Fourier movement primitives: an approach for learning rhythmic robot skills from demonstrations

Thibaut Kulak
Idiap Research Institute
thibaut.kulak@idiap.ch

João Silvério
Idiap Research Institute
joao.silverio@idiap.ch

Sylvain Calinon
Idiap Research Institute
sylvain.calinon@idiap.ch

***Abstract*—Whether in factory or household scenarios, rhythmic movements play a crucial role in many daily-life tasks. In this paper we propose a Fourier movement primitive (FMP) representation to learn such type of skills from human demonstrations. Our approach takes inspiration from the probabilistic movement primitives (ProMP) framework, and is grounded in signal processing theory through the Fourier transform. It works with minimal preprocessing, as it does not require demonstration alignment nor finding the frequency of demonstrated signals. Additionally, it does not entail the careful choice/parameterization of basis functions, that typically occurs in most forms of movement primitive representations. Indeed, its basis functions are the Fourier series, which can approximate any periodic signal. This makes FMP an excellent choice for tasks that involve a superposition of different frequencies. Finally, FMP shows interesting extrapolation capabilities as the system has the property of smoothly returning back to the demonstrations (e.g. the limit cycle) when faced with a new situation, being safe for real-world robotic tasks. We validate FMP in several experimental cases with real-world data from polishing and 8-shape drawing tasks as well as on a 7-DoF, torque-controlled, Panda robot.**

I. Introduction

Upper-body rhythmic movements play a crucial role in many daily-life tasks. Whether in factory scenarios (e.g. polishing, sawing) or household (e.g. whisking, hammering, wiping), such tasks require the use of repetitive patterns that should adapt to new situations. As opposed to discrete motions (e.g. reaching, picking, batting), where the final location is typically used as the parameter to adapt the task, rhythmic skills contain richer information pertaining to aspects like frequency, amplitude and phase, which can strongly depend on various types of inputs, such as the task context (e.g. wiping a small or large surface). The high number of aspects that need to be accounted for in rhythmic motions make them hard to pre-program. We propose to rely on learning from demonstration (LfD) [4] to learn these rich features.

The problem of learning rhythmic robot skills from demonstrations has received previous attention from the community, especially in the context of wiping/polishing tasks [3, 2, 15, 1], with results along two major research lines. The first one relies on dynamical system representations, through the popular dynamic movement primitives (DMP) [13]. Indeed, extensions of the original DMP [12, 8, 23, 7, 21] have exploited either periodic basis functions or non-linear oscillators to encode demonstrated robot motions. The second, and more recent, line of research leverages probabilistic approaches, either using probabilistic movement primitives (ProMP) [19] or kernelized movement primitives (KMP) [10]. In all cases, sinusoidal basis functions are used, capturing the periodic aspect, but limiting the applicability in cases of varying amplitude, frequency and phase (Section II).

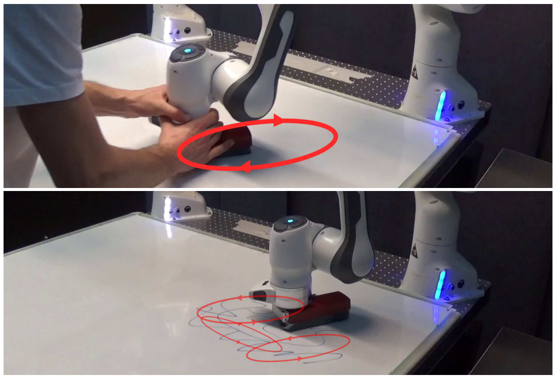


Fig. 1: Rythmic tasks such as wiping need to be demonstrated in few demonstrations (**top**), while the robot should extract the important motion features (e.g. amplitude, frequency and phase) and generalize it in a consistent, safe manner (**bottom**).

Fourier series have been used extensively during the past decades for synthesis and analysis of periodic signals (Section III). We here propose to leverage them in the context of LfD. The contribution of this paper is a model for learning rhythmic skills from demonstrations and adapting them to new situations based on Fourier movement primitives (FMP). We propose FMP as a movement primitive representation that relies on a superposition of *Fourier basis functions* (Section IV), or complex exponentials, as opposed to the typical choice of real-value sine/cosine basis functions. The main advantages of FMP over the state-of-the-art are:

1) **Extraction of multiple frequencies underlying demonstrations** - by relying on Fourier series as a basis representation, FMP can extract the superposition of various frequencies in a straightforward manner.
2) **No manual choice/tuning of the basis functions** - Fourier basis functions do not require hyperparameters, in contrast to Von-Mises or sinusoidal basis functions requiring centers, bandwidths and frequencies parameters. The use of the Fourier basis functions is also well motivated theoretically, as any periodic signal can be represented in the Fourier domain.
3) **Minimal preprocessing** - FMP requires very little pre-

processing. Namely, it does not require the demonstrations to be aligned, or the basis frequency of the signal to be identified.

4) **Unified magnitude and phase statistics** - the underlying processing with complex numbers allows the system to achive a statistical analysis over **amplitude**, **frequency** and **phase**.

We evaluate FMP in 3 different scenarios (Section V). First we consider data from a polishing task, requiring one single frequency per degree-of-freedom (DOF). Second we consider the drawing of an 8-shape, which needs a superposition of different frequencies. Finally, we use a 7-DOF Panda robot to perform a whiteboard-wiping task, showing that the robot can start from arbitrary locations in the workspace while smoothly converging to the demonstrations and perform the task. We close the paper with a discussion on the obtained results (Section VI) and conclusion (Section VII).

II. Related work

In this section we review related work on the representation and learning of periodic movement primitives by imitation and place our contribution in the context of the state-of-the-art.

A. Dynamical-system-based approaches

A prominent line of research based on dynamical systems stems from the seminal work of Ijspeert et al. [13] on DMP. The original DMP formulation [13] relies on simple second order dynamics to learn point-attractor movements, while exhibiting interesting properties such as convergence to a desired final state and resistance to perturbations. Owing to a non-linear term that shapes the dynamics, DMP can imitate the shape of demonstrations in a straightforward way. It can be used for both discrete and periodic movements [12], by considering non-linear oscillators and phase dynamics. Following from these results, more complex paradigms in robotics emerged, such as *central pattern generators* [6] and *adaptive frequency phase oscillators* [26].

