2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Macau, China, November 4-8, 2019

Mobile Robot Learning from Human Demonstrations with Nonlinear Model Predictive Control

Yingbai Hu[1], Guang Chen [1,2*], Xiangyu Ning[2], Jinhu Dong[2], Shu Liu[3], Alois Knoll[1]

Abstract— **Learning by imitation is a powerful way that can reduce the complexly in searching space. It could help the mobile robot to acquire new skills from interaction with a human-being in natural way. In this paper, the dynamic movement primitives (DMPs) is utilized to imitate the trajectory from human walking. DMPs is a modified formulation of virtual spring-dampers (VSD) system that enjoys better fitting performance in learning. Further, while dealing with the trajectory tracking problem of mobile robots, a novel nonlinear model predictive control (MPC) approach is proposed for motion control. The nonlinear MPC scheme applies a new neural network named Varying-parameter Lagrangian Neural Network (VP-LNN) to solve a Quadratic Programming (QP) problem by iterating over a finite receding horizon. The new network of VP-LNN can converge to the global optimal solution. Thus, a new human-robot interaction (HRI) scheme for mobile robot is proposed, which can reduce the complexity in motion planning in various applications.**

I. INTRODUCTION

If the robot can learn motor skills in a natural way like human, it will be able to complete tasks with more complexity. Previously, robot learning has attracted the attention of many researchers. In [1], [3], authors present the dynamic movement primitives (DMPs) for manipulator learning by imitation. The DMPs is a special virtual spring-dampers (VSD) system, which can represent the motion of a robot, even for the nonlinear movements.

However, most of the robot learning methods are applied to manipulators. In [4], dual-arm robot based on DMPs cooperate to cut a vegetable by human demonstrations. In [5], reinforcement learning fusion with DMPs is employed to learn the grasping actions with a number of iterations in external disturbance environment. In this paper, DMPs is applied to learn the human movement for mobile robot. The learning scheme is a novel human-robot interaction method which can imitate the motion from human demonstrations. Therefore, mobile robot is able to reproduce the demonstration by DMPs, which leads to the reduction of complexity for motion planning tasks.

Considering kinematic constraints, nonlinear model predictive control (MPC) is often used for controlling mobile robot [12]. MPC is an optimal control algorithm that can deal with the system which has nonlinear input-output over a finite horizon. It ensures the control sequences are always allowed within physical constraints. In this work, a new varying-parameter Lagrangian neural network (VP-LNN) method is proposed to solve optimization problem of MPC. The neural network method can solve QP problem in real time with physical constraints and it has good performance for applications of robot system. The new VP-LNN can obtain optimal solution for QP problems, while the network doesn't include matrix inversion, matrix-matrix multiplication or high-order nonlinear computation.

In the learning-control tasks, trajectories are demonstrated by human and recorded by Xsens mti-310. In the reproduction phase, DMPs is applied to reproduce the trajectory, where the nonlinear item of DMPs is obtained by locally weighted regression (LWR) in [14], and the new trajectory is generated by DMPs. Subsequently, VP-LNN based MPC method is employed to control the mobile robot for tracking the trajectory from reproduction. The framework of learning-based MPC scheme is shown in Fig. 1. The contributions of this paper can be summarized as:

1) A novel learning based human-robot interaction method is proposed for mobile robot, where mobile robot can learn complex motor skills from human demonstrations, the reduction of complexity for motion planning with the learning-control system is significant.
2) A new VP-LNN method is presented to solve QP optimization problem of MPC. It converges quickly to optimal value with physical constraints.

II. RELATED WORK

Robot learning is a special method that helps to obtain novel skills from environment or human being. In [6], Gaussian Mixture Model fusion dynamical system is used to learn the motions from demonstration. In [7], authors propose the DMPs based learning framework to learn the driving motions from human experience. However, most of the research focus on manipulators learning. We present a novel human-robot interaction (HRI) method for mobile robot learning.

Traditional controller do not consider the kinematic constraints in [8], [9]. In [10], an adaptive controller is proposed to track the trajectory in uncertainties of mobile robot. In [11], the nonlinear slide mode controller is designed for robot which can ensure the controller system from initial state to target equilibrium. In this paper, nonlinear MPC method is proposed to control the mobile robot.

In [15], the traditional numerical optimization method has high dimension and high computational cost for MPC, so it is not suitable for fast-changing robot system. In [16], [17], [18], a neural-dynamic based MPC method is proposed

* Guang Chen is the corresponding author.
[1]Chair of Robotics, Artificial Intelligence and Real-time Systems, Technische Universität München, München, 85748, Germany `yingbai.hu@tum.de, knoll@in.tum.de`
[2]College of Automotive Engineering, Tongji University, Shanghai, 201804, China `guangchen@tongji.edu.cn, sheldonnxy@gmail.com, 317684920@qq.com`
[3]D-MTEC, ETHZurich, 8092, Switzerland `liush@ethz.ch`
The research leading to these results has partially received funding from the European Union's Horizon 2020 Research and Innovation Program in accordance with Grant Agreement No. 785907 (HBP SGA2), and from the Shanghai AI Innovative Development Project.

