Learning Via-Point Movement Primitives with Inter- and Extrapolation Capabilities

You Zhou, Jianfeng Gao and Tamim Asfour

***Abstract*— Movement Primitives (MPs) are a promising way for representing robot motions in a flexible and adaptable manner. Due to the simple and compact form, they have been widely used in robotics. A major goal of the research activities on MPs is to learn models, which can adapt to changing task constraints, e. g. new motion targets. However, the adaptability of current MPs is limited to a small set of constraints due to their simple structures. It is indeed not a trivial task to maintain the simplicity of MPs representation and, at the same time, enhance their adaptability. In this paper, we discuss the adaptability of popular MPs such as Dynamic Movement Primitives (DMP) and Probabilistic Movement Primitives (ProMP) and propose a new simple but efficient formulation of MPs, the Via-points Movement Primitive (VMP), that can adapt to arbitrary via-points using a simple structured model that is based on the previous approaches but outperforms those in terms of extrapolation abilities.**

I. Introduction

Learning from human demonstrations (LfD) is a promising approach for intuitive robot programming [1]. In order to avoid repeated human demonstrations for specific tasks with variable task constraints, different approaches have been proposed to learn generalized representations of actions which allow adaptation to new situations. Due to their simplicity and compact representations, Movement Primitives (MPs) are one of the most used approaches for LfD. The adaptability of MPs is mainly dependent on their structure and basic assumptions. Hence, different types of MPs have been developed to enhance the adaptability and realize intuitive robot behaviors to new task constraints ([2], [3], [4], [5]). Except for the temporal adaptation, the adaptability of MPs can be reflected by their capability of via-points modulation. Via-points represent additional points through which e.g. the robot end-effector has to pass in order to fulfill certain task requirements, which may change also during the execution (see Fig. 1). In this work, we will refer to any newly required intermediate point as well as to new start and goal of the trajectories as via-points.

Regarding the via-points modulation, previous MPs have different limitations because of the assumptions based on which they are developed. For example, Dynamic Movement Primitives (DMP) (proposed and refined in [2], [6], [7]) can only adapt to new starts and goals, but cannot directly handle intermediate via-points. Although Probabilistic Movement Primitives (ProMP) that were introduced in [5] can adapt to via-points within the statistical distribution of the demonstrated trajectories, they generate infeasible motions for via-points beyond the original observation range (see Fig. 5). In many robotic applications, however, the robot is expected to apply the learned MPs for via-points outside the area of demonstrated trajectories. We refer to this as via-points extrapolation, which goes beyond via-points interpolation where the required via-points are in the area of the demonstrated trajectories. The lack of arbitrary via-points modulation limits the usability and adaptability of current MPs.

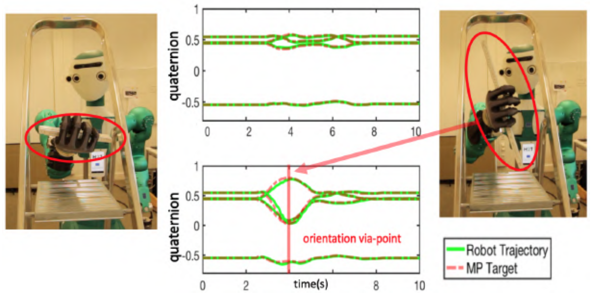

Fig. 1: An orientation via-point is specified to pass a ruler with different length through a narrow area.

In this work, we introduce a new formulation for MPs, the Via-points Movement Primitive (VMP), that can adapt to any via-points and resolve the limitations of previous methods. Based on VMP, we also propose a new learning framework, where the robot memorizes few via-points instead of observing repeated human demonstrations to accommodate new task constraints.

In section II, we compare the adaptability among previous MPs and clarify the reasons of their limitations regarding via-points modulation. In section III we introduce the basic VMP formulation and compare the via-point modulation among DMP, ProMP and VMP. In addition, we provide a task space VMP to allow the integration of via-points defined by both positions and orientations. In section IV, a learning framework based on VMP is presented allowing a robot to efficiently learn motion adaptations from a human teacher when encountering failures. Several robot experiments are conducted to prove the usability of VMP and the importance of via-points modulation in various robotic applications.

II. Representations of Movement Primitives

The work we describe in this paper is concerned with the representation of movement primitives. To clarify the

The research leading to these results has received funding from the European Union's Horizon 2020 Research and Innovation programme under grant agreement No 643950 (SecondHands) and grant agreement No. 731761 (IMAGINE).

The authors are with the Institute for Anthropomatics and Robotics, Karlsruhe Institute of Technology, Karlsruhe, Germany. {you.zhou, asfour}@kit.edu

contribution of our work in terms of via-points adaptability, we provide here an overview on three closely related approaches in the literature: Autonomous Dynamical System (DS), Dynamic Movement Primitive (DMP) and Probabilistic Movement Primitive (ProMP).

a) Autonomous Dynamical System: For the sake of completeness, we introduce DS here, because it is a popular approach for the robot motion generation. As described in [4] and [8], DS assumes that a robot motion is governed by an autonomous dynamical system defined over the robot state space. For trajectories defined in the Cartesian space, DS outputs the translational and rotational velocity for any specific input pose. In the pick place experiment in [9], authors moved the attractor of a DS to let the motion go through the via-points. In fact, this approach can be considered as running a sequence of dynamical systems one after an another. The via-points are indeed the goals of the motions. As shown in [9], DS can adapt to the new goal by changing its attractor, but it cannot directly adapt to the intermediate via-points.