In [8], Gams et al. exploit the capabilities of adaptive frequency oscillators proposed by Righetti et al. [26] in combination with periodic DMP. They propose a two-layered approach that relies firstly on a set of adaptive frequency oscillators to identify the *fundamental* frequency and phase of a demonstrated signal without prior knowledge of its frequency. In a second layer, a periodic DMP is trained using the previously extracted fundamental frequency and phase, to obtain the waveform of the signal, allowing for reproducing the skill with the aforementioned DMP properties. This approach has been further utilized by others in task generalization [29], human-robot collaboration [21, 22, 24], force control [9] and improved for automatic frequency extraction [23]. These approaches share the limitation that it is not straightforward to perform statistics on the learned model when there is access to multiple demonstrations. This consequently limits the potential of application in compliant control, especially at the level of *minimal intervention control* [18, 5, 27]. Compliant control is possible using such kind of dynamical systems, however the control policies do not reflect the structure of the data and are typically modulated by external signals, such as EMG [21].

Finally, [1, 16, 15] propose to use *autonomous* dynamical systems to learn polishing tasks, relying on formulations that share similarities with [14]. In these works, learning is done to the extent that the robot extracts surface normals [1] and adapts its behavior to new human intentions (either through different limit cycles [16] or task switches [15]). We, instead, focus on the learning of the spatiotemporal aspects of demonstrations, namely magnitude, frequency and phase.

B. Probabilistic approaches

While probabilistic approaches for motor primitive learning by imitation rose in popularity, two lines of approaches gain particular relevance for rhythmic skills. Paraschos et al. [19] propose a formulation relying on basis functions that can represent either discrete or periodic motions, named probabilistic movement primitives (ProMP). ProMP represents trajectories in a weight space where, provided enough data, statistics can be performed yielding important properties such as adaptation to new situations and the computation of various forms of uncertainties. By relying on cosine or Von-Mises basis functions, however, ProMP has limited adaptation capability in terms of frequency and phase.

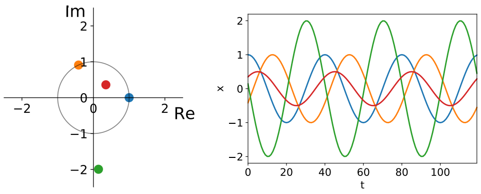
In another direction, following the spirit of non-parametric learning, Gaussian process regression (GPR) can also model periodic time series (see [25] ch. 4), and hence can also approximate well rhythmic robot skills, by relying on appropriate kernels. However, it is computationally expensive and it is not straightforward to adapt a demonstrated policy to a new situation. More recently, kernelized movement primitives (KMP) [10, 11] have been shown to permit the learning of periodic skills when using periodic kernel functions. Nonetheless, both KMP and GPR, despite allowing for statistics, share the same limitations as ProMP in that the kernels conventionally employed are not expressive enough to represent a wide range of frequencies and phases.

C. Constraint-based approaches

A third relevant line of research focuses on learning motion constraints [3, 2, 17] through the estimation of null space matrices from data. While [3, 17] perform polishing/wiping on flat surfaces, [2] extend the approach to be compatible with curved surfaces (which is also the motivation behind [1]). Similarly to [1, 16, 15], the focus is not on the learning of rhythmic motion primitives, hence application to tasks involving periodic motions (e.g. drumming, hammering) is not straightforward. However, these approaches rely on policy learning for generalizing the learned constraints. Hence, there is a high potential for combinations with FMP in the future.

III. Preliminaries

We briefly recall the concepts of discrete Fourier transform and inverse discrete Fourier transform, which are used to convert sequences from time domain to frequency domain, and the other way around.



(a) Weight in the complex plane

(b) Corresponding basis function multiplied by weight

Fig. 2: Illustration of reconstructed signals with one Fourier basis function (for $k = 3$). The orange and blue points have the same amplitude but not the same phase, which results in the same signals that are shifted in time.

A. Discrete Fourier transform

The discrete Fourier transform converts a one-dimensional sequence $\boldsymbol{y} = [y_0, \ldots, y_{T-1}]^\top$ of T equally-spaced samples into a same length sequence of complex coefficients corresponding to different frequencies. The basic idea is to consider the sequence $\boldsymbol{y}$ as a periodic signal of period T [1]. The sequence can be perfectly represented in the frequency domain with T complex coefficients:

$$\forall k \in [\![0; T-1]\!] : \tilde{w}_k = \sum_{n=0}^{T-1} y_n \exp\left(-\frac{2i\pi}{T}kn\right), \tag{1}$$

where i refers to the imaginary part of a complex number. By concatenating the T coefficients in a vector, we get the following matrix-form formula:

$$\begin{aligned} &\tilde{\boldsymbol{w}} = \boldsymbol{\Psi}\boldsymbol{y} \quad \text{with:} \\ &\forall (k,n) \in [\![0;T-1]\!]^2 : \Psi_{k,n} = \exp\left(-\frac{2i\pi}{T}kn\right). \end{aligned} \tag{2}$$

B. Inverse discrete Fourier transform

The discrete Fourier transform is an invertible, linear transformation. Therefore, we can map the frequency domain representation of the signal back to the time domain:

$$\forall n \in [\![0; T-1]\!] : y_n = \frac{1}{T}\sum_{k=0}^{T-1} \tilde{w}_k \exp\left(\frac{2i\pi}{T}kn\right). \tag{3}$$

This can also be expressed in matrix form as:

$$\boldsymbol{y} = \tilde{\boldsymbol{\Phi}}\tilde{\boldsymbol{w}} \quad \text{with} \quad \tilde{\boldsymbol{\Phi}} = \frac{1}{T}\boldsymbol{\Psi}^H, \tag{4}$$

where H denotes the Hermitian transpose operator. An interesting property of Fourier basis functions is that a single basis function represents variations of amplitude and phase, as we illustrated in Fig.2.

IV. Fourier movement primitives

In this section, we present Fourier movement primitives. First, we detail how we can compute statistics from demonstrations, then we explain how this is exploited for minimal intervention control in the Fourier domain.

