2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Madrid, Spain, October 1-5, 2018

Multi-modal Robot Apprenticeship: Imitation Learning using Linearly Decayed DMP+ in a Human-Robot Dialogue System

Yan Wu[1], Ruohan Wang[2], Luis F. D'Haro[3], Rafael E. Banchs[4] and Keng Peng Tee[1]

***Abstract*— Robot learning by demonstration gives robots the ability to learn tasks which they have not been programmed to do before. The paradigm allows robots to work in a greater range of real-world applications in our daily life. However, this paradigm has traditionally been applied to learn tasks from a single demonstration modality. This restricts the approach to be scaled to learn and execute a series of tasks in a real-life environment. In this paper, we propose a multi-modal learning approach using DMP+ with linear decay integrated in a dialogue system with speech and ontology for the robot to learn seamlessly through natural interaction modalities (like an apprentice) while learning or re-learning is done on the fly to allow partial updates to a learned task to reduce potential user fatigue and operational downtime in teaching. The performance of new DMP+ with linear decay system is statistically benchmarked against state-of-the-art DMP implementations. A gluing demonstration is also conducted to show how the system provides seamless learning of multiple tasks in a flexible manufacturing set-up.**

I. Introduction

Typically, robots in daily operations are programmed and controlled through either a teach pendant or a programming environment which is both time-consuming and requiring dedicated robotics engineers. One challenge for robots to be deployed deeper in real-world applications is the ability to learn like an apprentice [1]. Robot learning by demonstration (RLD) has been proposed and studied in recent years to address this issue [2], [3], [4]. However, RLD has mostly been implemented to learn tasks from a single demonstration modality, such as kinaesthetics [2] and video recording [3]. Such paradigms limit the range of real-world scenarios robots can be applied to, because human intervention is still required to programme the robot to transition among learning and executing a growing repertoire of learned tasks.

In this work, we propose a multi-modal RLD framework to learn, perform and adapt tasks through multiple interaction modalities. In order to cope with the growing repertoire of tasks, the framework employs an ontological representation for the knowledge learned while the capturing and validation of knowledge is managed by a dialogue system interacting through both speech and kinaesthetics. As many tasks are variants of one another (for instance, a new task is a combination of another two, a new task is a partial change to the original and so on), a task representation that can be flexibly adapted on the fly will make the interaction more natural, reduce user fatigue in demonstration as well as increase learning throughput. For this purpose, we introduce the Dynamic Movement Primitives Plus (DMP+) [4] with linear decay instead of the originally proposed exponential decay to allow online easy modifications of a learned task while keeping the learning outcome an accurate representation of the demonstration.

We experimentally validate the model using the UJI Pen Characters dataset to perform insertion of an additional character into an existing model of another character and statistically compare the results against state-of-the-art implementation of DMPs. We also conduct a real-world demonstration to showcase the merits of the full multi-modal learning system in performing a flexible manufacturing task of gluing pieces of customised products. The rest of this paper is organised as follows: We present the relevant literature in Section II followed by the overall architecture of the framework in Section III. The detailed implementations of each module are described in Section IV. We present the experimental results and a gluing demo in Section V and finally the conclusions and future work in Section VI.

II. Related Work

Traditionally, robot learning by demonstration (RLD) focuses on using a single modality to encode a particular task [2]. Limited literatures have discussed on a multi-modal approach in RLD. Multi-modality in robot learning has been proposed in terms of modelling stochastic policy [5] and reward function [6], however, limited literature has addressed the topic of multiple input modalities. As presented in [7], language plays a coordinating role in real-time learning, integrating the "show" and "tell" modalities forms a crucial part in RLD.

A. Dynamic Movement Primitives Plus

Dynamic Movement Primitives (DMPs) are a biologically inspired formulation of the human action behaviours [8]. It consists of a simple goal driven task governed by a PD controller and local attractors along the task represented by a group of forcing terms. It combines non-linear dynamical systems with statistical machine learning which give rise to many desirable outcomes such as ability to generalise, guaranteed convergence and to easily couple with other dynamical system [9].

[1]Yan Wu and Keng Peng Tee are with the A*STAR Institute for● Infocomm Research, Singapore {wuy, kptee}@i2r.a-star.edu.sg

[2]Ruohan Wang is with Imperial College London, United Kingdom r.wang16@imperial.ac.uk

[3]Luis F. D'Haro is with Universidad Politécnica de Madrid, Spain lfdharo@die.upm.es

[4]Rafael E. Banchs is with Nanyang Technological University, Singapore rbanchs@ntu.edu.sg

Although DMPs have been demonstrated on a wide range of applications [9], [10], [11], limited literature discusses the issues of model accuracy [4], [12], [13] and modifications of DMPs [14], [15], [16], [17]. Both qualities are of paramount importance in industrial applications [12], especially in flexible manufacturing where productivity improvement and operational downtime saving are critical for these high-mix low-volume production jobs.