We describe DMP and ProMP in more detail for a better understanding of our approach, which combines their advantages and resolves their limitations.

b) Dynamic Movement Primitive: DMP was originally developed as a deterministic motion representation for convenient speed and goal adaptation in [2]. In [10], it was extended to have probabilistic representations. In this paper, we consider a variant DMP formulation mentioned in [6] and [7] that outperforms the original one for goal adaptations. By reformulating the formulation in the paper, we obtain

$$\begin{aligned} \tau \dot{v} &= K((y_0 - g)x + g + xf(x) - y) - Dv; \\ \tau \dot{y} &= v, \end{aligned} \tag{1}$$

where y and v are the position and velocity, g and y_0 are the goal and start position of the motion. x is the canonical variable that serves as a virtual timer and controlled by a canonical system, which is a linear decay system going from 1 to 0 in this paper. This DMP can be considered as a PD controller with parameters K and D tracking a nonlinear attractor trajectory consisting of a linear trajectory with a constant velocity profile in the canonical space, namely $(y_0 - g)x + g$, and a non-linear force term $f(x)$. The force term $f(x)$ is usually represented by a linear model consisting of a parameter vector $\boldsymbol{w}$ and N radial basis functions (RBF) $\boldsymbol{\psi}(.)$ namely

$$f(x) = \frac{\sum_{i=1}^{N} \boldsymbol{\psi}_i(x)\boldsymbol{w}_i}{\sum_{i=1}^{N} \boldsymbol{\psi}_i(x)}. \tag{2}$$

If and only if K is big enough for perfect tracking, the attractor trajectory coincides with the real trajectory executed. In this case, the non-linear force term $f(x)$ can be regarded as the difference between the demonstrated motion and the linear trajectory. By changing the hyper-parameter g, a single DMP can generate motions which adapt to the new goal. Because DMP is trained on a single demonstration, any new goal refers to a via-point extrapolation. According to [11], it generates a topologically similar trajectory to the original demonstration for the unseen new goals. In [12], the authors proved that DMP minimizes a distance measure between the original trajectory and the adapted one.

As DS, DMP cannot directly adapt to intermediate via-points without extra learning methods. In [13], the authors used reinforcement learning to adjust the goal g and the parameter $\boldsymbol{w}$ of DMP for via-points adaptation. Reinforcement learning for via-points modulation requires a cost function and the process takes usually several iterations. The adapted motions are also difficult to predict. This is inefficient and not practical for many robotic applications. The reason why DMP does not support intermediate via-points is that it learns the attractor trajectory instead of the real trajectory for a PD tracking controller. It is not easy to adjust the motion by manipulating the attractor trajectory. According to [11], DMP usually serves as a kinematic planner that outputs a sequence of desired trajectory points in the Cartesian space, or in the joint configuration space. The robot needs to be equipped with a tracking controller that can accurately track the kinematic plan. For this reason, it is not necessary to learn the attractor trajectory.

c) Probabilistic Movement Primitives: ProMP [5] directly learns the position and velocity profiles of a real trajectory with a linear model and a noise term ϵ

$$y(x) = \boldsymbol{\psi}(x)^T \boldsymbol{w} + \epsilon_y \quad \epsilon_y \sim \mathcal{N}(0, \Sigma_y), \tag{3}$$

where $\boldsymbol{\psi}(.)$ is the radial basis function (RBF) of the canonical variable x. Learning a ProMP means learning the prior probability of $\boldsymbol{w}$ which is assumed to be a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_w, \Sigma_w)$. The likelihood function when observing multiple demonstrations with respect to the parameters $\boldsymbol{w}$ is given by:

$$p(D|\boldsymbol{w}) = \prod_{\xi \in D} p(\xi|\boldsymbol{w}), \quad p(\xi|\boldsymbol{w}) = \prod_{x} \mathcal{N}(y_x|\boldsymbol{\psi}(x)^T\boldsymbol{w}, \Sigma_y), \tag{4}$$

where $\xi \in D$ is a trajectory in the set of demonstrations D. The maximum likelihood estimator (MLE) of the parameters is that

$$\boldsymbol{\mu}_w = \frac{1}{M}\sum_{i=1}^{M} \boldsymbol{w}_i, \quad \Sigma_w = \frac{1}{M}\sum_{i=1}^{M} (\boldsymbol{w}_i - \boldsymbol{\mu}_w)(\boldsymbol{w}_i - \boldsymbol{\mu}_w)^T, \tag{5}$$

where $\boldsymbol{w}_i$ is the parameter vector of the linear model learned to represent the i-th trajectory. This probabilistic representation enables the via-point modulation with the conditional probability of $\boldsymbol{w}$ conditioning on the desired via-point (x^*, y^*) with an uncertainty Σ_y^*. The parameters of this conditional Gaussian distribution are calculated as follows:

$$\begin{aligned} \boldsymbol{\mu}_w^* &= \boldsymbol{\mu}_w + L(y^* - \boldsymbol{\psi}(x^*)^T \boldsymbol{\mu}_w), \\ \Sigma_w^* &= \Sigma_w - L\boldsymbol{\psi}(x^*)^T \Sigma_w, \\ L &= \Sigma_w \boldsymbol{\psi}(x^*)(\Sigma_y^* + \boldsymbol{\psi}(x^*)^T \Sigma_w \boldsymbol{\psi}(x^*))^{-1}. \end{aligned} \tag{6}$$

And the resulting mean trajectory has the form $y(x) = \psi(x)^T \boldsymbol{\mu}_w^*$. ProMP makes a strong assumption that the human demonstrations cover the whole possible workspace for a