A. Imitation learning

Let $(\boldsymbol{y}_i)_{i=1,\ldots,N}$ be a series of N demonstrations of length T. For clarity purposes, we assume that the demonstrations contain only one degree of freedom (we will discuss in subsection (IV-D) how it is extended to multiple ones). We compute using (2) the complex weights $(\tilde{\boldsymbol{w}}_i)_{i=1,\ldots,N}$ such that

$$\forall i \in [\![1; N]\!] : \boldsymbol{y}_i = \tilde{\boldsymbol{\Phi}}\tilde{\boldsymbol{w}}_i. \tag{5}$$

We then learn a distribution of $(\tilde{\boldsymbol{w}}_i)_{i=1,\ldots,N}$. The main difference here, with respect to standard ProMP, is that the weights are complex numbers. As we want to have correlations between real and imaginary parts of our weights (so that we can learn correlations in magnitudes or phases), we consider an expanded real version of our weights where the real and imaginary parts are concatenated as:

$$\boldsymbol{w}_i = \left[\mathrm{Re}(\tilde{\boldsymbol{w}}_i)^\top, \mathrm{Im}(\tilde{\boldsymbol{w}}_i)^\top\right]. \tag{6}$$

It is straightforward to see that $\boldsymbol{w}_i$ and $\tilde{\boldsymbol{w}}_i$ are linear in the complex space:

$$\tilde{\boldsymbol{w}}_i = \boldsymbol{A}\boldsymbol{w}_i \text{ with } \boldsymbol{A}_{T\times 2T} = \left[\, \boldsymbol{I}_T \;\; i\boldsymbol{I}_T \,\right]. \tag{7}$$

For notation simplicity, we define $\boldsymbol{\Phi} = \boldsymbol{A}\tilde{\boldsymbol{\Phi}}$, which implies:

$$\forall i \in [\![1; N]\!] : \boldsymbol{y}_i = \boldsymbol{\Phi}\boldsymbol{w}_i. \tag{8}$$

We learn the distribution of the weights $(\boldsymbol{w}_i)_{i=1\ldots N}$ by fitting a Gaussian mixture using the Expectation-Maximization algorithm, initialized with the K-means algorithm. We retrieve the weights, means and covariances $\boldsymbol{\theta} = (\pi_j, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)_{j=1,\ldots,M}$ of the Gaussian mixture, whose probability density function is expressed as:

$$\begin{gathered} p(\boldsymbol{w}|\theta) = \sum_{j=1}^{M} \pi_j \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j), \\ \text{with } \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) = \\ \frac{1}{(2\pi)^{(2T)/2}|\boldsymbol{\Sigma}_j|^{1/2}} \exp\left\{-\frac{1}{2}(\boldsymbol{w}-\boldsymbol{\mu}_j)^\top \boldsymbol{\Sigma}_j^{-1}(\boldsymbol{w}-\boldsymbol{\mu}_j)\right\}. \end{gathered} \tag{9}$$

We will use this distribution in the Fourier domain to perform minimal intervention control [28]. To do so, we need a way to transform a partial trajectory (e.g., the starting position of the robot) to the Fourier domain. In the context of ProMP, this is typically done by conditioning on the distribution (usually a single Gaussian). We observed that this is not suitable for the high number of dimensions we have, hence we propose a different approach that scales better with the number of dimensions.

B. Mapping partial trajectories to Fourier domain

Given a partial demonstration $\boldsymbol{y}_{1:K}$ of size $K \times 1$, we search $\boldsymbol{w}$ such that:

$$\boldsymbol{y}_{1:K} = \boldsymbol{\Phi}_{1:K}\boldsymbol{w}, \tag{10}$$

with $\boldsymbol{\Phi}_{1:K}$ of size $K \times T$ (containing the first K rows of $\boldsymbol{\Phi}$). It is important to note that the approach is also valid for

[1] For discrete movements, a periodic signal of period 2T can be constructed by symmetrizing the original signal of length T, so that the same method can be applied.

partial trajectories that do not occur at the beginning of the movement, or arbitrary keypoints.

A straightforward, but naive, solution would be to choose $\boldsymbol{w} = (\boldsymbol{\Phi}_{1:K})^{+}\boldsymbol{y}_{1:K}$, which, in practice, results in a value for $\boldsymbol{w}$ that is far from the distribution of demonstrated data, resulting in poor tracking. We instead propose to leverage the knowledge of the demonstrations distribution in the Fourier domain (as learned in Section IV-A) to find a set of weights $\boldsymbol{w}$ that is close to the demonstrations, while respecting (10).

This can be written as the optimization problem:

$$\max_{\boldsymbol{w}} p(\boldsymbol{w}|\boldsymbol{\theta}) \quad \text{s.t.} \quad \boldsymbol{y}_{1:K} = \boldsymbol{\Phi}_{1:K}\boldsymbol{w}, \tag{11}$$

which is equivalent to:

$$\min_{\boldsymbol{w}} \big(-\log p(\boldsymbol{w}|\boldsymbol{\theta}) \big) \quad \text{s.t.} \quad \boldsymbol{y}_{1:K} = \boldsymbol{\Phi}_{1:K}\boldsymbol{w}. \tag{12}$$

To solve this problem more efficiently, we use a Lagrangian relaxation:

$$\min_{\boldsymbol{w}} \big(\|\boldsymbol{y}_{1:K} - \boldsymbol{\Phi}_{1:K}\boldsymbol{w}\|^2 - \lambda \log p(\boldsymbol{w}|\boldsymbol{\theta}) \big), \tag{13}$$

where λ is the Lagrange multiplier. We could find the value of λ by solving the Lagrangian dual problem, but for simplicity purposes we fix λ to an arbitrary small value ($1e-8$) as it yields good results in all of our experiments. In practice, the weights $\boldsymbol{w}$ are of high dimensions and therefore the different Gaussians of the mixture have no overlap (formally, this means that the mutual information between any two Gaussians of the mixture is almost zero). The solution of (13) must verify that $p(\boldsymbol{w}|\boldsymbol{\theta})$ is not numerically zero (otherwise $-\lambda \log p(\boldsymbol{w}|\boldsymbol{\theta})$ tends to infinity). Under the hypothesis that the Gaussians have almost-zero mutual information, we can find candidate solutions by solving M least squares problems:

$$\begin{aligned} \boldsymbol{w}^j &= \arg\min_{\boldsymbol{w}} \big(\|\boldsymbol{y}_{1:K} - \boldsymbol{\Phi}_{1:K}\boldsymbol{w}\|^2 - \lambda \log \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \big) \\ &= \arg\min_{\boldsymbol{w}} \big(\|\boldsymbol{y}_{1:K} - \boldsymbol{\Phi}_{1:K}\boldsymbol{w}\|^2 + \lambda \|\boldsymbol{w} - \boldsymbol{\mu}_j\|^2_{\boldsymbol{\Sigma}_j^{-1}} \big) \\ &= \big(\boldsymbol{\Phi}_{1:K}^{H}\boldsymbol{\Phi}_{1:K} + \lambda \boldsymbol{\Sigma}_j^{-1} \big)^{-1} \big(\boldsymbol{\Phi}_{1:K}^{H}\boldsymbol{y}_{1:K} + \lambda \boldsymbol{\Sigma}_j^{-1}\boldsymbol{\mu}_j \big). \end{aligned} \tag{14}$$