In [4], the authors proposed to insert a bias term into the original DMP implementation to allow the overall system to learn a given task accurately. In the original DMP implementation, exponential decay system is chosen to ensure convergence towards the goal point [18], [19], [20]. However, in such systems, the duration of the task is limited by the numerical resolution of the computational platform which makes it difficult to modify primitives, such as joining and inserting any new primitive into an existing task. In the DMP+ formulation [4], the convergence mechanism has been provided using truncated kernels. This frees up the design of the canonical system to address the need of a system that can be joined by another one without having to worry about the vanishing exponential time decay problem. Thus, a linear decay system could be used in place of the exponential one to allow a DMP+ system to achieve arbitrary update to existing model on the fly.

B. Multi-modal Dialogue System

On the other hand, the incorporation of a multimodal dialogue system allows a more natural communication between an operator and the robot to setup, teach, correct or execute repetitive tasks. Thanks to recent improvements in robustness, accuracy, flexibility, and modularity in different technologies like speech and object recognition [21] [22], natural language understanding and dialogue [23] [24], grounding [25], integration of ontologies or knowledge bases [26], it has been possible to include these capabilities into many different kind of robots and applications [27] [28]. This way, by means of using natural language commands and a conversational style of interaction, an operator can request a robot to start a given task (direct commands), set the values of the different parameters needed for the task (e.g. speed of execution, use of a specific tool, tolerance margin, number of units to produce, etc.), then ask for recommendations for some parameters for which the operator may not be sure, followed by checking the validity of the final configuration (use of expert knowledge), and finally receiving feedback about the execution of the task while the operator could be performing other duties meanwhile.

III. System Architecture

Fig. 1 shows the overall organisation of our proposed architecture. Since this platform can be used for different applications, and can be integrated with other existing ROS components, some of our modules can be replaced for others or become optional depending on the specifications of the tasks to perform. In this section, we will provide a short

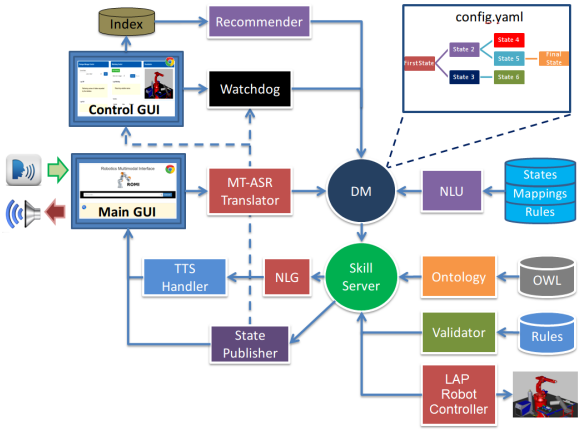


Fig. 1. System architecture of the multi-modal dialogue system

description of each module. For more detailed information please refer to [29].

Graphical User Interface and Control (GUI): The GUI is the main communication window with the robot operator allowing to collect or display input/output information by means of clickable buttons, a microphone, speakers, input forms for text, webcam video, graphs or animations. This way, the operator can load a previous project configuration, start or stop actions, or observe the execution of a task remotely. The control interface is mainly intended for administration and debugging purposes allowing fine tuning control of different configuration parameters for each module in the platform and checking their respective logs output. In both interfaces, we use the ROSbridge library and AJAX messages to connect the GUI to the ROS framework for low level access of the different modules.

Automatic Speech Recogniser (ASR) and Machine Translation (MT): The goal of the ASR is to transcribe the operator's speech when interacting with the robot in spite of noise or variations in accents and languages; our implementation is focused on English but we can handle different languages like Malay, Chinese, or Tamil by internally translating the transcribed sentences into English.

On the other hand, in order to quickly correct typical transcription errors and reduce the problem of recognizing out-of-vocabulary (OOV) words, in [30] we proposed the use of an MT system trained using words and phonetic encoding representations on the n-best lists of the ASR results. This approach improves the ASR performance by reducing the word error rate (WER), the correct transcription of domain specific words, and increasing the matching of the corrected transcriptions to the sentences that the NLU is trained to parse.

Natural Language Understanding (NLU) and Dialogue Management (DM): The NLU provides an interpretation of a given sentence or command (e.g. typed in a text box in the GUI or transcribed by the ASR) in a form that can be used by the different modules. Our platform allows using a rule-based approach for tasks with a closed set of grammar constructions, or a statistical approach for applications with

more variability and availability of training data. In addition, the module performs different pre-processing strategies to deal with numbers, abbreviations, aliases, and spelling corrections which are very frequent in most industrial tasks, and noisy or multilingual environments.

The DM is one of the most complex modules in the architecture since it asynchronously handles different sources of information (e.g. the NLU interpretation, direct inputs from the interface, the interchange of information between modules (skills server, watchdog, NLG), access to knowledge bases, confirmations, corrections, presentation of errors, etc.) where all the processing must be done in a prioritised sequence of actions in order to complete the task or to perform more important orders first. The implementation of our gluing task (see Section V-B follows a multi-slot frame-based approach defined in a finite state-machine flow although the platform allows single-slot filling approach or a multi-goal approach [31].