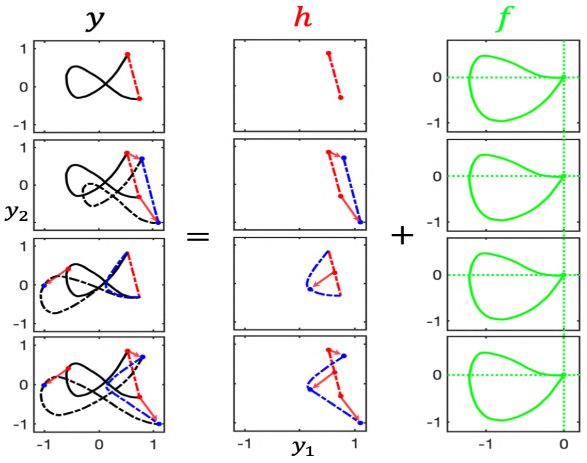

Fig. 2: Via-point modulation by changing the elementary trajectory h. The motion for drawing "α" in a 2D plane is divided into two parts: elementary trajectory h and shape modulation f. **Left:** The VMP is learned by only one demonstration. **Middle:** The via-points modulation is realized by manipulating the elementary trajectory h. **Right:** In this process, the learned force modulation is not changed.

specific task, which is hardly always the case. With a limited number of demonstrations, ProMP often learns an erroneous prior probability of $\boldsymbol{w}$ that works only in a narrow area surrounded by or near to demonstrations. Hence, it might generate infeasible motions when adapting to via-points outside the observation range. In the extreme case that there is only one demonstration, ProMP learns a Gaussian distribution with zero variance $\Sigma_w = 0$. If the goal is changed or any via-points are required to be exactly gone through, which means $\Sigma_y^* = 0$, we encounter a numerical problem when calculating L in the above formulation. Thus ProMP performs even worse than DMP for goal adaptation if only one demonstration is available.

Before introducing VMP, we will clarify the limitation of MP adaptability. In fact, the adaptation of any MP cannot be correct in all cases imaginable. The generated trajectories may not be suitable to accomplish some tasks. DMP assumes that the goal adaptation is changing the attractor of the damped spring system, while ProMP assumes that the adapted trajectory is a sample from a conditional Gaussian distribution. These assumptions may be wrong for specific tasks. We are not attempting to solve this fundamental problem by VMP. But VMP enhances the adaptability of MPs by enabling intermediate via-point modulation and extrapolating to generate feasible trajectories for via-points outside the observation range. Furthermore, we develop a learning framework based on VMP (see section IV) that allows us to teach a robot how to efficiently adapt when encountering failure, which can be a practical solution to the correctness problem.

III. Via-Point Movement Primitive

A. Basic Formulation

Inspired by both DMP and ProMP, VMP consists of two parts, an elementary trajectory h and shape modulation f:

$$y(x) = h(x) + f(x). \tag{7}$$

The shape modulation $f(x)$ is represented by a linear model with RBF $\boldsymbol{\psi}(.)$ and parameters $\boldsymbol{w}$, and a noise term ϵ_f such that

$$f(x) = \boldsymbol{\psi}(x)^T \boldsymbol{w} + \epsilon_f.$$

And we assume that the weights vector of the shape modulation follows a Gaussian distribution as ProMP does. If the variance is zero, the shape modulation is deterministic. The elementary trajectory $h(x)$ directly connects the start and the goal of the demonstrated motion. As an example depicted in Fig. 2, the motion that draws the letter "α" is separated into the linear trajectory h and the shape modulation f. We assume that the elementary trajectory h forms the basic structure for VMP, equivalent to the goal-directed damped spring system of DMP. Similar to the goal adaptation of DMP, manipulating the basic structure h provides a way for via-points modulation (see Fig. 2). Learning VMP is to learn the shape modulation f, or precisely the prior probability distribution of parameters vectors $\boldsymbol{w}$. The mean $\boldsymbol{\mu}_w$ is the representation of the empirical averaged shape modulation. Once the prior probability of $\boldsymbol{w}$ is learned, the probabilistic form of VMP becomes as follows:

$$y(x) \sim \mathcal{N}\left(h(x) + \boldsymbol{\psi}(x)^T \boldsymbol{\mu}_w, \boldsymbol{\psi}(x)^T \Sigma_w \boldsymbol{\psi}(x) + \Sigma_f\right), \tag{8}$$

which provides an another way for the via-point modulation. We calculate the conditional probability of $\boldsymbol{w}$ given an arbitrary via-point (see Eq. 17). In this case, VMP manipulates the shape f of the trajectory to let it go through some given via-points. VMP realizes the via-points modulation by either manipulating the elementary trajectory h or the shape modulation f. Before we discuss the via-points modulation in more detail, it is necessary to know what the elementary trajectory looks like and how to realize intermediate via-point modulation by manipulating $h(x)$.

We represent a via-point with a pair (x_{via}, y_{via}) where x_{via} denotes the canonical value when going through the via-point y_{via}.

B. Elementary Trajectory

Without loss of generality, in this section, we assume that MPs are trained on only one demonstration. As mentioned before, in this case, ProMP cannot adapt to any via point at all. Even though VMP has also zero variance of its shape modulation, it can avoid the numerical problem of ProMP mentioned before by changing only the elementary trajectory for via-points adaptation.

If no intermediate via-points are required, a linear elementary trajectory $h(x)$ directly connects the start and the goal in the spatial domain. Once an intermediate via-point (x_{via}, y_{via}) is required, $h(x)$ will be divided into two segments. One segment guarantees the connection between the previous via-point, which may be the start and the new via-point. And the other one ensures the connection of the new via-point with the next via-point, which may be the goal.