We can then solve (13) by finding the minimum over the finite set of solutions $(\boldsymbol{w}^j)_{j=1}^{M}$:

$$\begin{aligned} j^* = \arg \min_{j \in [\![1;M]\!]} \big(&\|\boldsymbol{y}_{1:K} - \boldsymbol{\Phi}_{1:K}\boldsymbol{w}^j\|^2 - \lambda \log(\pi_j) \\ &+ \lambda \|\boldsymbol{w} - \boldsymbol{\mu}_j\|^2_{\boldsymbol{\Sigma}_j^{-1}} \big), \end{aligned} \tag{15}$$

which allows us to map our partial trajectory to the Fourier domain with:

$$\boldsymbol{w}_K = \boldsymbol{w}^{j^*}. \tag{16}$$

The full process is summarized in Algorithm 1. Next, we propose a tracking controller in the Fourier domain, leveraging the distribution learned and the possibility to map partial trajectories to the Fourier domain.

```
Algorithm 1: Partial trajectory mapping
  Data: Partial observations $\boldsymbol{y}_{1:K}$ up to timestep $K$
  Result: Fourier weight $\boldsymbol{w}_K$ such that $\boldsymbol{y}_{1:K} \simeq \boldsymbol{\Phi}_{1:K}\boldsymbol{w}_K$
  Find $M$ candidate solutions $(\boldsymbol{w}^j)_{j=1}^{M}$ with Eq.14
  Compute minimum $\boldsymbol{w}_K$ with Eqs.15-16
```

C. Tracking in the Fourier domain

The ability to do minimal intervention control in the Fourier domain is a core component of our proposed method, as it permits to modulate both phase and amplitude by exploiting the variability of the provided demonstrations. We will track only one Gaussian for simplicity purposes (the solution of Eq.15). This seems to be a reasonable assumption because in high dimensions, the different Gaussians in the mixture are likely to have a very small overlap. We track this Gaussian in the Fourier domain with the given covariance.

We could do this by using model predictive control (MPC) in the Fourier domain, but, as the number of dimensions is high (T is the trajectory length), it would be too computationally expensive. We propose to use a simple proportional controller to track in the Fourier domain: an approach that proves satisfactory in practice. Given a current trajectory up to timestep t, represented as $\boldsymbol{w}_t$ in the Fourier space, we track the target $\boldsymbol{\mu}_{j^*}$ with precision matrix $\boldsymbol{\Sigma}_{j^*}^{-1}$. The update rule of the tracking controller is:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \text{dt}\ \beta\ \text{diag}(\boldsymbol{\Sigma}_{j^*}^{-1})(\boldsymbol{\mu}_{j^*} - \boldsymbol{w}_t), \tag{17}$$

where $\text{diag}(\cdot)$ is an operator zeroing all offdiagonal elements. We choose to weigh the updates by $\text{diag}(\boldsymbol{\Sigma}_{j^*}^{-1})$ and not $\boldsymbol{\Sigma}_{j^*}^{-1}$ since the latter made the controller unstable in practice. More sophisticated controllers could be used to leverage the full-rank structure of the precision matrix $\boldsymbol{\Sigma}_{j^*}^{-1}$, and we shall address this in future work.

Similarly to ProMP, we can go back from Fourier domain to time domain and find the next point to track as well as the appropriate tracking covariance:

$$\begin{aligned} \boldsymbol{y}_{t+1}^{\text{des}} &= \boldsymbol{\Phi}_{t+1}\boldsymbol{w}_{t+1}, \\ \boldsymbol{\Sigma}_{\boldsymbol{y}_{t+1}^{\text{des}}} &= \boldsymbol{\Phi}_{t+1}\boldsymbol{\Sigma}_{j^*}\boldsymbol{\Phi}_{t+1}^{H}, \end{aligned} \tag{18}$$

with $\boldsymbol{\Phi}_{t+1}$ of size $1 \times T$ (containing the $(t+1)^{\text{th}}$ row of $\boldsymbol{\Phi}$).

The pseudocode of the algorithm is given in Algorithm 2.

```
Algorithm 2: Tracking in Fourier domain
  Data: Partial observations $\boldsymbol{y}_{1:K}$ up to timestep $K$
  Result: desired trajectory $\boldsymbol{y}_{K+1:T}^{\text{des}}$ for timesteps $K+1$
          to $T$ and desired covariances
          $[\boldsymbol{\Sigma}_{\boldsymbol{y}_{K+1}^{\text{des}}}, \ldots, \boldsymbol{\Sigma}_{\boldsymbol{y}_{T}^{\text{des}}}]$
  Calculate $\boldsymbol{w}_K$ using Eqs.(14)-(16)
  for $t \leftarrow K$ to $T-1$ do
      Calculate $\boldsymbol{w}_{t+1}$ using Eq.(17)
      Calculate $\boldsymbol{y}_{t+1}^{\text{des}}$ and $\boldsymbol{\Sigma}_{\boldsymbol{y}_{t+1}^{\text{des}}}$ using Eq.(18)
  end
```

D. Multidimensional case

We discuss here the extension of our method to several degrees of freedom D. The extension is straightforward as it consists of concatenating along the dimensions. Following the previous notation, the data and partial data are written as such:

$$\boldsymbol{y}_i = \begin{pmatrix} \boldsymbol{y}_i^1 \\ \vdots \\ \boldsymbol{y}_i^D \end{pmatrix} \quad \text{and} \quad \boldsymbol{y}_{1:K} = \begin{pmatrix} \boldsymbol{y}_{1:K}^1 \\ \vdots \\ \boldsymbol{y}_{1:K}^D \end{pmatrix}, \tag{19}$$

where the superscript j of $\boldsymbol{y}_i^j$ denotes the j^{th} degree of freedom. And the $\boldsymbol{\Phi}$ matrix is used to construct a block-diagonal matrix with D entries:

$$\boldsymbol{\Phi}^D = \begin{pmatrix} \boldsymbol{\Phi} & \dots & \boldsymbol{0} \\ \vdots & \ddots & \vdots \\ \boldsymbol{0} & \dots & \boldsymbol{\Phi} \end{pmatrix}. \tag{20}$$

Similarly, $\boldsymbol{\Phi}_{1:K}$ and $\boldsymbol{\Phi}_t$ are concatenated D times block-diagonally. It is worth noting that in this case, $\boldsymbol{w}$ is a vector of length TD, which means that the Gaussian mixture learned captures correlations between the different degrees of freedom.