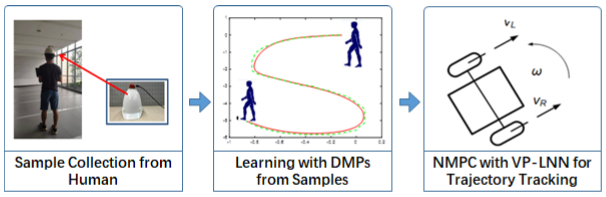
Sample Collection from Human
Learning with DMPs from Samples
NMPC with VP-LNN for Trajectory Tracking
v_L
ω
v_R

Fig. 1. The learning-control framework for mobile robot.

that does not rely on numerical optimization. Compared with traditional methods, this method can avoid some of the major pitfalls facing of traditional MPC and has global convergence, low computational complexity that can deal with the high online computational burden. Both of these methods are good to solve the QP problems of MPC. In this paper, the novel VP-LNN is proposed to optimize the QP problem of MPC. The novel neural network method converges to the global optimal solution, and has the simple structure.

III. IMITATION LEARNING WITH DMPS

Dynamical movement primitives system is a second-order nonlinear virtual spring dampers. DMPs can encode the nonlinear movement by learning the nonlinear item F. The formulation of DMPs is expressed as

$$\ddot{y}_t = K^{p1}(g - y_t) - K^{v1}\dot{y}_t + F \tag{1}$$

where K^{p1}, K^{p2} are the full stiffness matrix, F is the interaction forces, $y_t, \dot{y}_t, \ddot{y}_t$ and g are the position, velocity, acceleration and the attractor point respectively. The nonlinear items in (1) are defined as:

$$F = s_t f_t \tag{2}$$
$$\dot{s}_t = \alpha_s s_t \tag{3}$$

The system (1) is the transform system that generates the reference trajectories $[y_t, \dot{y}_t, \ddot{y}_t]$, including the position, velocity and acceleration, where g is the goal parameter of the attractor point. The equation (2) is the nonlinear item that can model the complex trajectory. The equation (3) is the canonical system, where α_s is time constants, and $s_t(0) = 1$ that means s_t asymptotically decays from initial state to 0. After multiplication of the canonical system, the nonlinear item F_t will converge to zero with the time increasing that means the DMPs system will become the linear spring damper system. If we get the desired trajectory, we can obtain the nonlinear item in DMPs from the equations (1)-(3),

$$F_{target} = \ddot{y}_d - K^{p1}(g - y_d) - K^{v1}\dot{y}_d \tag{4}$$

where $[y_d, \dot{y}_d, \ddot{y}_d]$ denotes the desired trajectory, and f_t can be obtained by LWR algorithm.

IV. MODEL PREDICTIVE CONTROL SCHEME

A. Mobile Robot Control System

The two wheels mobile robot and kinematic model are shown in Fig. 1. According to the relationship between robot velocity v and two driving wheels velocity (v_L, v_R), the robot velocity and angle velocity are expressed as: $v = (v_L + v_R)/2$, $\omega = (v_L - v_R)/D$, respectively. D denotes the distance of two wheels. The kinematics model formulation of mobile robot can be represented as:

$$\dot{X} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ \omega \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} u \tag{5}$$

where (x, y) is the mobile robot position in Cartesian space, and θ is the orientation angular; $u = (v, \omega)^T$ is the control input, and X is the state vector. From the formulation in (5), we can obtain kinematics model of the reference trajectory by state vector $X_r = (x_r, y_r, \theta_r)^T$ and control input $u_r = (v_r, \omega_r)^T$, and it is expressed as:

$$\dot{X}_r = \begin{bmatrix} \dot{x}_r \\ \dot{y}_r \\ \dot{\theta}_r \end{bmatrix} = \begin{bmatrix} v_r\cos\theta_r \\ v_r\sin\theta_r \\ \omega_r \end{bmatrix} = \begin{bmatrix} \cos\theta_r & 0 \\ \sin\theta_r & 0 \\ 0 & 1 \end{bmatrix} u_r \tag{6}$$

Therefore, we can get the robotic kinematic errors,

$$X_e = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r - x \\ y_r - y \\ \theta_r - \theta \end{bmatrix} \tag{7}$$

where $X_e = [x_e, y_e, \theta_e]^T$. The derivative of $\dot{X}_e$ can be obtained from error state in (7),

$$\begin{cases} \dot{x}_e = \omega y_e - v + v_r\cos\theta_e \\ \dot{y}_e = -\omega x_e + v_r\sin\theta_e \\ \dot{\theta}_e = \omega_r - \omega \end{cases} \tag{8}$$

The control input is reformulated as:

$$u_e = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} v_r\cos\theta_e - v \\ \omega_r - \omega \end{bmatrix} \tag{9}$$