Skill Server, Ontology and State Publisher: the skill server acts as a mediator between the low level primitives of the robot, the validator, and the ontology allowing the separation of the specific details of the robot implementation from the high level API calls used by the DM and other modules in the platform.

The ontology provides the formal representation of the properties, types, and relationships between the different domain entities in the task. This codification of the expert knowledge allows the robot to provide recommendations, validation of parameters and execution of the task. Our implementation uses SESAME for storing, Protègè for specification, and Tomcat for access through a RESTful interface.

The state publisher is a ROS publisher that publishes all the parameters from the ontology and broadcast them every second as a topic allowing any module to be subscribed to the topic and get access to the task information without depending only on having access through the DM. This feature is important specially for some tasks where a faster response is needed.

Validation, Recommendation and Watchdog: These three modules respectively allow: a) checking that the parameters provided by the user are correct and that the robot owns what is required to perform the task (e.g. supplies material, tools, etc.), b) to provide default or recommended parameter values for specific tasks or setup configuration, and c) to perform a periodical scanning or checking of the "robot health" in order to guarantee that it can execute the tasks required by the operator and automatically stopping the robot in case a safety break is detected.

Natural Language Generator (NLG) and Text-to-Speech (TTS): The NLG converts the high-level concept outputs from different modules into syntactic structures and words that the TTS can convert into an audible signal to be played to the operator. Our NLG is based on using prompt-templates that can generate new messages by modifying different parameters like tense, mode, subject, object, verb, etc. in the sentence.

LAP Robot Controller: Finally, the physical interaction and task learning component is implemented in this module. We adopt the Learn-Adapt-Production approach proposed in [12] with a Linearly Decayed DMP+ as the underlying learning-by-demonstration framework which will be detailed in the next section.

IV. Linearly Decayed DMP+

In [4], we demonstrated that DMP+ outperforms the original DMPs in trajectory accuracy. Extending DMP+, we propose an enhanced version which allows the system to perform effective trajectory joining and insertion by removing the problem of vanishing exponential time decay.

A. Modification to DMP+

The DMP+ formalisation is characterised by a second order dynamical system describing trajectory dynamics and a first order canonical system to replace the explicit time dependence as follows [4].

$$\tau^2 \ddot{y} = \alpha_z(\beta_z(g-y) - \tau\dot{y}) + f \tag{1}$$

$$f = \frac{\sum_{i=1}^{N} (w_i x + b_i)\psi_i}{\sum_{i=1}^{N} \psi_i + \varepsilon} \tag{2}$$

$$\psi_i = \begin{cases} \exp(-\frac{h_i}{2}(x-c_i)^2), & \text{if } -\theta_i \le x - c_i \le \theta_i \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

$$\tau\dot{x} = -\alpha_x x \tag{4}$$

where g is the goal state, τ the temporal scaling factor, and $\alpha_z, \beta_z, \alpha_x$ positive constants. h_i is the width of each kernel ψ_i centred at c_i, while w_i and b_i the learnable weights and biases. DMP+ represents a globally stable system with a unique attractor $(\dot{y}, y) = (0, g)$ as the forcing term f vanishes. DMP+ differs from the original DMPs by introducing learnable local bias terms and providing an alternative convergence mechanism through truncated kernels. DMP+ is capable of modelling arbitrarily complex trajectories by using learnable forcing terms, parameterised by N kernels ψ_i.

x in (4) is the phase variable to denote time implicitly, with the closed-form solution with respect to time t,

$$x = e^{-\alpha_x * t} \tag{5}$$

for trajectories with a large t, x may exceed the numerical resolution of computational platforms, thus constraining the length of the trajectories to be learned. Care must be taken in selection of α_x to ensure the numerical stability and convergence of phase variable to 0 at the end of trajectories. In [32], it is shown that the selection of α_z changes goal convergence in trajectories significantly. As DMP+ uses truncated kernels for convergence, we are free to choose an alternative canonical system suited for our purposes. To improve the numerical stability, and effectively solve the problem of trajectory joining and insertion, we replace the exponential decay system with a linear decay system

$$\dot{x} = -\frac{1}{T} \tag{6}$$

whereby T is the length of the trajectory for learning.

B. Primitives Learning with DMP+

The number of kernels N is determined empirically, such that the mean square error (MSE) of positional deviation is less than a constant threshold ϕ [4]. The exact value for ϕ depends on specific application requirements.

$$MSE = \frac{1}{P}\sum_{j=1}^{P}(y_d - y_{learned})^2 < \phi \tag{7}$$

Given N, kernels are placed evenly across the phase variable x as follows,

$$\begin{aligned} c_i &= \frac{i-1}{N-1}, i = 1,...,N \\ h_i &= (p(N-1))^2, i = 1,...,N \\ \theta_i &= \frac{q}{\sqrt{h_i}}, p < q \end{aligned} \tag{8}$$

where p and q are positive constants. (8) limits each kernel to only model samples within q standard deviations from the center c_i.