V. Experiments

In this section we show the performance of FMP on various datasets. First, we describe the data acquisition and preprocessing step. Then, a polishing task and the task of drawing a 8-shape are presented. Finally, the task of wiping a whiteboard is considered and applied on a real robot. When applicable, our method will be compared against the use of ProMP with Von-Mises basis functions. Videos of the experimental evaluation can be found at https://sites.google.com/view/fourier-movement-primitives.

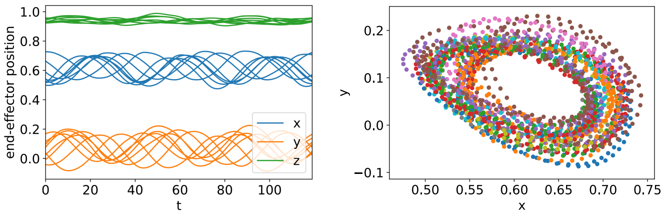
A. Data acquisition and preprocessing

For simplicity and visualization purposes, in all tasks the data consist of the position of the robot end-effector, and is therefore 3-dimensional. All demonstrations are obtained by kinesthetically teaching the robot. As the tasks are rhythmic, we propose to reduce the human burden by showing only one (long) demonstration, that is then preprocessed. The demonstration is acquired at 20Hz, and we cut it in subdemonstrations of length T, arbitrarily chosen to 120 in our experiments (corresponding to 6 seconds). To cut the demonstration, we let a sliding window slide across the demonstration by increments of 10 timesteps. By doing so, we exploit the fact that the task is rhythmic and can start anywhere.

B. Polishing task

The polishing task is a representative example because it can contain as low as one frequency for each degree of freedom. A 3-minute demonstration is recorded with the robot, which cuts in the demonstrations as explained above.

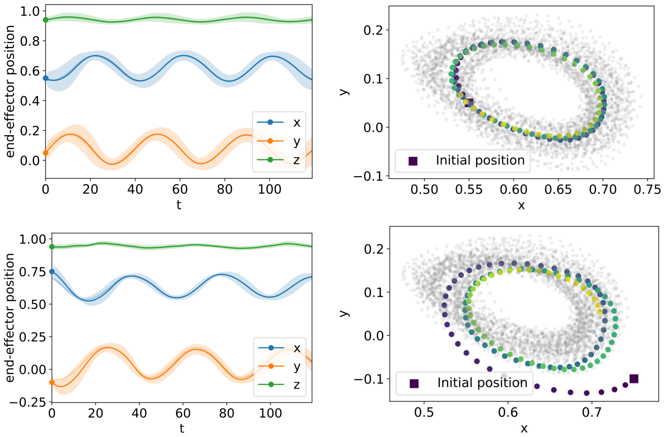
The demonstrations are shown in Fig.3. We learn the distribution of the data in the Fourier domain with $M = 10$ Gaussians. We show in Fig.4 the tracking for different starting positions: a position that belongs to the data distribution (interpolation), and a position outside of the data distribution (extrapolation).



(a) End-effector position as a function of time

(b) Demonstrations in the $x - y$ plane (different colors represent different demonstrations).

Fig. 3: Demonstrations of the polishing task.



(a) Tracking trajectory x, y, z in function of time, along with uncertainties

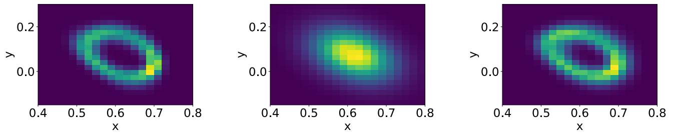
(b) Scatter plot of the tracking trajectory on dimensions x and y (color gradient for time)

Fig. 4: Polishing from different initial positions with FMP.

As we can see, our method permits both interpolation and extrapolation with respect to the starting position. We compare now our method to the standard ProMP with Von-Mises basis functions (later abbreviated ProMP-VM) [20]:

$$\begin{aligned} b_i^{\text{VM}}(z) &= \exp\left(\frac{\cos\left(2\pi f(z_t - c_i)\right)}{h}\right), \\ \Phi_i(z_t) &= \frac{b_i^{\text{VM}}(z)}{\sum_{j=1}^{n} b_j^{\text{VM}}(z)}, \end{aligned} \tag{21}$$

where f denotes the frequency of the signal, c_i the center of the basis function and h the width. This method requires the demonstrations to be aligned, and to contain exactly one period of the signal. For illustration purposes, we show how ProMP-VM performs after alignment of the data and cutting to contain only one period (roughly at $T = 40$), and therefore $f = 1$. We used 20 basis functions with the centers c_i uniformly placed between 0 and 2π. The hyperparameter h is selected so that the basis functions become cosine (high value of h, as the exponential function is locally equal to the identity around 0, up to a constant). We show in Fig.5a a heatmap of the learned distribution, where we can see that, in the case of careful data



(a) $T = 40$, alignment, $M = 1$

(b) $T = 120$, no alignment, $M = 1$

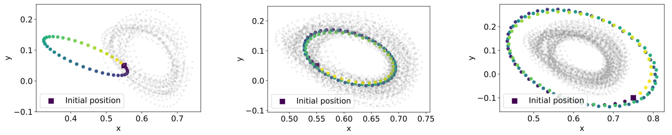
(c) $T = 120$, no alignment, $M = 10$

Fig. 5: Heatmaps of distribution learned with ProMP-VM.

alignment, ProMP-VM can approximate the distribution of polishing demonstrations well. In the original ProMP method, only one Gaussian is used to approximate the distribution of the demonstrations.

Even if, to the best of our knowledge, this has not been proposed in the ProMP literature, we will show that increasing the number of Gaussians can alleviate the need for demonstration alignment. Indeed, due to higher variability in the phase domain, the distribution of weights becomes multi-modal and hence is more accurately encoded by a mixture. In Fig.5b, we show the obtained results for ProMP-VM using more than one period (T=120), where we had to explicitly provide the frequency of the signal (in this case, f=3). As we can see, without alignment, ProMP-VM fails to approximate the distribution of the data. For a fairer comparison, we also extend ProMP-VM by learning the distribution with a mixture model (10 Gaussians) and show in Fig.5c that doing so permits to approximate the distribution well.