Substituting (9) into (8), the kinematics model of X_e can be rewritten as

$$\dot{X}_e = \begin{bmatrix} \dot{x}_e \\ \dot{y}_e \\ \dot{\theta}_e \end{bmatrix} = \begin{bmatrix} 0 & \omega & 0 \\ -\omega & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} + \begin{bmatrix} u_1 \\ v_r\sin\theta_e \\ u_2 \end{bmatrix} \tag{10}$$

The formulation in (10) can be reformulated as a following equation:

$$\dot{\bar{X}}_e = \begin{bmatrix} 0 & \omega_r & 0 \\ -\omega_r & 0 & v_r \\ 0 & 0 & 0 \end{bmatrix} \bar{X}_e + \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} u_e \tag{11}$$

Therefore, the motion control problem of mobile robot in (5) is converted as a stabilization problem of an nonlinear affine system in (11).

B. Nonlinear MPC Framework

The nonlinear MPC framework can be presented as:

$$X(k+1) = f(X(k)) + g(X(k))u(k) \tag{12}$$

the input and state vector constraints are defined as:

$$X(k) \in R^X, k = 1, 2, \ldots, N$$
$$u(k) \in R^U, k = 1, 2, \ldots N_u$$

where X and u denote state vector and input vector, respectively. In nonlinear affine system (12), $f(.)$ and $g(.)$ denote the nonlinear continuous functions. The initial conditions satisfy $f(0) = 0$, and $N \geq 1$ and $1 \leq N \leq N_u$ that denotes the prediction horizon.

In each sampling period, the optimal input vector can be obtained by a given cost function online optimization, thus the nonlinear MPC scheme is formulated by iterative solution of optimal control problems. The cost function $S(X, u)$ can be defined as a quadratic form as following:

$$S(k) = \sum_{j=1}^{N} \|X(k+j|k)\|_Q^2 + \sum_{j=0}^{N_u-1} \|\Delta u(k+j|k)\|_R^2 \quad (13)$$

where $X(k+j|k)$ is the predicted the future horizon state; $\Delta u(k+j|k) = u(k+j|k) - u(k-1+j|k)$ which is the increment of input vector; the parameters R and Q present constant design weight matrix; the symbol $\|\cdot\|$ represents the Euclidean norm of the corresponding vector. The equation (12) can be reformulated as:

$$X_e(k+1) = f(X_e(k)) + g(X_e(k))\, u_e(k) \quad (14)$$

$$\begin{aligned} \textit{subject to} \quad & u^- \le u_e(k) \le u^+ && (15)\\ & \Delta u^- \le \Delta u(k) \le \Delta u^+ && (16)\\ & X^- \le X_e(k) \le X^+ && (17) \end{aligned}$$

$$f(X_e) = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + T \begin{bmatrix} \omega_r x_2 \\ -\omega_r x_1 + v_r x_3 \\ 0 \end{bmatrix}$$

$$g(X_e) = T \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}$$

where $X_e = [x_1, x_2, x_3]^T = [x_e, y_e, \theta_e]^T$ is the state vector, and T is the sampling period; $u_e = [u_1, u_2]^T$, $u_1 = v_r cos(\theta_e) - v$, $v_2 = \omega_r - \omega$ is the input vector; (u^-, u^+) are the upper and lower limits of input variable, and $(\Delta u^-, \Delta u^+)$, (x^-, x^+) are also the upper and lower limits of input increment variable and state variable, respectively. The block diagram of MPC based control system is shown in Fig.1.

To construct the QP problem for online optimization, we introduce the vectors as:

$$\bar{X} = [X_e(k+1|k), ..., X_e(k+N|k)]^T \quad (18)$$

$$\bar{u}(k) = [u_e(k|k), ..., u_e(k+N_u-1|k)]^T \quad (19)$$

$$\Delta\bar{u}(k) = [\Delta u_e(k|k), ..., \Delta u_e(k+N_u-1|k)]^T \quad (20)$$

According to (14), and (18)–(20), we can get the predicted output,

$$\bar{X}(k) = A\Delta\bar{u}(k) + \hat{f} + \hat{g} \quad (21)$$

$$A = \begin{bmatrix} g(X_e(k|k-1)) & \cdots & 0 \\ g(X_e(k+1|k-1)) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ g(X_e(k+N-1|k-1)) & \cdots & g(X_e(k+N-1|k-1)) \end{bmatrix}$$

$$\hat{f} = \begin{bmatrix} f(X_e(k|k-1)) \\ f(X_e(k+1|k-1)) \\ \vdots \\ f(X_e(k+N-1|k-1)) \end{bmatrix}$$

$$\hat{g} = \begin{bmatrix} g(X_e(k|k-1)u(k-1)) \\ g(X_e(k+1|k-1)u(k-1)) \\ \vdots \\ g(X_e(k+N-1|k-1)u(k-1)) \end{bmatrix}$$

