2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Daejeon Convention Center
October 9-14, 2016, Daejeon, Korea

Dynamic Movement Primitives Plus: for Enhanced Reproduction Quality and Efficient Trajectory Modification Using Truncated Kernels and Local Biases

Ruohan Wang, Yan Wu, Wei Liang Chan, Keng Peng Tee[1]

Abstract— **Dynamic Movement Primitives (DMPs) are a generic approach for trajectory modeling in an attractor landscape based on differential dynamical systems. DMPs guarantee stability and convergence properties of learned trajectories, and scale well to high dimensional data. In this paper, we propose DMP+, a modified formulation of DMPs which, while preserving the desirable properties of the original, 1) achieves lower mean square error (MSE) with equal number of kernels, and 2) allows learned trajectories to be efficiently modified by updating a subset of kernels. The ability to efficiently modify learned trajectories i) improves reusability of existing primitives, and ii) reduces user fatigue during imitation learning as errors during demonstration may be corrected later without requiring another complete demonstration. In addition, DMP+ may be used with existing DMP techniques for trajectory generalization and thus complements them. We compare the performance of our proposed approach against DMPs in learning trajectories of handwritten characters, and show that DMP+ achieves lower MSE in position deviation. We demonstrate in a second experiment that DMP+ can efficiently update a learned trajectory by updating only a subset of kernels. The update algorithm achieves modeling accuracy comparable to learning the adapted trajectory with the original DMPs.**

I. Introduction

Dynamic Movement Primitives (DMPs) are a generic framework for motor representation based on nonlinear dynamic systems [1]. DMPs are capable of modeling both discrete and rhythmic movements and have been successfully applied to a wide range of tasks such as biped locomotion [2], drumming and tennis swings [3]. Motivated by imitation learning in robotics [4], [5], DMPs are formulated as a stable PD controller modulated by a learnable forcing term to represent arbitrarily complex movements. The forcing term is approximated with the weighted sum of radial basis kernels. DMPs have many desirable properties, such as stability and convergence of the underlying dynamical systems, and robustness against environment perturbation.

Numerous techniques have been developed for DMPs to adapt to novel situations. For instance, obstacle avoidance is achieved using potential fields to directs movements away from obstacles [1], [6]. A similar technique is used in [7] to enable interactive adaptation of learned trajectory by human gestures. In [8]. adaptive frequency modulation has been proposed to learn and modify periodic movements. In another work, *Ude et al.* proposed movement generation by the weighted sum of existing movement primitives [9]. Adaptation techniques such as obstacle avoidance and frequency modulation require carefully tuned dynamical coupling terms, and do not address the problem of permanently modifying existing behaviors. The technique in [7], though permanently adapts learned trajectories, requires updating all kernel weights. Most works on DMPs do not qualify the quality of trajectory learning as simply increasing the number of kernels improves the level of modeling accuracy.

To address the problems of reproduction quality and efficient adaptation, we propose DMP+, a modified formulation of DMPs that 1) efficiently update learned trajectories by updating only a subset of kernel weights, and 2) achieves lower reproduction error by approximating local gradients of the forcing term significantly better. In addition, DMP+ preserves the desirable properties of DMPs, such as the stability and convergence, and may be used with existing techniques in DMPs for generalization and adaptation. Therefore, our approach may be used as a drop-in replacement for DMPs to improve reproduction quality.

Our approach is partly inspired by the Learning-Adaptation-Production (LAP) approach [10], a general framework for iteratively adapting learned behaviors through multiple user demonstrations. Compared to the physical and visual interaction approach presented in [7], [11], further demonstration permits maximum granularity of control during adaptation. Further, as DMP kernels are local, we were inspired by the intuition that any trajectory modification should only affects a subset of kernels. The ability to efficiently adapt learned movements reduces user fatigue in performing demonstration by 1) improving reusability of existing primitives, and 2) allowing demonstration errors to be corrected later without having to start over again. Our method may be applied to Low-Volume-High-Mix manufacturing, by replacing tedious robot programmig with intuitive imitation learning and by reducing production downtime with improved reusability of learned robot motions.

We experimentally evaluate the efficacy of DMP+ by comparing it with the state-of-the-art implmentation for DMPs. On a handwritten character trajectory dataset, we demonstrate lower MSE in position deviation achieved with our method. In another experiment, we show that DMP+ efficiently adapts a learned trajectory and achieves modeling accuracy comparable to learning the adapted trajectory with DMPs.

