Dynamic movement primitives in robotics

A tutorial survey

Saveriano, Matteo; Abu-Dakka, Fares J.; Kramberger, Aljaž; Peternel, Luka

DOI
10.1177/02783649231201196

Publication date
2023

Document Version
Final published version

Published in
International Journal of Robotics Research

Article

The International Journal of Robotics Research
2023, Vol. 42(13) 1133–1184

DOI: 10.1177/02783649231201196
journals.sagepub.com/home/ijr

Dynamic movement primitives in robotics: A tutorial survey

Matteo Saveriano[1], Fares J Abu-Dakka[2], Aljaž Kramberger[3] and Luka Peternel[4]

Abstract

Biological systems, including human beings, have the innate ability to perform complex tasks in a versatile and agile manner. Researchers in sensorimotor control have aimed to comprehend and formally define this innate characteristic. The idea, supported by several experimental findings, that biological systems are able to combine and adapt basic units of motion into complex tasks finally leads to the formulation of the motor primitives' theory. In this respect, Dynamic Movement Primitives (DMPs) represent an elegant mathematical formulation of the motor primitives as stable dynamical systems and are well suited to generate motor commands for artificial systems like robots. In the last decades, DMPs have inspired researchers in different robotic fields including imitation and reinforcement learning, optimal control, physical interaction, and human–robot co-working, resulting in a considerable amount of published papers. The goal of this tutorial survey is two-fold. On one side, we present the existing DMP formulations in rigorous mathematical terms and discuss the advantages and limitations of each approach as well as practical implementation details. In the tutorial vein, we also search for existing implementations of presented approaches and release several others. On the other side, we provide a systematic and comprehensive review of existing literature and categorize state-of-the-art work on DMP. The paper concludes with a discussion on the limitations of DMPs and an outline of possible research directions.

Keywords

Motor control of artificial systems, movement primitives' theory, dynamic movement primitives, learning from demonstration

1. Introduction

> How do biological systems, like humans and animals, execute complex movements in a versatile and creative manner?

In the past decades, researchers of neurobiology and motor control have made a significant effort trying to answer this research question, and their experimental findings lead to the formulation of the *motor* or *motion primitives theory*. The motion primitives' theory explains the execution of complex motion with the ability of biological systems of sequencing and adapting units of actions, the so-called motion primitives (Flash and Hochner, 2005; Mussa-Ivaldi, 1999).

Dynamic Movement Primitives (DMPs) have their roots in the motor control of biological systems and can be seen as a rigorous mathematical formulation of the motion primitives as stable nonlinear dynamical systems (Schaal, 2006a, 2006b). In this respect, DMPs represent one of the first attempts to answer the research question:

> How artificial systems, like (humanoid) robots, can execute complex movements in a versatile and creative manner?

Beyond their biological motivation, DMPs have a simple and elegant formulation, guarantee convergence to a given target, are sufficiently flexible to create complex behaviors, are capable of reacting to external perturbations in real-time, and can be learned from data using efficient algorithms. These properties explain the "success" of DMPs in robotic applications, where they have been established as a prominent tool for learning and generation of motor commands. Since their formulation in the pioneering work from Ijspeert et al. (Ijspeert et al., 2002c; Schaal, 2006), DMPs have been successfully exploited in a variety of applications, becoming de facto the first approach that novices in the Imitation Learning (IL) field use on their robots.

[1]Department of Industrial Engineering (DII), University of Trento, Trento, Italy
[2]Munich Institute of Robotics and Machine Intelligence (MIRMI), Technical University of Munich, München, Germany
[3]SDU Robotics, The Maersk McKinney Moller Institute, University of Southern Denmark, Odense, Denmark
[4]Department of Cognitive Robotics, Delft University of Technology, Delft, The Netherlands

Corresponding author:
Fares J Abu-Dakka, Munich Institute of Robotics and Machine Intelligence (MIRMI), Technical University of Munich, Georg-Brauchle-Ring 60-62, München 80992, Germany.
Email: fares.abu-dakka@tum.de

1.1. Existing surveys and tutorials

The popularity of DMPs resulted in a large amount of work that use, modify, or extend the original formulation of Ijspeert and colleagues. The pioneering work (Ijspeert et al., 2001) computes the desired position and velocity as a mixture of multiple dynamical systems with different time scales. One year later, (Ijspeert et al., 2002c; Schaal, 2006) propose a different formulation that became popular and that we name, in this paper, the *classical DMPs*. As shown in Table 1, some tutorials and surveys already tried to categorize and review existing work on DMPs.

Schaal et al. (2007) presented the classical DMPs as an attempt to unify nonlinear dynamical systems and optimal control theory, *that is,* the two prominent frameworks used to derive computational models of neurobiological motor theories (Flash and Hochner, 2005; Mussa-Ivaldi, 1999). Ijspeert et al. (2013) presented a homogeneous formulation of rhythmic and discrete DMPs together with some extensions including coupling terms, generalization to different goals, and online adaptation for collision avoidance. They also described possible applications in IL and motion recognition methods. In the same year, Pastor et al. (2013) presented an extension to classical DMPs with a special focus on online adaptation of the DMP attractor landscape by integrating the perceptual information into the action generation process. Later on, Deniša et al. (2016b) reviewed the so-called Compliant Movement Primitive (CMP), which was first introduced by Petrič et al. (2014a). CMPs combine classical DMP to generate the desired kinematic path and *torque primitives*—a weighted summation of Gaussian basis functions—to generate task-specific dynamics. As shown in the Deniša et al. review, CMPs are capable of accurately tracking the kinematic path in a compliant manner, which makes them well suited for tasks that require interaction of the robot with the environment.

However, the abovementioned papers, reviews, and tutorials primarily focused on the methods and advancements within their respective research group and/or focused on a specific problem or field of application. On the other hand, the DMP-related literature is extensive and broad, with contributions from many research groups that made advancements in several important fields of application. Therefore, the proposed survey and tutorial on DMPs aim to scan a wider range and present a tutorial with unified and structured formulations for various DMP methods and advancements up to date. This should make it clearer for the users to see the differences and connections between various methods and can contribute to easier application. In addition, we provide a more comprehensive and categorized survey of all major DMP application areas in robotics. This can help to inspire the readers to apply the DMPs in various areas.

In the tutorial part, we present mathematical formulations, implementation details, and potential issues of existing DMP formulations starting from the classical DMPs presented in (Ijspeert et al., 2002c; Schaal, 2006) up to recent extensions of DMPs to Riemannian geometry and Symmetric Positive Definite (SPD) matrices (Abu-Dakka and Kyrki, 2020). In the survey part, we meticulously review existing literature on DMPs in a comprehensive and methodological manner by focusing on the quality and significance of their continuations without putting a bias on any particular research group. Details on the systematic review procedure are given as follows.

1.2. Systematic review process

We performed an automatic search for documents containing the string

Dynamic Movement Primitive

in Scopus on 25 November 2020 that returned 1223 papers. We started the search from 2001 when preliminary work on DMPs was published. We further refined the search on 20 June 2023 to include last-minute papers.

We manually inspect all the papers and removed the ones that do not explicitly use DMPs and that only compare against DMP in their literature review. The first and foremost selection criteria were the technical quality of work and the significance of the contribution with respect to the DMP state of the art prior to the publication of any particular paper. In other words, we asked the question "did the paper make a significant step change in the field?". Therefore, we discarded papers that presented similar (or same) ideas multiple times or that made insignificant improvements to the state of the art. If multiple papers presented the same/similar idea, we included the one with the most comprehensive technical quality, and if the quality was similar, the next deciding factors were publication in more prestigious journals/venues or the most cited ones. This manual selection led to 321 papers on DMPs (out of a total of 373 references) analyzed in this work.

1.3. A taxonomy of DMP-related research

The systematic review of DMP literature led to the taxonomy shown in Figure 1, which also describes the structure of this paper. DMPs are placed at the root of the tree and branch into two nodes, namely, the *tutorial* and the *survey*. In the tutorial part, we present different DMP formulations and extensions in rigorous mathematical terms.

The tutorial part spans Sections 2 and 3. Section 2 embraces DMP formulations for *discrete and periodic* motions, *orientation* trajectories, and *SPD* matrices. Section 3 discusses extensions of the DMP formalism to account for skills *generalization*, *joining* of multiple primitives, *online adaptation* based on force feedback, or reference velocity. The section ends with a short description of DMP-related formulations.

The survey part spans Sections 4 and 5. Section 4 presents DMP integration in larger executive frameworks for *manipulation* and *variable impedance* tasks, *reinforcement*, *deep*, and

Table 1. Comparison between existing papers, reviews, and tutorials about DMPs and our tutorial survey.

Paper/survey/ tutorial	Topics	Description
Schaal et al. (2007)	• Classical DMPs • Online adaptation • Optimization	A tutorial that provides a unifying view of the two main approaches used to develop computational motor control theories, namely, differential equations and optimal control. In this work, discrete and rhythmic DMPs (Ijspeert et al., 2002c; Schaal, 2006) are presented as a computational model of the motor primitives' theory (Mussa-Ivaldi 1999) that unifies nonlinear differential equations and optimal control. The tutorial has a section dedicated to DMP parameter optimization beyond ILs. Schaal et al. show how to optimize DMP parameters to minimize various costs describing, for instance, the total jerk of the trajectory or the end-point variance.
Ijspeert et al. (2013)	• Classical DMPs • Online adaptation • Coupling terms • Generalization	A paper on classical DMPs that in addition to its scientific contribution, it presents both discrete and rhythmic formulations, mostly developed in (Ijspeert et al., 2002c, 2002; Schaal, 2006), and their application in IL and movement recognition. The paper also presents extensions of the classical DMP formulation to prevent high accelerations at the beginning of the motion, to avoid collisions with unforeseen obstacles (Pastor et al. 2009), and to generalize both in space (*e.g.,* reach a different goal) and time (e.g., produce longer/shorter trajectories).
Pastor et al. (2013)	• Classical DMPs • Online adaptation • Coupling terms • Impedance learning	A paper on classical DMPs that in addition to its scientific contribution, it presents both discrete and rhythmic formulations, mostly developed in (Ijspeert et al., 2002c, 2002; Schaal, 2006). The paper also presents extensions of the classical DMP formulation to avoid collisions with unforeseen obstacles (Pastor et al. 2009) and to learn impedance control policies via Reinforcement Learning (RL) (Buchli et al. 2011b). The key difference between Ijspeert et al. (2013) and Pastor et al. (2013) is the section dedicated to the sensory association and online, context-aware adaptation of DMP trajectories using the associative skill memory framework developed in Pastor et al. (2011) and Pastor et al. (2011a).
Deniša et al. (2016b)	• Classical DMPs • Compliant Movement Primitives (CMPs)	A tutorial on CMPs, a framework developed to generate compliant robot behaviors that accurately track a reference trajectory. CMPs exploit classical DMPs to generate the desired kinematic landscape and encode task-dependent dynamics as a combination of Gaussian basis functions (torque primitives). The tutorial shows how to learn torque primitives from training data, how to generalize CMPs to new situations, and how to combine existing CMPs to synthesize new robot motions.
Survey and tutorial	**Topics**	**Description**
This paper	DMP tutorial • Classical • Orientation • SPD • Joining • Generalization • Online adaptation DMP survey • (Co-)Manipulation • Variable impedance • Physical interaction • Rehabilitation • Teleoperation • Motion recognition • Reinforcement, deep, and lifelong learning	This tutorial survey conducts a wide scan of the existing DMP literature with the aim of categorizing and presenting the published work in the field. The main objective of this comprehensive literature review is to give the reader an exhausting overview on DMP-related research, on its major achievements, as well as on open issues and possible research directions. Our tutorial survey also provides a structured and unified formulation for different methods developed starting from the classical DMPs proposed by (Ijspeert et al., 2002c; Schaal, 2006). We believe that such formulation contributes to an easier understanding of different methods and extension that can be found in the literature, clarifying connections and differences among the existing approaches. The tutorial survey also provides an analysis on pros and cons of various methods and a discussion with guidelines for different application scenarios.

life-long learning. Section 5 presents DMPs in different robotic applications including *physical interaction*, *co-manipulation*, *rehabilitation*, *teleoperation*, *motion recognition*, *humanoids* and *field robotics*, and *autonomous driving*.

The paper ends with a discussion (Section 6) of presented approaches with the aim of providing, where possible, guidelines to select the most suitable DMP approach for specific needs. We have also collected available DMP implementations (see Table 4) and contributed to the community with further open-source implementations available at https://gitlab.com/dmp-codes-collection. Section 6 terminates with a discussion on open issues and possible research directions.

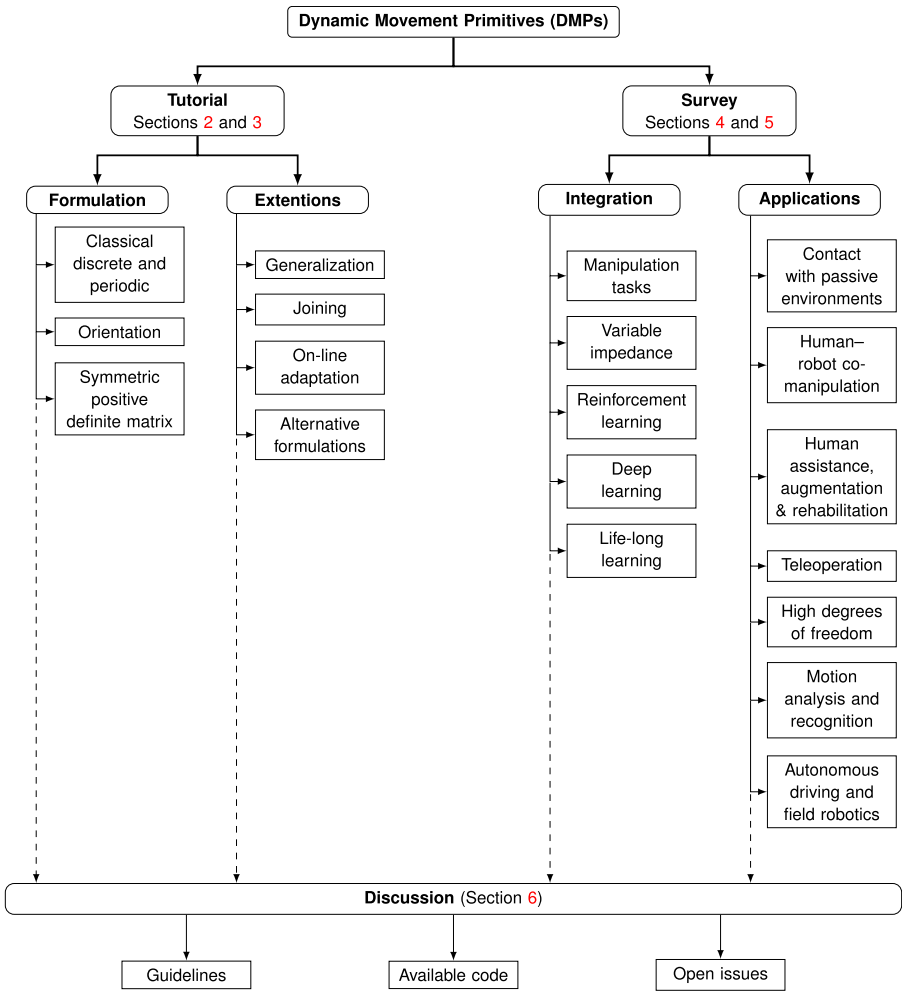

Figure 1. Structure of this tutorial survey on DMPs.

1.4. Contribution overview

Our paper has several key contributions that are summarized as follows.

Concerning the tutorial part:

- We present the classical DMP formulation and existing variations of this formulation in a unified manner with rigorous mathematical terms, providing implementation details and discussing the advantages and limitations of different approaches (Section 2).
- We describe *advanced* approaches where DMPs are integrated into sophisticated control and/or larger executive frameworks (Section 3).
- We release to the community several implementations of described approaches. Detailed information on these code repositories is provided in Table 4 and Section 6. Moreover, we search for existing open-source implementations of the presented formulations and list them in our repository (Section 6.2).

Concerning the survey part:

- We perform a systematic literature search to provide a comprehensive and unbiased review of the topic (Sections 4 and 5).
- We categorize existing work on DMPs into different streams and highlight prominent approaches in each category (Figure 1 and Sections 4 and 5).
- We present guidelines to select the most suitable approach for different applications, discuss limitations inherent to the DMP formalism, and highlight open issues and possible research directions (Section 6).

2. Formulation of DMP types

In this section, we will provide a complete description of the standard formulation of DMPs. Specifically, point attractors formulation—to encode discrete point-to point motions—in Section 2.1 and cycle attractors' formulation—to encode rhythmic-patterns motions—in Section 2.2. For a better

understanding, we have summarized the key notations and the used abbreviations in Table 2.

2.1. Discrete DMP

The discrete DMP is used to encode a point-to-point motion into a stable dynamical system. In the following subsections, we will go through the formulation and main features of discrete DMPs starting with the classical one operating in $\mathbb{R}$ space (Section 2.1.1), then passing by Cartesian space—$\mathcal{S}^3$ and $\mathcal{SO}(3)$—in Section 2.1.2, and ending by DMP formulation for SPD space ($\mathcal{S}^m_{++}$) in Section 2.1.3.

2.1.1. Classical DMP. The classical discrete DMPs, first introduced by Ijspeert et al. (2002c), encapsulate training data into a linear, second-order dynamics (a mass–spring–damper system) with an additive, nonlinear forcing term learned from a single demonstration. A DMP for a single DoF trajectory y of a discrete movement (point-to-point) is defined by the following set of nonlinear differential equations (Ijspeert et al., 2002c, 2013)

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

$$\tau \dot{y} = z, \tag{2}$$

$$\tau \dot{x} = -\alpha_x x \tag{3}$$

where x is the phase variable and z is an auxiliary variable. Parameters α_z and β_z define the behavior of the second-order system described by (1) and (2). With the choice $\tau > 0$, $\alpha_z = 4\beta_z$, and $\alpha_x > 0$, the convergence of the underlying dynamic system to a unique attractor point at $y = g$, $z = 0$ is ensured (Ijspeert et al., 2013). Alternatively, the gains α_z and β_z can be learned from training data while preserving the convergence of the system (Tan et al., 2016). In the DMP literature, equations (1) and (2), as well as their periodic counterpart (34)–(35), are called the *transformation system*, while (3) (or (36)) is the *canonical system*. $f(x)$ is defined as a linear combination of N nonlinear Radial Basis Functions (RBFs), which enables the robot to follow any smooth trajectory from the initial position y_0 to the final configuration g

$$f(x) = \frac{\sum_{i=1}^{N} w_i \Psi_i(x)}{\sum_{i=1}^{N} \Psi_i(x)} x, \tag{4}$$

$$\Psi_i(x) = \exp\left(-h_i(x - c_i)^2\right) \tag{5}$$

where c_i are the centers of Gaussian basis functions distributed along the phase of the movement and h_i their widths. For a given N and setting τ equal to the total duration of the desired movement, we can define $c_i = \exp(-\alpha_x i - 1/N - 1)$, $h_i = 1/(c_{i+1} - c_i)^2$, and $h_N = h_{N-1}$ where $i = 1, \ldots, N$. For each DoF, the weights w_i should be adjusted from the measured data so that the desired behavior is achieved. The selection of the number of weights should be based on the desired resolution of the trajectory. For controlling a robotic system with more than one DoF, we represent the movement of every DoF with its own equation system (1)–(2), but with the common phase (3) to synchronize them.

2.1.1.1. Learning the forcing term. For a discrete motion, given a demonstrated trajectory $y_d(tJ)$, $t_J = 1, \ldots, \mathfrak{T}$, and its time derivatives $\dot{y}_d(t_J)$ and $\ddot{y}_d(t_J)$, it is possible to invert (1) and approximate the desired shape f_d as follows

$$f_d(t_J) = \tau^2 \ddot{y}_d(t_J) - \alpha_z(\beta_z(g - y_d(t_J)) - \tau \dot{y}_d(t_J)) \tag{6}$$

By stacking each $f_d(t_J)$ and w_i into the column vectors $\mathfrak{F} = [f_d(t_1), \ldots, f_d(t_{\mathfrak{T}})]^\top$ and $\mathbf{w} = [w_1, \ldots, w_N]^\top$, we obtain the following linear system

$$\mathbf{\Phi} w = \mathfrak{F}, \tag{7}$$

where

$$\mathbf{\Phi} = \begin{bmatrix} \frac{\Psi_1(x_1)}{\sum_{i=1}^{N}\Psi_i(x_1)}x_1 & \cdots & \frac{\Psi_N(x_1)}{\sum_{i=1}^{N}\Psi_i(x_1)}x_1 \\ \vdots & \ddots & \vdots \\ \frac{\Psi_1(x_{\mathfrak{T}})}{\sum_{i=1}^{N}\Psi_i(x_{\mathfrak{T}})}x_{\mathfrak{T}} & \cdots & \frac{\Psi_N(x_{\mathfrak{T}})}{\sum_{i=1}^{N}\Psi_i(x_{\mathfrak{T}})}x_{\mathfrak{T}} \end{bmatrix} \tag{8}$$

Locally Weighted Regression (LWR) (Atkeson et al., 1997; Schaal and Atkeson, 1998; Ude et al., 2010) is a popular approach used to update the weights w_i. LWR with the recursive least squares method uses the error between the desired trajectory shape and the currently learned shape and a *forgetting factor* λ to incrementally update the weights as follows

$$\mathbf{P}_J = \frac{1}{\lambda}\left(\mathbf{P}_{J-1} - \frac{\mathbf{P}_{J-1}\boldsymbol{\varphi}_J\boldsymbol{\varphi}_J{}^\top\mathbf{P}_{J-1}}{\lambda + \boldsymbol{\varphi}_J{}^\top\mathbf{P}_{J-1}\boldsymbol{\varphi}_J}\right), \tag{9}$$

$$\mathbf{w}_J = \mathbf{w}_{J-1} + (f_d(\boldsymbol{t}_J) - \boldsymbol{\varphi}_J{}^\top\mathbf{w}_{J-1})\mathbf{P}_J\boldsymbol{\varphi}_J \tag{10}$$

In the previous equations, $\mathbf{w}_J = \mathbf{w}(t_J)$ and $\boldsymbol{\varphi}_J$ is the column vector obtained by transposing the J-th row of $\mathbf{\Phi}$. The initial value of the parameters is $\mathbf{P}_0 = \mathbf{I}$, $\mathbf{w}_0 = \mathbf{0}$. It is worth mentioning that the update rules in (9)–(10) represent an incremental version of LWR. A batch version of LWR is presented in Ijspeert et al. (2013), and the difference between the two approaches is highlighted in Schaal and Atkeson (1998) on page 9. A discrete DMP learned on synthetic data is shown in Figure 2.

LWR has been the standard method to learn the weights of DMPs and therefore $f(x)$. As an alternative to LWR, Krug and Dimitrovz (2013) have shown that learning a forcing term defined as in (4) can be formulated as a quadratic optimization problem and efficiently solved.

In general, the problem of learning and retrieving $f(x)$ can be in principle solved with any regression technique (Stulp et al., 2013). For instance, Wang et al. (2016) modified $f(x)$ in (4) by considering a bias term b_i, *that is,* $w_i x + b_i$, and used truncated kernels (Ψ_i vanishes if $x - c_i$ is smaller than a

Table 2. Description of key notations and abbreviations. Indices, super/subscripts, constants, and variables have the same meaning over the whole text.

N	$\triangleq$	# of nonlinear basis functions	i	$\triangleq$	index: $i = 1, 2, \ldots, N$
J	$\triangleq$	# of joints or Degree of Freedoms (DoFs)	j	$\triangleq$	index: $j = 1, 2, \ldots, J$
L	$\triangleq$	# of demonstrations or DMPs	l	$\triangleq$	index: $l = 1, 2, \ldots, L$
V	$\triangleq$	# of via-points or via-goals	v	$\triangleq$	index: $v = 1, 2, \ldots, V$
$\mathfrak{T}$	$\triangleq$	# of datapoints	j	$\triangleq$	index $: j = 1, 2, \ldots, \mathfrak{T}$
m	$\triangleq$	Dimensions of $\mathcal{S}^m_{++}$	n	$\triangleq$	Dimensions of $\mathbb{R}^n$
$\{\cdot\}_d$	$\triangleq$	Subscript for desired value	$\{\cdot\}_q$ or $\{\cdot\}^q$	$\triangleq$	Quaternion-related variable
$\{\cdot\}_R$ or $\{\cdot\}^R$	$\triangleq$	Rotation matrix-related variable	$\{\cdot\}_{++}$, $\{\cdot\}_+$ or $\{\cdot\}^+$	$\triangleq$	SPD-related variable
$\{\cdot\}_g$	$\triangleq$	Subscript for goal value	α_z, β_z, α_x, α_s, α_g, α_{yx}, α_{qg}	$\triangleq$	Positive gains
τ	$\triangleq$	Time modulation parameter	c_i, h_i	$\triangleq$	Centers and widths of Gaussians
T	$\triangleq$	Time duration	t	$\triangleq$	Continuous time
λ	$\triangleq$	Forgetting factor	r	$\triangleq$	Amplitude modulation parameter
x	$\triangleq$	Phase variable	$y, \dot{y}$	$\triangleq$	Trajectory data and its 1st derivative
s	$\triangleq$	Sigmoidal decay phase	$z, \dot{z}$	$\triangleq$	Scaled velocity and acceleration
p	$\triangleq$	Piece-wise linear phase	g, $\mathbf{g}_q$, $\mathbf{g}_+$	$\triangleq$	Attractor point (*goal*) in different spaces
$\boldsymbol{\omega}$	$\triangleq$	Angular velocity	$\widehat{g}$, $\widehat{\mathbf{g}}_q$ and $\tilde{g}$, $\tilde{\mathbf{g}}_q$	$\triangleq$	Moving target and delayed goal function in different spaces
$\mathcal{Q}_t, \dot{\mathcal{Q}}_t$	$\triangleq$	Joint position, its 1st time-derivative	g_v	$\triangleq$	Intermediate attractor (*via-goal*)
$\mathbf{q}, \dot{\mathbf{q}}$	$\triangleq$	Unit quaternion, its 1st time-derivative	$\mathbf{R}, \dot{\mathbf{R}}$	$\triangleq$	Rotation matrix, its 1st time-derivative
f, $\mathbf{f}_q$, $\mathbf{f}_R$, $\mathbf{f}_q$, $\mathcal{F}_+$	$\triangleq$	Forcing term for different spaces	w_i	$\triangleq$	Adjustable weights
Ψ_i	$\triangleq$	Basis functions	θ and $\boldsymbol{\vartheta}$	$\triangleq$	An angle and learnable parameters
$\mathcal{S}^m_{++}$	$\triangleq$	$m \times m$ SPD manifold	$\mathbf{Sym}^m$	$\triangleq$	$m \times m$ symmetric matrix space
$\mathcal{M}$	$\triangleq$	A Riemannian manifold	$\mathbf{X}$	$\triangleq$	An arbitrary SPD matrix
$\mathcal{T}_{\boldsymbol{\Lambda}}\mathcal{M}$	$\triangleq$	A tangent space of $\mathcal{M}$ at an arbitrary point $\boldsymbol{\Lambda}$	$\mathbf{M}$	$\triangleq$	The mean of $\{\mathbf{X}_t\}_{t=1}^{\mathfrak{T}}$
$\boldsymbol{\varrho} = \mathrm{Log}_{\boldsymbol{\Lambda}}(\boldsymbol{\Upsilon})$	$\triangleq$	$\mathcal{M} \mapsto \mathcal{T}_{\boldsymbol{\Lambda}}\mathcal{M}$, maps an arbitrary point $\boldsymbol{\Upsilon} \in \mathcal{M}$ into $\boldsymbol{\varrho} \in \mathcal{T}_{\boldsymbol{\Lambda}}\mathcal{M}$	$\boldsymbol{\Upsilon} = \mathrm{Exp}_{\boldsymbol{\Lambda}}(\boldsymbol{\varrho})$	$\triangleq$	$\mathcal{T}_{\boldsymbol{\Lambda}}\mathcal{M} \mapsto \mathcal{M}$, maps $\boldsymbol{\varrho} \in \mathcal{T}_{\boldsymbol{\Lambda}}\mathcal{M}$ into $\boldsymbol{\Upsilon} \in \mathcal{M}$
$vec\,(\cdot)$	$\triangleq$	A function transforms $\mathbf{Sym}^m$ into $\mathbb{R}^n$ using Mandel's notation.	$mat\,(\cdot)$	$\triangleq$	A function transforms $\mathbb{R}^n$ into $\mathbf{Sym}^m$ using Mandel's notation.
k, K, $\mathbf{K}^{\mathcal{P}}$, $\mathbf{K}^{\mathcal{O}}$	$\triangleq$	Different forms of stiffness gains	$D, \mathbf{D}^{\mathcal{V}}$, $\mathbf{D}^{\mathcal{W}}$	$\triangleq$	Different forms of damping gains
$\mathfrak{m}$ and $\boldsymbol{\mathcal{I}}$	$\triangleq$	Mass and inertia matrices	F, $\mathbf{f}^e$, and $\boldsymbol{\tau}^e$	$\triangleq$	Forces and external forces and torques