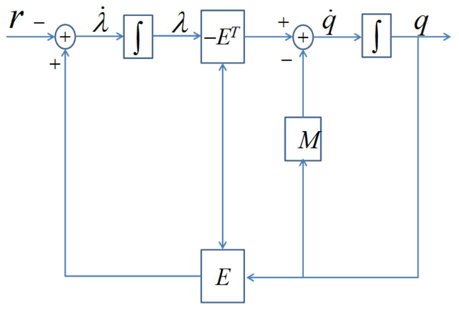

Fig. 2. Block diagram of VP-LNN.

where $A \in R^{3N \times 2N_u}$, $\hat{f} \in R^{3N}$, and $\hat{g} \in R^{3N}$. Then, the optimization problem in (13) can be rewritten as:

$$\begin{aligned} \textit{minimum} \quad & \left\|A\Delta\bar{u}(k) + \hat{f} + \hat{g}\right\|_Q^2 + \|\Delta u(k)\|_R^2 && (22)\\ \textit{subject to} \quad & \Delta\bar{u}^- \le \Delta\bar{u}(k+1) \le \Delta\bar{u}^+ && (23)\\ & \bar{u}^- \le \bar{u}(k-1) \le \bar{u}^+ && (24)\\ & \bar{u}^- \le \bar{u}(k-1) + H\Delta\bar{u}(k) \le \bar{u}^+ && (25)\\ & X^- \le \hat{f} + \hat{g} + A\Delta\bar{u}(k) \le X^+ && (26) \end{aligned}$$

where $H = \begin{bmatrix} I & 0 & 0 & 0 \\ I & I & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ I & I & \cdots & I \end{bmatrix} \in R^{2N_u \times 2N_u}$

The Optimization problem in (22)-(26) can be reformulated as a QP problem:

$$\begin{aligned} \textit{minimum} \quad & \tfrac{1}{2}\Delta\bar{u}^T M \Delta\bar{u} + c^T\Delta\bar{u} && (27)\\ \textit{subject to} \quad & E\Delta\bar{u} \le r && (28)\\ & \Delta\bar{u}^- \le \Delta\bar{u} \le \Delta\bar{u}^+ && (29) \end{aligned}$$

and the parameters details are given,

$$M = 2(A^T Q A + R) \in R^{2N_u \times 2N_u}$$

$$c = -2A^T Q(\hat{g} + \hat{f}) \in R^{2N_u}$$

$$\text{E} = \begin{bmatrix} -H \\ H \\ -A \\ A \end{bmatrix} \in R^{n_1}, r = \begin{bmatrix} -u^- + u(k-1) \\ u^+ - u(k-1) \\ -X^- + \hat{f} + \hat{g} \\ X^+ - \hat{f} - \hat{g} \end{bmatrix} \in R^{n_2}$$

where $n_1 = (4N_u + 6N) \times 2N_u$, $n_2 = 4N_u + 6N$.

V. VARYING-PARAMETER LAGRANGE NEURAL NETWORK

Firstly, we rewritten the QP problem as following:

$$\begin{aligned} \textit{minimum} \quad & \tfrac{1}{2}q^T M q + c^T q && (30)\\ \textit{subject to} \quad & Eq \le r && (31)\\ & q^- \le q \le q^+ && (32) \end{aligned}$$

where $q = \Delta\bar{u}$, $q^- = \Delta\bar{u}^-, q^+ = \Delta\bar{u}^+$, and $q \in [q^-, q^+]$. The Lagrangian of the time-varying quadratic programming problem subject to equality constraints described in (30) and (31) is defined as follows:

$$L(q, \lambda) = \frac{1}{2}q^T M q + c^T q + \lambda^T (Eq - r) \quad (33)$$

where λ is an n-dimensional column vector of Lagrangian multipliers at time t. By setting the partial derivatives of $L(q,\lambda)$ to zero, the Lagrange necessary condition gives rise to the following time-varying algebraic equations:

$$\begin{cases} \frac{\partial L(q,\lambda)}{\partial q} = Mq + c + E^T\lambda = 0 \\ \\ \frac{\partial L(q,\lambda)}{\partial \lambda} = Eq - r = 0 \end{cases} \tag{34}$$

Combining the above two partial derivatives together in (34), we have

$$Wz = \nu \tag{35}$$

where $W = \begin{bmatrix} -M & -E^T \\ E & 0 \end{bmatrix}$, $z = \begin{bmatrix} q \\ \lambda \end{bmatrix}$, $\nu = \begin{bmatrix} c \\ r \end{bmatrix}$

The varying-parameter Lagrange neural network model is defined as:

$$\dot{z} = \mu \cdot \exp(\eta t) \cdot (Wz - \nu) \tag{36}$$

where $\mu > 0$ is the positive constant parameter, $\mu \cdot exp(\eta t)$ is used to scaling the convergence rate of VP-LNN. VP-LNN is globally convergent to the optimal solution. Compared with other neural network models, the VP-LNN has no matrix inversion, matrix-matrix multiplication or high-order nonlinear computation, which has better efficiency for computing and can be easily implemented in hardware. The block diagram of VP-LNN is shown in Fig. 2.