Several methods exist for solving w_i and b_i, including locally weighted regression [33], globally weighted regression [34], and gradient descent with path integral approach [32]. We choose locally weighted regression over global regression as our testing revealed that global regression tend to overfit and produces more kinks in acceleration profile [4].

C. Joining of Movement Primitives

A simple way for joining primitives is to perform one DMP+ until it reaches its goal and then start off the next one at that point. However, the simple approach suffers from a undesirable discontinuity in acceleration [34] and trade-off between goal convergence and modelling accuracy [32]. We propose a method that joins two DMP+ representations A and B, such that B starts at the goal of A, and transition smoothly from A to B.

We first define two subroutines in Algorithm 1 and 2, which scales the range of the phase variable in a DMP+ representation while leaving the trajectory unchanged. The linear decay system makes such scaling straightforward. In all algorithms, we assume all DMP+ phase variables have range $[0,1]$.

Algorithm 1 *ScaleLeft*

Input: DMP+ representation $D = \{w_i, c_i, b_i, h_i | i = 1,...,N\}$, scale factor r
Output: scaled DMP+ representation D'

$w'_i = w_i / r$
$b_i = b_i + (r-1)w'_i$
$c'_i = 1 - (1-c_i)r$
$h'_i = h_i / r^2$
return $D' = \{w'_i, c'_i, b'_i, h'_i | i = 1,...,N\}$

ScaleLeft maps a trajectory to range $[r,1]$ while *ScaleRight* maps a trajectory to $[0,r]$.

Algorithm 2 *ScaleRight*

Input: DMP+ representation $D = \{w_i, c_i, b_i, h_i | i = 1,...,N\}$, scale factor r
Output: scaled DMP+ representation D'

$w'_i = w_i / r$
$b_i = b'_i$
$c'_i = rc_i$
$h'_i = h_i / r^2$
return $D' = \{w'_i, c'_i, b'_i, h'_i | i = 1,...,N\}$

The primitive joining algorithm is presented in Algorithm 3, to join trajectory D_2 after D_1, such that $goal_1$, the goal of D_1, is used as the starting position of D_2.

Algorithm 3 *JoinTrajectory*

Input: $D_1 = \{w_i^1, c_i^1, b_i^1, h_i^1 | i = 1,...,N_1\}, D_2 = \{w_i^2, c_i^2, b_i^2, h_i^2 | i = 1,...,N_2\}$, T_1, T_2
Output: joined DMP+ representation D'

$r1 = T1/(T1+T2)$;
$r2 = 1 - r1$
$D1$=ScaleLeft($D1, r1$); $D2$=ScaleRight($D2, r2$);
$\delta = goal_2 - s_2$
Find kernels to update:
$i_{min} = \arg\min_i r - c_i \leq \theta_i, i = 1,...,N$
$i_{max} = \arg\max_i c_i - r \leq \theta_i, i = 1,...,N$
Sample generation:
$x_{max} = c_{i_{max}} + \theta_{i_{max}}$; $x_{min} = c_{i_{min}} - \theta_{i_{min}}$
Generate sample points $\{y(x) | x_{min} <= x <= r\}$ from $D2$ and $\{y(x) + \alpha_z \beta_z \delta | r < t <= x_{max}\}$ from $D1$
Update the kernels:
Update the kernels $\{w_i^1, b_i^1 | i_{max} <= i < N_1\}$ and $\{w_i^2, b_i^2 | 1 <= i <= i_{min}\}$ with sample points generated
return $D' = \{D_1, D_2\} - \{w_{N_1}^1, c_{N_1}^1, b_{N_1}^1, h_{N_1}^1\}$

In Algorithm 3, s_2 denotes the starting position of D2, and T_1, T_2 the lengths of D_1, D_2 respectively. The joined trajectory D' has a shifted goal position $goal_2 + goal_1 - s_2$. The canonical system of D' is updated to

$$\dot{x} = -\frac{1}{T_1 + T_2} \tag{9}$$

We note that the last kernel of D_1 is discarded as it coincides with the first kernel of D_2.

In short, the algorithm first maps D_1 and D_2 onto the appropriate ranges of the phase variable using the scaling factor r. It then updates the kernels affected by the joining operation, namely those at the end of D_1 and at the start of D_2. The update ensures a smooth transition from D_1 to D_2. Lastly, we observe that $c_{i+1} - c_i = \frac{1}{N-1}$, therefore the number of kernels requiring update is a constant $\lfloor 2(N-1)\theta_i \rfloor = \left\lfloor \frac{2q}{p} \right\rfloor$, which results in an effective joining operation.