*This work is supported by Science and Engineering Research Council (SERC, A*STAR, Singapore) Grant No. 1225100001.

[1]Ruohan Wang, Yan Wu, Wei Liang Chan and Keng Peng Tee are with the A*STAR Institute of Infocomm Research, Singapore {wangrh, wuy, chanwl, kptee}@i2r.a-star.edu.sg

This paper is organized as follows: Section II presents the relevant literature in trajectory learning and DMPs. We formalizes DMP+ and the algorithms for trajectory learning and modification in Section III. Section IV presents our experiment results, followed by a discussion of the results in Section V and conclusion in Section VI.

II. Related Work

Trajectory learning is a fundamental problem in robotics. It enables robots to learn useful skills and solve specific tasks. Numerous representations for encoding observed behaviors have been proposed for trajectory learning. For instance, *Miyamoto et al.* proposed a spline-based approach [12] which has been employed in many contexts [13], though the explicit time dependence is considered undesirable [1]. Hidden Markov models (HMM) is another popular representation for trajectory learning. It has shown to be effective for both action imitation [14] and recognition [15]. Further, *Calinon et al.* presented Gaussian Mixture Models (GMMs) for movement encoding with advantages over HMM during trajectory reproduction [16].

Dynamic Movement Primitives [1], [3] (DMPs) adopt an alternative approach by combining nonlinear dynamical systems for trajectory modeling with statistical machine learning. DMPs is biologically inspired and can be used to explain phenomena in behavioral and brain imaging experiments [3]. DMPs formulation has many desirable properties such as stability and convergence, efficient coupling with other dynamical systems for trajectory modification, and robustness to environment perturbation.

May works on DMPs have investigated approaches for generalizing and adapting learned behaviors to novel situations. For instance, given a learned grasping behavior, online obstacle avoidance [6] is achieved by adding a forcing term that pushes away from the obstacles. Adaptive frequency modulation [8] is explored as a form of temporal coupling to achieve desired drumming behaviors. Reinforcement leaning has been applied to DMPs to guide policy search and improve learned behaviors [17], [18], [19]. Beyond modeling motion in Cartesian space, *Ude et al.* introduced a technique for incorporating end-effector orientation [20]. In [21], *Kulvicius et al.* proposed n modified formulation of DMPs to address the trade-off between trajectory accuracy and endpoint control. The modified formulation is also capable of joining a sequence of learned primitives efficiently. Few works on DMPs address the problem of modeling accuracy, as simply increasing the number of kernels improves reproduction quality. However, reproduction quality is important for tasks requiring accurate trajectory tracking. such as in industrial manufacturing. To the best of our knowledge, our proposed approach is the first to show that the quality of trajectory learning with DMPs can be significantly improved by using local biases to better approximate local gradients of the forcing term, while preserving the desirable properties of DMPs.

DMPs have also been widely used with learning-by-demonstration. *Nakanishi et al.* proposed a method for learning biped locomotion from demonstration [2].DMPs are also used to learn drumming pattern [1], [9] and tennis swings [3] from demonstrations. More recently, DMPs are used for adapting robot behaviors from human gestures [7] and via-points [19]. However, both adaptation techniques require all kernel weights to be updated for adaptation, which is inefficient for a complex trajectory learned with a large number of kernels.

III. Dynamic Movement Primitives Plus

In this paper, we focus on DMPs for discrete movements, which encode trajectory in external space (e.g. Cartesian). External space representation is advantageous over joint space representation as learned representations generalize across different robotics platforms. the formulation of DMPs is presented in the following equations [1].

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

$$\tau\dot{x} = -\alpha_x x \tag{2}$$

$$f = \frac{\sum_{i=1}^{N} \psi_i w_i}{\sum_{i=1}^{N} \psi_i} x \tag{3}$$

$$\psi_i = \exp(-h_i(x-c_i)^2) \tag{4}$$

where y_0 is the initial state, g the goal state, τ the temporal scaling factor, and $\alpha_z, \beta_z, \alpha_x$ positive constants. Further, h_i is the width of each kernel ψ_i, and w_i the learnable weight. (1) represents the dynamic system for trajectory modeling, combining a PD controller with a learnable forcing term f. if the forcing term $f = 0$, the equations represent a globally stable system with a unique attractor $(z, y) = (0, g)$. (2) is the canonical system for the phase variable x to replace the explicit time dependence. (3) and (4) consist of N kernels for learning the forcing term f to model arbitrarily complex trajectory.