VI. SIMULATION

A. Tracking the '8'-shape trajectory with LNN-MPC

In this section, the VP-LNN based MPC method is applied to track given trajectories by a two-wheel mobile robot, and we demonstrate the results as the performance of the proposed method in simulation.

The parameters of MPC and VP-LNN are set as $N = 3$, $N_u = 2$, $Q = 3I$, $R = 2I$, $\kappa = 0.001$, $\mu = 10^{-3}$, respectively. The sampling period $\Delta t = 0.1s$. In this simulation task, the mobile robot tracks an '8'-shape trajectory for periodic movement, and the reference trajectory is given as: $x = sin(0.1t); y = 2sin(0.05t)$. The initial position of the reference trajectory is $X_r(0) = [0; 0; 0.785]$, and the initial position of mobile robot is $X(0) = [0.3; 0; 0.7]$.

From Fig. 3(a), we can see that the exploration trajectories converge quickly to the desired trajectories, even when robot starts from a randomly chosen initial position. From Fig. 3(b), the actual values of orientation angular is extremely close to the desired value. Last, the tacking errors of the mobile robot quickly converge to zero as shown in Fig. 3(c)-3(d), indicating that the VP-LPNN based MPC can track the desired trajectory. It is obvious that the VP-LNN has good robustness and effectiveness.

B. Learning the trajectory from demonstration with learning-control Scheme

In this simulation, two tasks are tested, which includes the following of a simple 'S'-shape and of a more complex 'SS' shape path from human demonstrations. Moreover, some obstacles are set for 'S' and 'SS' shape tasks, where a human demonstrate how to pass through the obstacles, and then the robot passes through obstacles by imitation learning.

The parameters of DMPs are set as: $K^{P1} = diag[50, 50]$, $K^{v1} = diag[10, 10]$, $K^{P2} = diag[10, 10]$, $K^{v2} = diag[\sqrt{20}, \sqrt{20}]$. The sampling period is set as $\Delta t = 0.005s$. MPC and VP-OneLPNN parameters are set same as VI-A.

1) Learning the 'S'-shape path: In the first task, the mobile robot learns the 'S'-shape path from demonstrations. There are some obstacles randomly placed on the both sides of the 'S'-Shape path. Firstly, a human demonstrates the movement from the initial position to the target position in Cartesian space, where the path is recorded by Xsens mti-310. Then, the DMPs based learning framework is used to obtain the learning trajectory. Lastly, the VP-LNN based model predictive control method is applied to track the learning trajectory of the mobile robot.

In this task, we track the learning trajectory starting from $X_0 = [0, 0, 0]$. The learning-control result is shown in Fig. 4(a), and we can see that the tracking results are extremely close to the learning results. The position error and orientation angular error are shown in Fig. 5(a)–5(c), where the errors are close to zero. This also demonstrates the good performance of VP-LNN based MPC. The tracking results of linear velocity, and angular velocity are shown in Fig. 5(e)–5(f), respectively. For comparison, we randomly choose an initial position of the mobile robot $X_0 = [-0.06, 0.15, 0.01]$, and the simulations are shown in Fig. 4(b)–5(d). It is obvious that the robot trajectories coincide with the learning track after a period of time.

2) Learning the more complex 'SS' shape path: In the second task, the mobile robot learns a more complex task of 'SS' shape path from human demonstrations. In order to verify the effectiveness of the proposed method, we designed a more complex scenario, where the path is added more 'S'-turn, and also more obstacles are randomly placed on both sides of the path. In our experiment, a human demonstrates the movement through obstacles, and then the mobile robot learns to pass the obstacles in narrow space by imitation.

We also set two different initial position of mobile robot in this task, which are $X_0 = [0; 0; 0]$ and $[0.1; -0.1; 0.01]$. The imitation learning results are shown in Fig. 4(c)–4(d). In the motion control phase, the mobile robot can successfully cross the path with obstacles by imitation. The tracking errors of linear velocity and angular velocity are shown in Fig. 6(e), Fig. 6(f), respectively. The tracking errors are shown in Fig. 6(a)–6(c). The results also show good performance with learning-control scheme, even with different initial position which are shown in Fig. 4(d)–6(d).

C. Discussion

Overall, the above results prove that our method are able to converge to global optimal solution within small amount of time. In addition, the tracking errors, despite of minor turbulence in the initial states, are kept in a small range and also have a quick convergence towards zero. The mobile robot is able to successfully cross the path without colliding the obstacles, even when the tasks are challenging.