DMP	Dynamic Movement Primitive	IL	Imitation Learning
CMP	Compliant Movement Primitive	UAV	Unmanned Aerial Vehicle
RL	Reinforcement Learning	SPD	Symmetric Positive Definite
DoF	Degree of Freedom	RBF	Radial Basis Function
LWR	Locally Weighted Regression	GMM	Gaussian Mixture Model
GMR	Gaussian Mixture Regression	GP	Gaussian Process
NN	Neural Network	VMP	Via-points Movement Primitive
ProMP	Probabilistic Movement Primitives	LfD	Learning from Demonstration
GPR	Gaussian Process Regression	MoMP	Mixture of Motor Primitives
EMG	Electromyography	ILC	Iterative Learning Control
VIC	Variable Impedance Control	VILC	Variable Impedance Learning Control
PI^2	Policy Improvement With Path Integrals	CMAES	Covariance Matrix Adaptation-Evolution Strategies
CC-DMP	Coordinate Change-DMPs	RBF-NN	Radial Basis Function-Neural Network
AL-DMP	Arc-Length-DMP	HRL	Hierarchical RL
AEDMP	AutoEncoded DMP	CNN	Convolutional Neural Network
GPDMP	Global Parametric Dynamic Movement Primitive	PoWER	Policy Learning by Weighting Exploration with the Returns

threshold). This formulation, called DMP+, produces more accurate trajectories than the original DMP. Moreover, a learned trajectory can be modified by updating only a subset of the weights. However, DMP+ has double the parameters of the original DMP, and the truncated kernel introduces a discontinuity at the truncation point. To remedy this issue, Ginesi et al. (2021b) exploited the use of Mollifier-like and Wendland basis functions with promising results. Rouse and Daltorio (2021) substituted the underlying kernels of stable

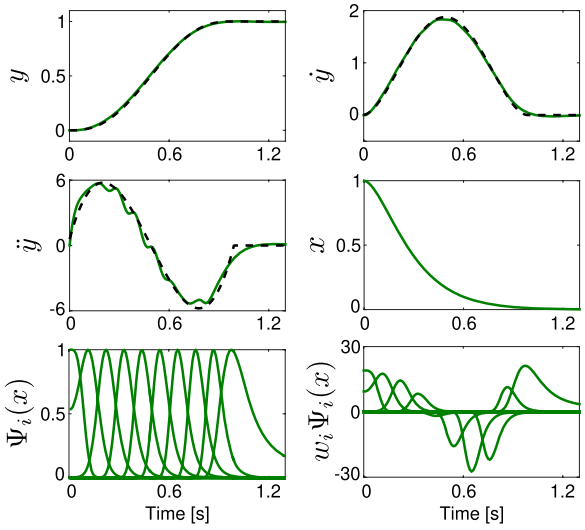

Figure 2. A classical DMP is used to generate a discrete motion connecting $x_0 = 0$ and $g = 1$ (green line in the top left panel). The training data (black dashed lines) are obtained from a minimum jerk trajectory connecting x_0 and g in $T = 1$s and used to learn the weights w_i of 10 Gaussian basis functions equally distributed in time. The results of the parameter learning procedure are shown in the bottom right panel. The exponentially decaying phase variable is used as shown in the middle right panel. Results are obtained with the open-source implementation available at https://gitlab.com/dmp-codes-collection/dmp-classical.

attractor points or limit cycles in DMPs with the underlying kernels of saddle points, known as Structurally Homogeneous Centers (SHCs). They demonstrated that the application of SHCs in the DMP framework was straightforward and yielded comparable results to the original approach in terms of ease of use and similarity of outcomes. Other work focused on using multiple demonstrations to increase the generalization power of the learned primitive. To learn a suitable forcing term from multiple demonstrations, some authors used Gaussian Mixture Model (GMM) (Yin and Chen, 2014; Pervez et al., 2017a) and Gaussian Mixture Regression (GMR) (Cohn et al., 1996), while others adopted Gaussian Process (GP) (Fanger et al., 2016; Rasmussen and Williams, 2006; Umlauft et al., 2017), or exploited a deep Neural Network (NN) (Pervez et al., 2017b; Pahič et al., 2020) developed originally in LeCun et al. (2015).

2.1.1.2. Phase stopping and goal switching. The phase variable x in (3) provides the ability to manipulate time during the execution of DMP equations. Moreover, DMP provides the ability to slow down or even stop the execution through the phase-stopping mechanism (Ijspeert et al., 2002c)

$$\tau \dot{x} = -\frac{\alpha_x x}{1 + \alpha_{yx}\|\tilde{y} - y\|} \tag{11}$$

where $\tilde{y}$ is the measured state and y is the one generated from the DMP. Alternatively, Anand et al. (2021) have shown that it is possible to modulate the time scaling factor τ as follows

$$\dot{\tau} = -k_\tau(\tau - u_\tau) \tag{12}$$

where control input u_τ is designed such that the decay rate of the canonical system in (3) is slowed down if the desired execution time is increased or sped up if the desired execution time is reduced.

DMPs also provide an elegant way to adapt the trajectory generation in real-time through goal-switching mechanisms (Ijspeert et al., 2013)

$$\tau \dot{g} = \alpha_g(g_0 - g) \tag{13}$$

where g_0 is the original goal and g is the new one to reach.

DMPs in their standard formulation are not suitable for direct encoding of skills with specific geometry constraints, such as orientation profiles (represented in either unit quaternions or rotation matrices), stiffness/damping, and manipulability profiles (encapsulated in full SPD matrices). For instance, direct integration of unit quaternions does not ensure the unity of the quaternion norm. Any representation of orientation that does not contain singularities is non-minimal, which means that additional constraints need to be taken into account during integration.

2.1.1.3. Alternative phase variables. Equation (3) describes an exponential decaying phase variable that has been widely used in the DMP literature. The main drawback of the exponential decaying phase is that it rapidly drops to very small values toward the end of the motion. This "forces" the learning algorithm to exploit relatively high weights w_i to accurately reproduce the last part of the demonstration (Samant et al., 2016). As an example, in Figure 3 the exponential decaying phase (brown dot-dashed line) is very small already after 0.6s, while the expected time duration of the motion is $T = 1$s.

To overcome this limitation, Kulvicius et al. (2011) propose the sigmoidal decay phase s (green solid line in Figure 3), obtained by integrating

$$\dot{s} = -\frac{\alpha_s e^{(\alpha_s/\delta_t)(\tau T - t)}}{\left[1 + e^{(\alpha_s/\delta_t)(\tau T - t)}\right]^2} \tag{14}$$

where α_s defines the steepness of s centered at time T and δt is the sampling time. As shown in Figure 3, $s = 1$ for $t < T - \delta_s$, where the time δ_s depends on the steepness α_s, and then it decays to $s = 0$.

The sigmoidal decay in Figure 3 has a tail effect since it vanishes after $T + \delta_s$s, where δ_s depends on the tunable parameter α_s. The piece-wise linear phase p (blue-dashed line in Figure 3), proposed by Samant et al. (2016), linearly decays from 1 to 0 in exactly Ts and then remains constant. p is obtained by integrating

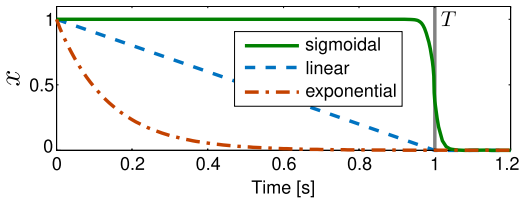

Figure 3. Possible phase variables used in different discrete DMP formulations. All the different possibilities ensure that $x, s, p \rightarrow 0$ for $t \rightarrow +\infty$ (for $t > T$ in practice).

$$\tau \dot{p} = \begin{cases} -\dfrac{1}{T}, & p \geq 0 \\ 0, & \text{otherwise} \end{cases} \tag{15}$$

where $p(0) = 1$ and T is the time duration of the motion.

2.1.2. Orientation DMP. The classical DMP formulation described in Section 2.1.1 applies to single DoF motions. Multidimensional motions are generated independently and synchronized with a common phase. In other words, equations (1) and (2) are repeated for each DoF while the phase variable in (3) is shared. This works when the evolution of different DoFs is independent, like for joint space or Cartesian position trajectories. Unlike Cartesian position, the elements of orientation representations like unit quaternion or rotation matrix are constrained. In this section, we present approaches that extend the classical DMP formulation to represent Cartesian orientations.

2.1.2.1. Quaternion DMP. Unit quaternion $\mathbf{q} = \nu + \mathbf{u} \in \mathcal{S}^3$ provides a representation of the orientation of the robot's end-effector (Chiaverini and Siciliano, 1999). $\mathcal{S}^3$ is a unit sphere in $\mathbb{R}^4$, $\nu \in \mathbb{R}$, and $\mathbf{u} \in \mathbb{R}^3$. Abu-Dakka et al. (2015a) rewrote DMP equations (1) and (2) for direct unit quaternion encoding as follows

$$\tau \dot{\boldsymbol{\eta}} = \alpha_z \left(\beta_z 2 \operatorname{Log}^q \left(\mathbf{g}_q * \overline{\mathbf{q}} \right) - \boldsymbol{\eta} \right) + \mathbf{f}_q(x), \tag{16}$$

$$\tau \dot{\mathbf{q}} = \frac{1}{2} \boldsymbol{\eta} * \mathbf{q} \tag{17}$$

where $\mathbf{g}_q \in \mathcal{S}^3$ denotes the goal orientation, the quaternion conjugation is defined as $\overline{\mathbf{q}} = \overline{\nu + \mathbf{u}} = \nu - \mathbf{u}$, and * denotes the quaternion product

$$\begin{aligned} \mathbf{q}_1 * \mathbf{q}_2 &= (\nu_1 + \mathbf{u}_1) * (\nu_2 + \mathbf{u}_2) \\ &= \left(\nu_1 \nu_2 - \mathbf{u}_1^\top \mathbf{u}_2 \right) + \left(\nu_1 \mathbf{u}_2 + \nu_2 \mathbf{u}_1 + \mathbf{u}_1 \times \mathbf{u}_2 \right). \end{aligned}$$

$\eta \in \mathbb{R}^3$ is the scaled angular velocity $\boldsymbol{\omega}$ and treated as unit quaternion with zero scalar ($\nu = 0$) in (17). The function $\operatorname{Log}^q(\cdot) : \mathcal{S}^3 \mapsto \mathbb{R}^3$ is given as follows

$$\operatorname{Log}^q(\mathbf{q}) = \begin{cases} \arccos(\nu) \dfrac{\mathbf{u}}{\|\mathbf{u}\|}, & \mathbf{u} \neq \mathbf{0} \\ [0\,0\,0]^\top, & \text{otherwise}, \end{cases} \tag{18}$$

where $\| \cdot \|$ denotes ℓ_2 norm. In practice, to avoid numerical instabilities, $\|\mathbf{u}\|$ is compared with a small ϵ before normalizing $\mathbf{u}$.

An early attempt to encode unit quaternion profiles using DMP was presented by Pastor et al. (2011). Unlike Abu-Dakka et al.'s formulation, Pastor et al.'s formulation does not take into account the geometry of $\mathcal{SO}(3)$ as they just used the vector part of the quaternion product $(\mathbf{g}_q * \overline{\mathbf{q}})$ in (16) instead of $2 \operatorname{Log}^q(\mathbf{g}_q * \overline{\mathbf{q}})$ which defines the angular velocity $\boldsymbol{\omega}$ that rotates quaternion $\mathbf{q}$ into $\mathbf{g}_q$ within a unit sampling time.

Equation (17) can be integrated as follows

$$\mathbf{q}(t + \delta t) = \operatorname{Exp}^q \left(\frac{\delta t}{2} \frac{\boldsymbol{\eta}(t)}{\tau} \right) * \mathbf{q}(t) \tag{19}$$

where $\delta_t > 0$ denotes a small constant. The function $\operatorname{Exp}^q(\cdot) : \mathbb{R}^3 \mapsto \mathcal{S}^3$ is given as follows

$$\operatorname{Exp}^q(\omega) = \begin{cases} \cos(\|\boldsymbol{\omega}\|) + \sin(\|\boldsymbol{\omega}\|) \dfrac{\boldsymbol{\omega}}{\|\boldsymbol{\omega}\|}, & \boldsymbol{\omega} \neq \mathbf{0} \\ 1 + [0\,0\,0]^\top & \text{otherwise}. \end{cases} \tag{20}$$

where $\|\boldsymbol{\omega}\|$ is compared with a small ϵ before normalizing $\boldsymbol{\omega}$ to avoid numerical instabilities.

Both mappings become one-to-one, continuously differentiable, and inverse to each other if the input domain of the mapping $\operatorname{Log}^q(\cdot)$ is restricted to $\mathcal{S}^3$ except for $-1 + [0\ 0\ 0]^\top$, while the input domain of the mapping $\operatorname{Exp}^q(\boldsymbol{\omega})$ should fulfill the constraint $\|\boldsymbol{\omega}\| < \pi$ (Abu-Dakka et al., 2015a). An exemplar unit quaternion DMP is shown in Figure 4.

Phase-stopping (11) can be rewritten as follows

$$\tau \dot{x} = -\frac{\alpha_x x}{1 + \alpha_{qx} \mathrm{d}(\tilde{\mathbf{q}}, \mathbf{q})} \tag{21}$$

where

$$\mathrm{d}(\tilde{\mathbf{q}}, \mathbf{q}) = \begin{cases} 2\ \pi, & \mathbf{q}_1 * \overline{\mathbf{q}_2} = 1 + [0\ 0\ 0] \\ 2\ |\operatorname{Log}^q(\mathbf{q}_1 * \overline{\mathbf{q}_2})|, & \text{otherwise} \end{cases}$$

Ude et al. (2014) extended DMP quaternion-based formulation by rewriting (13) to include goal switching mechanism

$$\tau \dot{\mathbf{g}}_q = \alpha_{qg} \operatorname{Log}^q \left(\mathbf{g}_{q,new} - \overline{\mathbf{g}}_q \right) * \mathbf{g}_q \tag{22}$$

so that $\mathbf{g}_q$ is continuously changing onto $\mathbf{g}_{q,new}$ in real-time. Equation (22) should be integrated using (20) along with (16) and (17).

As shown by Saveriano et al. (2019) using Lyapunov arguments, both the quaternion DMP formulations in Pastor et al. (2011) and in Abu-Dakka et al. (2015a) and Ude et al. (2014) asymptotically converge to the target quaternion $\mathbf{g}_q$ with zero velocity.

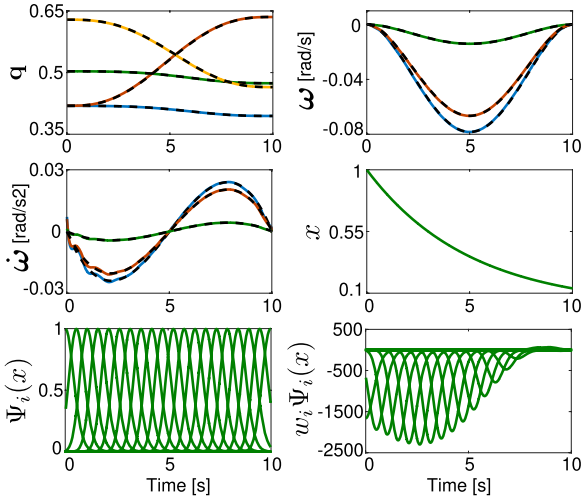

Figure 4. A unit quaternion DMP is used to generate a discrete motion connecting $\mathbf{q}_1$ and $\mathbf{g}_q$. The training data (black-dashed lines) are obtained from a minimum jerk trajectory connecting $\mathbf{q}_1$ and $\mathbf{g}_q$ in $T = 10$s and used to learn the weights w_i of 20 Gaussian basis functions equally distributed in time. The results of the parameter learning procedure are shown in the bottom right panel. The exponentially decaying phase variable is used as shown in the middle right panel. Results are obtained with the open-source implementation available at https://gitlab.com/dmp-codes-collection.

2.1.2.2. Rotation matrix DMP. In their work on orientation DMPs, Ude et al. (2014) extended DMP formulation in order to encode orientation trajectories represented in the form of rotation matrices $\mathbf{R}(t) \in \mathcal{SO}(3)$. Therefore, they rewrote (1) and (2) in the following form

$$\tau \dot{\boldsymbol{\eta}} = \alpha_z\left(\beta_z \operatorname{Log}^R\left(\mathbf{R}_g\mathbf{R}^\top\right) - \boldsymbol{\eta}\right) + \mathbf{f}_R(x), \tag{23}$$

$$\tau \dot{\mathbf{R}} = [\boldsymbol{\eta}]_\times \mathbf{R} \tag{24}$$

where $\mathbf{R}_g$ represents the goal orientation. $[\boldsymbol{\eta}]_\times$ is a skew symmetric matrix, such as $[\boldsymbol{\eta}]_\times^\top = -[\boldsymbol{\eta}]_\times$. The relation between the angular velocity and the 1st-time-derivative of the rotation matrix is given by

$$[\boldsymbol{\omega}]_\times = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} = \dot{\mathbf{R}}\mathbf{R}^\top \tag{25}$$

The function $\operatorname{Log}^R(\cdot) : \mathcal{SO}(3) \mapsto \mathbb{R}^3$ is given as follows

$$\operatorname{Log}^R(\mathbf{R}) = \begin{cases} [0,0,0]^\top, & \mathbf{R} = \mathbf{I} \\ \omega = \theta\mathbf{n}, & \text{otherwise,} \end{cases} \tag{26}$$

$$\theta = \arccos\left(\frac{\operatorname{trace}(\mathbf{R}) - 1}{2}\right), \mathbf{n} = \frac{1}{2\sin(\theta)}\begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{bmatrix}$$

The generated rotation matrices can be obtained by integrating (24) as follows

$$\mathbf{R}(t + \delta t) = \operatorname{Exp}^R\left(\delta t \frac{[\boldsymbol{\eta}]_\times}{\tau}\right)\mathbf{R}(t) \tag{27}$$

The function $\operatorname{Exp}^R(\cdot) : \mathbb{R}^3 \mapsto \mathcal{SO}(3)$ is given as follows

$$\operatorname{Exp}^R(t[\boldsymbol{\omega}]_\times) = \mathbf{I} + \sin(\theta)\frac{[\boldsymbol{\omega}]_\times}{\|\boldsymbol{\omega}\|} + (1 - cos(\theta))\frac{[\boldsymbol{\omega}]_\times^2}{\|\boldsymbol{\omega}\|^2}. \tag{28}$$

where $\theta(t) = t\|\boldsymbol{\omega}\|$ express the rotation angle within time t. An exemplar rotation matrix DMP is shown in Figure 5.

2.1.3. SPD matrices. Abu-Dakka and Kyrki (2020) generalized DMP formulation in order to encode robotic manipulation data profiles encapsulated in the form of SPD matrices. The importance of SPD matrices comes from the fact that many robotics data are encapsulated in such matrices, *for example,* full stiffness matrices, manipulability ellipsoids, and inertia matrices among others. By defining $\mathbf{X} \in \mathcal{S}^m_{++}$ as an arbitrary SPD matrix and $\boldsymbol{\Xi} = \{t_j, \mathbf{X}_j\}_{j=1}^{\mathfrak{T}}$ as the set of SPD matrices in one demonstration, where $\mathcal{S}^m_{++}$ defines the set of $m \times m$ SPD matrices. Afterward, we can rewrite (1) and (2) as follows

$$\tau\dot{\boldsymbol{\sigma}} = \alpha_z\left(\beta_z \mathbf{vec}\left(\mathbb{B}_{\mathbf{X}_j \mapsto \mathbf{X}_1}\left(\operatorname{Log}^+_{\mathbf{X}_j}(\mathbf{X}_g)\right)\right) - \boldsymbol{\sigma}\right) + \boldsymbol{\mathcal{F}}_+(x), \tag{29}$$

$$\tau\dot{\boldsymbol{\xi}} = \boldsymbol{\sigma} \tag{30}$$

where $\boldsymbol{\sigma} = \mathbf{vec}(\boldsymbol{\Sigma})$ is the Mandel representation of the symmetric matrix $\boldsymbol{\Sigma}$ and $\boldsymbol{\Sigma}$ is the time derivative of $\boldsymbol{\Xi}$ so that $\boldsymbol{\Sigma} \equiv \dot{\boldsymbol{\Xi}} = (\operatorname{Log}^+_{\mathbf{X}_{j-1}}(\mathbf{X}_j))/\delta t$. The function $\operatorname{Log}_{\mathbf{X}_{j-1}}(\mathbf{X}_j) : \mathcal{M} \mapsto \mathcal{T}_{\mathbf{X}_{j-1}}\mathcal{M}$ maps a point $\mathbf{X}_j$ in the manifold $\mathcal{M}$ to a point in the tangent space $\boldsymbol{\Delta} \in \mathcal{T}_{\mathbf{X}_{j-1}}\mathcal{M}$. *vec* $(\cdot)$ is a function that transforms a symmetric matrix into a vector using Mandel's notation, for example, a vectorization of a 2×2 symmetric matrix is as follows

$$vec\begin{pmatrix} a & b \\ b & d \end{pmatrix} = \begin{pmatrix} a \\ d \\ \sqrt{2}b \end{pmatrix} \tag{31}$$

$\boldsymbol{\xi}$ is the vectorization of $\boldsymbol{\Xi}$. $\mathbf{X}_g \in \mathcal{S}^m_{++}$ represents the goal of SPD matrix. $vec(\mathbb{B}_{\mathbf{X_j} \mapsto \mathbf{X}_1}(\operatorname{Log}^+_{\mathbf{X_j}}(\mathbf{X}_g)))$ is the vectorization of the transported symmetric matrix $\operatorname{Log}^+_{\mathbf{X_j}}(\mathbf{X}_g)$ over the geodesic from $\mathbf{X}_j$ to $\mathbf{X}_1$. Then we integrate (30) as follows

$$\widehat{\mathbf{X}}(t + \delta t) = \operatorname{Exp}^+_{\mathbf{X}(t)}\left(\frac{\mathbb{B}_{\mathbf{X}_1 \mapsto \mathbf{X}(t)}(mat(\boldsymbol{\sigma}(t)))}{\tau}\delta t\right) \tag{32}$$

where the function $mat(\cdot)$ is the inverse of $vec(\cdot)$ and denotes to the matricization using Mandel's notation. $\widehat{\mathbf{X}} \in \mathcal{S}^m_{++}$ represents the new SPD-matrices-based robot skills. The function $\operatorname{Exp}^+_{\mathbf{X}_{j-1}}(\boldsymbol{\Delta}) : \mathcal{T}_{\mathbf{X}_{j-1}}\mathcal{M} \mapsto \mathcal{M}$ maps a point $\boldsymbol{\Delta} \in \mathcal{T}_{\mathbf{X}_{j-1}}\mathcal{M}$ to a point $\mathbf{X}_j \in \mathcal{M}$, so that it lies on the

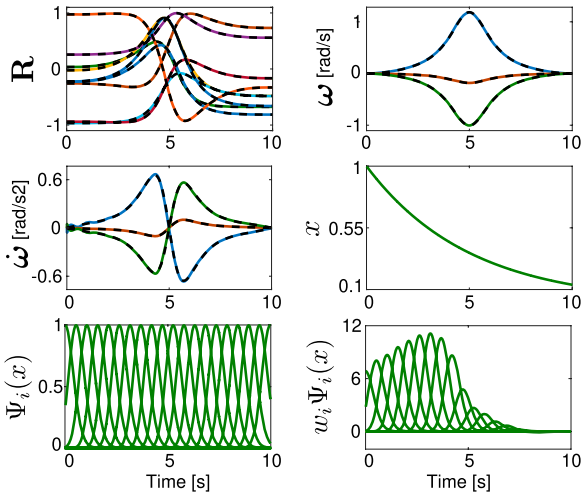

Figure 5. A rotation matrix DMP is used to generate a discrete motion connecting $\mathbf{R}_1$ and $\mathbf{R}_g$. The training data (black-dashed lines) are obtained from a minimum jerk trajectory connecting $\mathbf{R}_1$ and $\mathbf{R}_g$ in $T = 10$s and used to learn the weights w_i of 20 Gaussian basis functions equally distributed in time. The results of the parameter learning procedure are shown in the bottom right panel. The exponentially decaying phase variable is used as shown in the middle right panel.

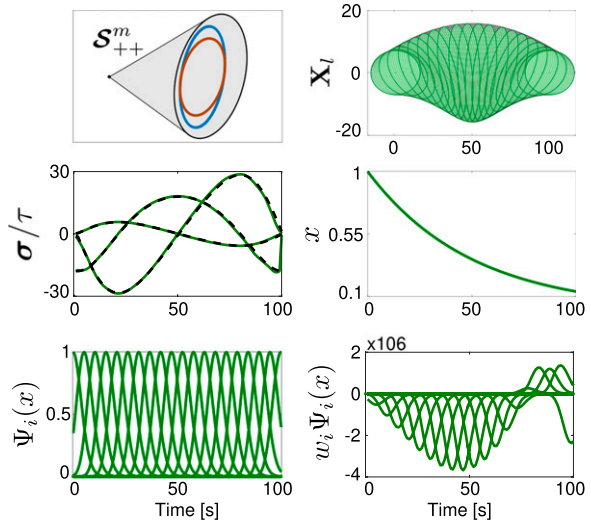

Figure 6. An SPD DMP is used to generate a discrete motion connecting $\mathbf{X}_1$ and $\mathbf{X}_g$. The training data (black-dashed lines) are obtained from a minimum jerk trajectory connecting $\mathbf{X}_1$ and $\mathbf{X}_g$ in $T = 100$s and used to learn the weights w_i of 20 Gaussian basis functions equally distributed in time. The cone in the upper left corner represents the manifold of SPD data and includes the geodesic of the SPD profile. The results of the parameter learning procedure are shown in the bottom right panel. The exponentially decaying phase variable is used as shown in the middle right panel. Results are obtained with the open-source implementation available at https://gitlab.com/dmp-codes-collection.

geodesic starting from $\mathbf{X}_{j-1} \in \mathcal{S}^m_{++}$ in the direction of $\mathbf{\Delta}$. An exemplar SPD DMP is shown in Figure 6.

Moreover, Abu-Dakka and Kyrki (2020) rewrote (13) for smooth goal adaptation in case of sudden goal switching as follows

$$\tau \dot{\mathbf{g}}_+ = \alpha_g \mathrm{Log}^+_{\mathbf{g}_{new}}(\mathbf{g}_+) \tag{33}$$

so $\mathbf{g}$ now is updated continually.

2.2. Periodic DMP

The periodic DMP (sometimes called rhythmic DMP) is used when the encoded motion follows a rhythmic pattern.

