2017.

[4] P. R. Florence, L. Manuelli, and R. Tedrake, "Dense object nets: Learning dense visual object descriptors by and for robotic manipulation," *arXiv preprint arXiv:1806.08756*, 2018.

[5] H. Ha and S. Song, "Flingbot: The unreasonable effectiveness of dynamic manipulation for cloth unfolding," in *Conference on Robot Learning*. PMLR, 2022, pp. 24–33.

[6] M. A. Lee, Y. Zhu, K. Srinivasan, P. Shah, S. Savarese, L. Fei-Fei, A. Garg, and J. Bohg, "Making sense of vision and touch: Self-supervised learning of multimodal representations for contact-rich tasks," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8943–8950.

[7] P. Sundaresan, S. Belkhale, and D. Sadigh, "Learning visuo-haptic skewering strategies for robot-assisted feeding," in *6th Annual Conference on Robot Learning*, 2022.

[8] E. Johns, "Coarse-to-fine imitation learning: Robot manipulation from a single demonstration," *CoRR*, vol. abs/2105.06411, 2021. [Online]. Available: https://arxiv.org/abs/2105.06411

[9] D. F. Prado, P. Ramachandrareddy, and E. Steinbach, "Demonstration quality-based teleoperated learning with visual and haptic data in bandwidth-limited environments," in *2024 IEEE 22nd Mediterranean Electrotechnical Conference - IEEE MELECON 2024*, Jun 2024.

[10] S. Calinon, F. Guenter, and A. Billard, "Goal-directed imitation in a humanoid robot," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, 2005, pp. 299–304.

[11] T. Asfour, F. Gyarfas, P. Azad, and R. Dillmann, "Imitation learning of dual-arm manipulation tasks in humanoid robots," in *2006 6th IEEE-RAS International Conference on Humanoid Robots*, 2006, pp. 40–47.

[12] S. Kumra and F. Sahin, "Collaborative robot learning from demonstrations using hidden markov model state distribution," *CoRR*, vol. abs/1809.10797, 2018. [Online]. Available: http://arxiv.org/abs/1809.10797

[13] A. K. Tanwani, J. Lee, B. Thananjeyan, M. Laskey, S. Krishnan, R. Fox, K. Goldberg, and S. Calinon, "Generalizing robot imitation learning with invariant hidden semi-markov models," *CoRR*, vol. abs/1811.07489, 2018. [Online]. Available: http://arxiv.org/abs/1811.07489

[14] M. Lauri, D. Hsu, and J. Pajarinen, "Partially observable markov decision processes in robotics: A survey," *IEEE Transactions on Robotics*, vol. 39, no. 1, p. 21–40, Feb. 2023. [Online]. Available: http://dx.doi.org/10.1109/TRO.2022.3200138

[15] M. Horowitz and J. Burdick, "Interactive non-prehensile manipulation for grasping via pomdps," in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 3257–3264.

[16] N. P. Garg, D. Hsu, and W. S. Lee, "Learning to grasp under uncertainty using pomdps," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 2751–2757.

[17] D. Silver and J. Veness, "Monte-carlo planning in large pomdps," in *Advances in Neural Information Processing Systems*, J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, Eds., vol. 23. Curran Associates, Inc., 2010. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2010/file/edfbe1afcf9246bb0d40eb4d8027d90f-Paper.pdf

[18] A. T. Le, M. Guo, N. v. Duijkeren, L. Rozo, R. Krug, A. G. Kupcsik, and M. Bürger, "Learning forceful manipulation skills from multi-modal human demonstrations," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 7770–7777.

[19] S. Calinon, "Mixture models for the analysis, edition, and synthesis of continuous time series," in *Mixture Models and Applications*, N. Bouguila and W. Fan, Eds. Springer, Cham, 2019, pp. 39–57.

[20] S. Calinon and D. Lee, "Learning control," in *Humanoid Robotics: a Reference*, P. Vadakkepat and A. Goswami, Eds. Springer, 2019, pp. 1261–1312, (in press).

[21] P. Song, Y. Yu, and X. Zhang, "Impedance control of robots: An overview," in *2017 2nd International Conference on Cybernetics, Robotics and Control (CRC)*, 2017, pp. 51–55.

[22] M. Mayr and J. M. Salt-Ducaju, "A c++ implementation of a cartesian impedance controller for robotic manipulators," *Journal of Open Source Software*, vol. 9, no. 93, p. 5194, 2024. [Online]. Available: https://doi.org/10.21105/joss.05194

[23] D. F. Prado, K. Larintzakis, J. Irsperger, and E. Steinbach, "Accelerating virtual fixture estimation for robot manipulation using reinforcement learning and human demonstrations," in *2024 IEEE 20th International Conference on Automation Science and Engineering (CASE)*, 2024, pp. 2013–2018.