D. Flexible Trajectory Insertion

In [4], we presented an algorithm to modify a learned trajectory based on new partial demonstrations. We extend

the previous technique to achieve trajectory substitution where the inserted segment may be of arbitrary length in comparison with the original one. The method alleviates our previous limitation that the replacement trajectory must be approximately the same length as the replaced segment.

We only require the start and end points of the replacement trajectory to match approximately the start and end points of the replaced trajectory segment to avoid discontinuity.

The insertion algorithm is presented in Algorithm 4.

Algorithm 4 *Modify Trajectory*

Input: DMP+ representation $D = \{w_i^1, c_i^1, b_i^1, h_i^1 | i = 1,...,N_1\}$, x_1, x_2, Trajectory modification M, T_D, T_M
Output: updated DMP+ representation D'

$T' = (1 + x_2 - x_1)T_D + T_M$
$r = T1/T'$;
Find kernels to be updated:
$i_{min} = \arg\min_i x_2 - c_i \leq \theta_i, i = 1,...,N$
$i_{max} = \arg\max_i c_i - x_1 \leq \theta_i, i = 1,...,N$
Find the range of trajectory needed for kernel update:
$x_{max} = c_{i_{max}} + \theta_{i_{max}}$; $x_{min} = c_{i_{min}} - \theta_{i_{min}}$
Generate sample points $S_{next} = \{y(x) | x_{min} <= x <= x_2\}$ and $S_{prev} = \{y(x) | x_1 <= x <= x_{max}\}$
Learn new kernels
create N_m new kernels D_M evenly placed from $r * c_{i_{min}}$ to $1 - r * (1 - c_{I_{max}})$
Learn D_M with sample points S_{prev}, M, S_{next}
$D_{next} = ScaleRight(D_{1..i_{min}-1}, r)$;
$D_{prev} = ScaleLeft(D_{i_{max}+1..N}, r)$;
return $D' = \{D_{prev}, D_M, D_{next}\}$

Here, x_1 and x_2 denote the implicit start and end of the replaced segment, while T_D and T_M denote the length of the original and replacement trajectories respectively. Again, only a constant number of existing kernels require re-learning. The N_m new kernels are used to model the replacement trajectory. We observe that Algorithms 3 and 4 share very similar strategies: 1) mapping the existing kernels to the appropriate ranges of the phase variable; 2) re-learning kernel parameters; and 3) combining the kernels to yield the new representation.

V. Experimental Evaluation

A. Statistical Analysis

To benchmark on model accuracy, we compare the performance of DMP+ with linear decay against DMP+ with exponential decay [4] and the original DMP (with 2 times the number of kernels used by both DMP+s) in modelling trajectories of single-stroke characters from the UJI Pen dataset[1]. Each character is scaled to within a $1cm^2$ and we set the kernel count to $N = 10$ for DMP+. Fig. 2 summarises the MSE in positional deviation achieved by the three formulations. It shows that both DMP+ formulations achieve almost identical trajectory accuracy and outperforms the original DMPs with twice number of kernels.

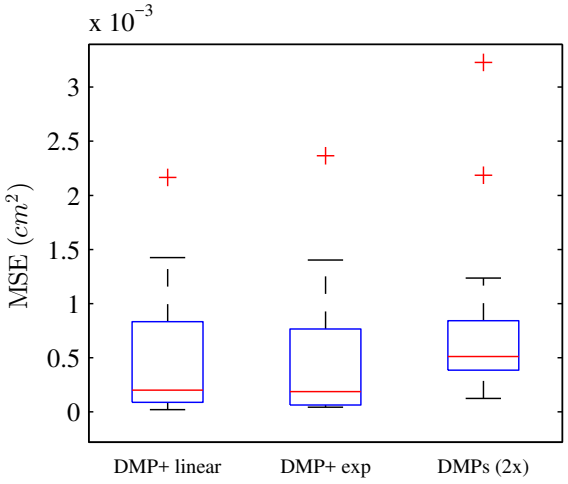


Fig. 2. Box plot of MSEs achieved by DMP+ (linear), DMP+ (exponential) and original DMPs with twice as many kernels. DMP+ still outperforms DMPs with a small margin.

Furthermore, we test the proposed trajectory joining algorithm on the UJI dataset. In this experiment, two characters are joined with the goal of the first character set as the starting point of the second character. To test the algorithm's modelling accuracy, we also learn the same character pair as a single trajectory with DMP+ using equal number of kernels. Fig. 3 shows a snapshot of the character pair *ab* joined using the joining algorithm and DMP+ batch learning.