It is important to note that, in scenario VI-A, we evaluate the algorithm running time and the amount of time that an algorithm requires until the largest error converges to below 0.01. In our setting, the sampling time is 0.1 second and we calculate the running time for 10000 samples. We compare the performance of our proposed VP-LNN with PDNN [18] and RNN [19]. The results are shown in Tab VI-C. The proposed VP-LNN method outperforms RNN in solving the QP

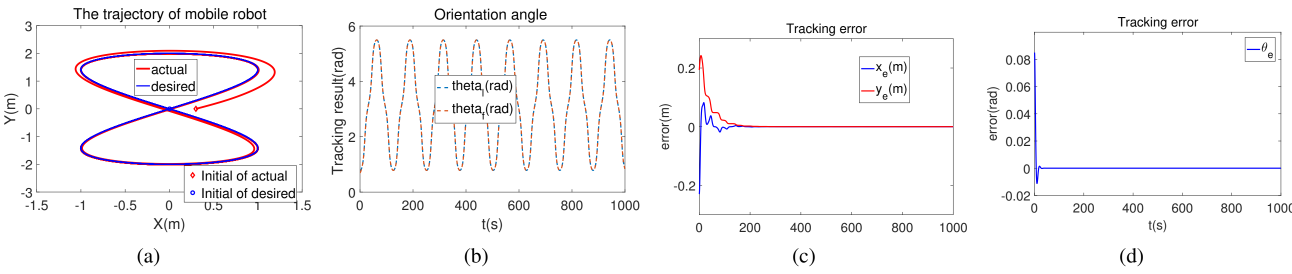

Fig. 3. Simulation results using nonlinear MPC based on VP-LNN. (a) Tracking results of '8'-shape trajectory for periodic movement. (b) The results of orientation angular. (c) Tracking error X_e and Y_e. (d) Tracking error θ_e.

TABLE I
SIMULATION RESULTS COMPARISON

Neural Network	Running time (s)	Convergence (s) ($\lvert \text{error} \rvert \leq 0.01$)
VP-LNN	11	113
PDNN [18]	16	95
RNN[19]	70	300

problem of MPC, from the perspective of both computation time and accuracy. Note that RNN could certainly deal with the nonlinear second-order optimization problem, for which VP-LNN and PDNN fail to address. However, we believe that VP-LNN trade-off minor convergence time for shorter running time as compared to others, which we consider is an advantage considering that in terms of implementation on mobile robots, higher computation efficiency is a key factor.

VII. CONCLUSION AND FUTURE WORK

In this paper, combined with dynamic movement primitives, we present a novel learning based human-interaction method. Mobile robot learns the motor skills from human demonstration, where robot can imitate human movement to complete complex tasks.

The varying-parameter Lagrange neural network is used to optimize the QP problem of MPC. Compared with previous work, the novel neural network has no matrix inversion, matrix-matrix multiplication or high-order nonlinear computation, which shows better efficiency for computing and can be easily implemented in hardware. Future work includes the implementation of the learning-based control system on our own created real mobile robot. We believe that the application of imitation learning on mobile robot platform, for example, on assistive robots, will bring new interactive experiences for human users.