In this paper, we will present two types of trajectory segments. One of them is the linear trajectory with a constant

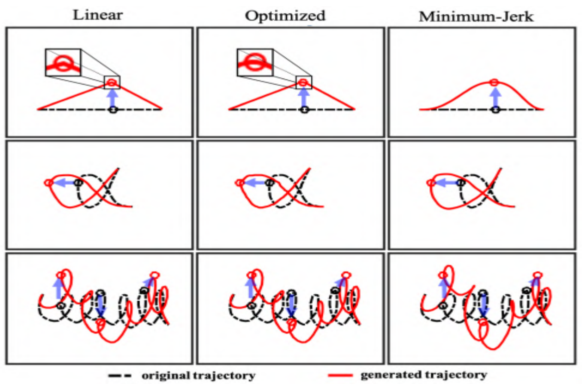

Fig. 3: The black dotted line is the original trajectory and the red line is the trajectory which adapts to the via-points (red circles) that have the same canonical values as the original points (black circles). The linear trajectory segments with a constant velocity generate a sudden acceleration at the via-point. This problem can be solved by an optimization problem defined in Eq. 13 or the minimum-jerk trajectory segments defined in Eq.14. The minimum-jerk trajectories generate more topologically similar trajectory because the velocities and accelerations can also be specified.

velocity profile that has a close relationship with DMP. The other one is the minimum-jerk trajectory that leads to a more flexible way of via-points modulations.

a) Linear trajectory with a constant velocity: A linear trajectory with a constant speed that goes through start $(1, y_0)$ and goal $(0, g)$ has the formula $h(x) = (y_0 - g)x + g$. Hence, the trajectory is represented as follows:

$$y(x) = (y_0 - g)x + g + f(x), \tag{9}$$

which is the same as the attractor trajectory described in the DMP formulation Eq. 1. As mentioned before, the attractor trajectory is similar to the real one if the PD tracking controller formed by DMP has a very high gain K. In this case, DMP is not different from VMP. In addition, the adaptability of DMP to new goals is also realized by changing g in $h(x)$. Hence, there is no difference between DMP and VMP when extrapolating trajectories for new goals.

In order to handle intermediate via-points, we divide the trajectory $h(x)$ into segments to form a piece-wise linear function. For only one via-point, we can have the elementary trajectory as follows:

$$h(x) = \begin{cases} -\frac{(h_{via} - y_0)x}{1 - x_{via}} + y_0 + \frac{(h_{via} - y_0)}{1 - x_{via}} & x \le x_{via} \\ \\ -\frac{(g - h_{via})x}{x_{via}} + g & x > x_{via}, \end{cases} \tag{10}$$

where $h_{via} = y_{via} - f(x_{via})$. This formulation results from the three constraints

$$h(0) = g, \quad h(1) = y_0, \quad h(x_{via}) = y_{via} - f(x_{via}). \tag{11}$$

Theoretically, it can adapt to as many via-points as required by introducing appropriate number of linear segments. However, the connection (turning point) of the piece-wise linear functions introduces potentially high accelerations, which is undesirable when controlling a real robot.

As mentioned before, in [12], the authors proved that DMP described in [6] and [7] gives the same trajectory as the solution of an optimization problem of the form

$$\begin{aligned} \hat{\boldsymbol{\xi}} \quad &= \quad \arg\min_{\boldsymbol{\xi} \in \Xi} ||\boldsymbol{\xi}_D - \boldsymbol{\xi}||_M \\ s.t. \quad & y(1) = y_0, \quad y(0) = g, \end{aligned} \tag{12}$$

where $\boldsymbol{\xi}_D$ is the demonstrated trajectory and $||\boldsymbol{\xi}||_M = \boldsymbol{\xi}^T M \boldsymbol{\xi}$. M is a semi-norm such that $M = A^T A$ with A being the finite differencing matrix. We can extend this problem by introducing the intermediate via-point and obtain the following optimization problem by canceling the shape modulation:

$$\begin{aligned} \hat{\boldsymbol{h}} \quad &= \quad \arg\min_{\boldsymbol{h} \in \mathbf{H}} \tfrac{1}{2} (\boldsymbol{h}_D - \boldsymbol{h})^T M (\boldsymbol{h}_D - \boldsymbol{h}) \\ s.t. \quad & h(1) = y_0, \quad h(0) = g, \\ & h(x_{via}) = y_{via} - f(x_{via}) \end{aligned} \tag{13}$$

The solution to this optimization problem is very similar to the case, where the elementary trajectory is piece-wise linear, but with a smoother turning point. The left and middle figures in Fig. 3 show the difference of their generated trajectories.

b) Minimum-Jerk Trajectory: A minimum-jerk trajectory is the trajectory of the movement that minimizes the third time derivative of the displacement. This trajectory is used extensively to model human reaching movement, largely motivated by [14]. We use a fifth order polynomial to model the minimum-jerk trajectory:

$$y(x) = \sum_{k=0}^{5} a_k x^k + f(x) \tag{14}$$

The parameters a_ks are obtained by the position, velocity and acceleration of the two adjacent via-points such that

$$\begin{aligned} h(x_0) &= y_0 - f_0 \quad & \dot{h}(x_0) &= \dot{y}_0 - \dot{f}_0 \quad & \ddot{h}(x_0) &= \ddot{y}_0 - \ddot{f}_0 \\ h(x_1) &= y_1 - f_1 \quad & \dot{h}(x_1) &= \dot{y}_1 - \dot{f}_1 \quad & \ddot{h}(x_1) &= \ddot{y}_1 - \ddot{f}_1. \end{aligned} \tag{15}$$

(x_0, y_0) and (x_1, y_1) are two adjacent via-points. $f_0 = f(x_0)$ and $f_1 = f(x_1)$ represent the shape modulation. The minimum-jerk trajectory segment is more powerful than the previous one because the velocities and accelerations at via-points can also be specified. For via-point modulation, thus, we can obtain a more similar trajectory when matching the velocity at the via-point with the corresponding one in the demonstrated trajectory. Moreover, the smoothness around the via-points can also be guaranteed by equating the velocities at the via-point of two connected trajectory segments(see Fig. 3).

Manipulating the elementary trajectory $h(x)$ enables the via-point modulation in areas outside the demonstrated trajectories. Note again that ProMP is not able to adapt to anything in the case of single demonstration. Hence, it is not necessary to compare ProMP with VMP in this section. In the next section, we will introduce the via-points modulation strategy of VMP by also considering its probabilistic representation, and compare VMP with ProMP in detail.