2.2.1. Classical DMP. The classical periodic (or rhythmic) DMPs were first introduced by Ijspeert et al. (2002b), where they redefined the second-order differential equation system described in (1) and (2) as follows

$$\dot{z} = \Omega(\alpha(\beta(-y) - z) + f(\phi)), \tag{34}$$

$$\dot{y} = \Omega z, \tag{35}$$

$$\tau \dot{\phi} = 1 \tag{36}$$

where Ω is the frequency and y is the desired periodic trajectory that we want to encode with a DMP. The main difference between periodic DMPs and point-to-point DMPs is that the time constant related to trajectory duration is replaced by the frequency of trajectory execution (refer to Ijspeert et al., 2013 and 2002b for details). In addition, the periodic DMPs must ensure that the initial phase ($\phi = 0$) and the final one ($\phi = 2\pi$) coincide in order to achieve smooth transition during the repetitions.

Similar to (4), $f(\phi)$ is defined with N Gaussian kernels according to the following equation

$$f(\phi) = \frac{\sum_{i=1}^{N} \Psi_i(\phi) w_i r}{\sum_{i=1}^{N} \Psi_i(\phi)}, \tag{37}$$

$$\Psi_i(\phi) = \exp(h(\cos(\phi - c_i) - 1)) \tag{38}$$

where the weights are uniformly distributed along the phase space and r is used to modulate the amplitude of the periodic signal (Gams et al., 2009; Ijspeert et al., 2002b) (if not used, it can be set to $r = 1$ [Peternel et al. 2016]).

Similar to discrete DMPs, LWR (Schaal and Atkeson, 1998) can be used to update the weight to learn a desired trajectory. In a standard periodic DMP setting (Gams et al., 2009; Ijspeert et al., 2002b), the desired shape f_d is approximated by solving

$$f_d(t_J) = \frac{\ddot{y}_d(t_J)}{\Omega^2} - \alpha_z\left(\beta_z(-y_d(t_J)) - \frac{\dot{y}_d(t_J)}{\Omega}\right) \tag{39}$$

where y_d is some demonstrated input trajectory that needs to be encoded. The weights w_i can be updated using the recursive least-squares method (Schaal and Atkeson, 1998)

with forgetting factor λ based on the error between the desired trajectory shape and the currently learned shape

$$w_i(t_{J+1}) = w_i(t_J) + \Psi_i P_i(t_{J+1}) r e_r(t_J), \tag{40}$$

$$e_r(t_J) = f_d(t_J) - w_i(t_J) r, \tag{41}$$

$$P_i(t_{J+1}) = \frac{1}{\lambda}\left(P_i(t_J) - \frac{P_i(t_J)^2 r^2}{\frac{\lambda}{\Psi_i} + P_i(t_J) r^2}\right) \tag{42}$$

The initial value of the parameters is w_i (0) = 0 and $P_i(0) = 1$. The forgetting factor determines the rate of weight changes. Refer to Schaal and Atkeson (1998) for details on parameter setting. An exemplar rhythmic DMP is shown in Figure 7.

The classical periodic DMP described by (34)–(36) does not encode the transient motion needed to start the periodic one. Transients are important in several applications like humanoid robot walking where usually the first step made from a rest position is a transient needed to start the periodic motion. To overcome this limitation, Nakanishi et al. (2004) presented a formulation of rhythmic DMPs including transient to achieve limit cycle with an application to biped locomotion. Ernesti et al. (2012) modified the classical formulation of periodic DMPs to explicitly consider transients as a motion trajectory that converges toward the limit cycle (i.e., periodic) one.

2.2.2. Orientation DMP. The same argument used in Section 2.1.2 is valid here too. Unlike Cartesian position, the elements of orientation representations like unit quaternion or rotation matrix are constrained. In this section, we present approaches that extend the classical periodic DMP formulation to represent periodic Cartesian orientations.

2.2.2.1. Quaternion periodic DMP. To encode unit quaternion trajectories accurately, the dynamic system in equations (34) and (35) is reformulated by Abu-Dakka et al. (2021), taking inspiration from the research on discrete quaternion DMPs (Abu-Dakka et al., 2015a; Ude et al., 2014).

$$\dot{\boldsymbol{\eta}} = \boldsymbol{\Omega}\left(\alpha_z\left(\beta_z 2\,\mathrm{Log}^q\left(\mathbf{g}_q * \bar{\mathbf{q}}\right) - \boldsymbol{\eta}\right) + \mathbf{f}_q(\phi)\right), \tag{43}$$

$$\dot{\mathbf{q}} = \boldsymbol{\Omega}\frac{1}{2}\boldsymbol{\eta} * \mathbf{q} \tag{44}$$

where $\mathbf{g}_q$ can take the form of either the identity orientation $1 + [0\ 0\ 0]^\top$ or the average of the demonstration quaternion profile. $\boldsymbol{\Omega}$ is the 3 × 3 diagonal matrix of frequencies, and the nonlinear forcing term is as follows

$$\mathbf{f}(\phi) = \mathbf{A}_r \frac{\sum_{i=1}^{N} \mathbf{w}_i \Psi_i(\phi)}{\sum_{i=1}^{N} \Psi_i(\phi)} \tag{45}$$

where $\mathbf{w}_i$ are the weights needed to follow any given rotation profile. We estimate the weights by

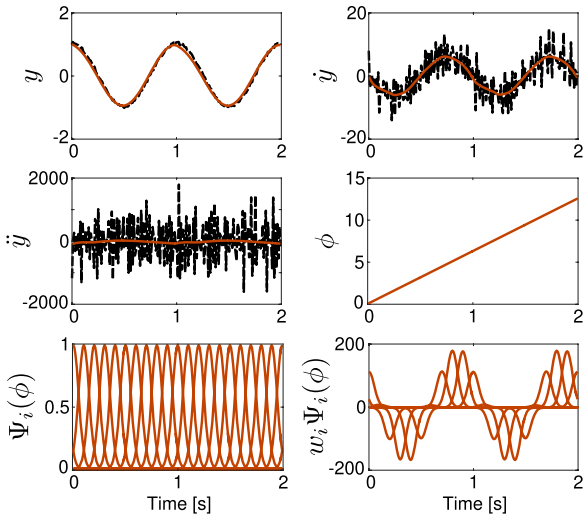

Figure 7. A classical DMP is used to reproduce a rhythmic motion (brown solid line in the top left panel). The desired trajectory is obtained by adding Gaussian noise to $y_d = \cos(2\pi t)$ with $t \in [0, 2]$ s and computing the numerical derivatives with $\delta t = 0.01$s (black-dashed lines). The forcing term is obtained as the weighted summation of 20 Gaussian bases equally distributed in time (bottom left panel). The results of the parameter learning procedure are shown in the bottom right panel. Results are obtained with the open-source implementation available at https://gitlab.com/dmp-codes-collection/dmp-classical.

$$\frac{\sum_{i=1}^{N} \mathbf{w}_i \Psi_i(\phi)}{\sum_{i=1}^{N} \Psi_i(\phi)} = \mathbf{A}_r^{-1}\left(\boldsymbol{\Omega}^{-1}\dot{\boldsymbol{\omega}} - \left(\alpha_z\left(\beta_z 2\,\mathrm{Log}^q\left(\mathbf{g}_q * \bar{\mathbf{q}}\right) - \omega\right)\right)\right), \tag{46}$$

where $\mathbf{A}_r$ is the 3 × 3 diagonal matrix of amplitude modulators.

The integration of (44) is done similarly as in (19), that is,

$$\mathbf{q}(t + \delta t) = \mathrm{Exp}^q\left(\frac{\delta t}{2}\boldsymbol{\Omega}\,\boldsymbol{\eta}(t)\right) \tag{47}$$

2.3. Formulation summary

A summary of the existence DMP formulations mentioned in the earlier sections is shown in Table 3. The table shows the variations of the formulation in its standard shape based on the space that they are applied to. However, the modifications of this standard shape (e.g., adding a coupling term) are discussed in the next section as an extension of the DMP formulations.

3. DMP extensions

3.1. Generalization

A desirable property of motion primitives is the ability to generalize to unforeseen situations. In this section, we present approaches that allow to adapt DMP motion trajectories to novel executive contexts.

Table 3. Summary of DMP basic formulations.

Type of movement	Space	System of equations	Reference	Short description
Discrete	$\mathbb{R}$	$\tau \dot{z} = \alpha_z(\beta_z(g-y)-z)+f(x)$ $\tau y = z$ $\tau x = -\alpha_x x$	Equations (1)–(3) (Ijspeert et al., 2002c)	A single DoF, discrete motion trajectory is encoded into a linear, second-order dynamical system with an additive, nonlinear forcing term. Convergence to the desired goal g is ensured by a vanishing phase variable x.
	$\mathcal{S}^3$	$\tau \dot{\boldsymbol{\eta}} = \alpha_z(\beta_z 2\,\mathrm{Log}^q(\mathbf{g}_q{}^*\overline{\mathbf{q}}) - \boldsymbol{\eta}) + \boldsymbol{f}_{\mathrm{q}}(x)$ $\tau \dot{\mathbf{q}} = 1/2\boldsymbol{\eta}^*\mathbf{q}$	Equations (16) and (17) (Abu-Dakka et al., 2015a)	A quaternion-based orientation trajectory (3 DoFs) is encoded into a second-order dynamical system with an additive, nonlinear forcing term. The error definition complies with the geometry of the unit quaternion space.
	$\mathcal{SO}(3)$	$\tau\dot{\boldsymbol{\eta}} = \alpha_z(\beta_z \mathrm{Log}^R(\mathbf{R}_g{}^*\mathbf{R}^\top) - \boldsymbol{\eta}) + \mathbf{f}_R(x)$ $\tau \dot{\mathbf{R}} = [\boldsymbol{\eta}]_\times \mathbf{R}$	Equations (23) and (24) (Ude et al., 2014)	A rotation matrix-based orientation trajectory (3 DoFs) is encoded into a second-order dynamical system with an additive, nonlinear forcing term. The error definition complies with the geometry of the rotation matrices space.
	$\mathcal{S}^m_{++}$	$\tau\dot{\boldsymbol{\sigma}} = \alpha_z(\beta_z \mathbf{vec}(\mathbb{B}_{\mathbf{X}_l \mapsto \mathbf{X}_1}(\mathrm{Log}^+_{\mathbf{X}_l}(\mathbf{X}_g))) - \boldsymbol{\sigma}) + \mathcal{F}(x)$ $\tau\dot{\boldsymbol{\xi}} = \boldsymbol{\sigma}$	Equations (29) and (30) (Abu-Dakka and Kyrki, 2020)	An SPD matrices trajectory, m (m + 1)/2 DoFs, is encoded into a second-order dynamical system with an additive, nonlinear forcing term. The error definition complies with the geometry of the SPD matrices space.
Periodic	$\mathbb{R}$	$\dot{z} = \Omega(\alpha(\beta(-y)-z)+f(\phi))$ $\dot{y} = \Omega z$ $\tau\dot{\phi} = 1$	Equations (34)–(36) (Ijspeert et al., 2002b)	A single DoF, periodic motion trajectory is encoded into a linear, second-order dynamical system with an additive, nonlinear forcing term. The resulting system generates a stable limit cycle.
	$\mathcal{S}^3$	$\dot{\boldsymbol{\eta}} = \boldsymbol{\Omega}(\alpha_z(\beta_z 2\,\mathrm{Log}^q(\mathbf{g}_q{}^*\overline{\mathbf{q}}) - \boldsymbol{\eta}) + \mathbf{f}_q(\phi))$ $\dot{\mathbf{q}} = \boldsymbol{\Omega} 1/2\boldsymbol{\eta}^*\mathbf{q}$	Equations (43) and (44) (Abu-Dakka et al., 2021)	A quaternion-based orientation trajectory (3 DoFs) is encoded into a second-order dynamical system with an additive, nonlinear forcing term. The error definition complies with the geometry of the unit quaternion space.

3.1.1. Start, goal, and scaling. Classical DMPs are time-invariant, meaning that time scaling $\varsigma\tau$ with $\varsigma > 0$ generates topologically equivalent trajectories (Ijspeert et al., 2013). Using a simple modification of the transformation system, namely, substituting (1) with

$$\tau \dot{z} = \alpha_z(\beta_z(g-y)-z) + (g-y_0)f(x) \tag{48}$$

where y_0 is the initial value of y, Ijspeert et al. (2013) show that DMPs are also scale-invariant, meaning that the scaling of the movement amplitude $\varsigma(g-y_0)$ with $\varsigma > 0$ generates topologically equivalent trajectories. The purpose of the green color used in (48) is to highlight differences *w.r.t.* (1). Kulvicius et al. (2011) use a similar strategy and scale the forcing term by the ratio between the start- and end-points of the demonstrated and desired trajectories. Apart from generating scaled—in time and space—versions of the demonstrated motion trajectory, classical DMPs also generalize to different initial/target states. However, the classical formulation—and its extension in (48)—may exhibit dangerous behaviors like over-amplification of the trajectory when reaching a different target and high accelerations when switching to a different target on-line (Ijspeert et al. 2013; Pastor et al. 2009). To alleviate the second issue, Ijspeert et al. (2013) replaced hard goal switches with the smooth switching law as in (13). However, the over-amplification issue still remains. Moreover, a DMP that uses (48) fails to learn motions with the same initial and target states (i.e., $g = y_0$, $z_0 = 0 \rightarrow y(t) = y_0 = g\ \forall t$).

In order to remedy those issues, Pastor et al. (2009) proposed to modify the transformation system as follows

$$\tau \dot{z} = \alpha_z(\beta_z(g - y - (g - y_0)x + f(x)) - z) \tag{49}$$

where the green color is used to highlight differences between (49) and (1). The most important change in this formulation is the term $(g - y_0)x$ that has several benefits. It prevents high accelerations at the beginning of the motion ($(g - y - (g - y_0)x = 0$ for $t = 0$) or when the goal is close to the initial state. It allows to reproduce motions with the same initial and target states and prevents over-amplifications and trajectory mirroring effects[1] when changing the goal. Hoffmann et al. (2009) derived a multidimensional representation of (49) from the behavior of the spinal force fields in frogs.

The goal can also change over time and, in this case, the tracking performance of the DMP mostly depends on the gains α_z and β_z. As proposed by Koutras and Doulgeri (2020b), the tracking performance can be improved by adapting the temporal scaling τ.

Dragan et al. (2015) showed that DMPs solve a trajectory optimization problem in order to minimize a particular Hilbert norm between the demonstration and the new trajectory subject to start and goal constraints. In this light, DMP adaptation capabilities to different start and goals can be improved by choosing (or learning) a proper Hilbert norm that reduces the deformation in the retrieved trajectory.

3.1.2. Via-points. A via-point can be defined as a point in the state space where the trajectory has to pass. Failing to pass a via-point may cause the robot to fail the task execution. Therefore, having a motion primitive representation with the capability of modulating the via-points is important in robotic scenarios. It is not surprising that researchers have extended the DMP formulation to consider intermediate via-points in the trajectory generation process.

Ning et al. (2011, 2012) extend the classical DMP to satisfy position and velocity constraints at the beginning and at the end of a sample trajectory. Their approach to traverse via-points consists of creating a sample trajectory by combining locally linear trajectories connecting the via-points. This sample trajectory is used to fit a DMP that is constrained to pass the via-points.

Weitschat and Aschemann (2018) considered each via-point as an intermediate goal (*via-goal*) g_v for $v = 1, \ldots, V$ to reach. The last via-goal g_V corresponded to the target state of the DMP. In their formulation, they defined a variable goal as follows

$$g_{via}(x) = \sum_{v=1}^{V} \Psi_v(x) g_v \tag{50}$$

where $\Psi_v(x)$ are the Gaussian basis functions centered at the time corresponding to the $v-$ th via-goal. The effectiveness of the approach is demonstrated in a task where the robot has to reach a different target while preventing possible self-collisions of the end-effector with the robot body. To this end, the authors placed the via-goals along the trajectory used to learn the DMP, forcing the generated trajectory to stay close to the demonstration while reaching the new target.

The problem of generalizing to via-point close (interpolation) and far (extrapolation) from the demonstration is faced by Zhou et al. (2019). Their approach, namely, *Via-points Movement Primitives (VMPs)*, combines the benefits of DMP and Probabilistic Movement Primitives (ProMPs) (Paraschos et al., 2013). Authors assumed that the motion trajectory is generated as follows

$$y_{vmp}(x) = e(x) + f_{vmp}(x) \tag{51}$$

where x is the phase variable defined as in (3) and the elementary trajectory $e(x)$ can be defined as the linear attractor $e(x) = (y_0 - g)x$. The shape modulation term $f_{vmp}(x)$ is defined as follows

$$f_{vmp}(x) = \sum_{i=1}^{N} w_i \Psi_i(x) + \epsilon_f \tag{52}$$

where the Gaussian kernels $\Psi_i(x)$ are defined as in (5), w_i are learnable weights, and ϵ_f is the Gaussian noise. As detailed in Paraschos et al. (2013), learning the shape modulation term $f_{vmp}(x)$ means Learning from Demonstrations (LfDs) the prior probability distribution of the weights w_i. Having separated the generated trajectory into two parts like in (51) allows to adopt different strategies to pass a via-point y_v at x_v. Zhou et al. (2019) proposed to modify the shape modulation term for interpolation cases—when the via-point is "close" to the demonstrations. In extrapolation cases, instead, the elementary trajectory $e(x)$ is rewritten as the polygonal line connecting y_0, y_v, and g. This approach easily generalizes to the case of multiple via-points. VMPs are experimentally compared with ProMPs, showing better performance especially in extrapolation cases.

3.1.3. Task parameters. Reaching a different goal, or passing through via-points, may not be enough to successfully execute a task in a different context. Approaches presented in this section adapt the DMP motion to new situations by adjusting the weights w_i of the forcing term (4), which modifies the entire DMP trajectory.

Weitschat et al. (2013) considered that L demonstrations are given, each encoded in a different DMP. In order to generalize, for instance, to a new goal g_{new}, they proposed to interpolate the weights of nearby DMPs, *that is.,* DMPs that reached points around g_{new}. In formulas

$$\mathbf{w}_{new} = \frac{\sum_{\forall o : d_o < d_{max}} \mathbf{w}_o d_o^{-1}}{\sum_{\forall o : d_o < d_{max}} d_o^{-1}} \tag{53}$$

where o represents the indices of the nearby DMPs for which it holds that $d_o < d_{max}$, d_o is the distance (or, more

generally, a cost) between g_{new} and g_o, d_{max} is the maximum distance to consider 2 DMPs close, and $\mathbf{w}_{new} = [w_{1,new}, \ldots, w_{N,new}]^\top$ and $\mathbf{w}_o = [w_{1,o}, \ldots, w_{N,o}]^\top$ are the new weights and the weight of nearby DMPs, respectively.

The approach by Forte et al. (2011, 2012) also assumes that L demonstrations are given and that each demonstration is encoded in a different DMP. Further, the authors exploited GP (Rasmussen and Williams, 2006) to learn a mapping between the query points q_l for $l = 1, \ldots, L$ (e.g., the goal of each DMP) and the DMP parameters $[\mathbf{w}_l, g_l, \tau_l]$. Given the new query point q_{new}, Gaussian Process Regression (GPR) (Rasmussen and Williams, 2006) is used to retrieve the new set of parameters $[\mathbf{w}_{new}, g_{new}, \tau_{new}]$, that can be used to generate a DMP motion. This approach builds on previous work (Gams and Ude, 2009; Ude et al., 2010) where raw data from the L demonstrations are stored in memory and LWR is used to generate new DMP weights. Alizadeh et al. (2016) extend the approach in Ude et al. (2010) to retrieve the DMP weights even when the task parameters are partially observable. Finally, Zhou and Asfour (2017) extend the approach in Ude et al. (2010) to consider task-specific costs while learning the mapping between query points and DMP weights.

The aforementioned approaches follow a 2-steps procedure where first the shape parameters $\mathbf{w}$ are estimated given new task parameters and then execute the DMP. Matsubara et al. (2010, 2011) augmented the forcing term with a style parameter used to capture human variability across multiple demonstrations. Stulp et al. (2013) proposed a 1-step procedure where the DMP forcing term (4) is reformulated to explicitly depend on the task parameters. Their experiments show that a 1-step approach gives more freedom *w.r.t.* the used regression technique and increases the generalization performance. Along the same line, Pervez and Lee (2018) embedded task parameters directly in the forcing term. The authors proposed to use a mixture of Gaussians (Cohn et al., 1996) to learn the mapping between the task parameters (e.g., new goal and the height of an obstacle) and the forcing term. Given a new query task parameter, regression over the mixture of Gaussians is used to retrieve the forcing term parameters and generate the DMP motion. The approach is tested on a variety of tasks including sweeping and striking and additionally compared with the approaches presented by Forte et al. (2012), Stulp et al. (2013), and Ude et al. (2010) showing better performance, especially in extrapolation.

A Mixture of Motor Primitives (MoMP) is proposed in Mülling et al. (2010, 2013) and used to generalize table tennis skills like hitting and batting a ball. MoMP uses an augmented state that contains robot position and velocity as well as the meta-parameters of the table tennis task like the expected hitting position and velocity. The adapted motion is generated by the weighted summation of L DMPs and the responsibility of each DMP, representing the probability that a particular DMP is the correct one for the sensed augmented state, is also learned from data.

In high DoF systems, like humanoid robots, it is non-trivial to find a relationship between the task and the DMP parameters. This is especially true when the DMPs are used to encode joint space trajectories. Bitzer and Vijayakumar (2009) showed that such a relationship is easier to find in a latent (lower-dimensional) space obtained from training data. Therefore, they used dimensionality reduction techniques to find the latent space where to fit a DMP and show that the interpolation of DMP weights in the latent space results in better generalization performance.

3.2. Joining multiple DMPs

An important and desired feature of any motion primitive representation is the possibility to combine basic movements to obtain more complex behaviors (Schaal, 1999). We review here three prominent approaches developed to smoothly join a sequence of DMPs. In this tutorial, we name the approach by Pastor et al. (2009) as *velocity threshold*, that in Kober et al. (2010b) as *target crossing*, and that in Kulvicius et al. (2011, 2012) as *basis functions overlay*. Some of the presented approaches modify the DMP formulations in Sections 2.1.1 and 2.1.2. The main differences are highlighted with green text. The 3 approaches have been implemented in MATLAB for both position (Section 2.1.1) and orientation (Section 2.1.2) DMPs. The source code is included in our public repository (see Table 4). Results on synthetic data are shown in Figures 8–11.

3.2.1. Velocity threshold. A properly designed DMP reaches the desired target with zero velocity and acceleration, *that is,* once a DMP is fully executed the robot comes to a full stop. This also implies that the velocity "close" to the target is continuously decreasing. Using this property, Pastor et al. (2009) propose to combine successive DMPs by simply terminating the current DMP when the velocity is below a certain threshold and then starting the following primitive. When executing a single DMP, it is a common practice to initialize its velocity to zero—the robot is assumed to be still. In principle, this initialization can be used to sequence multiple DMPs (Lioutikov et al., 2016; Xu and Wang, 2004), but it may generate discontinuities if the robot does not fully stop in between two consecutive primitives. To prevent these discontinuities, Pastor et al. initialized the state of the current DMP with that of the previous one.

The velocity threshold approach is simple and effective since it directly applies to the DMP formulations in Sections 2.1.1 and 2.1.2. For instance, Saveriano et al. (2019) showed how to join multiple quaternion DMPs[2] (see Section 2.1.2.1) with the velocity threshold approach.

Results in Figure 8 are obtained when the velocity threshold is applied to merge 2 DMPs separately trained to fit minimum jerk trajectories (black-dashed lines). Figures 8(a)–(e) show the position and Figures 8(f)–(j)

Table 4. Open-source implementations of DMP-based approaches that we have released to the community. The source code for each approach is available at https://gitlab.com/dmp-codes-collection.

Approach	Author	Language	Description
Discrete DMP	*Fares J. Abu-Dakka*	`C++`	An implementation for discrete DMP based on the work in Abu-Dakka et al. (2015a) and Ude et al. (2010, 2014).
Periodic DMP	*Luka Peternel*	`Python`	An implementation for periodic DMP based on the work in Peternel et al. (2016).
Unit quaternion DMP	*Fares J. Abu-Dakka*	`MATLAB and C++`	An implementation for unit quaternion DMP and goal switching based on the work in Abu-Dakka et al. (2015a) and Ude et al. (2014).
SPD DMP	*Fares J. Abu-Dakka*	`MATLAB`	An implementation for SPD DMP and goal switching based on the work in Abu-Dakka and Kyrki (2020).
Joining DMPs	*Matteo Saveriano*	`MATLAB`	An implementation for joining multiple DMPs based on the work in Saveriano et al. (2019).
Coupling-force DMPs	*Aljaz Kramberger*	`MATLAB`	An implementation for discrete DMPs and force coupling terms based on the work in Kramberger et al. (2018).

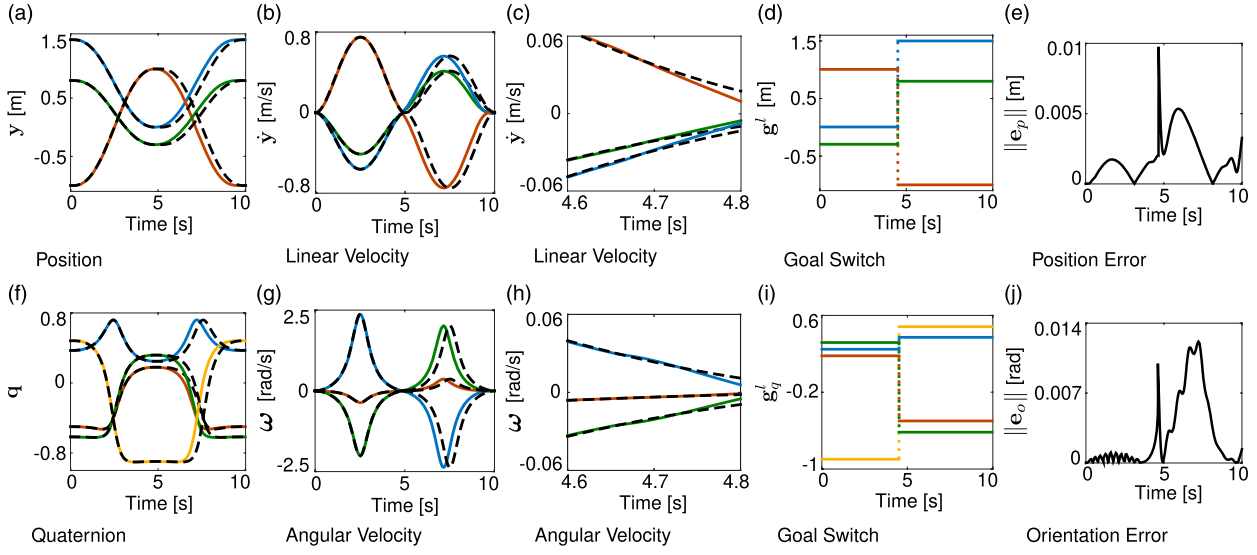

Figure 8. Results obtained by applying the zero velocity switch approach to join two DMPs trained on synthetic data. The training trajectory for the position and the orientation are shown as black-dashed lines in (a)–(b) and (f) –(g), respectively. Results are obtained with the open-source implementation available at https://gitlab.com/dmp-codes-collection.