DMP+ is formalized by modifying (3) and (4) to be (5) and (6) respectively.

$$f = \frac{\sum_{i=1}^{N} (w_i x + b_i)\psi_i}{\sum_{i=1}^{N} \psi_i + \varepsilon} \tag{5}$$

$$\psi_i = \begin{cases} \exp(-\frac{h_i}{2}(x-c_i)^2), & \text{if } x - c_i \leq \theta_i \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

In contrast with the formulation of DMPs, DMP+ differs by adding a bias term b_i to each kernel and by using truncated kernels. The motivation for the modifications are two folds: 1) The truncated kernels provide an alternative mechanism for the forcing term f to converge to zero as all w_i vanish eventually x progresses, therefore ensuring the stability and convergence of the system. Consequently, we are able to add a bias term for each kernel for better approximation of local gradients; 2) Modifying a learned trajectory only requires

updating a subset of kernel weights, as other kernels are unaffected by the modification.

To model a trajectory, we employ Locally Weighted Regression [22] (LWR) to approximate the appropriate forcing term f_{target}, obtained by:

$$f_{target} = \tau^2 \ddot{y}_d - \alpha_z(\beta_z(g - y_d) - \tau \dot{y}_d). \tag{7}$$

where $y_d, \dot{y}_d, \ddot{y}_d$ are position, velocity and acceleration of a single dimension from the desired trajectory. For each kernel ψ_i, LWR learns w_i and b_i to approximate a neighborhood of f to minimizes the following objective function,

$$J_i = \sum_{t=1}^{P} \psi_i(t)(f_{target}(t) - (w_i x + b_i))^2 \tag{8}$$

where P is the number of sample points from the desired trajectory. The closed-form solution is

$$\begin{pmatrix} w_i \\ b_i \end{pmatrix} = \begin{pmatrix} S_{x^2} & S_x \\ S_x & S_w \end{pmatrix}^{-1} \begin{pmatrix} S_{xy} \\ S_y \end{pmatrix} \tag{9}$$

where

$$S_w = \sum_{j=1}^{P} \psi_i(j)$$
$$S_x = \sum_{j=1}^{P} \psi_i(j)x(j)$$
$$S_{x^2} = \sum_{j=1}^{P} \psi_i(j)x(j)^2$$
$$S_{xy} = \sum_{j=1}^{P} \psi_i(j)x(j)f_{target}(j)$$

The regression performed with (9) is a batch solution. Recursive least squares with a forgetting factor λ may be used [22] for online approximation.

N, the number of kernels, is determined by the complexity of the desired trajectory, such that the mean square error (MSE) of position deviation is less than a constant threshold ϕ. The exact value for ϕ may be determined by the application requirements.

$$MSE = \frac{1}{P}\sum_{j=1}^{P}(y_d - y_{learned})^2 < \phi \tag{10}$$

For a given N, we set the centers c_i and widths h_i as outlined in [9]. In effect, the N kernels are evenly placed across the duration of the trajectory and each approximates the same number of points in its neighborhood.

$$\begin{aligned} c_i &= \exp(-\alpha_x \frac{i-1}{N-1}), i = 1,...,N \\ h_i &= \frac{S}{(c_{i+1} - c_i)^2}, i = 1,...,N \\ h_N &= h_{N-1} \end{aligned} \tag{11}$$

where S is a positive constant. Additionally, we set

$$\theta_i = \frac{K}{\sqrt{h_i}}. \tag{12}$$

(12) limits the number of points to be modeled by each kernel to within K standard deviations from the center c_i.

Given the DMP+ formulation, we present the algorithm for modifying a learned trajectory in Algorithm 1.