C. Via-Points Modulation

As DMP, VMP assumes that a goal-oriented basic structure always exists. As ProMP, VMP also supposes that the task-specific trajectories follow a Gaussian distribution that is, however, only valid near the demonstrations. VMP combines strategies from both DMP and ProMP for via-point modulation and determines when to manipulate $h(x)$ or $f(x)$ by calculating the probability density of the desired via-point given the current learned prior probability of $\boldsymbol{w}$:

$$p(y_{via}|\boldsymbol{\mu}_w, \Sigma_w) = \mathcal{N}(y_{via}; h(x_{via}) + \boldsymbol{\psi}(x_{via})^T\boldsymbol{\mu}_w, \\ \boldsymbol{\psi}(x_{via})^T\Sigma_w\boldsymbol{\psi}(x_{via}) + \Sigma_{via}) \quad (16)$$

Here, (x_{via}, y_{via}) is the required via-point with the uncertainty Σ_{via}. If this conditional probability is more than a specific threshold η, VMP will manipulate the shape modulation $f(x)$ with the conditional distribution parameters:

$$\begin{aligned} \boldsymbol{\mu}_w^* &= \boldsymbol{\mu}_w + L(y_{via} - h(x_{via}) - \boldsymbol{\psi}(x_{via})^T\boldsymbol{\mu}_w), \\ \Sigma_w^* &= \Sigma_w - L\boldsymbol{\psi}(x_{via})^T\Sigma_w, \\ L &= \Sigma_w\boldsymbol{\psi}(x)(\Sigma_{via} + \boldsymbol{\psi}(x)^T\Sigma_w\boldsymbol{\psi}(x))^{-1}. \end{aligned} \quad (17)$$

If the conditional probability is less than η, which means that the desired via-point is unlikely based on the learned prior probability of $\boldsymbol{w}$, VMP will manipulate the elementary trajectory $h(x)$ as discussed in the subsection III-B.

To compare VMP with ProMP, we first consider a simple case where a new goal out of range of observations is required. ProMP sticks to the low variance region, while VMP manipulates the elementary trajectory $h(x)$ to translate the whole trajectory distribution because $p(g_{new}|\mu_w, \Sigma_w) < \eta$ (see Fig. 4). In order to guarantee that the sampled trajectory goes through the new goal, VMP further manipulates the shape modulation $f(x)$ only to reduce the variance of the trajectories. VMP results in a trajectory with a similar shape to the original one. In contrast, ProMP generates infeasible motions because the learned prior probability is not suitable for required via-points any more.

In Fig. 5, we compare further the behavior of VMP with ProMP. In this case, both are trained on drawing eight figures. Two different intermediate via-points are required separately during two motion executions. For the via-point that is surrounded by the learned trajectory distribution where $p(y_{via}|\mu_w, \Sigma_w) > \eta$, VMP behaves the same as ProMP (blue curves). For the via-point that $p(y_{via}|\mu_w, \Sigma_w) < \eta$, VMP manipulates the elementary trajectory $h(x)$ and generates a scaled figure eight that goes through the via-point (see the green curves). In contrast, ProMP fails to generate any reasonable motions (see the most left-bottom figure). The parameter η is manually designed here. It is task specific to determine whether a motion generated by ProMP for via-points adaptation is good or not. Hence, it is not trivial to automatically decide η in a general way. In the extreme case, if it is very small, the behaviour of VMP has no big difference from ProMP. If it is very big, all via-points are adapted by adjusting the elementary trajectory, which can still meet requirements in many applications. With an η in the middle, we show that VMP is indeed a strategy combining ways of both ProMP and DMP for via-points adaptation.

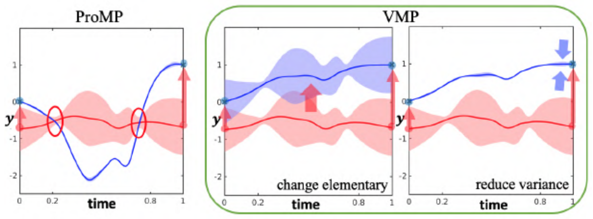

Fig. 4: Left: ProMP generates trajectories going through low variance region (red circle). **Right:** VMP adapts to the goal with two steps. Firstly it changes elementary trajectories (see the middle plot) and then reduces the variance (see the most right plot).

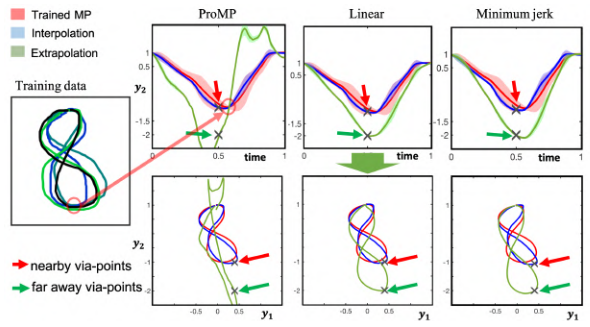

Fig. 5: Comparison between VMP (with $\eta = 0.8$) and ProMP for drawing a figure eight. We consider three cases: without via-points (red curves), with a via-point within the observation range (blue curves) and with a via-point outside the range (green curves). The **top-right** plots show the vertical axis values along the time line. The **bottom-right** plots show the result motions. Different elementary trajectory segments generate slightly different trajectories.

In the top row of the Fig. 5, we depict the trajectories for the vertical dimension in the time domain. It is clear to see that ProMP sticks to the region with a low variance. If carefully inspecting the training data, it is easy to find that the region of the low variance does not have any special meaning. It corresponds to a point at the bottom of figure eights which is coincidentally gone through by all demonstrations. This causes the erroneous prior probability learned in ProMP. In fact, it is to be expected to have an erroneous prior probability if we assume that all task-specific trajectories follow a Gaussian distribution because the number of demonstrations for a real robot application is always very limited and human demonstrations usually cannot cover all possible situations. In this case, VMP assumes that there is a goal-directed trajectory that stabilizes the generated motion as DMP does. Although it is still difficult to guarantee the correctness, VMP generates trajectories with more similar shapes to the original ones than ProMP does.