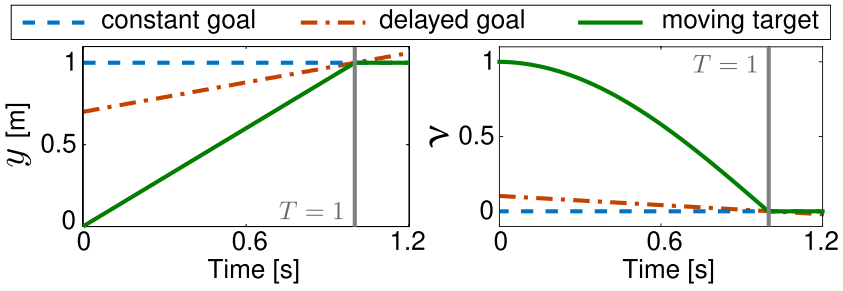

Figure 9. Constant goal, moving target, and delayed goal obtained with $y(0) = 0$ [m]; $g = 1$ [m]; $\dot{\hat{y}} = 0.3$ [m/s] (left); and $\mathbf{q}(0) = 1 + [0, 0, 0]^\top$, $\mathbf{g}_q = 0 + [1, 0, 0]^\top$, and $\hat{\boldsymbol{\omega}} = [0.2, 0.2, 0.2]^\top$ [rad/s] (right). The sampling time is $\delta t = 0.01$ [s]. Only the scalar part [*ieqn*7] of the quaternion is shown for a better visualization.

the orientation (unit quaternion) parts of the motion. The merged trajectory is generated by following the first DMP until the distance from the via-point is below 0.01 [m] and 0.01 [rad]. As shown in Figures 8(d) and (i), the switch occurs after about 4.7 [s]. Figures 8(e) and (j) show that the desired trajectory is accurately reproduced. More or less accurate trajectories can be obtained by tuning the distance from the via-point. However, the value of this

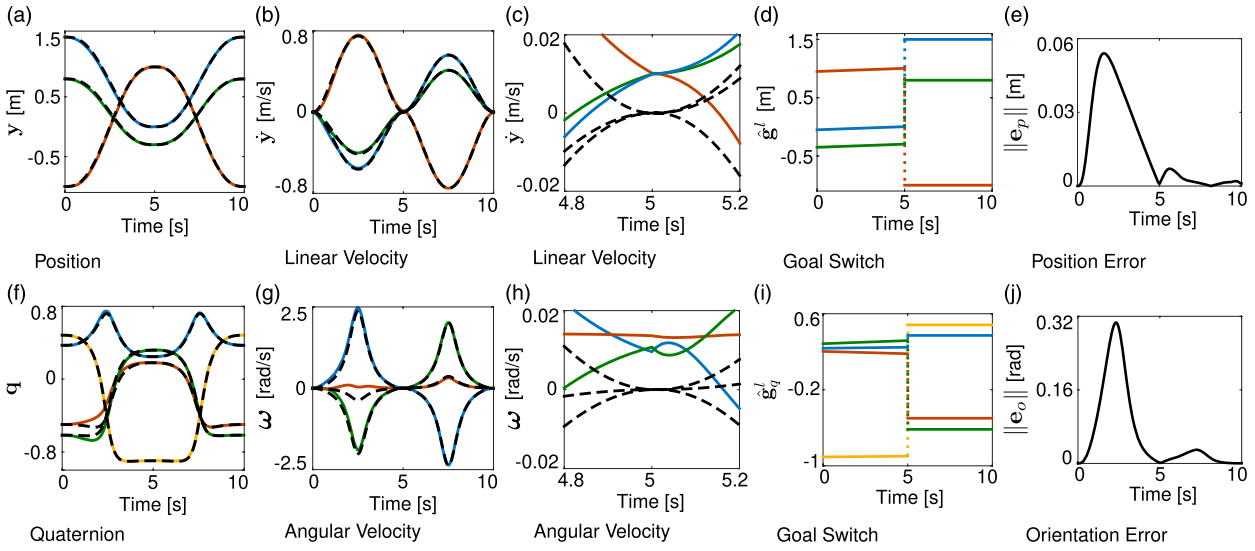

Figure 10. Results obtained by applying the target crossing approach to join two DMPs trained on synthetic data. The training trajectory for the position and the orientation are shown as black-dashed lines in (a)–(b) and (f)–(g), respectively. Results are obtained with the open-source implementation available at https://gitlab.com/dmp-codes-collection.

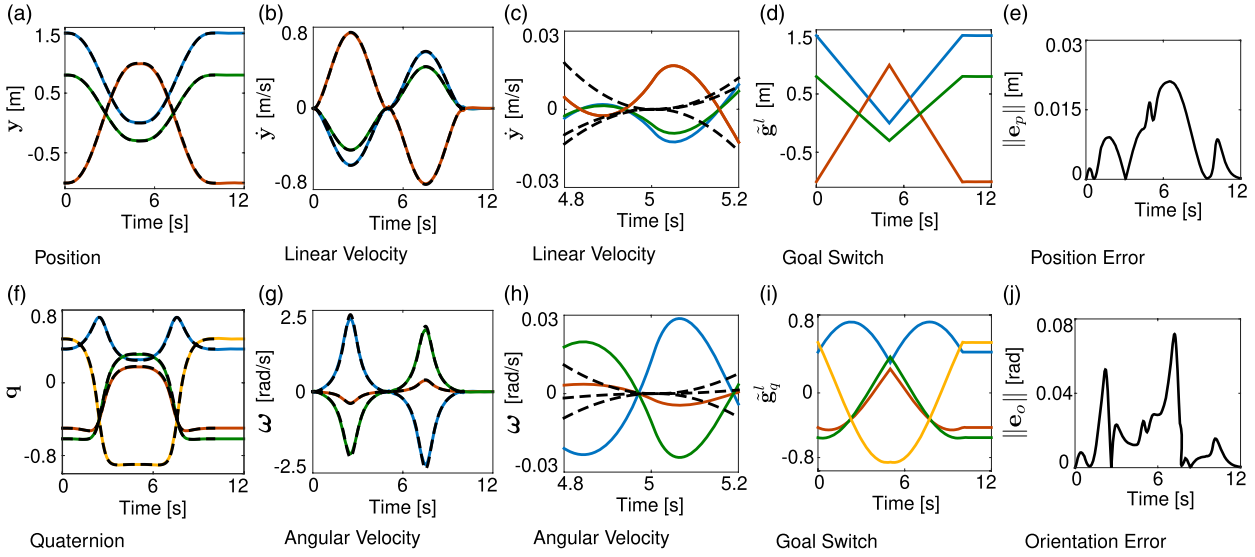

Figure 11. Results obtained by applying the basis functions overlay approach to join two DMPs trained on synthetic data. The training trajectory for the position and the orientation are shown as black dashed lines in (a)–(b) and (f)–(g), respectively. Results are obtained with the open-source implementation available at https://gitlab.com/dmp-codes-collection.

distance is the time duration of the generated trajectory—a bigger (smaller) distance results in a shorter (longer) trajectory. For instance, in the considered case, the total motion ends after 9.5 [s] while the demonstration lasts for 10 [s]. Depending on the application, the time difference may cause failures; therefore, it has to be taken into account. Finally, the velocity threshold approach may generate discontinuities if the target of the current DMP is far from the demonstrated initial point of the following primitive.

3.2.2. Target crossing. There exist movements like hitting or batting that are correctly executed only if the target is reached with a non-zero velocity. To this end, Kober et al. (2010b) extend the classical DMP formulation in Section 2.1.1 to let the DMP to track a target moving at a given velocity. In their approach, the DMP passes the target with a given velocity exactly after T seconds. To achieve this, the acceleration in (1) is re-written as follows

$$\tau\dot{z} = (1-x)\alpha_z\left(\beta_z(\widehat{g}-y)+\tau\left(\dot{\widehat{y}}-\dot{y}\right)\right)+f(x) \qquad (54)$$

where $\dot{\widehat{y}}_m$ is the desired velocity of the moving target $\widehat{g}$, which is defined as follows

$$\widehat{g} = \widehat{g}(0) - \dot{\widehat{y}}\frac{\tau \ln(x)}{\alpha_x}, \tag{55}$$

$$\widehat{g}(0) = g - \frac{T\dot{\widehat{y}}}{\tau} \tag{56}$$

By inspecting (55) and (56), and considering that the term $-\tau \ln(x)/\alpha_x$ represents the elapsed time if x is the phase defined in (3), it is possible to show that the moving target $\widehat{g}$ is designed to reach the goal g after T seconds, *that is,* $\widehat{g}(T) = g$ (Figure 9-*left*). The initial position of the moving target $\widehat{g}(0)$ is obtained by moving the goal position g for T seconds at constant velocity $-\dot{\widehat{y}}$. High accelerations at the beginning of the movement are avoided by the pre-factor $(1 - x)$ which is set to zero at the beginning of the motion ($x(0) = 1$). The approach by Nemec and Ude (2012) combines a moving target and a particular initialization of the subsequent DMP to ensure continuity of the movement up to second-order derivatives.

Saveriano et al. (2019) extended this idea to quaternion DMP. The angular acceleration in (16) is modified as follows

$$\begin{aligned}\tau \dot{\boldsymbol{\eta}} = (1 - x)\alpha_z\left(\beta_z 2\,\mathrm{Log}^q\left(\widehat{\mathbf{g}}_q * \overline{\mathbf{q}}\right) + \tau(\widehat{\boldsymbol{\omega}} - \boldsymbol{\omega})\right. \\ \left. + \mathbf{f}_q(x),\right.\end{aligned} \tag{57}$$

where $\widehat{\boldsymbol{\omega}}$ is the angular velocity of the moving quaternion target $\widehat{\mathbf{g}}_q$ and $2\,\mathrm{Log}^q(\widehat{\mathbf{g}}_q * \overline{\mathbf{q}})$ measures the error between the current orientation $\mathbf{q}$ and $\widehat{\mathbf{g}}_q$. The pre-factor $(1 - x)$ is used to avoid high angular accelerations at the beginning of the motion. The moving target for the quaternion DMPs is defined as follows

$$\begin{aligned}\widehat{\mathbf{g}}_q &= \mathrm{Exp}^q\left(-\frac{\tau \ln(x)}{2\alpha_x}\widehat{\boldsymbol{\omega}}\right) * \widehat{\mathbf{g}}_q(0), \\ \widehat{\mathbf{g}}_q(0) &= \mathrm{Exp}^q\left(-\frac{T}{2}\widehat{\boldsymbol{\omega}}\right) * \mathbf{g}_q,\end{aligned} \tag{58}$$

where $\mathbf{g}_q$ is the goal quaternion, T is the time duration of the DMP, and the exponential map $\mathrm{Exp}^q(\cdot)$ is defined in (20). As shown in Figure 9-*right*, the moving target $\widehat{\mathbf{g}}_q$ reaches the goal orientation after T seconds, *that is,* $\widehat{\mathbf{g}}_q(T) = \mathbf{g}_q$. This can be easily verified by considering that the initial value of the moving target $\widehat{\mathbf{g}}_q(0)$ is computed by moving the goal orientation $\mathbf{g}_q$ for T seconds at the desired velocity $-\widehat{\boldsymbol{\omega}}$.

The presented *target crossing* approach allows crossing the target after T seconds. Assuming to have two DMPs with time duration T^1 and T^2, respectively, one can join them by running the first DMP for T^1 seconds and then switching to the second one. As for the velocity threshold approach, possible discontinuities at the switching point are prevented by initializing the state of DMP_2 with the final state of DMP_1. This procedure can be repeated to join $L \geq 2$ consecutive DMPs.

Results in Figure 10 are obtained when the velocity threshold is applied for merging 2 separately trained DMPs to fit the minimum jerk trajectories (black-dashed lines). Figures 10(a)–(e) show the position and Figures 10(f)–(j) the orientation (unit quaternion) parts of the motion. The merged trajectory is generated by following the first DMP for $T^1 = 5$s and then switching to the second one. The required intermediate velocity is set to 0.01m/s (rad/s for the orientation) in each direction. The generated trajectory reaches the goal in 10s, *that is,* demonstration and execution times are the same. As required, the via-point is crossed at $T = 5$s with the desired velocity (Figures 10(c) and (h)). However, the non-zero crossing velocity introduces a deformation in the first part of the trajectory (Figures 10(e) and (j)).

3.2.3. Basis functions overlay. The approach by Kulvicius et al. (2011, 2012) combines multiple DMPs into a complex one, guaranteeing a smooth transition between the primitives by ensuring that the basis functions composing $f(x)$ in (4) overlap at the switching instances. First of all, Kulvicius et al. adopted a sigmoidal phase variable in (14) instead of the exponentially decaying one (3). As discussed in Section 2.1.1.3, the sigmoidal phase is ≈1 for the large part of the motion which makes it possible to use smaller forcing terms to reproduce the demonstrations. On the contrary, the exponential phase is close to zero already before Ts (Figure 3), which results in larger forcing terms.

The classical acceleration dynamics in (1) is modified as follows

$$\tau \dot{z} = \alpha_z(\beta_z(\tilde{g} - y) - z) + f(s) \tag{59}$$

Similar to target crossing, Kulvicius et al. used a moving target $\tilde{g}$ in the acceleration dynamics, but called it the *delayed goal function*. The $\tilde{g}$ term in (59) is obtained by integrating

$$\tau\dot{\tilde{g}} = \begin{cases} \frac{\delta t}{T}(g - y_0), & t \leq T \\ 0, & \text{otherwise} \end{cases} \tag{60}$$

with $\tilde{g}(0) = y_0$. The delayed goal function in Figure 9 moves linearly from y_0 to g in T seconds and then remains constant, *that is,* $\tilde{g}(t \geq T) = g$.

The nonlinear forcing term $f(s)$ is in green in (59) because it slightly differs from the classical one in (4). $f(s)$ is defined as follows

$$\begin{aligned} f(s) &= \frac{\sum_{i=1}^{N} w_i \Psi_i(t)}{\sum_{i=1}^{N} \Psi_i(t)} s, \\ \Psi_i(t) &= \exp\left(-\frac{\left(\frac{t}{\tau T} - c_i\right)^2}{2\sigma_i^2}\right), \end{aligned} \tag{61}$$

where σ_i is the width and c_i is the center of the i-th basis function, and s is obtained by integrating (14). The term $t/\tau T$ is used in (61) instead of the phase variable x. Being $0 \leq t/$

$\tau T \leq 1$, the basis functions are equally spaced between 0 and 1. Finally, σ_i are the widths of each kernel. They are constant and depend on the number of kernels.

Having presented the main differences with the canonical approach, it is possible to focus on how Kulvicius et al. (2012) solved the problem of joining $L \geq 2$ DMPs. In general, each of the L DMPs has a different time duration T^l, desired target g^l, and initial position y_0^l, from which it is possible to compute the delayed goal functions by integrating

$$\tau \dot{\tilde{g}}^l = \begin{cases} \frac{\delta t}{T^l}\left(g^l - y_0^l\right), & \sum_{\kappa=1}^{l-1} T^\kappa \leq t \leq \sum_{\kappa=1}^{l} T^\kappa \\ 0, & \text{otherwise} \end{cases} \tag{62}$$

Note that, being $\tilde{g}^l(0) = y_0$, the acceleration (59) is s. For this reason, the term $(1 - x)$ used in (54) is not needed in (59).

Assuming that L DMPs have been trained and that each DMP has N kernels, we can merge them into one DMP as follows. The centers of the joined DMP are computed as follows

$$\breve{c}_i^l = \begin{cases} \frac{T^1(i-1)}{T_{join}(N-1)}, & l = 1 \\ \frac{T^l(i-1)}{T_{join}(N-1)} + \frac{1}{T_{join}} \sum_{\kappa=1}^{l-1} T^\kappa, & \text{otherwise} \end{cases} \tag{63}$$

where T^l is the duration of the l-th DMP and $T_{join} = \sum_{l=1}^{L} T^l$ (duration of the joined motion). The widths of the joined DMP are computed as follows

$$\breve{\sigma}_i^l = \frac{\sigma_i^l T^l}{T_{join}} \tag{64}$$

The centers and widths computed in (63) and (64), respectively, overlap at the transition points allowing for smooth transitions between consecutive DMPs. The weights of the joined DMP are obtained by stacking the N weights of the L DMPs. Therefore, the joined DMP has $N*L$ kernels and $N*L$ weights. The phase variable (14) is modified to run for the duration T_{join} of the joint motion.

Saveriano et al. (2019) extended the basis functions overlay approach to quaternion DMPs. Assuming that a sequence of L quaternion DMPs is given. The angular acceleration in (16) is reformulated for each DMP as follows

$$\tau \dot{\boldsymbol{\eta}}^l = \alpha_z\left(\beta_z 2\,\mathrm{Log}^q\left(\tilde{\mathbf{g}}_q^l * \overline{\mathbf{q}}^l\right) - \boldsymbol{\eta}^l\right) + \mathbf{f}_q^l(s) \tag{65}$$

where l indicates the l-th quaternion DMP and $\mathbf{f}_q^l(s)$ is defined as in (61). The term $\tilde{\mathbf{g}}_q^l$ is the *quaternion delayed goal function,* and it ranges from $\mathbf{q}^l(0)$ to $\mathbf{g}_q^l$ in T^l seconds (see Figure 9 [right]). To generate this moving target while preserving the geometry of $\mathcal{S}^3$, it is needed that $\tilde{\mathbf{g}}_q^l$ moves along the geodesic connecting $\mathbf{q}^l(0)$ to $\mathbf{g}_q^l$. Therefore, $\tilde{\mathbf{g}}_q^l$ is defined as follows

$$\tilde{\mathbf{g}}_q^l(t + \delta t) = \mathrm{Exp}^q\left(\frac{\tau \tilde{\boldsymbol{\omega}}^l(t)}{2}\right) * \tilde{\mathbf{g}}_q^l(t) \tag{66}$$

where

$$\tilde{\omega}^l(t) = \begin{cases} \frac{2}{T^l}\mathrm{Log}^q\left(\mathbf{g}_q^l * \overline{\mathbf{q}}^l(0)\right) & \sum_{\kappa=1}^{l-1} T^\kappa \leq t \leq \sum_{k=1}^{l} T^\kappa \\ [0,0,0]^\top, & \text{otherwise} \end{cases} \tag{67}$$

The angular velocity in (67) is computed for each l. The term $2\mathrm{Log}^q(\mathbf{g}_q^l * \overline{\mathbf{q}}^l(0))$ represents the angular velocity that rotates $\mathbf{q}^l(0)$ into $\mathbf{g}_q^l$ in a unit time. Note, that the mappings $\mathrm{Log}^q(\cdot)$ and $\mathrm{Exp}^q(\cdot)$ are defined in (18) and (20), respectively. The delayed goal $\tilde{\mathbf{g}}_q^l$ crosses all the via-goals $\mathbf{g}_q^l, l = 1, \ldots, L-1$ and then reaches the goal $\mathbf{g}_q^L$.

Results in Figure 11 are obtained when the velocity threshold is applied to merge 2 DMPs separately trained to fit the minimum jerk trajectories (black-dashed lines). Figure 11(a)–(e) show the position and Figure 11(f)–(j) the orientation (unit quaternion) parts of the motion. This approach does not require a switching rule and automatically generates a smooth trajectory—with continuous velocity as shown in Figure 11(c) and (h)—that passes close to the via-point which favors the overall reproduction accuracy (Figure 11(e) and (j)). However, the distance from the via-point depends on the weights of the joined primitives and cannot be separately decided. The trajectory generated with this approach tends to last longer than the demonstrations. This is due to the sigmoidal phase that vanishes after $T + \delta_s$s (Figure 3). Depending on the application, the time difference may cause failures and has to be taken into account.

3.3. Online adaptation

The standard periodic DMP learning approach approximates the shape $f_d(t)$ of the input trajectory y_d in (39) by changing the weights of the Gaussian kernel functions (Ijspeert et al., 2013). Updating of the weights is performed in such a way that the difference between the reference trajectory and the DMP is reduced at every control step and gradually throughout the periodic repetitions. However, the DMP can also be reshaped by some external feedback function to achieve different functionalities for different applications, for instance, tasks that require trial-and-error approach (Kober et al., 2008), obstacle avoidance (Ginesi et al., 2021a; Hoffmann et al., 2009; Park et al., 2008; Tan et al., 2011), coaching (Gams et al., 2016; Petrič et al., 2014b) for robots, and adaptation of assistive exoskeleton behavior (Peternel et al., 2016). Alternatively, the frequency of the existing periodic DMPs can be modulated online (Gams et al., 2009; Petrič et al., 2011).

3.3.1. Robot obstacle avoidance and coaching. In Hoffmann et al. (2009), Park et al. (2008), and Tan et al. (2011), the detected obstacle was fitted with a potential field function to change the shape of the DMP to avoid it. More in detail, Tan et al. (2011) used the potential field to compute a time-varying goal and modified the resulting DMP trajectory, while Hoffmann et al. (2009), Park et al. (2008), and Zhai et al. (2022) added an extra forcing term to the DMP. Similarly in Gams et al. (2016), the human arm was fitted with a potential field function, which was used to reshape the DMP to perform coaching. The potential field was coupled to the position of the human hand to make pointing gestures and indicate the direction in which the robot arm position trajectory should change:

$$\dot{z} = \Omega(\alpha(\beta(-y) - z) + C_{\mathcal{O}} + f) \tag{68}$$

The added coupling term $C_{\mathcal{O}}$ is the obstacle avoidance term that contains the potential field and is given in a simplified form for the sake of explanation as follows:

$$C_{\mathcal{O}} = d_s(\|\mathcal{O} - y\|)\exp(-\zeta(\mathcal{O} - y)) \tag{69}$$

where $\mathcal{O}$ is the obstacle (or human pointing gesture) and y is the robot position. Exponential and ζ functions determine the potential field, while function d_s controls the distance at which the perturbation field should start affecting the DMP. For the full formulation of $C_{\mathcal{O}}$ and its parameters, see Gams et al. (2016). In Rai et al. (2017), the method was extended to include generalization of the obstacle avoidance formulation in (68). In a recent study, Ginesi et al. (2021a) proposed a novel approach for implementing volumetric obstacle avoidance by utilizing the theory of super quadratic potential functions.

Alternatively, the faulty segment of collision DMP trajectory can also be directly adjusted online by the human demonstrator (Karlsson et al., 2017b). On the other hand, the method in Kim et al. (2015) considers obstacle avoidance as a constraint of an optimization problem, which modifies the DMP trajectory to prevent collisions.

3.3.2. Robot adaptation based on force feedback. Similarly, as for obstacle avoidance, task dynamics can also be incorporated into DMP as coupling terms. In Gams et al. (2014), task dynamics were coupled on the acceleration and velocity level of the DMP. The presented method was utilized for interaction tasks, where the human changed the behavior of the robot based on the exerted dynamics on the manipulator

$$\tau \dot{z} = \alpha_z(\beta_z(g - y) - z) + \dot{C}_f + f(x), \tag{70}$$

$$\tau \dot{y} = z + C_f \tag{71}$$

whereas the force coupling term $C_f = \varsigma F$ is defined as a virtual or measured force F and ς is a scaling factor, which essentially changes the dynamic behavior of the DMP, enabling the motion primitive to instantly react to the coupled force. Later, Zhou et al. (2016b) introduced a PD controller-based coupling term formulation $C_{PD} = \varsigma(K^{\mathcal{P}}(F_d - F^e) - D^{\mathcal{V}}\dot{F}^e)$ coupled to the velocity part of the DMP (71). In the formulation F_d represents the desired force, F^e is the measured force, ς is a scaling factor, and $K^{\mathcal{P}}$ and $D^{\mathcal{V}}$ are the proportional and derivative gains of the Proportional Derivative (PD) controller, respectively. The coupling term formulation allows for a controlled adaptation of robot motion to changes in the environment.

In Kramberger et al. (2018) this approach was extended, with a force feedback loop coupled to the velocity (2) and the goal g of the DMP. The outcome of this approach is a similar behavior as an admittance controller (Villani and De Schutter, 2008), with a difference that the execution is directly on the trajectory generation level.

$$\tau \dot{z} = \alpha_z(\beta_z((g + C_a) - y) - z) + f(x), \tag{72}$$

$$\tau \dot{y} = z + \dot{C}_a \tag{73}$$

Here $\dot{C}_a = \varsigma(F_d - F^e)$ is the first time-derivative of the admittance coupling term, which changes the velocity and consequently the integrated coupling term, the position output of the DMP. The described approach can be used for Cartesian space motion, where the forces have to be substituted for desired and measured torques. This approach can be implemented in robot tasks involving contact with the environment as well as contact with humans. The constrained DMP formulation in Lu et al. (2021) unifies the formulations described in Sections 3.3.1 and 3.3.2 while ensuring stability of the generated motion.

3.3.3. Exoskeleton joint torque adaptation. In Peternel et al. (2016), human effort was used to provide information about the direction in which the assistive exoskeleton joint torque DMP should change in order to minimize it. The human was included into the robot control loop by replacing the error calculation in (41) with the human effort feedback term $U(E)$:

$$w_i(t_{j+1}) = w_i(t_j) + \Psi_i P_i(t_{j+1}) U(E) \tag{74}$$

where $E(t)$ is the current effort measured by human muscle activity through Electromyography (EMG) signals.[3] Equations (34)–(38) and (42) are used in the original form. Equations (39)–(41) are not used, since (74) is used to modulate the weights in (37) instead.

The effort feedback term $U(E)$ closes the loop and acts as feedback for adapting the weights of Gaussian kernels that define the shape of the trajectory. A positive $U(E)$ increases, while a negative $U(E)$ decreases the values of weights at a given section of the periodic DMP that encodes joint torque. If the shape of the DMP does not provide enough assistive power, the human has to exert effort (i.e., muscle activity) to produce the rest of the power required to achieve the desired task under given dynamics. In turn, muscle activity feedback then increases the magnitude of the DMP until the human effort term $U(E)$ is minimized. Note that each joint has its own torque DMP and $U(E)$ term (Peternel et al., 2016). After that point, the DMPs do not change unless the task, dynamics, or conditions change. If they change, the

human has to compensate for the change by an additional muscle activity, which in turn adapts the DMPs to the new required joint torques.

3.3.4. Trajectory adaptation based on reference velocity. In many LfD scenarios, it is desired to modify both the spatial motion and the speed of the learned motion at any stage of the execution. Speed-scaled dynamic motion primitives first presented in Nemec et al. (2013a) are applied for the underlying task representation. The original DMP formulations from (1) and (2) were extended by adding a temporal scaling factor v on the velocity level of the DMP

$$v(x)\tau \dot{z} = \alpha_z(\beta_z(g - y) - z) + f(x), \tag{75}$$

$$v(x)\tau \dot{y} = z \tag{76}$$

From (75) and (76), it is evident that the velocity term is a function of phase and therefore encoded with a set of RBFs similarly as in (4). This method allows for modification of the spatial motion as well as the speed of the execution at any stage of the trajectory execution. The authors demonstrated the proposed method in a learning scenario, where after every learning cycle (using Iterative Learning Control [ILC]) a new velocity profile was encoded based on the wrench feedback and thus converged to an optimal velocity for the specific task. Vuga et al. (2016) extended the approach by incorporating a compact representation for non-uniformly accelerated motion as well as simple modulation of the movement parameters.

Later on, in Nemec et al. (2018) the authors extended the previous approach to also incorporate velocity scaling of the encoded orientation trajectories represented with unit quaternions. The outcome of the presented work is a unified approach to velocity scaling for tasks executed in Cartesian space. Furthermore, a reformulation of the velocity approach called Arc-Length-DMPs (AL-DMPs) was presented by Gašpar et al. (2018). In this work, they present a method, where the spatial and temporal components of the motion are separated, by means of the arc-length based on the time-parameterized trajectory. Arc-length, based on the differential geometry of curves, is related to the speed of the movement, given as the time derivative of the demonstrated trajectory. The approach is well suited when multiple demonstrations are compared for the extraction of relevant information for learning. Following the AL-DMPs idea, Simonič et al. (2021) introduced a constant speed DMPs to fully decouple the spatial and temporal part of the task. Pahic et al. (2021) used deep NN to map images into spatial paths represented by AL-DMPs. Weitschat and Aschemann (2018) added an extra forcing term to keep the velocity within a certain predefined limit. The aim of this work is to guarantee a safe execution of the robot task when interacting with humans, as well as provide a framework for safe interaction in a changing environment where the robot position and velocity have to change over time. For a full formulation of the coupling term, see Weitschat and Aschemann (2018).