Algorithm 1 DMP+ Update

Require: DMP+ representation $D = \{w_i, c_i, b_i, h_i | i = 1,...,N\}$
Input: Trajectory modification $M = \{y'(t) | l \le t \le m\}$
Output: Adapted DMP+ representation D'

Find kernels to be replaced:
1: $i_{min} = \arg\min_i x(l) - c_i \le \theta_i, i = 1,...,N$
2: $i_{max} = \arg\max_i x(m) - c_i \le \theta_i, i = 1,...,N$
Find the range of trajectory needed for kernel update:
3: $x_{max} = c_{i_{min}} + \theta_{i_{min}}$; $x_{min} = c_{i_{max}} - \theta_{i_{max}}$
4: $t_{min} = t(x_{max})$; $t_{max} = t(x_{min})$
5: Generate sample points $\{y(t) | t_{min} < t < t_{max}\}$ from D
Update the kernels:
6: Update the kernels $\{w_i, c_i, b_i, h_i | i = i_{min},...,i_{max}\}$ using (9) and (11) with sample points $\{z(t) | t_{min} \le t \le t_{max}, z(t) = y'(t)$ if $l \le t \le m, z(t) = y(t)$ otherwise$\}$
7: **return** D

The DMP+ update algorithm first determines the range of kernels from i_{min} to i_{max} requiring update given the modification. It is straightforward to find i_{min} and i_{max} by evaluating each kernel at the end points of the modification. We in turn extend the modification segment to include all points modeled by the kernels i_{min} to i_{max}. Each kernel in the range is then relearned using the extended segment as input.

We note that N, the number of kernels needed to learn a trajectory, is determined by the complexity of the trajectory. N in turn determines the number of sample points to extend at both ends of the modification for trajectory update. As N increases, fewer number of points need to be extended. Lastly, we note that the algorithm uses $i_{max} - i_{min} + 1$ kernels to represent the adaptation, which may underfit the modified segment if the modification contains complex trajectory. In this case, the number of kernels may be increased until the MSE drops below a desirable level.

IV. Experimental Evaluation

To verify the efficacy of our proposed approach, we implemented DMP+ algorithms in Matlab and on a KUKA LBR iiwa. We set $\alpha_z = 25$, $\beta_z = \alpha_z/4$, and $\alpha_x = \alpha_z/3$ according to [1]. $\beta_z = \alpha_z/4$ guarantees that the globally stable system in (1) is critically damped, allowing y to monotonically converge to g. ε is an arbitrarily small value of 10^{-8}. We set $K = 3$ such that each kernel only omits points more than 3 standard deviations away from the center. For all experiments, we use Euler integration to integrate the differential equations to calculate trajectory positions.

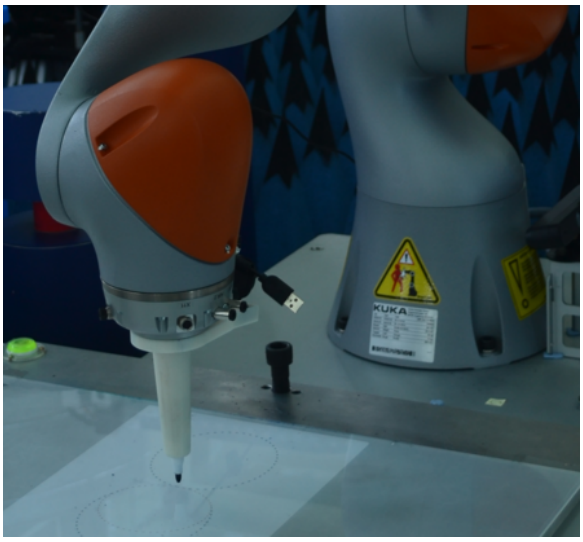


Fig. 1. The experiment setup. Pen mounted on a Kuka LBR-iiwa draws figure of eight on a planar surface

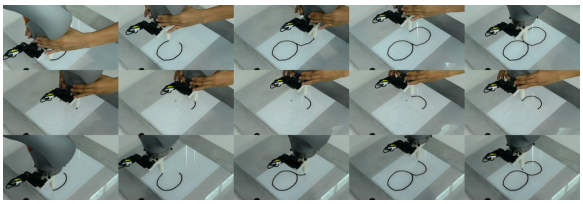


Fig. 2. The three rows correspond with learning, adaptation, and production phase respectively. Learning phase: demonstration of figure eight. Adaptation phase: partial trajectory update. Production Phase: reproduction of the adapted trajectory

As an initial test, we compare DMP+ against the state-of-the-art DMP implementation[1] to learn a minimum-jerk trajectory. Minimum-jerk trajectory resembles human reaching trajectories [23], and is used in [24] as a testing benchmark.