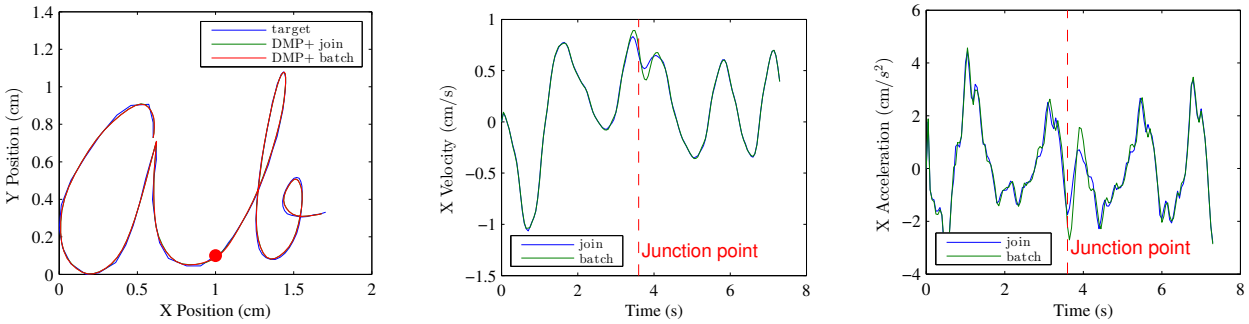
The results illustrate that the joining algorithm effectively and accurately joins two trajectories together, with modelling accuracy comparable to batch learning. The junction point is also correctly modelled by the joining algorithm. Fig. 3b shows that velocity profile is smooth for each method with a slight variation at the junction point. Fig. 3c also shows that both methods create continuous acceleration profiles. We attribute the kinks in the acceleration to noise from the numerical differentiation used in trajectory learning, as the dataset does not provide instantaneous velocity and acceleration at each sample.

B. Gluing Demo

The overall multi-modal learning by demonstration framework is implemented in ROS on the KUKA LBR iiwa robot[2]. This prototype system is used to perform some gluing tasks with a mock-up glue head for simplicity. The experimental setup is shown in Fig. 4. The system learns the outlines of two toy figurines shown in Fig. 5 through speech dialogue and kinaesthetic teaching. It first learns the outline of one figurine, and uses the proposed insertion algorithm to adapt the learned task with a partial demonstration of the head segment of the other figurine. The kinaesthetic demonstration is sampled at $50Hz$.

[1]dataset available at `https://archive.ics.uci.edu/ml/datasets/UJI+Pen+Characters`

[2]Video can be viewed at https://youtu.be/jK3LgbcNmGU



(a) (b) (c)

Fig. 3. a: Trajectory for character *a*, reproduction by DMP+ joining algorithm and batch learning. b: Learned velocity in X dimension for DMP+ joining and batch learning. c: Learned acceleration in X dimension for DMP+ joining and batch learning

For the dialogue part, we defined a state machine with 13 states that allows getting 3 different process parameters (i.e. speed, type of gluing: continuous or point, and the part to glue), execute the gluing task, start and stop the teaching process, load a pre-defined model, and repeat or finish the task; we also defined 42 rules for the NLU grammar in order to parse salutations, confirmations, starting or stopping the teaching mode, to reconfigure and re-start the execution, to stop the robot, switch among tasks and figurines, ask for recommendations, and set the parameters using different grammatical constructions (e.g. one or several parameters at once, using abbreviations). Lastly, we trained our MT system by asking 7 people from different nationalities to repeat at least 3 times the 40 most common sentences used in the demo and storing the top 10 ASR n-best results for each, accounting a total of 16K parallel sentences. Check [29] and [35] for more details about the design and evaluation.

For the DMP+ with linear decay implementation, we set $\alpha_z = 25$, $\beta_z = \alpha_z/4$ according to [9]. ε is an arbitrarily small value of 10^{-8}. We set $p = 2$ and $q = 3$ such that each kernel only omits samples more than 3 standard deviations away from the centre. For all experiments, we use Euler integration to calculate trajectory dynamics with the step size corresponding to the sampling frequency of demonstrations.

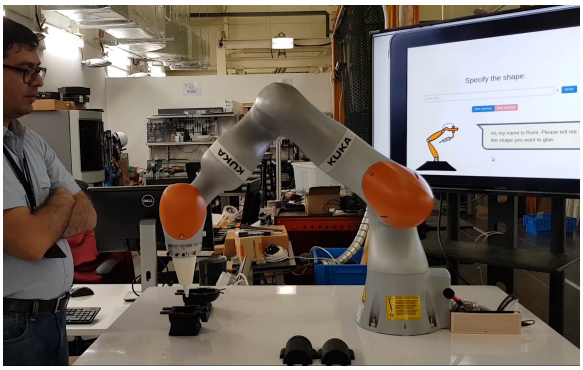


Fig. 4. The experiment setup to simulate gluing task on KUKA LBR-iiwa.

Fig. 6 shows that the update algorithm achieves comparable MSE against DMP+ batch learning. We note that the modification segment is almost twice as long in duration as the segment to be replaced. The update algorithm successfully alleviates the trajectory update limitation of DMP+ with exponential decay.