To maintain consistency along the trajectory while ensuring a bounded velocity for each DoF, it is essential to decrease the velocities uniformly across all dimensions. This rationale prompts the consideration of temporal coupling, which involves modifying the DMP time constant, equivalent to the duration of path traversal, rather than spatial coupling. By increasing the time constant, the temporal evolution of all DoFs can be simultaneously reduced by the same factor. A temporal coupling approach based on tracking error was introduced in Ijspeert et al. (2013) and an enhanced version was proposed in Karlsson et al. (2017a). However, this method requires distorting the path in order to slow down the trajectory, and therefore, it cannot be directly utilized to limit the velocity or acceleration. One year later, the authors provided stability analysis of temporally coupled DMPs in Karlsson et al. (2018). Dahlin and Karayiannidis (2020) in their work proposed a temporal coupling based on a repulsive potential, keeping the DMP velocity within the predefined velocity limits while ensuring the path shape invariance. Subsequently, Dahlin and Karayiannidis (2021) introduced a temporal coupling method for DMPs that enables control over the velocity and acceleration constraints of the generated trajectory. This approach incorporates a filtering mechanism that proactively decelerates the trajectory as the acceleration constraints are approached, thereby mitigating the potential risk of infeasibility.

3.4. Robots with flexible joints

DMPs are commonly employed for generating trajectories in manipulators with noticeable elasticity. This choice is often justified by the relatively low execution speeds, resulting in lower acceleration and jerk levels. However, when rapid motions with high acceleration and jerk are required, combining DMPs for trajectory generation with inverse dynamics for feedforward controls can lead to oscillations in robots with flexible modes. In industrial settings, where execution speed is typically faster than the speed during demonstration due to cycle time requirements, this becomes a significant challenge. To overcome this issue, Wahrburg et al. (2021) proposed an extended framework for DMPs toward flexible joint robots, dominated as FlexDMP, by introducing a fourth-order system for generating trajectories as follows

$$\tau \dddot{z} = \alpha_1(\alpha_2(\alpha_3(\alpha_4(g - y) - z) - \dot{z}) - \ddot{z}) + f(x), \tag{77}$$

$$\tau \dddot{y} = \ddot{z}, \tag{78}$$

$$\tau \ddot{y} = \dot{z}, \tag{79}$$

$$\tau\ \dot{y} = z \tag{80}$$

where $y, \dot{y}, \ddot{y}, \dddot{y}$ are position, velocity, acceleration, and jerk, respectively. The canonical system and forcing terms are similar to the ones in (3) and (4). For critically damped system $\alpha_1 = 4\beta$, $\alpha_2 = 3/2\beta$, $\alpha_3 = 2/3\beta$, and $\alpha_4 = 1/4\beta$, where $\beta > 0$.

3.5. Alternative formulations

LfD is a wide research area, and many different approaches have been developed to reproduce human demonstrations (Billard et al., 2016). As already mentioned, the aim of this tutorial survey is to provide a comprehensive overview of DMP research, and we intentionally skip the rich literature in the field of LfD. However, we found some representations that are closely related to the DMP formulation. This section briefly reviews them.

Calinon et al. (2009) computed an acceleration command for the robot in a PD-like form

$$\ddot{\mathbf{y}} = \mathbf{K}^{\mathcal{P}}(\mathbf{y}_d - \mathbf{y}) + \mathbf{D}^{\mathcal{V}}(\dot{\mathbf{y}}_d - \dot{\mathbf{y}})$$

where $\mathbf{K}^{\mathcal{P}}$ is a stiffness and $\mathbf{D}^{\mathcal{V}}$ a damping gain, $\mathbf{y}$ is the measured state of the robot and $\dot{\mathbf{y}}$ its time derivative (velocity), and $\mathbf{y}_d$ and $\dot{\mathbf{y}}_d$ are desired position and velocity retrieved with GMR, respectively. Authors then showed that the acceleration command $\ddot{\mathbf{y}}$ can be seen as a mixture of linear dynamics, each converging to a certain attractor. Despite later work like Kormushev et al. (2010) referred to this representation as "a modified version" of DMPs, there are significant differences with the DMP formulation properly highlighted by Calinon et al. (2012).

Herzog et al. (2016) computed an acceleration command for the robot from the linear system

$$\ddot{\mathbf{y}} = \mathbf{u} = \mathbf{K}^{\mathcal{P}}(\mathbf{y}_d - \mathbf{y})$$

where $\mathbf{y}$ is the measured state of the robot, $\mathbf{y}_d$ is a human demonstration, and $\mathbf{K}^{\mathcal{P}}$ is a control gain computed using the linear-quadratic regulator method. Then, a compact representation of the control input trajectory $\mathbf{u}$ is computed by means of Chebyshev polynomials. This representation does not require a vanishing phase variable to ensure convergence, but the generalization to different start/goal positions requires the application of the linear-quadratic regulator method to find a new sequence of control inputs.

Regarding periodic motions, Ajallooeian et al. (2013) proposed a dynamical system-based framework to learn rhythmic movements with an arbitrary shape and basin of attraction. They exploit phase-based scaling functions to represent the mapping between a known, base limit cycle and a desired periodic orbit. The basic limit cycle can be, for example, the one generated by periodic DMPs, which makes the approach of Ajallooeian et al. (2013) a more general formulation of periodic primitives.

4. DMPs integration in complex frameworks

This section reviews approaches where DMPs have been integrated into bigger executive frameworks. We categorize these approaches into five main research areas, namely, *grasping and manipulation*, *impedance learning*, *reinforcement learning*, *deep learning*, and *incremental and life-long learning*.

4.1. Manipulation tasks

4.1.1. Grasping and tool usage. Successfully grasping an object is the first step toward robotic manipulation. Grasping necessitates the perception of the environment, particularly visual perception, to locate the object and determine suitable grasping points based on its geometric characteristics. In this context, even slight uncertainties can lead to object slippage and failed grasps. To improve the robustness of vision-driven grasping, Krömer et al. (2010a) augmented DMPs with a potential field based on visual descriptors to adapt hand and finger trajectories according to the local geometry of the object. This grasping strategy was integrated within a hierarchical control architecture where the upper level determines the object's grasp location and the lower level locally adjusts the motion to achieve a robust grasp of the object (Krömer et al., 2010b). Stein et al. (2014) proposed a point cloud segmentation approach based on the convexity and concavity of surfaces. This approach is particularly well-suited for recognizing object handles, and DMPs are employed to execute grasping with a real robot.

The ability to grasp and use tools is also desirable to perform daily-life manipulation. In this respect, Guerin et al. (2014) proposed the so-called *tool movement primitives* that transform the demonstrations in a tool affordance frame. The result is a motion that generalizes to different tool poses and to tools that share the same affordance(s). Li and Fritz (2015) considered tool usage with low-cost, non-dexterous grippers and propose a framework to learn bi-manual strategies for tool usage and compensate for the lack of dexterity. Bi-manual robotic manipulation is a challenging task that requires precise coordination between hand movements and adherence to spatial constraints. Thota et al. (2016) developed a DMP-based control framework for bi-manual manipulation that ensures time synchronization of the two hands while being robust to spatial perturbations and goal changes.

4.1.2. Motion primitives sequencing. Beyond object grasping, everyday manipulation requires a precise execution of complex movements. Often such complex movements are hard to encode into a single motion primitive, but they can be conveniently split into simpler motions (e.g., reach and grasp) that can be properly sequenced and executed (Figure 12).

The possibility of exploiting DMPs as the building blocks of complex tasks was investigated in Caccavale et al. (2018, 2019) and Ramirez-Amaro et al. (2015). In these works, a human teacher demonstrated a relatively complex task consisting of several actions performed on different objects. The demonstration was then automatically segmented into M basic motions used to fit M DMPs. While Ramirez-Amaro et al. (2015) exploit semantic rules (e.g., *reach an object with a knife means cut*) to infer high-level human activities, Caccavale et al. built a hierarchical structure to schedule the execution of the complex task by selecting the proper DMP for the current executive context. They used kinesthetic teaching and verbal cues (open/close gripper commands) to provide task demonstrations. Schwaner et al. (2021) used DMPs to build a

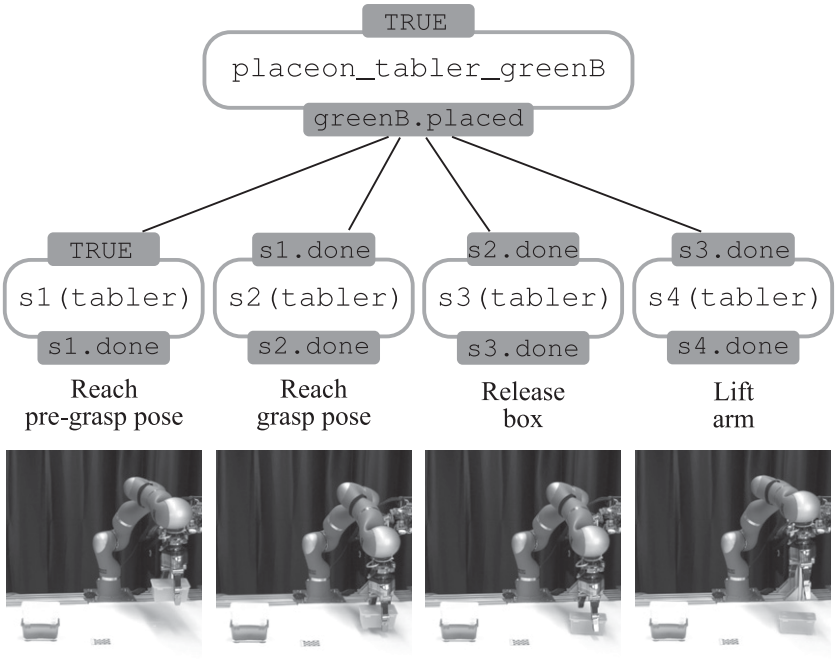

Figure 12. An example of hierarchical task decomposition and motion primitives sequencing from Agostini et al. (2020).

library of surgical skills which used to autonomously execute part of a suturing task. Lemme et al. (2014) organize segmented task demonstrations into a motion primitives' library learned from self-generated trajectory patches. They also introduced a mechanism to remove unused skills and update the library. Kinesthetic teaching and haptic feedback were also used by Eiband et al. (2019) to segment and recognize basic motions or *skills* and to build a tree describing geometric relationships—like reference frames and goal poses—between consecutive skills. At run time, the robot performed haptic exploration to locate objects in the scene and update the skill tree. The transformations in the skill tree were then used to define the initial and goal pose of the DMPs and execute the task. Finally, Wu et al. (2018) integrated DMPs into a dialogue system with speech and ontology to learn or re-learn a task using natural interaction modalities.

4.1.3. Data collection. Collecting demonstrations becomes an issue of kinesthetic teaching or marker-based motion trackers cannot be used. The latter requires an expensive sensor infrastructure that is hard to build in real-world scenarios like factory floors. Kinesthetic teaching needs torque-controlled/collaborative robots that are still uncommon in industrial scenarios. To remedy this issue, Mao et al. (2015) exploited a low-cost RGB-D camera and track the human hand using the markerless approach proposed by Oikonomidis et al. (2011). Collected data were then segmented into basic motions and used to fit DMPs. Also, the approach in Yang et al. (2022) does not require tracking markers or manual annotations. Instead, authors exploit videos of random unpaired interactions with objects by the robot and human for unsupervised learning of a keypoint model of visual correspondences. Bayesian optimization is then used to find the parameters of rhythmic DMPs from a single human video demonstration within a few robot trials.

The described approaches assume that human teachers always provide consistent and noiseless task demonstrations. Ghalamzan E. et al. (2015) encoded noisy demonstrations into a GMM and computed a noise-free trajectory using GMR. The noise-free trajectory was then used to fit a DMP that generalized to different start, goal, and obstacle configurations. Dong et al. (2023) propose to fit a DMP from correct (positive) and incorrect (negative) demonstrations to increase the representation and generalization capabilities of the model. Niekum et al. (2012, 2015) designed a framework that learns from unstructured demonstrations by segmenting the task demonstrations, recognizing similar skills, and generalizing the task execution. Interestingly, a user study on 10 volunteers conducted by Gutzeit et al. (2018) showed that existing strategies for segmentation and learning are sufficiently robust to enable automatic transfer of manipulation skills from humans to robots in a reasonable time.

4.1.4. Task learning and execution. Some work (Deniša and Ude, 2013a, 2013b, 2015; Denisa et al., 2021) exploited transition graphs and trees to embed parts of a trajectory and search algorithms to discover a sequence of partial parts and generate motions that have not been demonstrated. Approaches that rely on a hierarchical, tree-like structure to represent the task have limited task generalization capabilities. Lee and Suh (2013) used probabilistic inference and object affordances to infer the adequate skill that can handle uncertainties in the executive context. Beetz et al. (2010) learned stereotypical task solutions from observation

and used task planning and symbolic reasoning to execute novel mobile manipulation tasks. A generative learning framework was proposed by Wörgötter et al. (2015) to augment the robot's knowledge base with missing information at different levels of the cognitive architecture, including symbolic planning as well as object and action properties. Paxton et al. (2016) used task and motion planning to generalize the execution of complex assembly tasks and proposed a learning by demonstration approach to ground symbolic actions. Agostini et al. (2020) performed task and motion planning by combining an object-centric description of geometric relations between objects in the scene, a symbol to motion hierarchical decomposition depending on three consecutive actions in the plan, and the LfD approach developed in Caccavale et al. (2019) (Figure 12). A manipulation task was described at three different levels by Aein et al. (2013). The top level provides symbolic descriptions of actions, objects, and their relationships. The mid-level uses a finite state machine to generate a sequence of action primitives grounded by the lower level. A common point among these approaches is that they use DMP to execute the task on real robots.

4.2. Variable impedance learning control

Impedance control can be used to achieve compliant motions, in which the controller resembles a virtual spring–damper system between the environment and robot end-effector (Hogan, 1985). Such an approach permits smooth, safe, and energy-efficient interaction between robots and environments (possibly humans). A standard model for such interaction is defined as follows

$$\mathfrak{M}\ddot{\mathbf{y}}_t = \mathbf{K}_t^{\mathcal{P}}\left(\mathbf{y}_g - \mathbf{y}_t\right) - \mathbf{D}_t^{\mathcal{V}}\dot{\mathbf{y}}_t + \mathbf{f}_t^e, \tag{81}$$

$$\mathcal{I}\dot{\boldsymbol{\omega}}_t = \mathbf{K}_t^{\mathcal{O}}\left(\mathrm{Log}^R\left(\mathbf{R}_g\mathbf{R}_t{}^\top\right)\right) - \mathbf{D}_t^{\mathcal{W}}\boldsymbol{\omega}_t + \boldsymbol{\tau}_t^e \tag{82}$$

where (81) and (82) correspond to translational and rotational cases, respectively, $\mathfrak{M}, \mathbf{K}_t^{\mathcal{P}}$, and $\mathbf{D}_t^{\mathcal{V}}$ are the mass, stiffness, and damping matrices, respectively, for translational motion, while $\mathcal{I}, \mathbf{K}_t^{\mathcal{O}}$, and $\mathbf{D}_t^{\mathcal{W}}$ are the moment of inertia, stiffness, and damping matrices, respectively, for rotational motion. $\widehat{\mathbf{R}}, \mathbf{R}_t \in \mathcal{SO}(3)$ are rotation matrices and correspond to desired rotation goal and actual orientation profile of the end-effector, respectively. $\mathbf{f}_t^e$ and $\boldsymbol{\tau}_t^e$ represent the external force and torque applied to the robot end-effector, respectively.

In fact, Variable Impedance Control (VIC) plays an important role when a robot needs to interact with any environment in order to avoid high impact forces and damage for the environment or the robot (i.e., change to low stiffness) (Ajoudani et al., 2012; Abu-Dakka et al., 2018; Peternel et al., 2018a). On the other hand, it is important in rejecting unexpected and unpredictable perturbations from the environment to achieve the desired position tracking precision (i.e., change to high stiffness) (Yang et al., 2011). In addition, it is also important in the coordination of human–robot collaborative movements (Peternel et al., 2017b). However, a robotic system still needs to learn how to adapt such VIC to unseen situations while avoiding hard-coding. Such a paradigm of learning is called Variable Impedance Learning Control (VILC). Interested readers can refer to our recent survey on VILC (Abu-Dakka and Saveriano, 2020) or teleimpedance (Peternel and Ajoudani, 2022).

In this review, we will mention some of the works that integrate DMP with VIC in a VILC framework. Figure 13 shows a simple generic example where DMP is integrated in a VIC control scheme.

Buchli et al. (2011a) proposed one of the earliest approaches that integrates DMP with Policy Improvement with Path Integrals (PI^2) algorithm (Theodorou et al., 2010) to learn movements (position and velocity presented by DMP) while optimizing impedance parameters. Later, the authors exploited a diagonal stiffness matrix and expressed the variation (time derivative) of each diagonal entry as follows

$$\dot{k}_{\vartheta_j,t} = \alpha_j\left(\curlywedge_j{}^\top\left(\boldsymbol{\vartheta}_j + \boldsymbol{\epsilon}_{j,t}\right) - k_{\vartheta_j,t}\right), j = 1, \ldots, J \tag{83}$$

where j indicates the j-th joint, $k_{\vartheta_j,t}$ is the stiffness of joint j, $\boldsymbol{\epsilon}_{j,t}$ is a time-dependent exploration noise, each $\curlywedge_j$ is a vector of N Gaussian basis functions, and $\boldsymbol{\vartheta}_j$ are the learnable parameters for joint j. The stiffness parameterization in (83) is also linear in the parameters, and PI^2 can be applied to find the optimal policy. Later, authors used PI^2 to learn VIC in deterministic and stochastic force fields (Stulp et al., 2012a). Nakanishi et al. (2011) proposed a method that optimizes a periodic motion along with a time-varying joint stiffness.

Basa and Schneider (2015) introduced an extension to DMP formulation by adding a second nonlinear function to cope with elastic robots as follows

$$\tau\dot{z} = \alpha_z(\beta_z(g - y) - z) + f(x) + f_2 \tag{84}$$

where f_2 is defined as (4) but without the phase variable x. The main purpose of f_2 is to compensate for the gravitational influence on the moved DoF at the end of the movement time and beyond. Differently, Haddadin et al. (2016) used optimal control to execute near-optimal motion of elastic robots.

Nemec et al. (2016) proposed a cooperative control scheme that enables a dual-arm robot to adapt its stiffness online along with the executed trajectory in order to provide accurate evolution. Umlauft et al. (2017) used GP along with DMPs (as proposed in Fanger et al. (2016)) to predict the trajectories. During the execution, their admittance controller adapts both stiffness and damping online. The energy-tanks passivity-based control method has been integrated with DMPs to enforce passivity in order to stably adapt to contacts in unknown environments by adapting the stiffness online (Shahriari et al., 2017; Kramberger et al., 2018; Kastritsi et al., 2018).

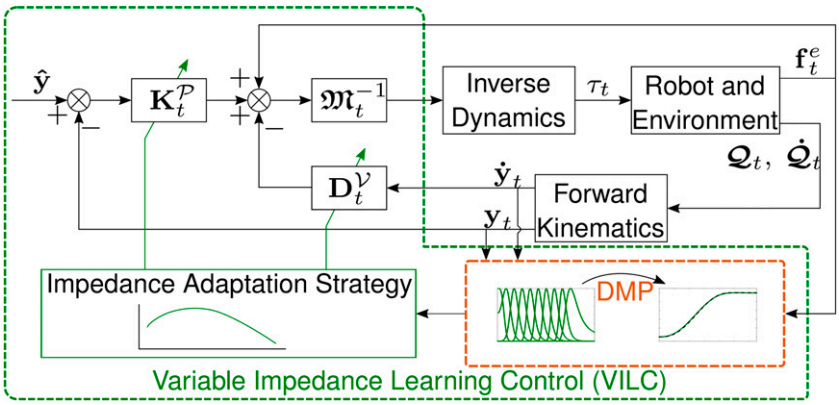

Figure 13. General control scheme of VIC and DMP.

Methods in Bian et al. (2019), Peternel et al. (2014, 2018b, 2018a), and Yang et al. (2018, 2019) designed different multi-modal interfaces to let the human to explicitly teach an impedance behavior to the robot. Most of them combined EMG-based variable impedance skill transfer with DMP-based motion sequence planning, inheriting the merits of these two aspects for robotic skill acquisition. Hu et al. (2018) used Covariance Matrix Adaptation-Evolution Strategies (CMA-ES) to update the parameters of DMPs and variable impedance controller in order to reduce the impact during the robot motion in noisy environments. Dometios et al. (2018) integrated a Coordinate Change-DMPs (CC-DMP) with a vision-based motion planning method to adapt the reference path of a robot's end-effector and allow the execution of washing actions.

Travers et al. (2016, 2018) proposed a shape-based compliance controller for the first time in locomotion, by implementing amplitude compliance on a snake robot moving in a complex environment with obstacles. Their approaches allow snake-like robots to blindly adapt to such complex unstructured terrains, thanks to their proprioceptive gait compliance techniques.

Recently, an adaptive admittance controller is proposed (Wang et al., 2020) which integrates GMR for the extraction of human motion characteristics, DMP to encode a generalizable robot motion, and an RBF-NN-based controller for trajectory tracking during the reproduction phase. In their work, Spector and Zacksenhouse (2021) introduced a residual admittance policy within the framework of DMPs. This policy explicitly learned full asymmetric stiffness matrices and aimed to correct the movements generated by a baseline policy. The effectiveness of the learned policy was demonstrated through successful peg insertion tasks involving various shapes and sizes of pegs. Additionally, the policy exhibited robustness in handling uncertainties in hole location and peg orientation and showed good generalization to new shapes. Moreover, the learned policy demonstrated successful transferability from simulations to real-world scenarios.

Novel LfD approaches explicitly take into account that training data are possibly generated by certain Riemannian manifolds with associated metrics. Abu-Dakka and Kyrki (2020) reformulated DMPs based on Riemannian metrics, such that the resulting formulation can operate with SPD data in the SPD manifold. Their formulation is capable to adapt to a new goal-SPD-point.

Recently, biomimetic controller has been integrated with DMPs (Zeng et al., 2021) in order to learn and adapt compliance skills.

4.3. Reinforcement Learning (RL)

In RL, an agent tries to improve its behavior via trial-and-error by exploring different strategies (*actions*) and receiving a feedback (*reward*) on the outcome of its actions. Actions a are drawn from a *policy* $\pi(s, a)$ that represents a mapping between *states* s and actions a. The goal of RL is to find an optimal policy $\pi^\star$ that maximizes the cumulative expected reward, *that is,* the sum of expected rewards over a possibly infinite time interval. When the agent is a robot performing tasks in the real world, the state and action spaces are inherently continuous. Moreover, the robotic agent is affected by imperfect (e.g., noisy) perception and inaccurate models (e.g., contacts). Finally, performing a large amount of interactions with the real world (*rollouts*) is expensive and possibly dangerous. As discussed by Kober et al. (2013), robotic-specific challenges require specific solutions to make the RL problem feasible.

4.3.1. DMPs as control policies. One possibility is to use parameterized policy and use RL to search for an optimal, finite set of policy parameters. In this respect, DMPs have been widely used as policy parameterization. The general idea is shown in Figure 14. More in detail, (Peters and Schaal, 2008a; Schaal, 2006) showed that various policy gradient and actor-critic RL approaches can be effectively applied to improve robotic skills parameterized as DMPs. Other research focused on developing policy search algorithms specifically for parameterized policies. Inspired by stochastic optimal control, Theodorou et al. (2010)

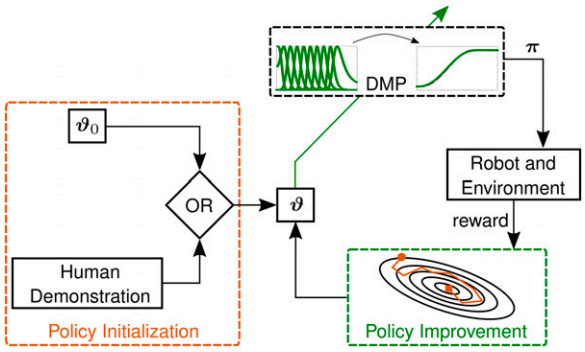

Figure 14. General block scheme of DMP-based policy improvement.

proposed PI^2 which is an application of path integral optimal control to DMPs. PI^2 and DMPs have been successfully applied in several domains including VILC (Buchli et al., 2011a, 2011b) and in-contact tasks (Hazara and Kyrki, 2016), grasping under-state estimation uncertainties (Stulp et al., 2011a), bi-manual manipulation (Zhao et al., 2020), nonprehensile manipulation (Sun et al., 2022), and robot-assisted endovascular intervention (Chi et al., 2018). Kober and Peters (2011) derived from expectation-maximization the so-called Policy Learning by Weighting Exploration with the Returns (PoWER). PoWER and DMPs have been successfully applied to perform highly dynamic tasks including ball-in-a-cup (Kober and Peters, 2011) and pancake flipping (Kormushev et al., 2010).

4.3.2. Limit the search space. Even with parameterized policies the number of rollouts needs to search for optimal policy parameters may become large, especially for robots with many DoFs. Dimensionality reduction techniques can be exploited to perform policy search in a reduced space (Colomé and Torras, 2014). The effectiveness of this approach was demonstrated in the challenging task of clothes (i.e., soft tissues) manipulation (Colomé and Torras, 2018). IL arises as an effective approach to policy initialization and to speed up policy search by reducing the number of rollouts (Kober and Peters, 2010). In this respect, Kober et al. (2008, 2010a) augmented DMPs with a perceptual coupling term and propose to initialize the DMP via human imitation and to refine the motor skill via RL. IL can be eventually combined with dimensionality reduction (Tan and Kawamura, 2011), and several rollouts can be performed firstly in simulation (Cohen and Berman, 2014) to further speed up the policy search. When multiple demonstrations are given, one can learn a mapping between policy parameters and query points (e.g., goal positions) and use the mapping to generalize to new situations (Section 3.1.3). This strategy was used by Nemec et al. (2011, 2012, 2013b) to provide a good initial policy for a new situation which is then further refined using RL. Being the mapping estimated using example query points, the search space can be effectively constrained within query points making the policy search more efficient. Vuga et al. (2015a, 2015b) combined this approach with a different DMP formulation to optimize the velocity of execution. The approach was tested on diverse tasks including pouring water into a cup, where it prevented the water to split from the cup during the motion. Schroecker et al. (2016) provided demonstrations in the form of soft via-points (Section 3.1.2) which reduce the search space to the neighborhood of the taught via-points. Multiple demonstrations were used by Reinhart and Steil (2014, 2015) to build a parameterized skill memory that connects low-dimensional skill parameterization to motion primitive parameters. This low-dimensional embedding is then leveraged for efficient policy search. Instead of learning a mapping from task to policy parameters, Queißer et al. (2016) used data from the rollouts to incrementally learn a parametric skill (bootstrapping) and used it to generate a good initial policy for a new task.

4.3.3. DMP generalization and sequencing. Instead of using generalization to provide a better initial policy, some researchers exploit RL to improve and generalize the motion primitive. André et al. (2015) adapted DMP policies to walk on sloped terrains. Mülling et al. (2010) generalized to new situations using a mixture of DMPs. In their approach, RL was used to estimate the shape parameters as well as to estimate the optimal responsibility of each DMP. Mülling et al. (2013) used episodic RL to estimate meta-parameters like the temporal and spatial interception point of the ball and the racket typical of table tennis tasks. Lundell et al. (2017) used parameterized kernel weights and RL to search for optimal parameters, while Forte et al. (2015) augmented the given demonstration using RL-based state space exploration to autonomously expand the robot's task knowledge. Metric RL was exploited by Hangl et al. (2015) to smoothly switch between learned DMP policies and execute a task in new situations.

RL can be also applied to sequence multiple motion primitives and perform more complex tasks; a successful strategy when the robot has to perform, for instance, a manipulation task (Section 4.1). To sequence multiple primitives, it is also of importance to learn the goal of each motion. Tamosiunaite et al. (2011) used continuous value function approximation to optimize the goal parameters of a DMP used to perform a pouring task. Kober et al. (2011, 2012) learned a meta-parameter function that maps the current state to a set of meta-parameters including the goal and duration of the movement. Instead of separating shape and goal learning into different processes, Stulp et al., 2011b, 2012) extended PI^2 to simultaneously learn the shape and goal of a sequence of DMPs. Wang et al. (2022) describe both complex manipulation tasks and user execution preferences as logic and temporal constraints and use RL to find a set of DMP parameters that fulfill the constraints.