In Fig. 3, the desired trajectory is a minimum jerk trajectory with a goal state $g = 1$. 1000 sample points evenly spaced in time domain are used for training. Both DMPs and DMP+ use 10 kernels respectively. It is visually clear that DMP+ tracks the desired trajectory more closely. We quantify the reproduction quality in terms of MSE in Table I by varying the number of kernels used. For all cases, DMP+ achieves lower MSE than DMPs.

We attribute the lower MSE to additional bias terms b_i used in DMP+, which significantly improves the approximation of the forcing term f. Without the bias term, each kernel in DMPs must pass through the origin, therefore significantly limiting its ability to model local gradients. The comparison in Fig. 3c clearly shows that DMP+ approximates local gradients of the forcing term significantly better. Consequently, kernels in DMP+ provides reasonable approximation for larger regions, therefore reducing the number of kernels needed to accurately model the entire trajectory. An alternative comparison using Table I considers models with

[1]code from `http://www-clmc.usc.edu/software/git/gitweb.cgi?p=matlab/dmp.git;a=summary`

TABLE I

MSE FOR MINIMUM JERK TRAJECTORY REPRODUCTION: DMP VS. DMP+

# of kernels	DMP+	DMP
5	7.5940	15.6579
10	0.1288	5.9882
20	0.0011	0.6432
40	0	0.0504

the same number of free parameters, i.e. comparing a DMP+ model with N kernels against a DMP model with $2N$ kernels. We note that DMP+ still outperforms DMPs significantly, except for $N = 5$. The poorer performance is explained by severe underfiting when N is too small.

To further compare the two approaches, we tested both methods in learning handwriting trajectory from the UJI Pen dataset[2]. We choose from the dataset a subset of characters written with a single stroke, as the dataset does not record the pen-up trajectories for multi-stroke characters. For the experiment, both DMPs and DMP+ use 10 kernels for trajectory learning.

As an example, we plot the forcing term and trajectory learned of character a in Fig. 6 and 7 respectively. Fig. 6 shows the training samples for the forcing term f in x and y axes, and the approximation by both DMP+ and DMPs. The training samples are challenging to model due to significant noise stemming from the numerical differentiation of position data.

Fig. 7 shows the trajectory reproduction by DMP+ and DMPs. It is clear that DMP+ tracks training samples more closely in both x and y axes. Consequently, DMP+ reproduces a trajectory that better resembles the training data. Due to poorer approximation, DMPs converges to the goal position way past the duration of the target trajectory. The results also show that DMP+ is robust against noise in training data.

To quantify our results, we compare the MSE on trajectory reproduction between DMP+ and DMPs in Table II. DMP+ achieves consistently 10 fold decrease in MSE.

In the second experiment, we evaluate the performance of DMP+ update algorithm in adapting a learned trajectory on Kuka LBR-iiwa. The experimental setup is shown in Fig. 1. In this experiment, we kinesthetically operate a KUKA LBR iiwa to draw a figure of eight. Trajectotry demonstration is sampled at $50Hz$ as input. The trajectory is first learned with DMP+, and then adapted partially with a new demonstration. The original reference trajectory and the adapted segment are shown in Fig. 4. We note that the reference trajectory may be obtained via other methods such as tele-operation. Given the modification to the original trajectory, we use Algorithm 1 to update the learned model. We compare the MSE achieved by the update algorithm against learning the