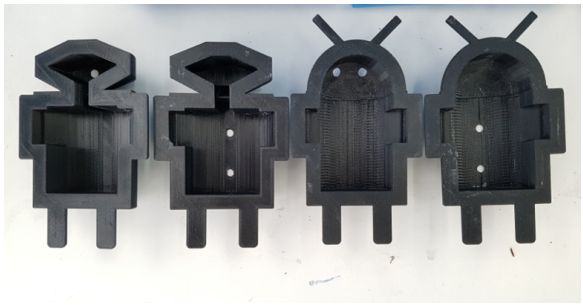


Fig. 5. Two figurines with different head structures. Task modification is done to learn only the difference (i.e. the trajectory of the head).

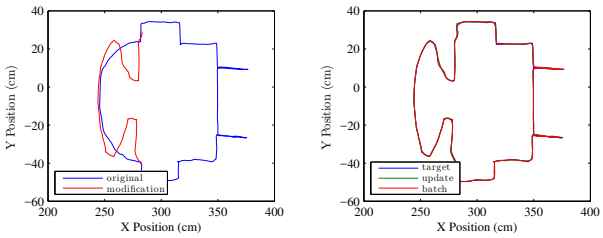


Fig. 6. Left: The captured adaptation to the original demonstration. Right: The 2D plots of the demonstrated task adaptation, the learning outcome using DMP+ with update algorithm and the outcome using batch learning.

The experiment results show that DMP+ with a linear decay system achieves 1) comparable performance with the original DMP+ formulation and 2) effective trajectory joining and insertion on-the-fly. The linear decay system allows us to effectively scale trajectories to desired ranges of the phase variable and perform joining and insertion afterwards. We further exploit the use of truncated kernels in our model to allow alternative design choice of the canonical system for the phase variable.

VI. Conclusion

We proposed a multi-modal robot apprenticeship system to learn multiple tasks from demonstration through natural interaction modalities. This system made use of a linearly decayed DMP+ model integrated in a dialogue system with speech and ontology. The new DMP+ model achieved comparable trajectory accuracy against DMP+ and significantly outperformed the original DMPs formulation while allowing partial reuse of previously learned task. This reduced user fatigue and operational downtime due to teaching redundancy. A gluing demonstration was implemented using this system to showcase how the KUKA robot was able to learn multiple tasks seamlessly.

References

[1] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004, p. 1.

[2] B. D. 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.

[3] Y. Wu, Y. Su, and Y. Demiris, "A morphable template framework for robot learning by demonstration: Integrating one-shot and incremental learning approaches," *Robotics and Autonomous Systems*, vol. 62, no. 10, pp. 1517–1530, 2014.

[4] R. Wang, Y. Wu, W. L. Chan, and K. P. Tee, "Dynamic movement primitives plus: for enhanced reproduction quality and efficient trajectory modification using truncated kernels and local biases," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2016)*. IEEE, 2016.

[5] K. Hausman, Y. Chebotar, S. Schaal, G. Sukhatme, and J. J. Lim, "Multi-modal imitation learning from unstructured demonstrations using generative adversarial nets," in *Advances in Neural Information Processing Systems*, 2017, pp. 1235–1245.

[6] M. González-Fierro, C. Balaguer, N. Swann, and T. Nanayakkara, "A humanoid robot standing up through learning from demonstration using a multimodal reward function," in *2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2013, pp. 74–79.

[7] M. Petit, S. Lallee, J. D. Boucher, G. Pointeau, P. Cheminade, D. Ognibene, E. Chinellato, U. Pattacini, I. Gori, U. Martinez-Hernandez, H. Barron-Gonzalez, M. Inderbitzin, A. Luvizotto, V. Vouloutsi, Y. Demiris, G. Metta, and P. F. Dominey, "The coordinating role of language in real-time multimodal learning of cooperative tasks," *IEEE Transactions on Autonomous Mental Development*, vol. 5, no. 1, pp. 3–17, March 2013.

[8] S. Schaal, "Dynamic movement primitives-a framework for motor control in humans and humanoid robotics," in *Adaptive Motion of Animals and Machines*. Springer, 2006, pp. 261–280.

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

[10] J. Peters and S. Schaal, "Reinforcement learning of motor skills with policy gradients," *Neural networks*, vol. 21, no. 4, pp. 682–697, 2008.

[11] J. Nakanishi, J. Morimoto, G. Endo, G. Cheng, S. Schaal, and M. Kawato, "Learning from demonstration and adaptation of biped locomotion," *Robotics and Autonomous Systems*, vol. 47, no. 2, pp. 79–91, 2004.

[12] W. Ko, Y. Wu, K. Tee, and J. Buchli, "Towards industrial robot learning from demonstration," in *3rd International Conference on Human-Agent Interaction (HAI)*, 2015.

[13] Y. Schroecker, H. Ben Amor, and A. Thomaz, "Directing policy search with interactively taught via-points," in *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, ser. AAMAS '16. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2016, pp. 1052–1059.

[14] T. Kulvicius, K. Ning, M. Tamosiunaite, and F. Worgotter, "Joining movement sequences: Modified dynamic movement primitives for robotics applications exemplified on handwriting," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 145–157, 2012.