It is possible that the low variance region learned from human demonstrations is essential to accomplish the task, such as for avoiding some fixed obstacles. In this case, the way to generate motions with VMP for a far away via-point might be erroneous, while ProMP can still meet some hard constraints, even though it provides motions with unnatural shapes. In practice, this essential region is better guaranteed by a via-point instead of a low variance region which can still be violated if the variance is not exactly zero and some other hard task constraints are required.

In order to use VMP for robotic applications, encoding the position trajectories is not enough. In the next section, we propose a way to handle orientation and realize the task space VMP considering the via-point with both position and orientation.

D. Task Space VMP

In the task space MP, 3D position trajectories can be separately encoded and coupled by a canonical system. However, it is not good to split the orientation into separate dimensions due to the fact that these dimensions are affected by each other and can only be controlled in a coherent way. First, we consider the case of the deterministic VMP and use the same symbols h and f to denote orientation trajectories. Compared to Eq. 7 for the position trajectory, the orientation trajectory of VMP is defined by

$$y(x) = h(x) * f(x) \tag{18}$$

where $*$ is the quaternion multiplication. In order to encode the orientation trajectory, we need to first define the elementary trajectory $h(x)$. For the position, we showed that the elementary trajectory consisting of segments that directly connect two adjacent via-points is an intuitive choice. For orientation, we also need to have an intuitive trajectory segments that connect two adjacent via-point orientations. This trajectory can be obtained by the spherical linear interpolation (SLERP):

$$h(x) = \frac{sin((1-\alpha(x))\beta)}{sin(\beta)} q_0 + \frac{sin(\alpha(x)\beta)}{sin(\beta)} q_1, \tag{19}$$

where $cos(\beta) = q_0 \cdot q_1$ (here, $\cdot$ is the dot product of quaternion elements). q_0 and q_1 are calculated by

$$q_0 = y_0 * f(x_0)^{-1}, \quad q_1 = y_1 * f(x_1)^{-1}, \tag{20}$$

where (x_0, y_0) and (x_1, y_1) are two adjacent orientation via-points. $\alpha(x)$ is the ratio of SLERP ranging from 0 to 1. By controlling the ratio α, we can have different velocity profiles. For example, the constant velocity profile of rotation is realized by $\alpha(x) = \frac{1}{x_1 - x_0}(x - x_0)$. And a bell-shaped velocity profile corresponding to the minimum-jerk trajectory for position can be implemented by $\alpha(x) = \sum_{k=1}^{5} a_k (\frac{x-x_0}{x_1-x_0})^k$. All results obtained from the discussion of elementary trajectories in the subsection III-A can be applied here. It is trivial to manipulate the elementary trajectory $h(x)$ for via-point modulation.

For the shape modulation, we use three linear model to represent the trajectories for q_x, q_y and q_z separately. The first component q_w is calculated by the fact that a unit quaternion is required. The initial value of q_w is determined by the start quaternion, and we track its value continuously to guarantee that there is no jump caused by the antipodally symmetric property. For the probabilistic formulation of the weights, one simple way is to assume that the components are independent and identically distributed, then we have a three dimensional gaussian distribution to represent the shape modulation. We can use the same strategy to calculate the conditional probability for orientation via-points(see Eq. 16).

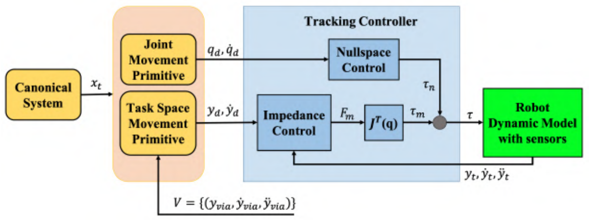

Fig. 6: The VMP control framework is shown here.

In reality, it is less accurate and more complicated to calculate the conditional probability for quaternion shape modulation parts than directly integrating via-points into the elementary trajectories.

IV. VMP for Robot Applications

In this section, we present different robot applications with VMP and show the necessity of via-points modulation. First, a learning framework based on VMP is introduced. Compared to DMP and ProMP, VMP provides an efficient way to learn from human when encountering the failure. Then, we describe the return property of VMP, which is very practical and useful in the robot application. Finally, we show how to use VMP for obstacle avoidance.

We trained the robots (such as ARMAR-6 [15]) with multiple demonstrations, many times with only one single demonstration in the kinesthetic manner. In all the following experiments, we integrated via-points into the elementary trajectory, because it is the main advantage of VMP over others.

The robot learned both task space and joint space VMP. The task space VMP provides a sequence of desired positions and velocities $(y_d, \dot{y}_d)$, while the joint space VMP outputs desired joint values $(q_d, \dot{q}_d)$, that are used for the nullspace control. The generalized force F_m given by the impedance control based on the desired task space positions is finally transferred to the torque command τ with inverse dynamics (see Fig. 6).

A. VMP based Learning by Demonstrations

As mentioned before, it is hard to always generate correct motions for via-points modulation. Hence, we need a robotic learning framework which can not only generalize its learned motions, but can also learn from human when encountering the failure. The traditional way to accommodate new task constraints that cannot be handled by the current learned DMP/ProMP is to ask human to demonstrate a new movement which the robot can follow to fulfill the task. However, this process results in multiple tedious and repeated demonstrations for even a simple task and a large number of MPs stored for a specific skill. Instead of a new trajectory and multiple MPs, VMP requires only one MP with a few via-points to solve the new task constraints.