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

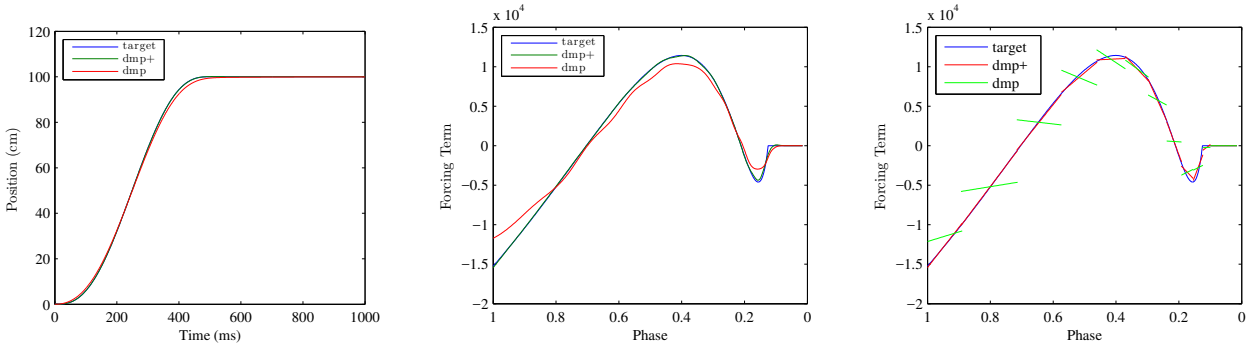


Fig. 3. a: Target minimum trajectory, reproduction by DMP+ and DMPs. b: Functional approximation of the forcing term. c: Local gradient approximation of the forcing term. DMP+ significantly outperforms DMP with the bias terms.

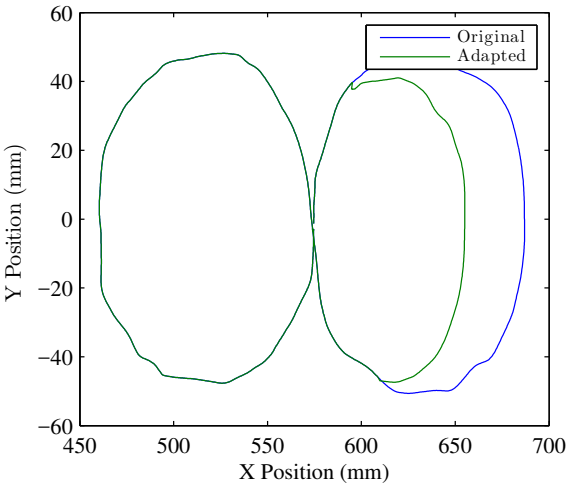


Fig. 4. The kinesthetic demonstrations of the original and adapted trajectories for figure of eight. The adapted trajectory draws an eight with smaller right half.

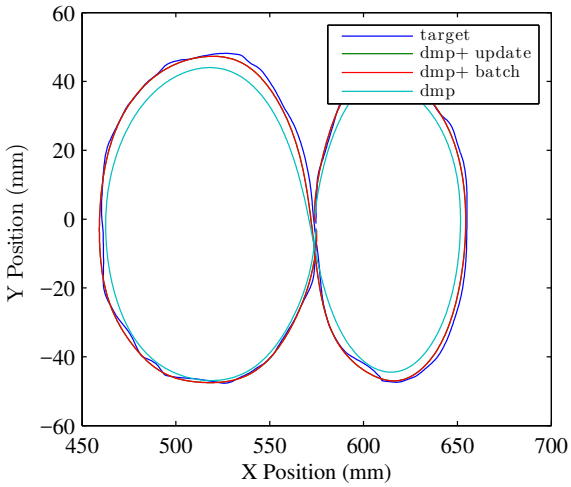


Fig. 5. The figure shows the adapted trajectory, the reproduction by DMP+ update algorithm, by DMP+ batch learning, and by DMPs batch learning respectively.

full adapted trajectory by DMPs, using same number of kernels. We outline the learning, adaptation and reproduction phases of the experiment in Fig. 2, and plot the reproduction trajectories in Fig. 5.