4.3.4. Skills transfer. Learned skills can be potentially transferred across different tasks to speed up the learning process and increase robot autonomy. To this end, Fabisch and

Metzen (2014) considered the case where the robot can actively choose which task to learn to make the best progress in learning. The process of actively selecting the task was considered as a non-stationary bandit problem for which a suitable algorithmic solution exists while intrinsic motivation heuristics were exploited to reward the agent after the selection. Cho et al. (2019) defined the complexity of a motor skill based on the temporal and spatial entropy of multiple demonstrations and used the measured complexity to generate an order for learning and transferring motor skills. Their experimental findings provided useful guidelines for skill learning and transfer. In short, humans have to demonstrate, when possible, the most complex task and then the robot is able to transfer the motor skills. Vice versa, if demonstrations are not given, it is more effective to start learning simple skills first and then transfer the simpler skills to more complex tasks.

4.3.5. Learning hierarchical skills. RL often lacks scalability to high-dimensional continuous state and action spaces. To remedy this issue, hierarchical RL exploits a *divide et impera* approach by decomposing an RL problem into a hierarchy of sub-tasks in order to reduce the search space. Different levels in the hierarchy represent information at different times and/or spatial scales.

Stulp and Schaal (2011) proposed to represent different options as DMPs to sequence. PI^2 was extended to optimize shape and (sub-)goal of each DMP at different levels of temporal abstraction. In particular, the shape was adjusted based on the cost up to the next primitive in the sequence, while the sub-goal considers the cost of the entire sequence of two DMPs. Layered direct policy search in End et al. (2017) did not rely on a set of predefined sub-policies and/or sub-goals but instead used information theoretic principles to uncover a set of diverse sub-policies and sub-goals.

Reducing the number of rollouts required to discover optimal policies is also important in Hierarchical RL (HRL). As already mentioned, IL is a valuable option to find good initial policies. However, there are applications like manipulation with multi-fingered robotics hands for which it is hard or impossible to provide expert demonstrations. To make policy search more efficient, Ojer De Andres et al. (2018) used HRL where the upper level considers discrete action and state spaces to search for optimal finger gaiting and synchronization among the fingers. This information was passed to the lower level where rhythmic DMPs and PI^2 generated continuous commands for the fingers. Another possibility to increase data-efficiency is to use model-based approaches for RL. Colome et al. (2015) exploited a friction model to improve a DMP policy and manipulate soft tissues (a scarf). A model-based HRL approach was proposed by Kupcsik et al. (2017) for data-efficient learning of upper level policies that generalize well across different executive contexts. Li et al. (2018) proposed a hybrid hierarchical framework where the higher level computes optimal plans in Cartesian space and converts them to desired joint targets using an efficient solver. The lower level is then responsible to learn joint space trajectories under uncertainties using RL and DMPs. Recently, Davchev et al. (2022) proposed residual LfD, a framework that combines DMPs and RL to learn a residual correction policy for assembly tasks. In the paper, they show that applying residual learning directly in task space and operating on the full pose of the robot can significantly improve the overall performance of the DMPs.

4.4. Deep learning

A popular method of machine learning is NNs. Due to their non-parametric nature, they can effectively represent non-linear mappings. A major drawback of NNs in the past was their computational complexity of learning. In recent years, there is a renewed interest in NNs. New deep learning approaches were successfully (LeCun et al., 2015) applied in machine vision and language processing.

In recent years, deep learning has been applied also in robotics to learn task dynamics (Yang et al., 2016) and movement dimensionality reduction (Chen et al., 2015). The authors (Chen et al., 2015, 2016) introduced a framework called AutoEncoded DMP (AEDMP) which uses deep auto-encoders to find movements represented in latent feature space. In this space DMPs can optimally be generalized to new tasks, as well as the architecture enables the DMPs to be trained as a unit. Pervez et al. (2017b) in their work coupled the vision perception data for object calcification with task-specific movement definitions represented with DMPs. The data was modeled with Convolutional Neural Networks (CNNs), where the images and the associated movements were directly processed by the deep NN, thus preserving the associated DMP properties and eliminating the need for extracting the task parameters during motion reproduction. Later on, Kim et al. (2018b) combined deep RL with DMPs to learn and generalize robotic skills from demonstration. The framework builds on an RL approach to learn and optimize a new DMP skill based on a demonstration. The RL approach is backed up with a hierarchical search strategy, reducing the search space for the robot, which allows for more efficient learning of complex tasks. Furthermore, Pan and Manocha (2018) presented a deep learning approach for motion planning of high-dimensional deformable robots in complex environments. The locomotion skills are encoded with DMPs, and an NN is trained for obstacle avoidance and navigation. The data is further optimized with deep Q-Learning showing that the learned planner can efficiently plan and navigate tasks for high-dimensional robots in real-time.

Pahic et al. (2018) proposed a deep learning approach for perception-action couplings, demonstrating the coupling between the vision-based images and associated movement trajectories. Later on, they extended the approach to incorporate CNNs and give a distinguishing property formulation for the approach (Pahič et al., 2020), which utilizes a loss function to measure the physical distance between the

movement trajectories as opposed to measuring the distance between the DMPs parameters which have no physical meaning, leading to better performance of the algorithm. Recently, they extended the usage of GPR to create a database needed to train autoencoder NNs for dimensionality reduction (Lončarević et al., 2021). Mavsar et al. (2022) in their work presented a recurrent neural architecture, capable of transforming variable-length input motion videos into a set of parameters describing a robot trajectory which is later encoded with DMP, where predictions can be made after receiving only a few frames, in addition, a simulation environment is utilized to expand the training database and to improve the generalization capability of the network, which is used for handover robotic tasks. Furthermore, Jaques et al. (2021) in their study introduced the Newtonian Variational Autoencoder (Newtonian VAE), a framework for learning latent dynamics. Drawing inspiration from Newton's second law, they define a linear dynamic system in a hidden space. This system is based on a rigid-body model with J DoFs. The Newtonian VAE framework enables the formation of a PD-controllable state space, which facilitates robust path following based on visual demonstrations using DMPs within the learned latent space.

4.5. Lifelong/Incremental learning

Lifelong (incremental) learning is a framework which provides continuous learning of tasks arriving sequentially (Chen and Liu, 2018; Fei et al., 2016; Thrun, 1996). The essential component of this framework is a database that maintains the knowledge acquired from previously learned tasks TSK_1, TSK_2, …, TSK_{N-1}. Incremental learning starts from the task manager assigning a new task TSK_N to a learning agent. In this case, the agent exploits the knowledge in the database as prior data for enhancing the generalization performance of its model on the new task. After the new task TSK_N is learned, the database is updated with the knowledge obtained from learning TSK_N. In fact, the incremental learning framework provides an agent with three capabilities: (*i*) continuous learning, (*ii*) knowledge accumulation, and (*iii*) re-using previous knowledge for future learning enhancements. Figure 15 shows a general structure of DMP integrated with a lifelong framework.

Churchill and Fernando (2014) proposed a cognitive architecture capable of accumulating adaptations and skills over multiple tasks in a manner which allows recombination and reuse of task-specific competences. Lemme et al. (2014) segmented demonstrations based on geometric similarities and subsequently created a motion primitives library. The library is updated by removing unused skills and including new ones. Multiple demonstrations are used by Reinhart and Steil (2014, 2015) to build a parameterized skill memory that connects low-dimensional skill parameterization to motion primitive parameters. This low-dimensional embedding is then leveraged for efficient policy search. Piece-wise linear phase is used to improve incremental learning performance (Samant et al. 2016).

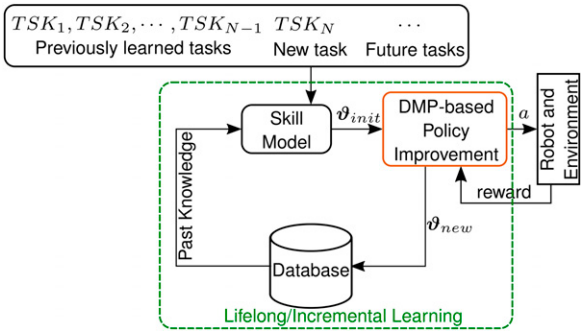

Figure 15. General framework of lifelong/incremental learning approach.

Duminy et al. (2017) designed a framework for learning which data collection strategy is most efficient for acquiring motor skills to achieve multiple outcomes and generalize over its experience to achieve new outcomes for cumulative learning.

A generative learning framework is proposed to augment the robot's knowledge-base with missing information at different levels of the cognitive architecture including symbolic planning as well as object and action properties (Wörgötter et al., 2015). Aforementioned approaches use DMPs to represent the learned skills and execute them on real robots.

Wang et al. (2016) proposed a modified formulation of DMPs called DMP+ which is capable of efficiently modifying learned trajectories by improving the usability of existing primitives and reducing user fatigue during IL. Later, DMP+ had been integrated into a dialogue system with speech and ontology to learn or re-learn a task using natural interaction modalities (Wu et al., 2018).

In the literature, it has been shown that incremental learning provides better generalization than the isolated learning approaches in terms of interpolation, extrapolation, and the speed of learning (Hazara and Kyrki, 2017). Hazara and Kyrki (2018) improved their Global Parametric Dynamic Movement Primitive (GPDMP) (Lundell et al., 2017) in order to construct, incrementally, a database of motion primitives, which aims to improve the generalization to new tasks. Furthermore, it has been transferred incrementally from simulation to the real world (Hazara and Kyrki, 2019). Moreover, authors endow incremental learning with a task manager, which is capable of selecting a new task by maximizing future learning while considering the current task performance (Hazara et al., 2019).

5. DMPs in application scenarios

We categorize the applications into several subsections based on different topics. We first separate the use of DMPs for robot interaction with the *passive environment* (e.g., tools, objects, and surfaces) and for interaction with an agent that involves *co-manipulation* (e.g., human and another robot). Additionally, we examine several other major application areas, such as *human body augmentation/rehabilitation* with exoskeletons, *teleoperation*, *motion*

Figure 16. Human operators teach the robot how to perform different tasks. *Left* scenarios use robots' gravity compensation mode to enable kinesthetic guiding, where a human operator guides the robot's tool center point along the desired trajectory in such a way that the desired task is successfully executed (Abu-Dakka et al., 2015a, 2018; Caccavale et al., 2019; Sloth et al., 2020). *Right* scenarios use a teleoperation system to demonstrate appropriate robot movements either through a haptic interface (Peternel et al., 2018a) or magnetic trackers (Abu-Dakka et al., 2015a).

analysis/recognition, *high DoF robots*, and *autonomous driving and field robotics*.

5.1. Robots in contact with passive environment

Most of the daily tasks that the robots perform involve some kind of physical interaction with the environment that requires control of forces or positions. Nevertheless, simultaneous control of force and position in the same axis is not possible (Stramigioli, 2001)[4], and thus the control approaches have to make a compromise between prioritizing position control or force control (Schindlbeck and Haddadin, 2015). The key to such control is for the robot to learn appropriate force or position reference trajectories that can lead to the desired task performance in interaction with the environment. Factors such as positional information, muscle stiffness of the human arm, and contact force with the environment are crucial in comprehending and generating robot manipulation behaviors that resemble those of humans. In Wang et al. (2021), both positional and contact force profiles are represented using DMPs to facilitate the transfer of human–robot skills.

5.1.1. Demonstration of interaction tasks. A common approach to teaching robot motion trajectories is kinesthetic guidance (Figure 16-*Left*), where the human operator holds the robot arm and shows the appropriate movements to be encoded by DMPs (Abu-Dakka et al., 2015a; Joshi et al., 2017; Kormushev et al., 2011; Papageorgiou et al., 2020; Schaal, 2006). Recently, the technology is protruding into high-risk fields such as invasive surgery, where high-dimensional fine human-like manipulation skills are being demonstrated (Su et al., 2021) and executed with robots (Ginesi et al., 2019; Su et al., 2020). In Kormushev et al. (2011), the human held the robot arm and used kinesthetic guidance to teach the position and orientation trajectories necessary to perform ironing and door-opening tasks. In the second stage, the corresponding forces and torques were recorded with a haptic device in a teleoperation setup. For setups where the robot arm is equipped with multiple force/torque sensors, the two demonstration steps with additional control policies can be combined into one (Montebelli et al., 2015; Steinmetz et al., 2015).

An alternative to learning force trajectories is to learn the impedance of the robot by learning the desired stiffness trajectories. The ability to change the impedance of the arm is crucial to simplify the physical interaction in unpredictable and unstructured environments (Burdet et al., 2001; Hogan, 1984). In Peternel et al. (2018a), teleoperation was used with a push-button interface to command the robot impedance, which was learned by DMPs that enabled the robot to perform various collaborative assembly tasks. For example, the learned position and stiffness DMPs were used to insert a peg in a groove to bind the two parts or to screw a bolt (Peternel et al., 2018a). A similar approach was used in Yang et al. (2018) to learn DMPs used for a vegetable cutting task.

While teleoperation-based methods are very effective to teach the robot DMPs for interaction tasks, it usually involves a complex and expensive system. The method in Abu-Dakka et al. (2018) enabled the robot to learn stiffness profiles through measurement of interaction force with the environment to perform a valve turning task. The method in Peternel

et al. (2017a) used human demonstration and EMG to learn stiffness DMPs from human muscle activity measurements in order to perform sawing and wiping (Figure 17) tasks.

Nevertheless, adaptation of a single trajectory is unlikely to generate an appropriate solution for more general cases, where the task execution needs to change significantly. After learning the initial DMP motion trajectories through kinesthetic guidance, the robot can then adapt them based on the measured force of interaction while performing the task. Pastor et al. (2011) introduced a method for real-time adaptation of demonstrated DMP trajectories depending on the measured sensory data. They developed an adaptive regulator for trajectory adaptation based on estimated and actual force data. Prakash et al. (2020) extended the real-time adaptation approach incorporating a fuzzy fractional-order sliding mode controller in order to efficiently and stably adapt the demonstrated DMP trajectory to fast movements, such as a ping pong swing. Recently, Cui et al. (2022) presented a method for coupling multiple DMPs for modeling robot tasks for transportation tasks of deformable objects.

Sutanto et al. (2018) presented a data-driven framework for learning a feedback model from demonstrations. They used an RBF-NN to represent the feedback model for the movement primitive. Similar to this research, Gams et al. (2010) proposed a method for adaptation of demonstrated movements depending on the desired force, with which the robot should act on the environment. Thus, they ensured the adaptation of the learned movements to different surfaces. This approach was later expanded (Pastor et al., 2011) to provide the statistically most likely force–torque profile (Pastor et al., 2012), and furthermore, force–torque data was used for training a classifier (Straizys et al., 2020) in order to modulate the demonstrated trajectory for the use with delicate tasks such as tissue or fruit cutting.

Moving onward from policy learning, Do et al. (2014) presented an adaptation framework, where not only the desired adaptation force or trajectory but also the entire skill can be learned. They demonstrated the method with a wiping task under different environmental conditions.

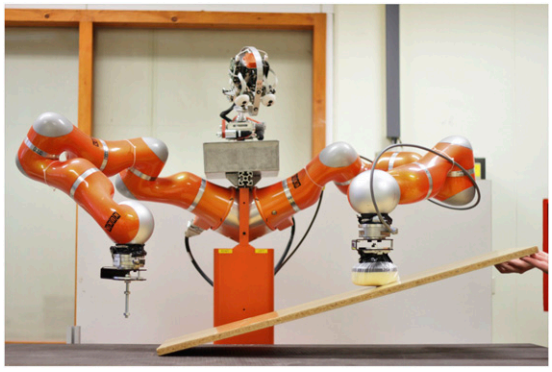

Figure 17. Using DMPs for adapting to changing surfaces (e.g., wiping task) (Kramberger et al., 2018).

5.1.2. Assembly tasks. Assembly presents one of the more challenging tasks to automate, where not only position trajectories but also task dynamics have to be taken into account. To deal with this challenge, various methods were proposed. Abu-Dakka et al. (2015a) proposed a method that can learn the orientation aspect of the complex physical interaction, like the peg-in-the-hole assembly tasks (Figure 18). The proposed method was integrated into an industrial assembly framework where the key challenge was to adapt to uncertainties presented by the assembly task (Abu-Dakka et al., 2014; Krüger et al., 2014).

Complex assembly tasks that are subject to change cannot be demonstrated and executed on the fly; therefore, adaptation methods are required for ensuring a successful execution. Nemec et al. (2020) used exception strategies, modeled as DMPs, for dealing with complex assembly cases. Sloth et al. (2020) presented an exception strategy framework, combining discrete and periodic DMP, coupled with force control to learn an assembly task under tight tolerances. Angelov et al. (2020) incorporated several different control policies by taking into account the

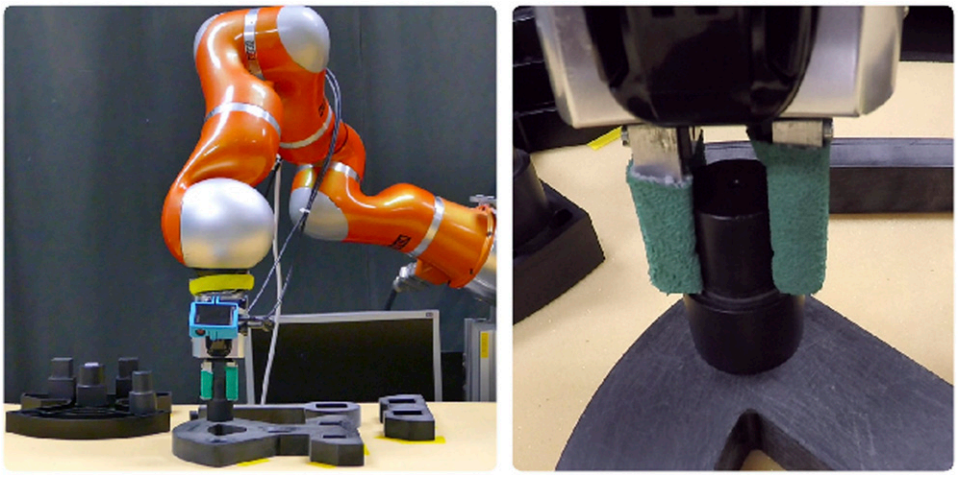

Figure 18. An example of using DMPs in assembly tasks (e.g., peg-in-the-hole) (Kramberger et al., 2016b).

dynamics and sequencing of the task. The approach uses DMPs to generate free motions and convolutional neural networks for the assembly.

In some cases, active exploration and autonomous database expansion can be used for learning assembly policies automatically. In Petric et al. (2015), the proposed algorithm can build and combine CMP motion knowledge from a database in an autonomous manner.

Complementary to assembly tasks, disassembly is also challenging by solely using the demonstrated trajectories. As described in Ijspeert et al. (2013), DMPs have a unique point attractor in the specified goal parameter of the movement, essentially repelling the idea of reversibility. Therefore, Nemec et al. (2018) proposed a framework, where the disassembly challenge was tackled by learning two separate DMPs from a single demonstrated motion: one forward and one backward. Iturrate et al. (2019) took the idea further and reformulated the DMP phase system with a logistic differential equation to obtain two stable point attractors. This Reversible Dynamic Movement Primitive (RevDMP) approach provided a reversibility formulation of the dynamical system and demonstrated the effectiveness of the algorithm on a peg-in-hole assembly task.

5.1.3. Learning methods for contact adaptation. Desired force–torque profiles can be tracked using ILC (Gams et al., 2014, 2015b). In repetitive robotic tasks, iterative learning has been gaining increased popularity (Bristow et al., 2006) due to its effectiveness and robustness. However, in order to achieve effective results, a careful tuning of learning parameters is required. Norrlöf (1991) and Tayebi (2004) presented an adaptive learning approach for automated tuning of learning parameters.

Another approach is to use RL to adapt DMPs. For example, in Buchli et al. (2011b, 2011a), stiffness parameters were adjusted during the task execution by RL.

Alternatives to the feedback-based adaptation of DMPs and RL are scalability and generalization approaches. Matsubara et al. (2011) proposed an algorithm for the generation of new control policies from existing knowledge, thereby achieving an extended scalability of DMPs, while a mixture of motor primitives was used for generation of table tennis swings (Mülling et al., 2010). On the other hand, generalization of DMPs was combined with model predictive control by Krug and Dimitrov (2015) or applied to DMP coupling terms by Gams et al. (2015a), which were learned and later added to a demonstrated trajectory to generate new joint space trajectories.

Stulp et al. (2013) proposed to learn a function approximator with one regression in the full space of phase and task parameters, bypassing the need for two consecutive regressions. Forte et al. (2012) performed a comparison study of LWR and GPR for trajectory generalization. This work shows that higher accuracy can be achieved with LWR trajectory approximation. Koropouli et al. (2015) presented a generalization approach for force control policies. By learning both the policy and the policy difference data using LWR, they could estimate the policy at new inputs through superposition of the training data.

Deniša et al. (2016a) used GPR-based generalization over combined joint position trajectories and torque commands in the framework of CMPs. To showcase the versatility of the approach, Petric et al. (2018) applied it for robot-based assembly tasks. Finally, Kramberger et al. (2017) extended the approach to account for variations of the desired tasks, *for example,* assembly of similar objects. This enables the robot movements to be automatically generated with the use of LWR from a demonstrated database of successful task executions, which include kinematic and dynamic demonstrated trajectories encoded with DMPs. The newly obtained data is used to account for the changes in the work-space. Nevertheless, a major problem in statistical learning is how to efficiently deal with singularity-free representations of orientation trajectories. To resolve this issue, Kramberger et al. (2016a) proposed a formulation for Cartesian space DMPs where orientations are represented with unit quaternion.

5.2. Human–robot co-manipulation

While control of robot interaction with the passive environment can solve the majority of the tasks, in some cases the robot needs to interact with an active agent (e.g., human and another robot). Human–robot collaboration is becoming one of the key fields in robotics (Ajoudani et al., 2018). To perform a successful physical human–robot collaboration, the robot must be able to control complex movements in coordination with the human partner. In this direction, the ability to modulate the impedance is important to coordinate the physical interaction during human–robot co-manipulation of tools (Peternel et al., 2017b). DMPs offer an elegant solution to encode such coordinated dynamic movements.

In Peternel et al. (2014), the collaborative robot was taught online through teleoperation how to perform collaborative sawing with a human co-worker. The impedance was commanded to the robot through the muscle activity measurement using EMG. DMPs were used to encode coordinated phase-dependent motion and impedance as demonstrated by the human teleoperator. Teaching through teleoperation is an effective way to convey the physical interaction skill to the collaborative robot; however, the setup can be expensive and is not widely available.

An intuitive alternative to teleoperation is for the robot to learn the skill directly through physical interaction with the human partner while they are collaborating. Numerous methods have focused on learning the synchronized motion between collaborative partners (Gams et al., 2014; Kulvicius et al., 2013; Lu et al., 2022; Peternel et al., 2018b; Prada et al., 2013; Sidiropoulos et al., 2019, 2021; Ugur and Girgin, 2020; Umlauft et al., 2014; Wu et al., 2022; Zhou et al., 2016a). For example, in Kulvicius et al. (2013), the interactive movements were encoded with DMPs and adapted based on the measured force arising from the disagreements

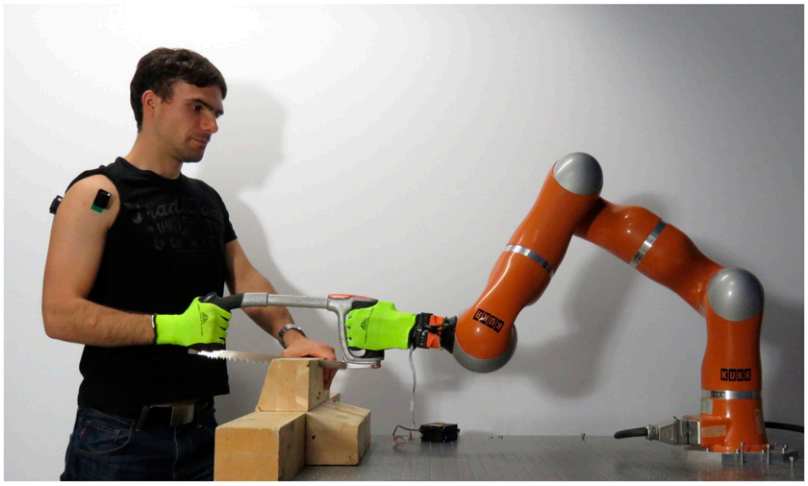

Figure 19. An example of using DMPs for collaborative human–robot sawing from Peternel et al. (2018b).

between agents during co-manipulation. Similarly, in Gams et al. (2014), the collaborative movements were encoded with DMPs and adapted using force feedback and ILC. The approach in Zhou et al. (2016a) combines two DMPs to encode the movements of each partner's arm, which are coupled in a leader-follower manner.

Besides adapting the collaborative movements, in Peternel et al. (2018b), the robot used DMPs to also learn the impedance online directly from the co-manipulation with the human (Figure 19). The robot started with a basic skill set that enabled it to collaborate with the human in a pure follower role. Through the collaborative task execution, the robot then learned the motion and impedance trajectories online and encoded them with DMPs. When the human became fatigued, the robot used the learned advanced skill to take over the majority of the task execution.

The method in Ben Amor et al. (2014) proposed an upgraded version of standard DMPs called *Interaction Primitives* that can account for a probabilistic nature of collaborative movements. Rather than having a single value of weights, the DMP includes weight distributions. This distribution enabled the robot to learn the inherent correlations of cooperative actions and infer the behavior of the human partner during the cooperation. Cui et al. (2016, 2019) used visual information to extract context-related parameters that augment the interaction primitives to increase the robustness during the task execution.

There are also other types of co-manipulation scenarios, such as within-hand bi-manipulation or human–robot object handover. For example, in Amadio et al. (2022), Gao et al. (2019), and Koene et al. (2014) DMPs were used to perform bi-manipulation, while in Abdelrahman et al. (2020), Iori et al. (2023), Lafleche et al. (2019), Prada et al. (2014), and Solak and Jamone (2019) DMPs were used for human–robot object handover.

When the environment is hazardous for human workers or when there are too many robots compared to the number of human workers, the obvious solution is to make robots collaborate between themselves. The method in Peternel and Ajoudani (2017) used DMPs to make novice robots learn from the expert robot through co-manipulation. Initially, the novice robot remained compliant to let the expert robot lead the task execution. In the first stage, the novice robot learned the reference motion through DMPs. In the second stage, it became stiff to perform the newly learned motion, while the expert robot initiated stiff/compliant phases expected in the collaborative task execution. Finally, the novice robot then learned in which phases of the task to increase or decrease the impedance and encoded this impedance behavior with DMPs.

5.3. *Human assistance, augmentation, and rehabilitation*

The most common type of co-manipulation is the classic human–robot collaboration, where a human and a robotic agent are physically performing industrial or daily tasks. Another type of co-manipulation occurs when a human is using a wearable robot such as an exoskeleton. There are different types of functions that the exoskeleton can be used for. One function is augmentation where the current human motion is amplified to augment existing (healthy) human capabilities such as in tasks involving heavy loads. When human capabilities are impaired, the exoskeleton has to act in an assistance function. If human capabilities are impaired to a larger degree, the exoskeleton can be employed in a rehabilitation function to perform physical therapy. In the augmentation function, DMPs can be employed on the robot to offload a hard and/or repetitive motion of healthy human workers, while in the assistance and rehabilitation functions, they can be used to assist impaired humans in their daily tasks or perform repetitive physical therapy on patients that would lead to recovery.

Besides the type of exoskeleton function, another important aspect is the shared load between the exoskeleton and the human during physical human–robot interaction. For example, in the case of highly impaired patients in

early-stage rehabilitation, DMPs generated by the exoskeleton may take almost all the load of the movement and assume the role of the leader to perform passive physiotherapy. As the recovery progresses and more active exercise is preferred, the majority of the load can be shifted to the human who leads the movements, while the DMP system can act in a support capacity as a follower. When full recovery is not possible, DMP can provide assistance in daily tasks where the load can be partially shared. In case of augmentation of existing (healthy) human capabilities, the DMP system can adapt to human movement and add extra power on the top of the human effort that is needed to perform a specific heavy-load task. In some cases, the exoskeleton can learn DMP from human movements to take over a repetitive action completely.