REFERENCES

[1] E. Theodorou, J. Buchli, S. Schaal, "A generalized path integral control approach to reinforcement learning," *Journal of Machine Learning Research*, vol.11, no.11, pp.3137-3181, 2010.
[2] A. D. Dragan, K. Muelling, J. A. Bagnell, S. S. Srinivasa, "Movement primitives via optimization," *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp.2339-2346, 2015.
[3] Y. Hu, X. Wu, P. Geng, Z. Li, "Evolution Strategies Learning With Variable Impedance Control for Grasping Under Uncertainty," *IEEE Transactions on Industrial Electronics*, vol.66, no.10, pp.7788-7799, 2018.
[4] R. Lioutikov, O. Kroemer, G. Maeda, et al, "Learning manipulation by sequencing motor primitives with a two-armed robot," *Intelligent Autonomous Systems 13. Springer, Cham*, pp.1601-1611, 2016.
[5] Z. Li, T. Zhao, F. Chen, Y. Hu, C. Y. Su, T. Fukuda, "Reinforcement Learning of Manipulation and Grasping using Dynamical Movement Primitives for a Humanoid-like Mobile Manipulator," *IEEE/ASME Transactions on Mechatronics*, vol.23, no.1, pp.121-131, 2018.
[6] M. J. A. Zeestraten, I. Havoutis, J. Silverio, S. Calinon, D. G. Caldwell, "An approach for imitation learning on Riemannian manifolds," *IEEE Robotics and Automation Letters (RA-L)*, vol.2, no.3, pp.1240-1247, June 2017.
[7] K. C. Mbanisi, H. Kimpara, T. Meier, M. Gennert, Z. Li, "Learning Coordinated Vehicle Maneuver Motion Primitives from Human Demonstration," *In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp.6560-6565, October 2018.
[8] Z. Li, C. Y. Su, G. Li, H. Su, "Fuzzy approximation-based adaptive backstepping control of an exoskeleton for human upper limbs," *IEEE Transactions on Fuzzy Systems*, vol.23, no.3, pp.555-566, 2014.
[9] H. Su, C. Yang, G. Ferrigno, E. De Momi, "Improved HumanRobot Collaborative Control of Redundant Robot for Teleoperated Minimally Invasive Surgery," *IEEE Robotics and Automation Letters*, vol.4, no.2, pp.1447-1453, 2019.
[10] B. S. Park, S. J. Yoo, J. B. Park, Y. H. Choi, "A simple adaptive control approach for trajectory tracking of electrically driven nonholonomic mobile robots," *IEEE Transactions on Control Systems Technology*, vol.18, no.5, pp.1199-1206, 2010.
[11] H. Fukushima, K. Muro, F. Matsuno, "Sliding-mode control for transformation to an inverted pendulum mode of a mobile robot with wheel-arms," *IEEE Transactions on Industrial Electronics*, vol.62, no.7, pp.4257-4266, 2015.
[12] Y. Pan, J. Wang, "Model predictive control of unknown nonlinear dynamical systems based on recurrent neural networks. IEEE Transactions on Industrial Electronics," vol.59, no.8, pp.3089-3101, 2012.
[13] Y. Zhang, J. Wang, "A dual neural network for convex quadratic programming subject to linear equality and inequality constraints," *Physics Letters A*, vol.298, no.4, pp.271-278, 2002.
[14] L. Peternel, E. Oztop, J. Babič, "A shared control method for online human-in-the-loop robot learning based on Locally Weighted Regression," *In 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp.3900-3906, October, 2016.
[15] P. Glotfelter, M. Egerstedt, "A Parametric MPC Approach to Balancing the Cost of Abstraction for Differential-Drive Mobile Robots," *In 2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp.732-737, May, 2018.
[16] H. Xiao, Z. Li, C. P. Chen, "Formation control of leader-follower mobile robots' systems using model predictive control based on neural-dynamic optimization," *IEEE Transactions on Industrial Electronics*, vol.63, no.9, pp.5752-5762, 2016.
[17] H. Su, N. Enayati, L. Vantadori, et al, "Online human-like redundancy optimization for tele-operated anthropomorphic manipulators," *International Journal of Advanced Robotic Systems*, vol.15, no.6, 2018.
[18] F. Ke, Z. Li, C. Yang, "Robust Tube-Based Predictive Control for Visual Servoing of Constrained Differential-Drive Mobile Robots," *IEEE Transactions on Industrial Electronics*, vol.65, no.4, pp.3437-3446, 2018.
[19] Z. Yan, X. Le, J. Wang, "Tube-based robust model predictive control of nonlinear systems via collective neurodynamic optimization," *IEEE Transactions on Industrial Electronics*, vol.63, no.7, pp.4377-4386, 2016.

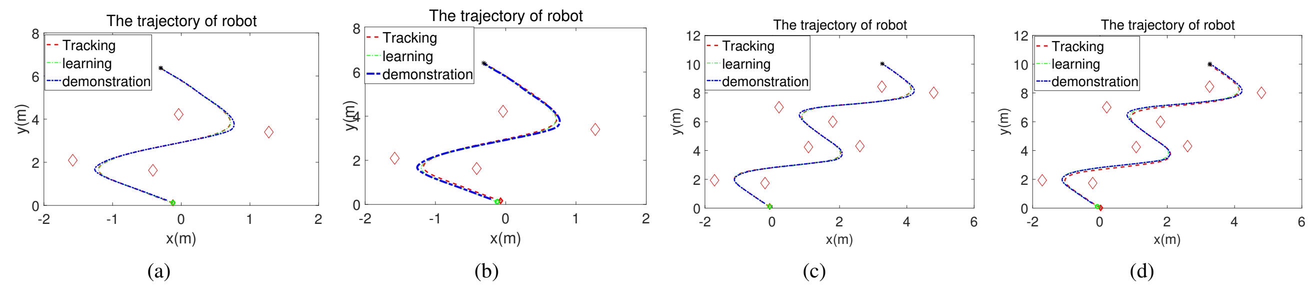

Fig. 4. Imitation results of two different initial positions using nonlinear MPC based on VP-LPNN. (The symbol '◇' represents the obstacles. The red and blue '○' denote the initial positions, and the black '*' denotes the end attractor). (a) Tracking results of 'S'-shape trajectory. (b) Tracking results of 'S'-shape trajectory with different initial positions. (c) Tracking results of 'SS'-shape trajectory. (d) Tracking results of 'SS'-shape trajectory of different initial positions.