In reality, we have experience when learning a skill from experts, who sometimes prefer adjusting our pose (via-points) instead of showing us the whole motion again and again. During the pose adjustment, the experts can also take individual difference into the consideration. Thus,

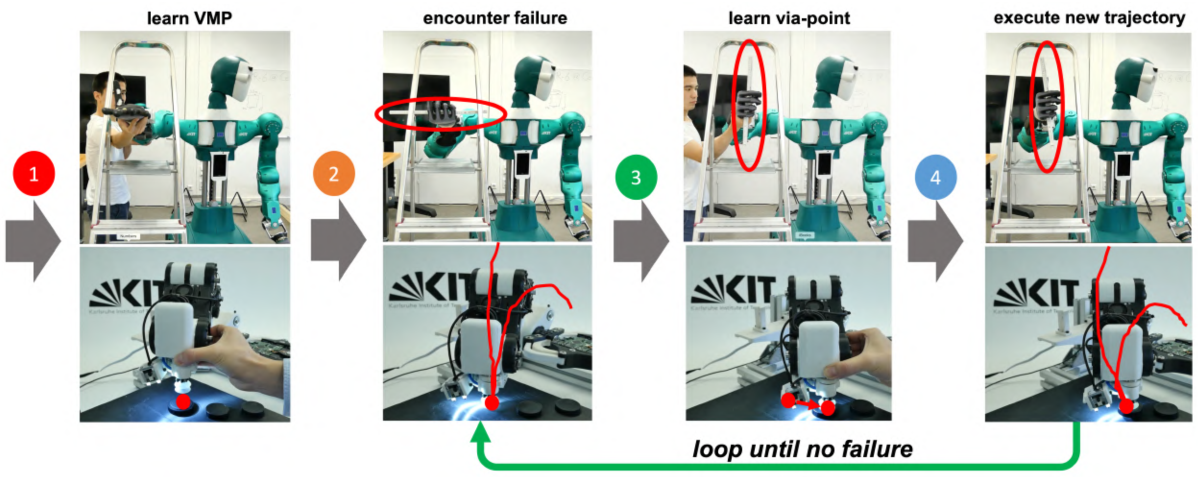

Fig. 7: The robot 1) learns a motion by kinesthetic teaching, and reproduces the motion. The robot 2) encounters a failure. 3) A via-point is memorized. Finally, the robot 4) succeeds with a new trajectory generated by VMP with a via-point. The first row shows the experiment with ARMAR-6 and the second row is for the sucker gripper.

remembering some important via-points is much simpler and sometimes essential for learning a skill. Based on VMP, we present a concrete learning framework shown in Fig.7, which enables the robot learning from the expert when encountering the failure.

The top row of the Fig. 7 shows that the learning framework is used for obstacle avoidance with different task constraints. The learned motion for passing a shorter ruler does not succeed for a longer one. After the failure is detected based on the force sensor, the human shows the robot a via-point with both position and orientation, which leads to the success. With DMP or ProMP, however, new MPs are required, which is less efficient than memorizing a via-point.

One more example is shown in the bottom row of the Fig.7 where a suck gripper learns how to move an object from one position to the other. The whole motion that is encoded by only one VMP includes both reaching and transporting the object. After demonstrating how to move the first object, the gripper tries to move the second one that is located at a different position and fails. The human shows the correct via-point that the gripper should go through. With VMP, the problem is solved by only one via-point. With DMP, however, we need to either encode the motion with two separate parts (one goes to the location of the object, the other goes to the position to release the object) or train a new DMP for the second object that is located at a different place. With ProMP, we need multiple demonstrations which should cover the whole workspace surrounding the possible locations of the target object.

B. Return Property of VMP

VMP has one more important benefit, namely return property, that fully utilizes the knowledge learned from the demonstrations and releases the burden of a local motion planner that designs the via-points or an expert who demonstrates the via-points. If we set a via-point on the original demonstrated mean trajectory, the rest of the trajectory will

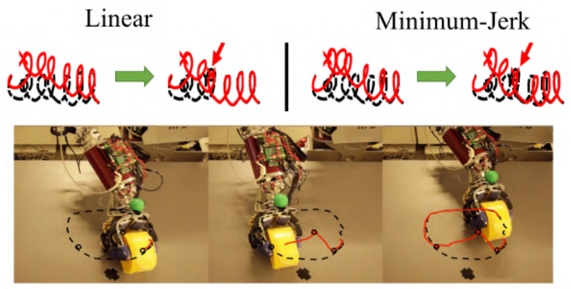

Fig. 8: Top: a via-point on the demonstrated trajectory (red arrows) guides the generated trajectory back to the original motion. **Bottom**: example with the humanoid ARMAR-IIIb [16]. Three via-points are used to avoid the black tape. Two of them are on the demonstrated trajectory (the black dotted line). The last one ensures that the movement after obstacle avoidance is the same as the original motion.

coincide with the demonstrated one. In Fig. 8, we illustrate the return property. Fig. 9 shows an another example of how to use this property to guarantee the safety of the motion. For robot learning by demonstration, no cost function exists to keep the robot away from failures like in reinforcement learning. Hence, if no special requirements such as new via-points exist in the rest of the trajectory, it is beneficial to keep the generated trajectory near to the demonstrated one in order to reduce the risk of failure. More failures require a better motion planner or more human demonstrations for via-points. Therefore, one extra via-point on the demonstrated trajectory for the return property of VMP is sometimes very useful, especially for obstacle avoidance. This return property only works when the elementary trajectory is adapted for new via-points.

C. Obstacle Avoidance

In order to be able to use VMP to avoid obstacles, a learning framework mentioned before or a local motion planner can be used. Flexibly integrating via-points connects MP with the task space motion planning.