The methods in Lauretti et al. (2017, 2018) obtained DMPs offline by learning by demonstration, which were then used by an arm exoskeleton to support human movements. In Peternel et al. (2016), the control method employed DMPs to interactively adapt the joint torques required to perform the arm exoskeleton assistance and compensate all the underlying dynamics for periodic movements (Figure 21-*Left*). The phase-dependent toque trajectory was updated online in real-time in order to minimize the muscle activity feedback measured by EMG. A similar adaptation capability was achieved for discrete exoskeleton movements by the DMP-based method proposed in Lanotte et al. (2021). In Petrič et al. (2016), the robot encoded the assistive motion with DMPs and then adapted it by taking into account aspects of human motor control through the Fitts' law. Li et al. (2021a) introduced a hierarchical control strategy that enables the learning of human–machine cooperative operations by integrating DMPs and GMM. The proposed approach utilizes well-defined primitives, which offer comprehensive mathematical formulations and rigorous analyses of stability and convergence for control methods based on DMPs.

Gait-related assistance and rehabilitation with exoskeletons is a very common application of DMPs, and there are numerous examples (Abu-Dakka et al., 2015; Amatya et al., 2020; Escarabajal et al., 2023; Huang et al., 2016a; Hwang et al., 2019, 2021; Schaal, 2006; Yuan et al., 2020; Zou et al., 2021). In Abu-Dakka et al., 2015, 2020), a parallel robot was used for ankle rehabilitation, where the movements were generated by DMPs (Figure 20). A similar parallel robot was applied in Escarabajal et al. (2023) for knee rehabilitation, where RevDMPs was used to enable a patient to reverse the movement in order to maintain their own desired pace. In Huang et al. (2016a), DMPs were used to learn the gait motion trajectories for a lower body exoskeleton. This approach was then extended with an RL method to adapt a force coupling term (similar to earlier approaches presented in Section 3.3.2) to enable online adaption of motion trajectories (Huang et al., 2016b). In Luo et al. (2022), DMPs were adapted to the different starting and ending locations of the foot in the swing phase of gait. The method in Xu et al. (2023) used DMPs for leg exoskeleton movements in a mirror therapy concept where the motion of the healthy limb is transferred to the impaired limb. Hong et al. (2023) used DMPs to plan obstacle-avoidance leg movements during walking with a prosthesis.

Figure 20. An example of using DMPs for teaching passive exercises for ankle rehabilitation (Abu-Dakka et al., 2015, 2020).

Besides normal gait, DMPs were also applied for stair-ascend (Xu et al., 2020) and sit-to-stand (Kamali et al., 2016) assistive movements of lower body exoskeletons. In Joshi et al. (2019), a robotic arm was used to assist humans with putting the clothes on their body, where the movements were generated by DMPs. Ding et al. (2022) developed a framework for assistance of older adults combining DMPs admittance control for mobility assistance and manipulation support. The framework was implemented on a mobile platform with a robotic arm utilized for LfD.

Besides assistive body movement and rehabilitation, DMPs were also applied for relaxation purposes. For example, in Li et al. (2020), a robotic arm provided massage movement through DMPs.

5.4. Teleoperation

Teleoperation is one of the major fields of robotics and enables a human to have a direct and real-time control over a (remote) robot. Typically, the control is done through interfaces that can capture the human commands to be sent to the robot and that can provide haptic feedback from the robot. While teleoperation focuses on giving the human operator a full or shared control over the robot, DMPs are used to encode autonomous robot behaviors. Therefore, here we mostly examine cases where teleoperation is used to teach the robot new autonomous behavior encoded by DMPs.

In Kormushev et al. (2011), a combination of kinesthetic teaching and teleoperation was employed to form the DMP-based robot skill for ironing. After the motion trajectories were learned through kinesthetic guidance, the corresponding forces were recorded by using a haptic device and a teleoperation system. In Peternel et al. (2014),

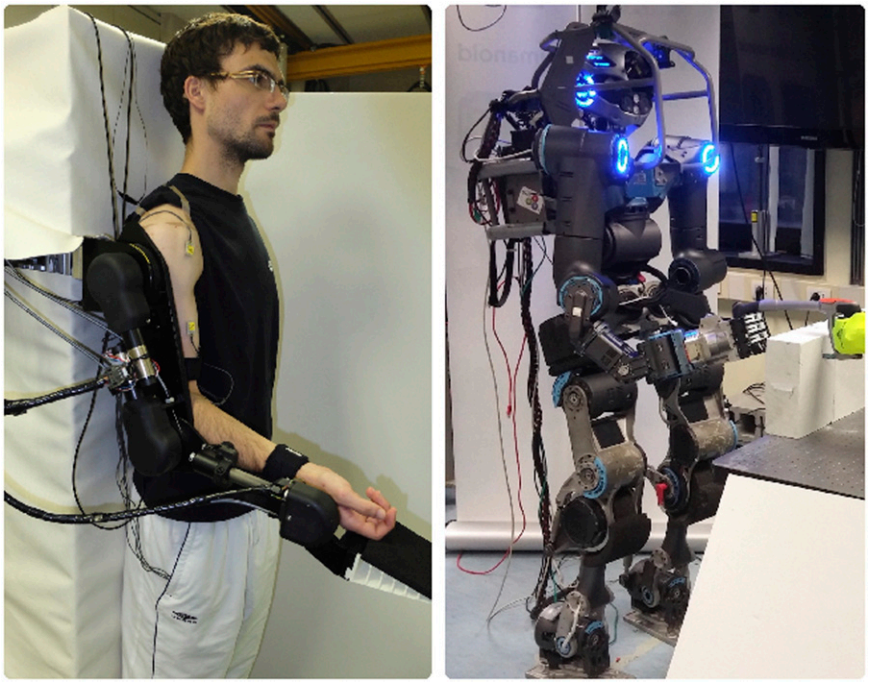

Figure 21. Left photo shows arm exoskeleton application from Peternel et al. (2016). The right photo shows high-DoF humanoid robot Walk-man (Tsagarakis et al., 2017) performing sawing in Peternel and Ajoudani (2017).

teleoperation was used to teach the robot how to physically collaborate with another human. Since there was no haptic feedback, the teleoperation setup was unilateral, but the human was able to teach also the impedance of the robot in addition to motion. The former was commanded by muscle activity measurement through EMG, while the latter was commanded by the movement of the human operator's arm as measured by an optical motion capture system. Similar teleoperation-based DMP learning approaches were used in Lentini et al. (2020) and Yang et al. (2018).

In Peternel et al. (2018a), the human operator taught the robot through teleoperation how to perform autonomous assembly actions (Figure 16-*right*). DMPs were used to encode the commanded impedance and motion; however, a more practical push-button impedance command interface was employed instead of EMG. More importantly, the teleoperation setup was bilateral, and the haptic interface provided the human operator with feedback about the forces the robot felt. Learned DMPs through teleoperation can also be generalized to improve adaptability (Yang et al., 2018). In Luo et al. (2023), DMPs were employed to encode skilled operator movements, which were then used for training novice operators. The method in Zhang et al. (2023) used DMPs to partially automate the surgical process of appendectomy.

A real robot is not always necessary to acquire new skills. In Beik-Mohammadi et al. (2020), the robot and the environment were simulated and the human operator used a virtual reality system. A combination of DMPs and RL was used to form an adaptive skill. The scenario proposed in Abu-Dakka et al. (2015a) was teleoperation in its basis; however, the human demonstrator did not just pretend that he/she is embodied in the robot, but the robot task environment was cloned at the human side (Figure 16-*right*). This removed the need for force feedback and a haptic device, since the human felt the real environment on his/her side, while the motion was captured by a non-contact-based sensory system (i.e., magnetic trackers) and then mirrored on the robot.

Multiple demonstrations through teleoperation can be inconsistent, especially if done in a multi-agent shared-control setting. The method proposed in Pervez et al. (2019) can synchronize inconsistent demonstrations through shared-control teleoperation and encode them with DMPs. Maeda (2022) investigated the possibility of using DMPs to implicitly blend human and robot policies without requiring the design of task-specific arbitration functions or the need to provide multiple (possibly inconsistent) demonstrations.

5.5. High DoF robots

DMPs provide an elegant and fast way to deal with systems with high-dimensional space by sharing one canonical system (3) among all DoFs and maintain only a separate set of transformation systems. By high-dimensional space, we are referring to systems with 10 or more DoFs (i.e., Walk-man humanoid robot in Figure 21-*right*). In this section, we will quickly mention some of the potential works with a high number of DoFs.

Ijspeert et al. (2002b, 2002a) used DMPs in an IL framework to learn tennis forehand, a tennis backhand, and rhythmic drumming using 30-DoFs humanoid robot. Pastor et al. (2009) used DMPs to encode a 10-DoFs exoskeleton

robot arm. Luo et al. (2015) integrated DMPs with stochastic policy gradient RL and GPR in order to design an online adaptive push recovery control strategy. The approach had been applied to PKU-HR5 humanoid robot with 20-DoFs. André et al. (2015, 2016) implemented a predictive model of sensor traces that enable early failure detection for humanoids based on an associative skill memory to periodic movements and DMPs. They applied their algorithm on DARwIn-OP with 20-DoFs in simulation. Pfeiffer and Angulo (2015) represented gestures by applying DMPs on REEM robotic platform with 23-DoFs. Nah et al. (2020) proposed an approach to optimize DMP parameters in order to deal with the complexity of high DoF system like a whip. They tested their approach in simulation for 10-, 15-, 20-, and 25-DoF systems. In order to reduce the number of required rollouts for adaptation to new task conditions, Queißer and Steil (2018) used CMA-ES to optimize DMP parameters. In addition, they introduced a hybrid optimization method that combines a fast coarse optimization on a manifold of policy parameters with a fine-grained parameter search in the unrestricted space of actions. The approach was successfully illustrated in simulation using a 10-DoF robot arm. Liu et al. (2020) proposed DMP-based trajectory generation to enable a full-body humanoid robot with 10-DoFs (for the two legs) to realize adaptive walking. Liang et al. (2021) developed an efficient approach to enable service robots with 26 DoFs with the skill of performing sign language motions.

Travers et al. (2016, 2018) proposed a framework that integrates DMP with Gaussian-shaped spatial activation windows in order to plan the motion for high DoF robotic systems (e.g., snake-like robot) in complex environments (with obstacles) by linking low-level controllers to high-level planners. DMPs can also be applied to soft continuum robots where the separation between DoFs is more blurred (Seleem et al., 2023).

5.6. Motion analysis and recognition

DMPs tend to fit topologically similar trajectories with similar shape parameters w_i (Ijspeert et al., 2013). This behavior, due to the temporal and spatial invariance of DMPs, makes the shape parameters a useful descriptor to recognize similar motions. Indeed, Strachan et al. (2004) have shown that the shape parameters computed for 5 repetitions of 4 classes of discrete hand gestures—measured with a 3 DoF accelerometer—are linearly separable, *that is,* easy to classify. Lantz and Murray-Smith (2004) draw similar conclusions for 10 classes of periodic hand gestures. Xu et al. (2005) used the correlation between the parameter vectors of two DMPs to measure the similarity between the original motion and recognize gait patterns. Similarly, Ijspeert et al. (2013) used the correlation between parameter vectors to recognize the 26 letters of the Graffiti alphabet.

The shape parameters w_i are also suitable to fit more sophisticated classifiers like support vector machines. This strategy was used to successfully classify gestures observed with a monocular (Liu et al., 2014) or a binocular (Wang and Payandeh, 2015) camera. Instead of considering a fixed number of basis functions (number of shape parameters), Zhang et al. (2017) used fast dynamic time warping (Salvador and Chan, 2007) to align parameter vectors of different lengths and then used K-nearest neighbors to classify different motions.

Motion recognition can also be used to determine whether the robot is correctly executing a task by comparing sensed data with a movement template. In this respect, André et al. (2016) used an associative skill memory, like the one in Pastor et al. (2011), as a predictive model of sensor traces that enables early failure detection. In this work, DMPs were used to compactly encode the associative skill memory and speed up the failure detection. Described approaches demonstrate that DMPs are a valuable option for gesture recognition especially for systems with limited computational power. Liang et al. (2021) presented a solution to the motion retargeting problem for generating dual-arm sign language motions. Their approach involves an offline-constrained optimization technique that minimizes the deviation from trajectories generated by DMPs, which encodes the human demonstrations. It should be noted that their approach was exclusively applied and tested in an offline setting.

Humans tend to perform the same task in slightly different manners. Sometimes, differences in the execution style contain useful information to adapt the motion to different executive contexts. This is the case, for instance, of a reaching motion with and without an obstacle on the way. To capture the execution style, Matsubara et al. (2010) augmented the forcing term of the DMP with a style parameter learned from multiple demonstrations. At run time, different style parameters can be used to smoothly interpolate between demonstrated behaviors. Zhao et al. (2014) not only employed movements with different styles but also learned a smooth mapping between style parameters and goal to improve the generalization.

When humans provide seamless demonstrations, DMPs can be used for online segmentation and recognition. To this end, Meier et al. (2011) assumed that a library of DMPs is given and used it to recognize motion segments during a task demonstration. Instead of using exemplar templates for each class of primitives, Chang and Kulić (2013) segmented a video stream using motion to non-motion transitions, fitted DMPs on segmented data, and performed clustering to group similar motion segments in an unsupervised fashion. Song et al. (2020) performed unsupervised trajectory segmentation using the concept of key points, *that is,* shared features across different task demonstrations. Mandery et al. (2016) segmented whole-body motions by detecting contacts with the environment and used them to build a probabilistic language model where words represent the poses and sentence sequences of poses. The learned language model was used to plan whole-body motion trajectories executed by joining multiple DMPs (see Section 3.2). Kordia

and Melo (2021) introduced a method for recognizing an observed trajectory from a library of pre-learned motions and predicting its target position. They leveraged distinctive features of the observed trajectory to aid in the recognition and utilized DMPs for representing the movement.

DMPs have been developed as a computational model of the neurobiological motor primitives (Schaal et al., 2007). Experimental findings from neurophysiology related to the spinal force fields in frogs have inspired the modification of DMP formulation in Hoffmann et al. (2009). As discussed in Section 3.1.1, this multidimensional representation overcomes limitations of classical DMPs like trajectory overshooting and dependence of the trajectory from the reference frame used to describe the motion. Hoffmann et al. (2009) also derived a collision avoidance strategy for DMPs, inspired by the way humans avoid collisions during arm motion. DeWolf et al. (2016) investigated the human ability to cope with changes in the arm dynamics and kinematic structure during motion control. They proposed a spiking neuron model of the motor control system that uses DMPs to implement the preparation and planning functionalities of the premotor cortex. The effects of changes in the robot's dynamic parameters on the tracking performance of a DMP trajectory were studied in Kuppuswamy and Alessandro (2011). Their findings suggest that the change in the body parameters should be explicitly considered in the DMP learning process. Hotson et al. (2016) augmented a brain–machine interface that captures neural signals with a DMP model of the endpoint trajectories executed by a non-human primate. The system was used to decode real trajectories from a primate manipulating four different objects.

5.7. Autonomous driving and field robotics

DMPs can be utilized in various autonomous non-stationary fields of robotics. Perk and Slotine (2006) utilized DMPs for defining flight paths and obstacle avoidance for Unmanned Aerial Vehicles (UAVs), where the trajectories were generated based on the joystick movements controlling the throttle of the UAV motors. Later, Fang et al. (2014) extended the approach to encode user-demonstrated UAV data, extracting and encoding the rhythmic and linear segments of the flight trajectory, and combining them into a flight control skill. Furthermore, Tomić et al. (2014) formulated the UAV movements as an optimal control problem. The output of the optimal control solver was encoded with DMPs, enabling them to generalize and apply in-flight modifications to the UAV flight trajectories in real-time. Similarly, Lee et al. (2018) and Kim et al. (2018a) presented a framework for UAV cooperative areal manipulation tasks, based on an adaptive controller which adapts the movement of the UAV in relation to the mass and inertial properties of the payload. In addition, DMPs were incorporated in the control scheme to modify the flight trajectories and avoid obstacles on the fly. The approach was later extended to incorporate path optimization, where DMPs play a significant role in real-time obstacle avoidance (Lee et al., 2020).

As mentioned before, DMPs represent a versatile movement representation, which can be implemented in various tasks and scenarios. One of the recent applications in this field is also Autonomous Underwater Vehicles (AUVs). Carrera et al. (2015) integrated the DMPs in a learning by demonstration scenario for an AUV. The demonstrated data consisted of the manipulator and vehicle sensory outputs, which were efficiently used to demonstrate an underwater valve-turning task.

DMPs are also represented in the autonomous driving domain. In the recent work of Wang et al. (2018, 2019), the authors propose a framework which decomposes the complex driving data into a more elementary composition of driving skills represented as motion primitives. In the proposed framework, DMPs are utilized to represent the driver's trajectory with acceptable accuracy and can be generalized to different situations.

6. Discussion

This section provides guidelines to choose, among the several discussed in this work, the most appropriate approach for a given application. A useful criterion to decide whether to use a particular approach is the availability of code that greatly simplifies the implementation. We have searched for open-source DMP implementations and listed them in a Git repository (see Section 6.2). To further contribute to the community, we have also released the implementations listed in Table 4. This section ends with a discussion on the limitations inherent to the DMP formulation, the open issues, and the possible research directions. These are summarized in Table 5.

6.1. Guidelines for different applications

Previous sections present different DMP formulations and extensions together with possible application scenarios. As usual, there is not a single formulation that serves all the scopes and purposes, and the suitable approach to use depends on the goal to achieve and the conditions of application. For this reason, we present some guidelines to guide the user in the process of selecting the formulation to use.

6.1.1. Discrete versus periodic. For a task with distinct starting and ending points, discrete DMPs are a logical option to encode the movement trajectories between them. Examples of these tasks include reaching and pick-and-place (Caccavale et al., 2019; Deniša et al., 2016a; Forte et al., 2012; Stulp et al., 2009), specific actions of assembly (Krüger et al., 2014; Abu-Dakka et al., 2014; Nemec et al., 2020; Angelov et al., 2020), and cutting (Straizys et al., 2020; Yang et al., 2018).

When the starting and ending points coincide, periodic DMPs are the logical option, since the encoded movements can be repeated over and over again. Good examples of their application are repetitive tasks such as locomotion (Nakanishi et al., 2004; Rückert and d'Avella, 2013; M. Wensing and Slotine, 2017), human body augmentation/

Table 5. A summary of DMP features and limitations that have been solved (✓) or partially solved (▮).

Limitation	Related work	Status
Via-points	Ning et al. (2011, 2012), Weitschat and Aschemann (2018), Saveriano et al. (2019), and Zhou et al. (2019)	✓
Start-point	Hoffmann et al. (2009), Ijspeert et al. (2013), Weitschat et al. (2013), and Dragan et al. (2015)	✓
Goal-point	Ijspeert et al. (2013), Weitschat et al. (2013), Abu-Dakka and Kyrki (2020), Dragan et al. (2015), and Weitschat and Aschemann (2018)	✓
Obstacle avoidance	Park et al. (2008), Hoffmann et al. (2009), Tan et al. (2011), Kim et al. (2015), and Rai et al. (2017)	✓
Geometry-constrained data	Pastor et al. (2009), Abu-Dakka et al. (2015a), Ude et al. (2014), Saveriano et al. (2019), and Abu-Dakka and Kyrki (2020)	▮[5]
Probabilistic	Ben Amor et al. (2014)	▮
Extrapolation	Pervez and Lee (2018) and Zhou et al. (2019)	▮
High-dim input	Pervez et al. (2017a) and Pahič et al. (2020)	▮
Closed-loop	Peternel et al. (2016) and Kramberger et al. (2018)	▮
Multi-attractor	Nemec et al. (2018) and Iturrate et al. (2019)	▮

rehabilitation (Peternel et al., 2016), wiping a surface (Gams et al., 2016; Kramberger et al., 2018; Peternel et al., 2017a), and sawing (Peternel et al., 2018b). Nevertheless, even typically non-repetitive tasks that are executed just once every now and then can still be encoded with periodic DMPs when the starting and ending points coincide (Peternel et al., 2018a).

There are cases where it is not possible to clearly distinguish if the motion is periodic or discrete. For instance, Ernesti et al. (2012) have shown that the first step in a gait of a humanoid robot is a transient toward a periodic motion. Their representation is a good candidate to encode transients converging to limit cycle trajectories. Finally, in some cases like in complex assembly, the task requires a combination of discrete and periodic DMPs (Sloth et al., 2020).

6.1.2. Space representation. The original formulations of DMPs were and are still successfully applied to multidimensional independent data with each DoF $\in\mathbb{R}$ (Sections 2.1.1 and 2.2.1). These data can be joint or Cartesian positions, forces, torques, *etc.*, where every DoF of the data can be evolved independently from the rest. However, such formulation is not enough to successfully encode data with specific geometry constraints without pre- and/or post-processing the data. Examples of such data are *i*) orientation, where data are tight up by additional constraints (i.e., the orthogonality in case of rotation matrix representation or the unit norm of the quaternion representation); *ii*) full stiffness/damping matrices and manipulability matrices are encapsulated in SPD matrices.

In many early works, orientation trajectories were learned and adapted without considering their geometry constraints (Pastor et al., 2009), leading to improper orientation and hence requiring an additional re-normalization. In a different example, Umlauft et al. (2017) used eigendecomposition for impedance adaptation.

In order to comply with such geometry constraints, researchers provided a new formulation of DMPs that ensures proper unit quaternions or rotation matrices over the course of orientation adaptation (Abu-Dakka et al., 2015a; Koutras and Doulgeri, 2020a; Saveriano et al., 2019; Ude et al., 2014 and proper SPD matrices over the course of the adaptation of SPD profiles (e.g., stiffness or manipulability ellipsoids) (Abu-Dakka and Kyrki, 2020). We believe that using these geometry-aware DMPs is preferable to encode data with underlying geometry constraints.

6.1.3. Weights learning method. DMPs represent motion trajectories as stable dynamical systems with learnable weights that define the shape of the motion. In the LfD paradigm, DMP weights are usually learned in a supervised manner using human demonstrations. The procedure used to transform human demonstrations into training data for the DMP forcing term is highlighted in Section 2.1.1.1. The number of weights, which corresponds to the number of RBFs used to approximate the forcing term, is a hyper-parameter that is typically provided by the user. As practical tuning guidelines, one has to consider that the number of RBFs increases with the length of the trajectory and with its complexity, which depends on changes in concavity and frequency and magnitude of picks. Given the training data, and the number of RBFs, different techniques can be used to fit the weights.

LWR is widely used when the forcing term is a combination of RBFs as in (4). However, in the literature one can use RBF-NN as in Si et al. (2021) or if multiple demonstrations are given, one can exploit GMM/GMR as in Li et al. (2021b) and Pervez et al. (2017a) or GPR as in Fanger et al. (2016) to represent the forcing term and use expectation–maximization to fit the (hyper-)parameters. Deep NNs, typically trained via back-propagation, seem an appealing possibility to map input images into forcing terms (Pervez et al., 2017b), mimicking the human perception-action loop. Although appealing, the possibility of exploiting deep learning techniques as motion primitives requires further investigations.

In real applications, there can be a misplacement between the DMP trajectory and the robot motion. Typical examples include assembly or other tasks that require physical interaction with the environment (see Section 5.1). In this

situation, the DMP motion can be incrementally adjusted to improve the robot's performance. ILC arises as an interesting approach to iteratively update the DMP weights as it ensures a rapid convergence to the desired performance (Abu-Dakka et al., 2015a; Gams et al., 2014; Kramberger et al., 2018). However, ILC assumes that a target behavior to reproduce is given. When the target behavior cannot be easily specified and the robot performance is not satisfactory, RL solutions have to be adopted. As detailed in Section 4.3, DMPs are effective control policies and, combined with policy search algorithms like PI^2 or PoWER, are able to solve complex and highly dynamic tasks.

6.1.4. Online adaptation. Performing robotic tasks in the real world requires adaptation capabilities. When adaptation of DMPs based on some feedback is required, one of the extension methods should be applied. For example, to change the existing movement based a detected obstacle, the method in Gams et al. (2016), Hoffmann et al. (2009), Park et al. (2008), and Tan et al. (2011) can be used (see Section 3.3.1). If it is necessary to adaptively learn the movement dynamics based on real-time effort feedback, the method in Peternel et al. (2016) can be employed (see Section 3.3).

Furthermore, for industrial tasks, such as assembly or polishing, adaptation strategies combining force control with demonstrated trajectories can be applied (Abu-Dakka et al., 2015a; Gams et al., 2010; Kramberger et al., 2016), ensuring the system will follow the predefined trajectory and adapt to the environmental uncertainties. For online adaptation DMPs can be used as a trajectory generator, which output represents an input to the force control algorithm, on the other hand, force feedback can directly be incorporated as a coupling term in the DMP formulation (see Section 3.3.2), eliminating the need for an additional force controller. A similar approach can also be utilized for velocity-based adaptation of the movements (see Section 3.3.4).

6.1.5. Impedance versus force. In physical interaction tasks, DMPs can be used to either learn force or impedance (Peternel et al., 2017a). If the task requires position control, then the impedance should be learned with DMPs in combination with the reference position. If the task requires to control a specific force, *for example,* pushing on a surface during the wiping and drilling, either force or impedance is feasible. However, if safety is the most critical aspect, the DMPs should be used to learn impedance control so that the robot can be made soft.

Furthermore, to overcome any undesirable movements, the control policy can be augmented with a tank-based passivity approach (Shahriari et al., 2017). This approach monitors the energy flow between the modeled sub-systems, *for example,* DMP trajectory generation, impedance control, and environment. In an event of an energy violation, the system will first try to passively compensate for the violation and subsequently, if the violation cannot be compensated, for example, the energy tank is depleted, stop the system. In cases, where the task characteristics are not fully known, a learning policy can be added on the top of the passivity approach (Kramberger et al., 2018) in order to learn the overall energy requirements for the task.

6.2. Resources and codes

The availability of code and datasets is useful to speed up the setup of novel applications without the need of re-implementing a promising approach from scratch. We have searched for available DMP implementation and found out that several researchers published their DMP codes in various open-source repositories. We decided to list the available implementations on the Git repository that accompanies this paper (https://gitlab.com/dmp-codes-collection/third-party-dmp). For each implementation, we mention the type of DMP, the author, the URL to download the code, and the used programming language. We also provide a short description of the key features.

Apart from listing existing approaches, the Git repository that accompanies this paper contains an implementation that we decided to release to the community. The list of provided implementations is given in Table 4.

6.3. Limitations and open issues

As any motion primitive representation, DMPs have strengths but also inherent limitations. The advantages of the DMPs have been widely discussed in previous sections. Here, we present the main limitations of the DMPs and discuss open issues that require further investigation. A summary of these limitations is presented in Table 5.

6.3.1. Implicit time dependency. The phase variable used to suppress the nonlinear forcing term and ensure convergence to a given goal introduces an implicit time dependency in the DMP formulation. The reason for representing the time dependency implicitly as a dynamical system is that such a phase variable can be conveniently manipulated. For example, in Section 2.1.1.2, we have seen how to manipulate the phase variable to slow down (or even stop) the execution. A drawback of the time dependency is that the shape of the DMP motion is significantly affected by the time evolution of the phase variable. If the phase vanishes too early, the last part of the trajectory is executed with a linear dynamics converging to the goal. If the phase lasts too long, the trajectory may overshoot and fail to reach the goal within the desired time. In both cases, the DMP motion may significantly deviate from the demonstration. A properly designed phase-stopping mechanism can remedy the issue, but the proper phase-stopping to adopt depends on the specific application.

In order to overcome this limitation, several authors focused on learning stable and time-independent (or autonomous) dynamical systems from demonstrations. A globally stable and autonomous system generates a vector field that converges to the given goal from any initial state. Without the need for a phase variable, the generated motion

depends only on the current state of the system. Notable approaches to learn stable and autonomous systems exploit Lyapunov theory (Khansari-Zadeh and Billard, 2011, 2014), contraction theory (Blocher et al., 2017; Ravichandar and Dani, 2015), diffeomorphic transformations (Perrin and Schlehuber-Caissier, 2016; Neumann and Steil, 2015), and passivity considerations (Kronander and Billard, 2015). These approaches have been effectively used to learn complex movements from demonstrations.