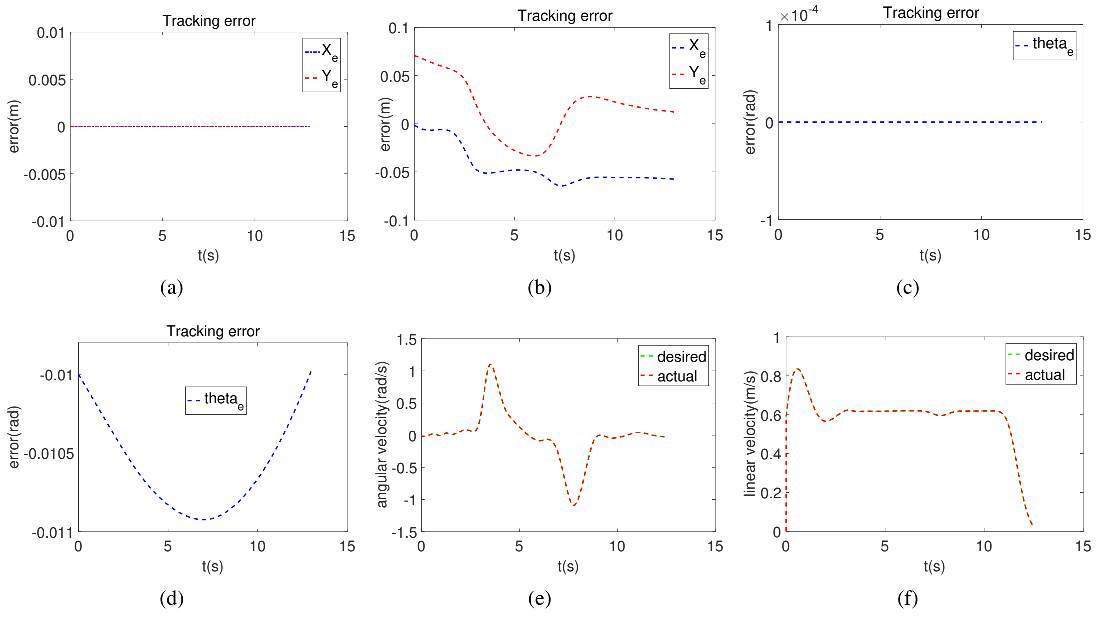

Fig. 5. Results of 'S'-shape trajectory tracking. (a) Tracking errors of X_e and Y_e. (b) Tracking errors of X_e and Y_e of different initial positions. (c) Tracking errors of θ_e. (d) Tracking errors of θ_e of different initial positions. (e) Mobile robot angular velocity ω. (f) Mobile robot linear velocity v.

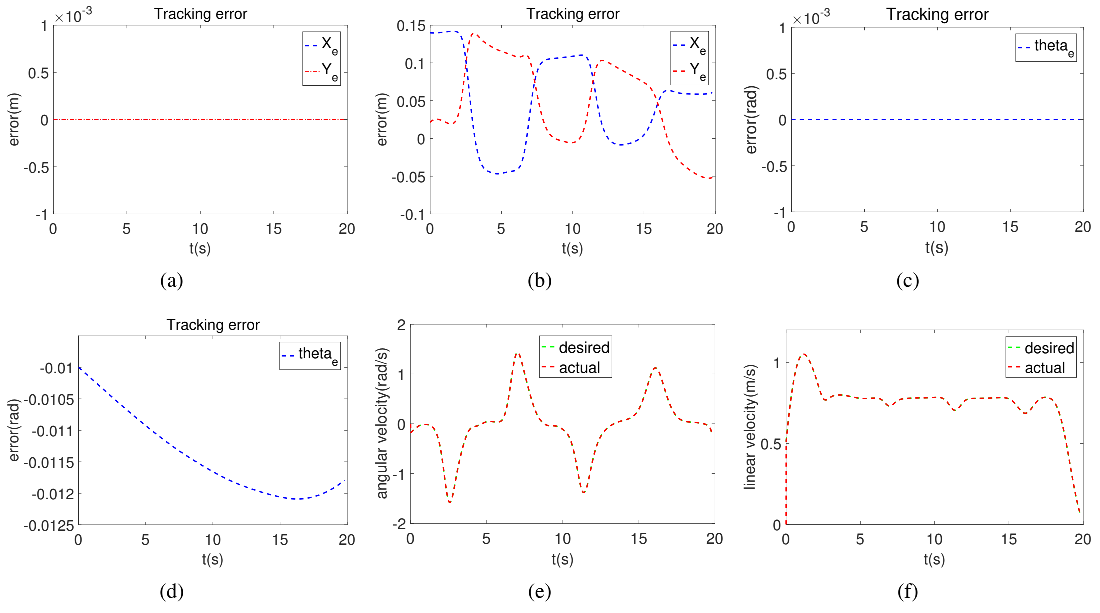

Fig. 6. Results of 'SS'-shape trajectory tracking. (a) Tracking errors of X_e and Y_e. (b) Tracking errors of X_e and Y_e of different initial positions. (c) Tracking errors of θ_e. (d) Tracking errors of θ_e of different initial positions. (e) Mobile robot angular velocity ω. (f) Mobile robot linear velocity v.