The results in Fig.5 show that, by bringing ProMP closer to FMP, the original ProMP formulation can be greatly improved. However, a major difference between ProMP-VM and FMP lies in the way we generate trajectory distributions that go through keypoints (see IV-B). With ProMP-VM, it is done via conditioning, whereas in FMP is is achieved by mapping the keypoint to the Fourier domain, and tracking in the Fourier domain (here, the keypoint that we evaluate is the starting point, but it is applicable to any keypoint or partial trajectory). Indeed, as shown in Fig.6a, when using ProMP-VM with alignment it is not possible to start the movement from a different region than the one observed in the demonstrations. When we use multiple Gaussians without alignment, this adaptation capability becomes possible (Fig.6b). Since ProMP-VM represents the demonstrations with periodic basis functions, it can only generate periodic signals that will pass through the initial point. As seen in Fig.6c, this mechanism does not allow to cope well with perturbations that require to extrapolate outside of the demonstrations, while following the demonstrations in the next cycles. Indeed, when conditioning outside of the training data, it tends to produce overconfident trajectory distributions (because the Gaussian mixture is learned by maximizing the log-likelihood) that do not return to the demonstrations. In contrast, FMP can generate trajectory distributions that return back to the training data in a way that is compatible with the variations that were observed in the demonstrations, as we can see in Fig.4.

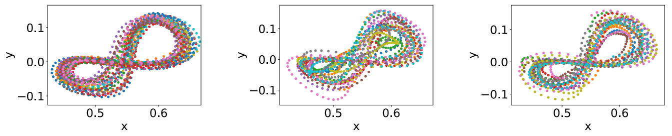


(a) $T = 40$, alignment, $M = 1$

(b) $T = 120$, no alignment, $M = 10$

(c) $T = 120$, no alignment, $M = 10$

Fig. 6: Conditioning on initial position with ProMP-VM.



(a) Demonstrations

(b) ProMP-VM-Mult

(c) FMP

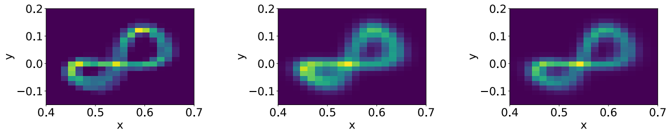
Fig. 7: Samples of the learned 8-shape distribution.

C. 8-shape drawing

We demonstrate a 3-minute drawing of an 8-shape, used as a standard benchmark task [8, 7]. This task is interesting because it involves a superposition of different frequencies. In the standard ProMP-VM, only one frequency can be approximated. For a better analysis of the performances of FMP, we propose here to benchmark FMP against an extension of ProMP-VM that can approximate a superposition of frequencies. We include basis functions for different frequencies f, namely for f from 1 to 5. For each f, 20 offset basis functions are used, as previously. We use this extension of ProMP-VM on the same data as FMP (no alignment), and with $M = 10$ for a fair comparison. We denote this extension as ProMP-VM-Mult.

Fig.7 shows demonstration samples, ProMP-VM-Mult samples, and FMP samples. We observe that FMP samples are smoother and closer to the demonstrations than ProMP-VM-Mult samples, which suggests that the distribution has been better learned with FMP than ProMP-VM-Mult. To verify this observation, we computed a heatmap of the learned distribution. For ProMP-VM-Mult and FMP, we sample 10000 trajectories from the learned distribution, and compute the heatmap. Those are shown in Fig.8, next to the demonstrations heatmap.

We can see that the heatmap of ProMP-VM-Mult is more blurred compared to the FMP heatmap (more samples seem to fall inside the 8 holes). To confirm this, we propose to evaluate quantitatively the learned distribution. We compare the distributions learned with ProMP-VM-Mult and FMP to the ground truth (obtained from the demonstrations). We note Q the ground truth distribution, and P the approximate distribution (respectively obtained with ProMP-VM-Mult or FMP). The distributions are discrete probability distributions,



(a) Demonstrations

(b) ProMP-VM-Mult

(c) FMP

Fig. 8: Heatmaps of the learned 8-shape distribution.

	ProMP-VM-Mult	FMP
Forward KL	0.20	**0.11**
Reverse KL	0.53	**0.28**

TABLE I: Quantitative comparison of distributions learned with ProMP-VM-Mult and FMP versus ground truth distribution for 8-shape task.

defined over the finite set of cases $\mathcal{X}$ of the heatmap. We considered two different metrics:

- the Forward Kullblack-Leibler divergence:

$$D_{\mathrm{KL}}(P||Q) = \sum_{x \in \mathcal{X}} P(x) \log \frac{P(x)}{Q(x)}.$$

  Forward KL is known as *zero avoiding*, as it penalizes $Q(x) = 0$ when $P(x) > 0$. This therefore quantifies if the distribution learned covers well the ground truth distribution.
- the Reverse Kullblack-Leibler divergence:

$$D_{\mathrm{KL}}(Q||P) = \sum_{x \in \mathcal{X}} Q(x) \log \frac{Q(x)}{P(x)}.$$

  Reverse KL is known as *zero forcing*, as it does not penalize $Q(x) = 0$ when $P(x) > 0$. This therefore measures how well our distribution Q approximates a part of the ground truth distribution.

The results are presented in Table I. We observe that FMP has learned a distribution that is about twice closer to the ground truth distribution compared to ProMP-VM-Mult. This can be interpreted easily, as ProMP-VM-Mult has several basis functions for a given frequency, which gives many more basis functions for the same given number of frequencies, resulting in poor statistics. Finally, we evaluate how FMP can generate trajectories that start at any given position. In Fig.9, we can see that, even for tasks that involve a superposition of different frequencies, FMP can generate trajectory distributions that get back to the training data in a way that is compatible with the variations observed in the demonstrations. The results with ProMP-VM-Mult were unsatifactory, consistently with Fig.7 (these results are not included in the manuscript due to space constraint).

D. Real-world wiping task

Finally, we apply FMP to a real-world robotic task of whiteboard wiping. Our robot is a 7-DoF torque-controlled Panda robot. We record a 2-minute demonstration of whiteboard wiping with kinesthetic teaching. The demonstration is then split into subdemonstrations of length $T = 120$ (6s) as explained previously. For simplicity purposes, only the position of the robot end-effector is recorded, the statistics are therefore made on end-effector position trajectories (with $M = 10$ Gaussians). The robot is then controlled with an impedance controller that tracks the desired trajectory with manually specified gains, with a fixed orientation (we allow the robot to be compliant around the normal to the plane by setting low orientation gains around that axis). An overview of the setup is shown in Fig.1. In this experiment we show that we can generate movements of arbitrary durations with FMP. While this should be trivial because we have periodic basis functions over the duration T, this is not in practice as we did not preprocess the data so that the beginning and end of the demonstrations are equal. We alleviate this by recomputing at timestep T the Fourier weights $\boldsymbol{w}$ given the partial trajectory from $T - K$ to T (in practice, we use K=10), and subsequently can use the desired trajectory between timesteps T and $2T - K$. We then repeat this process (it is interesting to note that every time we recompute $\boldsymbol{w}$ using the partial trajectory mapping, we allow the trajectory to change the Gaussian that is tracked). While this might appear cumbersome, this is in practice very efficient, and much easier than having to align the demonstrations. To evaluate the quality of the learned distribution, we propose to show two movements given a desired initial position:

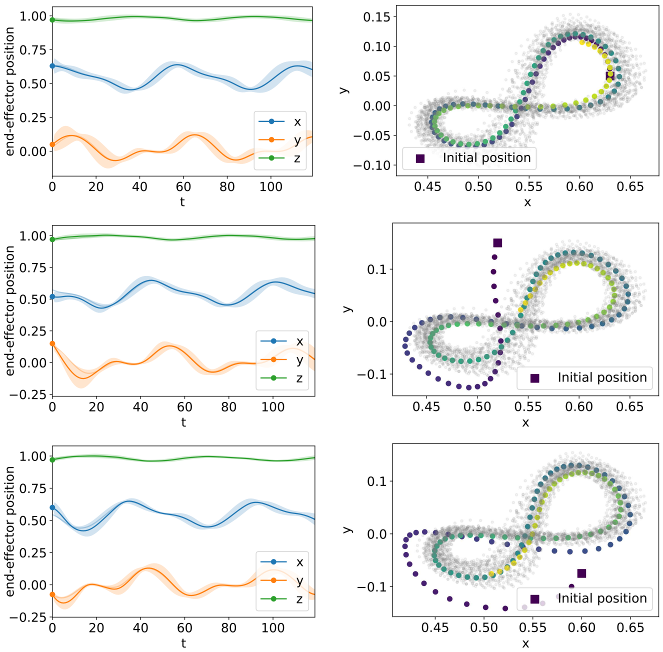


Fig. 9: 8-shape from different initial positions with FMP.

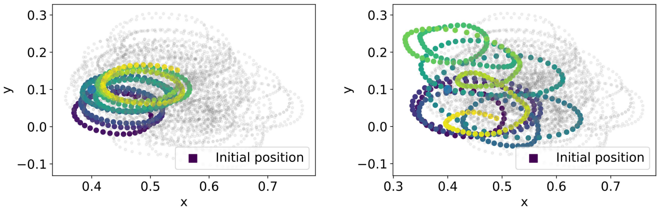
- One where we track the Gaussian mean as explained in Section IV-C.
- One where we sample from the Gaussian distribution and track this sample instead of the mean.

We observe in Fig.10 that FMP is successful at generating trajectories of arbitrary lengths. In addition, sampling instead of tracking a Gaussian provides an interesting possibility, as we can see that the generated trajectory shows much more variability. This is useful for tasks that require some (co)variations in the movement (such as wiping tasks where we do not want artifacts to arise from a movement that repeats itself exactly).

VI. Discussion

We now discuss the results from Section V and emphasize the advantages of FMP over other state-of-the-art methods.

We have shown that **FMP does not require demonstration alignment**, as it performs statistics directly over phase shifts



(a) Tracking the Gaussian mean

(b) Tracking a Gaussian sample

Fig. 10: Generated trajectories of length 400 (20s) for a given initial position.

in the complex weight space. It therefore goes beyond ProMP with cosine or Von-Mises basis functions, which fails when demonstrations are not aligned (see Fig.5). However, we have seen that increasing the number of Gaussians in ProMP can also permit to alleviate the need for demonstrations alignment. Moreover, when using ProMP-VM, only one frequency can be approximated, and it additionnally requires the extraction of this frequency as an external preprocessing step. FMP does not require such preprocessing.

FMP can learn tasks that involve a superposition of signals of different frequencies. This could not be done with the standard ProMP-VM. However, for a fairer comparison, we proposed to extend ProMP-VM to different frequencies by adding basis functions of different frequencies. We have shown that doing so permits to learn tasks that require different frequencies, but that the distribution learned is not as accurate as the one learned with FMP (we identified a factor 2 in terms of performance for our experiment, see Table I). Also, FMP can represent variations of phase and amplitude for a given basis function in a single weight, by exploiting complex number properties. Furthermore, as many basis functions need to be placed for each frequency for ProMP-VM-Mult, this would not scale with the number of basis functions needed. We observed empirically that we could not include higher frequencies in ProMP-VM-Mult, as the redundancy and number of the basis functions led to numerical instabilities when learning the Gaussian mixture. In contrast, FMP scales well with the number of basis functions, as we use all of them in our experiments. Better statistics might be obtained by performing dimensionality reduction of the number of basis functions and we plan to address this in future work.

Defining appropriate hyperparameters for basis functions in ProMP can be cumbersome. This holds true for ProMP-VM with periodic signals as well. Indeed, the number and centers of the basis functions need to be appropriately chosen (too few would make a very coarse discretization of the phase shifts, too many would lead to a very high number of basis functions, and hence poor statistics and/or numerical instabilities). **With FMP, no such choice is required**, as the complex exponentials form a basis and can approximate any signal. We therefore have a theoretical guarantee that demonstrations can be represented by weights.

In practice, one of the few hyperparameters that needs to be chosen with FMP is the length of the signal T to cut the demonstration(s). We noticed empirically that it had no effect on the final solution, as long as T is big enough to contain one or more periods. FMP does not require T to be set such that subdemonstrations are equal at the beginning and at the end. FMP just uses higher frequencies to compensate for this, but we did not observe any problem in our experiments.

We also showed that FMP has interesting extrapolation capabilities. While theoretically possible, conditioning to find a distribution that goes through a keypoint is not applicable in high dimensions, as it collapses to the mean of the distribution and hence does not go through the desired keypoint. We therefore proposed another way that is fast (solving of a least squares problem) and applicable to our high-dimensional setting (see Section IV-B). Additionnaly, we showed that it is safe when faced with a new situation (see Section V-C). Not only does it return to the demonstrations, but it does so in a way that exploits the variations of magnitude and phase that were observed in the demonstrations. In the first two experiments (polishing and 8-shape), this means that the generated trajectories return back to the limit cycle. This is a property that is usually desirable for dynamical systems, which is not satisfied by ProMP-VM, as we saw in Fig.6. Moreover extrapolation with ProMP-VM might not be safe, as conditioning far from the Gaussian mean can result in overconfident trajectory predictions and hence highly stiff control around a potentially poor generalized trajectory.