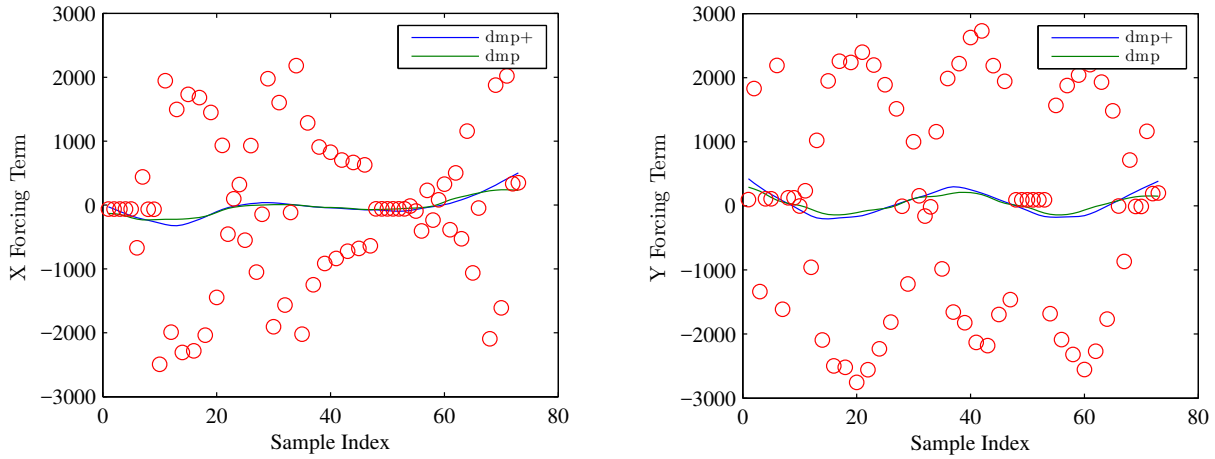
The results show DMP+ update algorithm achieves lower MSE against DMPs batch learning, and comparable MSE against DMP+ batch learning. The advantage over DMPs can be attributed to better modeling accuracy demonstrated in the previous experiments. The update algorithm is equivalent to DMP+ batch learning as the learned representation remain the same. We note that if a different number of kernels were used to represent the modified segment, or any kernel parameter changes (e.g. c_i for kernel function placement), we achieve comparable performance between DMP+ batch learning and the adaptation algorithm.

V. Discussion

The experiment results show that DMP+ achieves 1) improved modeling performance and 2) efficient adaptation of learned trajectories. We have shown the significance of local biases in improving model accuracy and applied the technique to adapt the trajectory for a figure of eight.

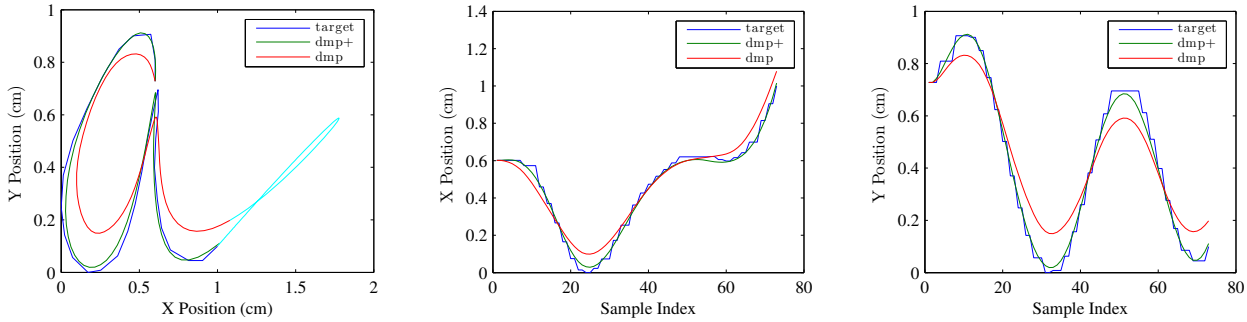
The key contributions from our formulation are the use of truncated kernels and the addition of local biases. They truncated kernels 1) provides a mechanism for the forcing term to converges to zero and thus ensuring the stability of the dynamical system used in the formulation, and 2) limits the number of kernels affected by trajectory modification. Consequently, we are able to add local biases to achieve better modeling accuracy and efficient updates of learned trajectories.

Our approach is complementary to the existing techniques for DMP modification. For instance, temporal and spatial coupling could be used with DMP+ by simply adding the corresponding terms to the dynamical equations. In contrast with online adaptation presented in [7], whereby the adaptation is based on obstacle avoidance, DMP+ is appropriate in situations whereby an arbitrary and static modification is required. DMP+ offers exact control over



(a) X-axis Forcing Term (b) Y-axis Forcing Term

Fig. 6. Comparison on the forcing term approximation between DMP+ and DMPs in x and y axes for character a. The training samples contain significant noise.



(a) (b) (c)

Fig. 7. a) Trajectory reproduction of character a. b) Trajectory reproduction in x dimension against time. c) Trajectory reproduction in y dimension against time.

TABLE II

MSE FOR UJI PEN DATA SET TRAJECTORY REPRODUCTION: DMP VS. DMP+. 10 KERNELS USED FOR BOTH LEARNING ALGORITHMS.