[15] T. Petric, A. Gams, L. Zlajpah, A. Ude, and J. Morimoto, "Online approach for altering robot behaviors based on human in the loop coaching gestures," in *IEEE International Conference on Robotics and Automation (ICRA), 2014*. IEEE, 2014, pp. 4770–4776.

[16] A. G. Billard, S. Calinon, and F. Guenter, "Discriminative and adaptive imitation in uni-manual and bi-manual tasks," *Robotics and Autonomous Systems*, vol. 54, no. 5, pp. 370–384, 2006.

[17] T. Inamura, I. Toshima, H. Tanie, and Y. Nakamura, "Embodied symbol emergence based on mimesis theory," *The International Journal of Robotics Research*, vol. 23, no. 4-5, pp. 363–377, 2004.

[18] H. Hoffmann, P. Pastor, D.-H. Park, and S. Schaal, "Biologically-inspired dynamical systems for movement generation: automatic real-time goal adaptation and obstacle avoidance," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, 2009, pp. 2587–2592.

[19] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal, "Learning and generalization of motor skills by learning from demonstration," in *IEEE International Conference on Robotics and Automation (ICRA), 2009*. IEEE, 2009, pp. 763–768.

[20] A. Ude, A. Gams, T. Asfour, and J. Morimoto, "Task-specific generalization of discrete and periodic dynamic movement primitives," *IEEE Transactions on Robotics*, vol. 26, no. 5, pp. 800–815, 2010.

[21] W. Xiong, J. Droppo, X. Huang, F. Seide, M. Seltzer, A. Stolcke, D. Yu, and G. Zweig, "Achieving human parity in conversational speech recognition," *arXiv preprint arXiv:1610.05256*, 2016.

[22] R. Girshick, I. Radosavovic, G. Gkioxari, P. Dollár, and K. He, "Detectron," https://github.com/facebookresearch/detectron, 2018.

[23] R. Sarikaya, G. E. Hinton, and A. Deoras, "Application of deep belief networks for natural language understanding," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 4, pp. 778–784, 2014.

[24] H. Chen, X. Liu, D. Yin, and J. Tang, "A survey on dialogue systems: Recent advances and new frontiers," *arXiv preprint arXiv:1711.01731*, 2017.

[25] R. Paul, J. Arkin, N. Roy, and T. M. Howard, "Efficient grounding of abstract spatial concepts for natural language interaction with robot manipulators." in *Robotics: Science and Systems*, 2016.

[26] K. Toutanova, D. Chen, P. Pantel, H. Poon, P. Choudhury, and M. Gamon, "Representing text for joint embedding of text and knowledge bases," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, pp. 1499–1509.

[27] H. Cuayáhuitl, "Robot learning from verbal interaction: a brief survey," *Proceedings of the New Frontiers in Human-Robot Interaction*, 2015.

[28] I. Maurtua, I. Fernandez, J. Kildal, L. Susperregi, A. Tellaeche, and A. Ibarguren, "Enhancing safe human-robot collaboration through natural multimodal communication," in *Emerging Technologies and Factory Automation (ETFA), 2016 IEEE 21st International Conference on*. IEEE, 2016, pp. 1–8.

[29] L. F. D'Haro, A. I. Niculescu, C. Cai, S. Nair, R. E. Banchs, A. Knoll, and L. Haizhou, "An integrated framework for multimodal human-robot interaction," in *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference 2017 (APSIPA ASC)*, Dec 2017.

[30] L. F. D'Haro and R. E. Banchs, "Automatic correction of asr outputs by using machine translation." Interspeech, 2016, pp. 3469–3473.

[31] R. Jiang, R. E. Banchs, S. Kim, L. F. D'Haro, A. I. Niculescu, and K. H. Yeo, "Configuration of dialogue agent with multiple knowledge sources," in *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2015 Asia-Pacific*. IEEE, 2015, pp. 840–849.

[32] T. Kulvicius, K. Ning, M. Tamosiunaite, and F. Wörgötter, "Modified dynamic movement primitives for joining movement sequences," in *IEEE International Conference on Robotics and Automation (ICRA), 2011*, May 2011, pp. 2275–2280.

[33] C. G. Atkeson, A. W. Moore, and S. Schaal, "Locally weighted learning for control," in *Lazy learning*. Springer, 1997, pp. 75–113.

[34] B. Nemec, M. Tamošiūnaitė, F. Wörgötter, and A. Ude, "Task adaptation through exploration and action sequencing," in *2009 9th IEEE-RAS International Conference on Humanoid Robots*, Dec 2009, pp. 610–616.

[35] A. I. Niculescu, L. F. D'Haro, and R. E. Banchs, "When industrial robots become social: on the design and evaluation of a multimodal interface for welding robots," in *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference 2017 (APSIPA ASC)*, Dec 2017, pp. 83–89.