VII. Conclusion

We proposed a method based on discrete Fourier transform and Probabilistic Movement Primitives, which we call Fourier Movement Primitives (FMP) for the learning of rhythmic movements from demonstrations. Our basis functions are theoretically well motivated and no demonstrations alignment is required, which reduces the engineering burden. We have shown that FMP can learn tasks that involve a superposition of basis functions of different frequencies. The extrapolation capabilities of FMP are also relevant, generating trajectories that go back to the demonstrations when faced with a situation different from what was observed.

Future work will consider dimensionality reduction in the space of weights, which could enable the use of better control strategies in the Fourier domain, by using for example a Linear Quadratic Regulator (LQR). We will also study the possibility to perform statistics separately for the phase and magnitude of the weights, as it could yield richer compliance control strategies with an adaptive modulation of phase and amplitude.

Acknowledgements

This work was supported by the ROSALIS project (Swiss National Science Fundation) and by the CoLLaboratE project (European Commission's Horizon 2020 Programme, https://collaborate-project.eu/, grant agreement 820767).

References

[1] W. Amanhoud, M. Khoramshahi, and Aude Billard. A dynamical system approach to motion and force genera-

tion in contact tasks. In *Robotics: Science and Systems*, Freiburg im Breisgau, Germany, June 2019.

[2] L. Armesto, J. Bosga, V. Ivan, and S. Vijayakumar. Efficient learning of constraints and generic null space policies. In *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, pages 1520–1526, May 2017.

[3] L. Armesto, V. Ivan, J. Moura, A. Sala, and S. Vijayakumar. Learning constrained generalizable policies by demonstration. In *Robotics: Science and Systems*, Cambridge, Massachusetts, July 2017.

[4] A. Billard, S. Calinon, and R. Dillmann. *Handbook of Robotics*, chapter Learning from Humans, pages 1995–2014. Springer Berlin Heidelberg, 2016.

[5] S. Calinon, D. Bruno, and D. G. Caldwell. A task-parameterized probabilistic model with minimal intervention control. In *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, pages 3339–3344, Hong Kong, China, May-June 2014.

[6] S. Degallier, L. Righetti, S. Gay, and A. J. Ijspeert. Toward simple control for complex, autonomous robotic applications: combining discrete and rhythmic motor primitives. *Autonomous Robots*, 31(2-3):155–181, 2011.

[7] J. Ernesti, L. Righetti, M. Do, T. Asfour, and S. Schaal. Encoding of periodic and their transient motions by a single dynamic movement primitive. In *Proc. IEEE Intl Conf. on Humanoid Robots (Humanoids)*, pages 57–64, Nov 2012.

[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*, 27(1): 3–23, 2009.

[9] A. Gams, M. Do, A. Ude, T. Asfour, and R. Dillmann. On-line periodic movement and force-profile learning for adaptation to new surfaces. In *Proc. IEEE Intl Conf. on Humanoid Robots (Humanoids)*, pages 560–565, December 2010.

[10] Y. Huang, F. Abu-Dakka, J. Silvério, and D. G. Caldwell. Towards orientation learning and adaptation in cartesian space. *arXiv:1907.03918*, pages 1–14, 2019.

[11] Y. Huang, L. Rozo, J. Silvério, and D. G. Caldwell. Kernelized movement primitives. *International Journal of Robotics Research*, 38(7):833–852, 2019.

[12] A. Ijspeert, J. Nakanishi, P. Pastor, H. Hoffmann, and S. Schaal. Dynamical movement primitives: Learning attractor models for motor behaviors. *Neural Computation*, 25(2):328–373, 2013.

[13] A. J. Ijspeert, J. Nakanishi, and S. Schaal. Movement imitation with nonlinear dynamical systems in humanoid robots. In *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, pages 1398–1403, 2002.

[14] S. M. Khansari-Zadeh and A. Billard. Learning stable non-linear dynamical systems with Gaussian mixture models. *IEEE Trans. on Robotics*, 27(5):943–957, 2011.

[15] M. Khoramshahi and A. Billard. A dynamical system approach to task-adaptation in physical human–robot interaction. *Autonomous Robots*, 43(4):927–946, 2019.

[16] M. Khoramshahi, A. Laurens, T. Triquet, and A. Billard. From human physical interaction to online motion adaptation using parameterized dynamical systems. In *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, pages 1361–1366, October 2018.

[17] H. Lin, M. Howard, and S. Vijayakumar. Learning null space projections. In *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, pages 2613–2619, May 2015.

[18] J.R. Medina Hernández, D. Lee, and S. Hirche. Risk-sensitive optimal feedback control for haptic assistance. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

[19] A. Paraschos, C. Daniel, J. Peters, and G. Neumann. Probabilistic movement primitives. In *NIPS*, pages 2616–2624, 2013.

[20] A. Paraschos, C. Daniel, J. Peters, and G. Neumann. Using probabilistic movement primitives in robotics. *Autonomous Robots*, 42(3):529–551, 2018.

[21] L. Peternel, T. Petrič, E. Oztop, and J. Babič. Teaching robots to cooperate with humans in dynamic manipulation tasks based on multi-modal human-in-the-loop approach. *Autonomous Robots*, 36(1-2):123–136, 2014.

[22] L. Peternel, T. Petrič, and J Babič. Human-in-the-loop approach for teaching robot assembly tasks using impedance control interface. In *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, pages 1497–1502. IEEE, 2015.

[23] T. Petrič, A. Gams, A. J. Ijspeert, and L. Žlajpah. On-line frequency adaptation and movement imitation for rhythmic robotic tasks. *International Journal of Robotics Research*, 30(14):1775–1788, 2011.

[24] T. Petrič, A. Gams, L. Žlajpah, and A. Ude. Online learning of task-specific dynamics for periodic tasks. In *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, pages 1790–1795, September 2014.

[25] C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning*. MIT Press, Cambridge, MA, USA, 2006.

[26] L. Righetti, J. Buchli, and A. J. Ijspeert. Dynamic hebbian learning in adaptive frequency oscillators. *Physica D: Nonlinear Phenomena*, 216(2):269–281, 2006.

[27] E. Todorov. Optimality principles in sensorimotor control. *Nature Neuroscience*, 7(9):907–915, 2004.

[28] E. Todorov and M. I Jordan. A minimal intervention principle for coordinated movement. In *Advances in neural information processing systems*, pages 27–34, 2003.

[29] A. Ude, A. Gams, T. Asfour, and J. Morimoto. Task-specific generalization of discrete and periodic dynamic movement primitives. *IEEE Trans. on Robotics*, 26(5): 800–815, 2010.