Character	DMP+	DMP
a	0.0014	0.0117
b	0.0007	0.0079
c	0.0000	0.0015
e	0.0001	0.0050
0	0.0008	0.0074
2	0.0001	0.0029
6	0.0002	0.0047

trajectory adaptation and is suitable for applications requiring high accuracy such as in industrial manufacturing . DMP+ may be applied to High-Mix-Low-Volume manufacturing, whereby large variants of similar tasks need to be performed by robots. In addition to improved modeling accuracy, DMP+ enables efficient reuse of learned robotic motions, which may reduce production downtime and improve productivity.

VI. CONCLUSION

We present DMP+, a variant of DMP that achieves better trajectory learning, reduces computational cost, and allows easy adaptation of learned trajectories. We have experimentally verified its performance and outlined its application in industrial manufacturing. In our paper, we chose a simple method to place kernels evenly across the duration of trajectories in effect. Techniques such as Receptive Field Weighted Regression (RFWR) [25] exists to automatically determine the number of kernels and their placements. However, we note that RFWR does not perform well in our experiments, and we intend to investigate automatic optimization of kernel placement in future works.

REFERENCES

[1] 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.

[2] 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.

[3] 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.

[4] A. J. Ijspeert, J. Nakanishi, and S. Schaal, "Learning rhythmic movements by demonstration using nonlinear oscillators," in *IEEE/RSJ International Conf on Intelligent Robots and Systems (IROS), 2002*, no. BIOROB-CONF-2002-003, 2002, pp. 958–963.

[5] ——, "Movement imitation with nonlinear dynamical systems in humanoid robots," in *IEEE International Conference on Robotics and Automation (ICRA), 2002*, vol. 2. IEEE, 2002, pp. 1398–1403.

[6] D.-H. Park, P. Pastor, S. Schaal, *et al.*, "Movement reproduction and obstacle avoidance with dynamic movement primitives and potential fields," in *8th IEEE-RAS International Conference on Humanoid Robots (Humanoids), 2008*. IEEE, 2008, pp. 91–98.

[7] 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.

[8] A. Gams, A. J. Ijspeert, S. Schaal, and J. Lenarčič, "On-line learning and modulation of periodic movements with nonlinear dynamical systems," *Autonomous robots*, vol. 27, no. 1, pp. 3–23, 2009.

[9] 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.

[10] 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.

[11] A. Gams, T. Petrič, M. Do, B. Nemec, J. Morimoto, T. Asfour, and A. Ude, "Adaptation and coaching of periodic motion primitives through physical and visual interaction," *Robotics and Autonomous Systems*, vol. 75, pp. 340–351, 2016.

[12] H. Miyamoto, S. Schaal, F. Gandolfo, H. Gomi, Y. Koike, R. Osu, E. Nakano, Y. Wada, and M. Kawato, "A kendama learning robot based on bi-directional theory," *Neural networks*, vol. 9, no. 8, pp. 1281–1302, 1996.

[13] L. Sciavicco and B. Siciliano, *Modelling and control of robot manipulators*. Springer Science & Business Media, 2012.

[14] 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.

[15] 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.

[16] S. Calinon, F. Guenter, and A. Billard, "On learning, representing, and generalizing a task in a humanoid robot," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 37, no. 2, pp. 286–298, 2007.

[17] J. Kober and J. Peters, "Learning motor primitives for robotics," in *IEEE International Conference on Robotics and Automation (ICRA), 2009*. IEEE, 2009, pp. 2112–2118.

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

[19] 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.

[20] A. Ude, B. Nemec, T. Petri, and J. Morimoto, "Orientation in cartesian space dynamic movement primitives," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 2997–3004.

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

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

[23] T. Flash and N. Hogan, "The coordination of arm movements: an experimentally confirmed mathematical model," *The journal of Neuroscience*, vol. 5, no. 7, pp. 1688–1703, 1985.

[24] A. Gams and A. Ude, "Generalization of example movements with dynamic systems," in *9th IEEE-RAS International Conference on Humanoid Robots (Humanoids), 2009*. IEEE, 2009, pp. 28–33.

[25] S. Schaal and C. G. Atkeson, "Receptive field weighted regression," Tech. Rep., 1997.