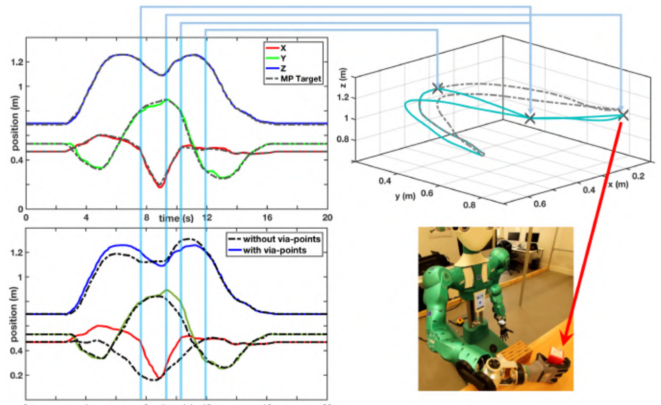

Fig. 9: Four via-points are integrated to a learned VMP to avoid an obstacle during grasping. The **top-left** plot shows the tracking accuracy. The **bottom-left** plot shows the difference between VMP with (colored) and without (dotted) via-points. Four via-points are corresponding to the black crosses in the **top-right** diagram. One of the via-point specifies the grasping pose shown in the **bottom-right** image.

Fig. 1 shows that an orientation via-point is integrated to avoid collision. This valid orientation via-point is easily found when the robot can observe the boundary of the ladder as the narrow area. Fig. 9 shows that four via-points are used to generate a collision-free trajectory. One via-point is integrated twice before and after grasping to avoid the obstacle. The location of this via-point is simply calculated by adding a certain offset from the obstacle. Grasping point is an another via-point chosen based on the robot manipulability when approaching the target object. The last via-point pulls the robot end-effector back to the original learned trajectory to ensure the safety when retracting the hand back from above the table. This benefits from the return property mentioned in the last section. VMP can be combined with some task space motion planner and provides an another way to avoid collisions. Note that the capability of via-points modulation is not restricted to obstacle avoidance as shown in Fig. 7, but provides a more compact representation of a specific skill.

V. Conclusion and Future Works

In this paper, we proposed a new formulation for movement primitive, the Via-points Movement Primitive (VMP), which inherits the advantages from both DMP and ProMP, and relax their limitations, and a VMP based learning framework that allows efficient teaching the robot how to accommodate new task constraints when encountering failure with generalized motions. While DMPs cannot directly adapt to intermediate via-points due to the underlying dynamical system, ProMPs sometimes generate motions that are less similar to the demonstrated ones. The main advantage of our VMP representation is the adaptation to arbitrary via-points by changing the elementary trajectory. Note that VMP is different from some geometry and optimization based methods that generate trajectories going through pre-defined via-points as e. g. described in [17]. The latter one focuses more on accurate human motion modeling, while the via-points for VMP are usually task specified. Further, VMPs, similar to DMPs, can be combined with extra learning algorithms such as supervised learning for motion generalization and reinforcement learning for motion refinement. In the future works, we consider also learning how to automatically set via-points and the threshold η for specific tasks.

References

[1] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, *Robot Programming by Demonstration*. Springer Berlin Heidelberg, 2008, pp. 1371–1394.

[2] A. Ijspeert, J. Nakanishi, and S. Schaal, “Learning attractor landscapes for learning motor primitives,” in *Advances in Neural Information Processing Systems 15*, 2003, pp. 1547–1554.

[3] 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. Special issue on robot learning by observation, demonstration and imitation*, vol. 37, no. 2, pp. 286–298, 2007.

[4] E. Gribovskaya, S. Khansari-Zadeh, and A. Billard, “Learning nonlinear multivariate dynamics of motion in robotic manipulators,” *The International Journal of Robotics Research*, vol. 30, no. 1, pp. 80–117, 2011.

[5] A. Paraschos, C. Daniel, J. R. Peters, and G. Neumann, “Probabilistic movement primitives,” in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2013, pp. 2616–2624.

[6] 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)*, Kobe, Japan, 2009.

[7] 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 *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2009, pp. 1534–1539.

[8] S. Khansari-Zadeh and A. Billard, “Learning stable nonlinear dynamical systems with gaussian mixture models,” *Robotics, IEEE Transactions on*, vol. 27, no. 5, pp. 943 –957, 2011.

[9] M. Khoramshahi, A. Laurens, T. Triquet, and A. Billard, “From human physical interaction to online motion adaptation using parameterized dynamical systems,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2018, pp. 1361–1366.

[10] F. Meier and S. Schaal, “A probabilistic representation for dynamic movement primitives,” *ArXiv*, 2016.

[11] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, “Dynamical movement primitives: Learning attractor models for motor behaviors,” *Neural Comput.*, vol. 25, no. 2, pp. 328–373, Feb. 2013.

[12] A. D.Dragan, K. Muelling, J. Bagnell, and S. S.Srinivasa, “Movement primitives via optimization,” in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 2339–2346.

[13] F. Stulp, E. Theodorou, M. Kalakrishnan, P. Pastor, L. Righetti, and S. Schaal, “Learning motion primitive goals for robust manipulation,” in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sep. 2011, pp. 325–331.

[14] N. Hogans, “An organizing principle for a class of voluntary movements,” *Journal of neuroscience*, vol. 4, pp. 2745–54, 1984.

[15] T. Asfour, L. Kaul, M. Wachter, S. Ottenhaus, P. Weiner, S. Rader, R. Grimm, Y. Zhou, M. Grotz, F. Paus, D. Shingarey, and H. Haubert, “Armar-6: A collaborative humanoid robot for industrial applications,” in *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, 2018.

[16] T. Asfour, K. Regenstein, P. Azad, J. Schröder, and R. Dillmann, “ARMAR-III: A Humanoid Platform for Perception-Action Integration,” in *2nd International Workshop on Human-Centered Robotic Systems (HCRS)*, Munich, Germany, 2006.

[17] Y. Meirovitch, D. Bennequin, and T. Flash, “Geometrical invariance and smoothness maximization for task-space movement generation,” *IEEE Transactions on Robotics*, vol. 32, no. 4, pp. 837–853, Aug 2016.