In general, autonomous systems have the potential to represent much more complex movements than DMPs. For example, autonomous systems can encode different motions in different regions of the state-space. In this respect, DMPs can only generate a stereotypical trajectory connecting the start to the goal, regardless of where the initial state is placed in the state-space. However, the stereotypical motion generation is also an advantage of DMPs since it makes it easier to predict the generated motion in regions of the state-space poorly covered by training data. On the contrary, it is hard to predict how an autonomous system generalizes where only a few or no training data are available. DMPs are known to scale well in high-dimensional spaces since the learned forcing term always depends on a shared, scalar phase variable. Autonomous systems perform learning directly on the high-dimensional state-space, which poses numerical challenges and requires much more training data. In synthesis, each representation has its own advantages and disadvantages and the choice between time-dependent and autonomous motion primitives depends on the specific application.

6.3.2. Stochastic information. Representing the demonstrated motion as a probability distribution has several advantages. For example, in a probabilistic framework, the generalization to new a goal (or a via-point) is achieved using conditioning on the new goal (via-point), while the covariance computed from the probability distribution can represent couplings between different DoFs (Paraschos et al., 2013). As a matter of fact, classical DMPs are deterministic and lack stochastic information on the modelled motion.

Ben Amor et al. (2014) proposed an approach to estimate the predictive distribution $\mathcal{P}(\mathbf{w}|y_{1:\mathfrak{T}})$ that relates the DMP weights $\mathbf{w}$ and a partial trajectory $y_{1:\mathfrak{T}}$ observed for $\mathfrak{T}$ time instants. $\mathcal{P}(\mathbf{w}|y_{1:\mathfrak{T}})$ is used to estimate the most likely weights given a partial movement and to reconstruct the missing part of the trajectory. However, a full probabilistic characterization of DMPs is still missing.

The ProMP framework (Paraschos et al., 2013) proposed an alternative movement primitive representation that contains information about the variability across different demonstrations as well as different DoFs in the form of a covariance matrix. This enables to explicitly encode the couplings between different directions and to increase the generalization by conditioning on a desired goal, via-point, or intermediate velocity. The covariance computed by ProMPs represents the variability and the correlation in the demonstrations. In other representations, like GPR, the covariance is a measure of the model uncertainty due to the lack of training data. An attempt to unify ProMP and DMP formulations was made in Li et al. (2023). Kernelized Movement Primitives (KMPs) (Huang et al., 2019; Silvério et al., 2019) offer the possibility of modelling variability, correlation, and uncertainty in the same framework. However, KMP's computational cost can be elevated compared to DMP in longer trajectories due to the computation of the inverse of the kernel matrix.

6.3.3. Closed-loop implementation and issues. A vast majority of methods employ DMPs only as a reference trajectory generator for the closed-loop controller, which then actually executes it. However, the DMPs can also be used as a part of the closed-loop controller itself where the sensor measurements, for example, forces and torques, are used as a coupling term in the DMP for changing its behavior. In other words, in the open-loop case, the DMP serves as the plan and does not change online during the execution (perhaps iteratively after each execution), while in the closed-loop case, DMP serves as the action generator and changes online during the execution. Until now, only a few methods explored the closed-loop concept. For example, in Peternel et al. (2016), the DMPs are directly torque generators for exoskeleton actuators in the control loop, which is closed by feedback from the human user's muscle activity. Nevertheless, in such scenarios, the closed-loop stability and passivity become crucial considerations that have to be addressed and resolved before the widespread application (Kramberger et al., 2018).

6.3.4. Coping with high-dimensional inputs. One of the main limitations of DMP is that it encodes human and robot trajectories explicitly with the time (i.e., 1–D input) which may lead to synchronization issues since human motions in the new evaluations could be significantly different (e.g., faster/slower velocity) from the demonstrated ones. In order to avoid synchronization problem, Ben Amor et al. (2014) designed a time-alignment strategy, while Pervez et al. (2017a) estimated the phase signal during the training using expectation-maximization (Bishop, 2006).

As the DMP models trajectories using basis functions, this works effectively when learning time-driven trajectories (i.e., 1–D input). However, when demonstrations comprise high-dimensional inputs, specifying the center vectors and widths of basis functions becomes quite cumbersome. Specifically, as discussed in Bishop (2006) the number of basis functions often increases exponentially when the dimension of inputs increases. To alleviate this limitation, some approaches investigated modern deep-learning techniques. Pahič et al. (2020) used a deep NN to synthesize DMP weights from an input image. The classical DMP formulation is then used to generate motion trajectories. Pervez et al. (2017b) used a CNN (LeCun et al., 2015) to predict 2–D task parameters (e.g., the position of a target) from an input image and a fully

connected NN to retrieve the forcing term from the 2–D parameters and the phase variable. The CNN and the fully connected NN are trained in two separate stages. The approach is promising, but the separate training of the two networks increases the pre-processing and complicates the learning process.

Alternative approaches in the literature, such as GMM/GMR (Calinon, 2016), Task-Parameterized GMM (TP-GMM) (Calinon, 2016), and KMP (Huang et al., 2019, 2021), can be directly applied for learning demonstrations comprising high-dimensional inputs.

6.3.5. Multi-attractor systems. The well-known second-order dynamic properties of the DMPs strive toward a single attractor system (Ijspeert et al., 2002a). The properties, *for example,* convergence and modulation of the motion, are well studied and implementations can be found in many research papers. Because of the second-order dynamics, the system becomes unstable if, for example, the motion is reversed during the execution. In the past years, two main approaches describing the reversibility problem have been introduced. In the first approach (Nemec et al., 2018), reversibility is considered as learning two separate primitives, one for each direction of the motion. The approach is promising, but does not reflect true reversibility, because it uses one attractor point for each primitive.

On the other hand, Iturrate et al. (2019) introduced an alternative formulation with two stable attractor systems. The first attractor is defined at the starting point y_0 of the trajectory and the subsequent one at the goal g, and the dynamical system between them guarantees a stable convergence depending on the selected attractor. The approach demonstrated true reversibility, while keeping all the DMP properties. Nevertheless, all questions have not been resolved yet, and the approach was evaluated on tasks and joint space position trajectories. A proper formulation for dealing with orientations, for example, quaternions in task space, is still missing.

7. Concluding remarks

Since their introduction in the early 2000s, DMPs have established as one of the most used and popular approaches for motor command generator systems in robotics. Several authors have exploited and extended the classical formulation to overcome some limitations and fulfill different requirements. Their research resulted in a large number of papers published over the last two decades.

One of the aims of this paper is to categorize and review the vast literature on DMPs. We took a systematic review approach and automatically searched for DMP-related papers in a popular database. A manual inspection of the resulting papers, guided by clear and unbiased criteria, led to the papers included in this tutorial survey.

Another aim of our work is to provide a tutorial on DMPs that presents the classical formulation and the key extensions in rigorous mathematical terms. We made an effort to unify the notation among different approaches in order to make them easier to understand. Moreover, we provide useful guidelines that guide the reader to select the right approach for a given application. In the tutorial vein, we have also searched for open-source implementations of the described approaches and released to the community several implementations of DMP-based approaches.

Advantages of DMPs have been discussed as well as their limitations and the open issues. We have summarized them in Table 5 where we also indicate the solved issues and the ones that require further investigation. In this respect, as research on DMP is still very active, we provide a comprehensive discussion that will help the readers to understand what has been done in the field and where they can put their research focus.

Acknowledgments

Part of the research presented in this work has been conducted when M. Saveriano was at the Department of Computer Science, University of Innsbruck, Innsbruck, Austria, and when Fares J. Abu-Dakka was at the Department of Electrical Engineering and Automation, Aalto University, Espoo, Finland.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by the euROBIN under grant agreement no. (101070596), CHIST-ERA project IPALM (Academy of Finland decision 326,304), The Austrian Research Foundation (Euregio IPN 86-N30, OLIVER), Innovation Fund Denmark (Research and innovation project MADE FAST), and European Union, Next-GenerationEU project iNest (ECS 00000043).

ORCID iDs

Matteo Saveriano https://orcid.org/0000-0002-9784-3973
Fares J Abu-Dakka https://orcid.org/0000-0001-9062-9416
Aljaž Kramberger https://orcid.org/0000-0002-4830-4885
Luka Peternel https://orcid.org/0000-0002-8696-3689

Notes

1. As discussed by Pastor et al. (2009), a transformation system that uses (48) generates a mirrored trajectory while reaching a new goal g_{new} every time the signs of $(g_{new} - y_0)$ and $(g - y_0)$ differ.
2. Saveriano et al. used the multi-dimensional DMP formulation developed in Hoffmann et al. (2009) for both position and quaternion DMPs. In this review paper, we reformulate the merging approaches in Saveriano et al. (2019) to comply with the formulations in Section 2.1.1 and 2.1.2.1.
3. Note that other feedback that measures human effort can be used instead of EMG, such as joint torque or limb forces.

4. There is a duality in impedance-admittance, *that is*, the force produces motion and motion produces force; therefore if one is the input, the other can only be the output of the control system (Peternel et al. 2017a).
5. The referred work extended the classical DMP to different space like $\mathcal{SO}(3)$ or $\mathcal{S}^m_{++}$. Although formally similar, the extention to other Riemannian manifolds like the Grassmannian or the Hyperbolic manifolds is non-trivial and still not fully addressed.

References

Abdelrahman A, Mitrevski A and Ploger P (2020) Context-aware task execution using apprenticeship learning. In: IEEE international conference on robotics and automation, Paris, France, 31 May 2020–31 August 2020, pp. 1329–1335.

Abu-Dakka FJ and Kyrki V (2020) Geometry-aware dynamic movement primitives. In: IEEE international conference on robotics and automation. Paris, France, 31 May 2020–31 August 2020, pp. 4421–4426.

Abu-Dakka FJ and Saveriano M (2020) Variable impedance control and learning – a review. *Frontiers in Robotics and AI* 7: 177.

Abu-Dakka FJ, Nemec B, Jørgensen JA, Savarimuthu TR, Krüger N and Ude A (2015a) Adaptation of manipulation skills in physical contact with the environment to reference force profiles. *Autonomous Robots* 39(2): 199–217.

Abu-Dakka F, Nemec B, Kramberger A, Buch AG, Krüger N and Ude A (2014) Solving peg-in-hole tasks by human demonstration and exception strategies. *Industrial Robot* 41(6): 575–584.

Abu-Dakka FJ, Rozo L and Caldwell DG (2018) Force-based variable impedance learning for robotic manipulation. *Robotics and Autonomous Systems* 109: 156–167.

Abu-Dakka FJ, Valera A, Escalera J, et al. (2015b) Trajectory adaptation and learning for ankle rehabilitation using a 3-prs parallel robot. In: H Liu, N Kubota, X Zhu, et al. (eds.) *Intelligent Robotics and Applications*. Cham: Springer International Publishing, 483–494.

Abu-Dakka FJ, Valera A, Escalera JA, Abderrahim M, Page A and Mata V (2020) Passive exercise adaptation for ankle rehabilitation based on learning control framework. *Sensors* 20(21): 6215.

Abu-Dakka FJ, Saveriano M and Peternel L (2021) Periodic DMP formulation for quaternion trajectories. In: IEEE international conference on advanced robotics, Ljubljana, Slovenia, 06–10 December 2021, pp. 658–663.

Aein M, Aksoy E, Tamosiunaite M, et al. (2013) Toward a library of manipulation actions based on semantic object-action relations. In: IEEE/RSJ international conference on intelligent robots and systems, Tokyo, Japan, 03–07 November 2013, pp. 4555–4562.

Agostini A, Saveriano M, Lee D, et al. (2020) Manipulation planning using object-centered predicates and hierarchical decomposition of contextual actions. *IEEE Robotics and Automation Letters* 5(4): 5629–5636.

Ajallooeian M, Van Den Kieboom J, Mukovskiy A, et al. (2013) A general family of morphed nonlinear phase oscillators with arbitrary limit cycle shape. *Physica D: Nonlinear Phenomena* 263: 41–56.

Ajoudani A, Tsagarakis N and Bicchi A (2012) Tele-impedance: teleoperation with impedance regulation using a body–machine interface. *The International Journal of Robotics Research* 31(13): 1642–1656.

Ajoudani A, Zanchettin AM, Ivaldi S, et al. (2018) Progress and prospects of the human–robot collaboration. *Autonomous Robots* 42(5): 957–975.

Alizadeh T, Malekzadeh M and Barzegari S (2016) Learning from demonstration with partially observable task parameters using dynamic movement primitives and Gaussian process regression. In: IEEE/ASME international conference on advanced intelligent mechatronics, Banff, Alberta, Canada, 12–15 July 2016, pp. 889–894.

Amadio F, Laghi M, Raiano L, et al. (2022) Target-referred DMPS for learning bimanual tasks from shared-autonomy telemanipulation. In: IEEE-RAS international conference on humanoid robots, Ginowan, Japan, 28–30 November 2022, pp. 496–503.

Amatya S, Rezayat Sorkhabadi S and Zhang W (2020) Human learning and coordination in lower-limb physical interactions. In: Proceedings of the american control conference, Denver, CO, USA, 01–03 July 2020, volume 2020-July, pp. 557–562.

Anand AS, Ostvik A, Grotli EI, et al. (2021) Real-time temporal adaptation of dynamic movement primitives for moving targets. In: International conference on advanced robotics, Ljubljana, Slovenia, 06–10 December 2021, pp. 261–268.

André J, Teixeira C, Santos C, et al. (2015) Adapting biped locomotion to sloped environments: combining reinforcement learning with dynamical systems. *Journal of Intelligent and Robotic Systems: Theory and Applications* 80(3–4): 625–640.

André J, Santos C and Costa L (2016) Skill memory in biped locomotion: using perceptual information to predict task outcome. *Journal of Intelligent and Robotic Systems: Theory and Applications* 82(3–4): 379–397.

Angelov D, Hristov Y, Burke M, et al. (2020) Composing diverse policies for temporally extended tasks. *IEEE Robotics and Automation Letters* 5(2): 2658–2665.

Atkeson CG, Moore AW and Schaal S (1997) Locally weighted learning. *Artificial Intelligence Review* 11: 11–73.

Basa D and Schneider A (2015) Learning point-to-point movements on an elastic limb using dynamic movement primitives. *Robotics and Autonomous Systems* 66: 55–63.

Beetz M, Stulp F, Esden-Tempski P, et al. (2010) Generality and legibility in mobile manipulation: learning skills for routine tasks. *Autonomous Robots* 28(1): 21–44.

Beik-Mohammadi H, Kerzel M, Pleintinger B, et al. (2020) Model mediated teleoperation with a hand-arm exoskeleton in long time delays using reinforcement learning. In: IEEE international conference on robot and human interactive communication, Naples, Italy, 31 August 2020–04 September 2020, pp. 713–720.

Ben Amor H, Neumann G, Kamthe S, et al. (2014) Interaction primitives for human-robot cooperation tasks. In: IEEE international conference on robotics and automation. Hong Kong, China, pp. 2831–2837.

Bian F, Ren D, Li R, et al. (2019) An extended DMP framework for robot learning and improving variable stiffness manipulation. *Assembly Automation* 40(1): 85–94.

Billard A, Calinon S and Dillmann R (2016) Learning from humans. In: B Siciliano and O Khatib (eds) *Handbook of Robotics*, Chapter 74. Secaucus, NJ, USA: Springer, pp. 1995–2014. 2nd edition.

Bishop CM (2006) *Linear Models for Regression*. Springer, pp. 172–173.

Bitzer S and Vijayakumar S (2009) Latent spaces for dynamic movement primitives. In: IEEE-RAS international conference on humanoid robots, Paris, France, 07–10 December 2009, pp. 574–581.

Blocher C, Saveriano M and Lee D (2017) Learning stable dynamical systems using contraction theory. In: International conference on ubiquitous robots and ambient intelligence, Jeju, Korea (South), 28 June 2017–01 July 2017, pp. 124–129.

Bristow DA, Tharayil M and Alleyne AG (2006) A survey of iterative learning control. *Control Systems Magazine* 26(3): 96–114.

Buchli J, Stulp F, Theodorou E, et al. (2011a) Learning variable impedance control. *The International Journal of Robotics Research* 30(7): 820–833.

Buchli J, Theodorou E, Stulp F, et al. (2011b) Variable impedance control: a reinforcement learning approach. *Robotics: Science and Systems VI* : 153.

Burdet E, Osu R, Franklin DW, et al. (2001) The central nervous system stabilizes unstable dynamics by learning optimal impedance. *Nature* 414(6862): 446–449.

Caccavale R, Saveriano M, Fontanelli G, et al. (2018) Imitation learning and attentional supervision of dual-arm structured tasks. In: Joint IEEE international conference on development and learning and on epigenetic robotics. Lisbon, Portugal, pp. 66–71.

Caccavale R, Saveriano M, Finzi A, et al. (2019) Kinesthetic teaching and attentional supervision of structured tasks in human–robot interaction. *Autonomous Robots* 43(6): 1291–1307.

Calinon S (2016) A tutorial on task-parameterized movement learning and retrieval. *Intelligent Service Robotics* 9(1): 1–29.

Calinon S, D'halluin F, Caldwell DG, et al. (2009) Handling of multiple constraints and motion alternatives in a robot programming by demonstration framework. In: IEEE-RAS international conference on humanoid robots, Paris, France, 07–10 December 2009, pp. 582–588.

Calinon S, Li Z, Alizadeh T, et al. (2012) Statistical dynamical systems for skills acquisition in humanoids. In: IEEE-RAS international conference on humanoid robots, Osaka, Japan, pp. 323–329.

Carrera A, Palomeras N, Hurtós N, et al. (2015) Cognitive system for autonomous underwater intervention. *Pattern Recognition Letters* 67: 91–99.

Chang G and Kulić D (2013) Motion learning from observation using affinity propagation clustering. In: IEEE international symposium on robot and human interactive communication, Gyeongju, Korea (South), 26–29 August 2013, pp. 662–667.

Chen Z and Liu B (2018) Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 12(3): 1–207.

Chen N, Bayer J, Urban S, et al. (2015) Efficient movement representation by embedding dynamic movement primitives in deep autoencoders. In: IEEE-RAS international conference on humanoid robots, Seoul, Korea, November 2015, pp. 434–440.

Chen N, Karl M and Van Der Smagt P (2016) Dynamic movement primitives in latent space of time-dependent variational autoencoders. In: IEEE-RAS international conference on humanoid robots, Cancun, Mexico, 15–17 November 2016, pp. 629–636.

Chi W, Liu J, Abdelaziz M, et al. (2018) Trajectory optimization of robot-assisted endovascular catheterization with reinforcement learning. In: IEEE/RSJ international conference on intelligent robots and systems, Madrid, Spain, 01–05 October 2018, pp. 3875–3881.

Chiaverini S and Siciliano B (1999) The unit quaternion: a useful tool for inverse kinematics of robot manipulators. *Systems Analysis Modelling Simulation* 35(1): 45–60.

Cho N, Lee S, Suh I, et al. (2019) Relationship between the order for motor skill transfer and motion complexity in reinforcement learning. *IEEE Robotics and Automation Letters* 4(2): 293–300.

Churchill A and Fernando C (2014) An evolutionary cognitive architecture made of a bag of networks. *Evolutionary Intelligence* 7(3): 169–182.

Cohen A and Berman S (2014) Integrating simulation with robotic learning from demonstration. In: European conference on modelling and simulation, Brescia, Italy, May 27th–30th, 2014, pp. 421–427.

Cohn DA, Ghahramani Z and Jordan MI (1996) Active learning with statistical models. *Journal of Artificial Intelligence Research* 4: 129–145.

Colomé A and Torras C (2014) Dimensionality reduction and motion coordination in learning trajectories with dynamic movement primitives. In: IEEE/RSJ international conference on intelligent robots and systems, Chicago, IL, USA, 2014, pp. 1414–1420.

Colomé A and Torras C (2018) Dimensionality reduction for dynamic movement primitives and application to bimanual manipulation of clothes. *IEEE Transactions on Robotics* 34(3): 602–615.

Colome A, Planells A and Torras C (2015) A friction-model-based framework for reinforcement learning of robotic tasks in non-rigid environments. In: IEEE international conference on robotics and automation, Seattle, WA, USA, 26–30 May 2015, pp. 5649–5654.

Cui Y, Poon J, Matsubara T, et al. (2016) Environment-adaptive interaction primitives for human-robot motor skill learning. In: IEEE-RAS international conference on humanoid robots, Cancun, Mexico, 15–17 November 2016, pp. 711–717.

Cui Y, Poon J, Miro JV, et al. (2019) Environment-adaptive interaction primitives through visual context for human–robot motor skill learning. *Autonomous Robots* 43(5): 1225–1240.

Cui Z, Ma W, Lai J, et al. (2022) Coupled multiple dynamic movement primitives generalization for deformable object manipulation. *IEEE Robotics and Automation Letters* 7(2): 5381–5388.

Dahlin A and Karayiannidis Y (2020) Adaptive trajectory generation under velocity constraints using dynamical movement primitives. *IEEE Control Systems Letters* 4(2): 438–443.

Dahlin A and Karayiannidis Y (2021) Temporal coupling of dynamical movement primitives for constrained velocities and accelerations. *IEEE Robotics and Automation Letters* 6(2): 2233–2239.

Davchev T, Luck KS, Burke M, et al. (2022) Residual learning from demonstration: adapting DMPS for contact-rich manipulation. *IEEE Robotics and Automation Letters* 7(2): 4488–4495.

Deniša M and Ude A (2013a) Discovering new motor primitives in transition graphs. *Advances in Intelligent Systems and Computing* 193(1): 219–230.

Deniša M and Ude A (2013b) New motor primitives through graph search, interpolation and generalization. *Studies in Computational Intelligence* 466: 137–148.

Deniša M and Ude A (2015) Synthesis of new dynamic movement primitives through search in a hierarchical database of example movements. *International Journal of Advanced Robotic Systems* 12(10).

Deniša M, Gams A, Ude A, et al. (2016a) Learning compliant movement primitives through demonstration and statistical generalization. *IEEE/ASME Transactions on Mechatronics* 21(5): 2581–2594.

Deniša M, Petrič T, Gams A, et al. (2016b) A review of compliant movement primitives. In: EG Hurtado (ed) *Robot Control*, Chapter 1. Rijeka: IntechOpen, 1–17.

Denisa M, Schwaner KL, Iturrate I, et al. (2021) Semi-autonomous cooperative tasks in a multi-arm robotic surgical domain. In: IEEE international conference on advanced robotics, Ljubljana, Slovenia, 06–10 December 2021, pp. 134–141.

DeWolf T, Stewart T, Slotine JJ, et al. (2016) A spiking neural model of adaptive arm control. *Proceedings of the Royal Society B: Biological Sciences* 283(1843).

Ding L, Xing H, Torabi A, et al. (2022) Intelligent assistance for older adults via an admittance-controlled wheeled mobile manipulator with task-dependent end-effectors. *Mechatronics* 85.

Do M, Schill J, Ernesti J, et al. (2014) Learn to wipe: a case study of structural bootstrapping from sensorimotor experience. In: IEEE international conference on robotics and automation, Hong Kong, China, May 31–June 7, 2014, pp. 1858–1864.

Dometios A, Zhou Y, Papageorgiou X, et al. (2018) Vision-based online adaptation of motion primitives to dynamic surfaces: application to an interactive robotic wiping task. *IEEE Robotics and Automation Letters* 3(3): 1410–1417.

Dong S, Yang Z, Zhang W, et al. (2023) Dynamic movement primitives based on positive and negative demonstrations. *International Journal of Advanced Robotic Systems* 20(1).

Dragan AD, Mülling K, Bagnell JA, et al. (2015) Movement primitives via optimization. In: IEEE international conference on robotics and automation. Seattle, WA, USA, pp. 2339–2346.

Duminy N, Nguyen S and Duhaut D (2017) Strategic and interactive learning of a hierarchical set of tasks by the poppy humanoid robot. In: IEEE international conference on development and learning and epigenetic robotics, Cergy-Pontoise, France, 19–22 September 2016, pp. 204–209.

Eiband T, Saveriano M and Lee D (2019) Learning haptic exploration schemes for adaptive task execution. In: IEEE international conference on robotics and automation, Montreal, QC, Canada, pp. 7048–7054.

End F, Akrour R, Peters J, et al. (2017) Layered direct policy search for learning hierarchical skills. In: IEEE international conference on robotics and automation, Singapore, 29 May 2017–03 June 2017, pp. 6442–6448.

Ernesti J, Righetti L, Do M, et al. (2012) Encoding of periodic and their transient motions by a single dynamic movement primitive. In: IEEE-RAS international conference on humanoid robots, Osaka, Japan, pp. 57–64.

Escarabajal RJ, Pulloquinga JL, Zamora-Ortiz P, et al. (2023) Imitation learning-based system for the execution of self-paced robotic-assisted passive rehabilitation exercises. *IEEE Robotics and Automation Letters* 8(7): 4283–4290.

Fabisch A and Metzen J (2014) Active contextual policy search. *Journal of Machine Learning Research* 15: 3371–3399.

Fang Z, Wang G, Li W, et al. (2014) Control-oriented modeling of flight demonstrations for quadrotors using higher-order statistics and dynamic movement primitives. In: IEEE international symposium on industrial electronics, Istanbul, Turkey, 01–04 June 2014, pp. 1518–1525.

Fanger Y, Umlauft J and Hirche S (2016) Gaussian processes for dynamic movement primitives with application in knowledge-based cooperation. In: IEEE/RSJ international conference on intelligent robots and systems, Daejeon, Korea (South), 09–14 October 2016, pp. 3913–3919.

Fei G, Wang S and Liu B (2016) Learning cumulatively to become more knowledgeable. In: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, San Francisco, USA, August 2016, pp. 1565–1574.

Flash T and Hochner B (2005) Motor primitives in vertebrates and invertebrates. *Current Opinion in Neurobiology* 15(6): 660–666.

Forte D, Ude A and Gams A (2011) Real-time generalization and integration of different movement primitives. In: IEEE-RAS international conference on humanoid robots. Bled, Slovenia, pp. 590–595.

Forte D, Gams A, Morimoto J, et al. (2012) On-line motion synthesis and adaptation using a trajectory database. *Robotics and Autonomous Systems* 60(10): 1327–1339.

Forte D, Nemec B and Ude A (2015) Exploration in structured space of robot movements for autonomous augmentation of action knowledge. In: The 17th international conference on advanced robotics, Istanbul, Turkey, 27–31 July 2015, pp. 252–258.

Gams A and Ude A (2009) Generalization of example movements with dynamic systems. In: IEEE-RAS International Conference on Humanoid Robots, Paris, France, pp. 28–33.

Gams A, Ijspeert A, Schaal S, et al. (2009) On-line learning and modulation of periodic movements with nonlinear dynamical systems. *Autonomous Robots* 27(1): 3–23.

Gams A, Do M, Ude A, et al. (2010) On-line periodic movement and force-profile learning for adaptation to new surfaces. In: IEEE-RAS international conference on humanoid robots. Nashville, TN, USA, pp. 560–565.

Gams A, Nemec B, Ijspeert A, et al. (2014) Coupling movement primitives: interaction with the environment and bimanual tasks. *IEEE Transactions on Robotics* 30(4): 816–830.

Gams A, Deniša M and Ude A (2015a) Learning of parametric coupling terms for robot-environment interaction. In: IEEE-RAS international conference on humanoid robots. Seoul, South Korea, pp. 304–309.

Gams A, Ude A and Morimoto J (2015b) Accelerating synchronization of movement primitives: dual-arm discrete-periodic

motion of a humanoid robot. In: IEEE/RSJ international conference on intelligent robots and systems, Hamburg, Germany, 28 September 2015–02 October 2015. pp. 2754–2760.

Gams A, Petrič T, Do M, et al. (2016) Adaptation and coaching of periodic motion primitives through physical and visual interaction. *Robotics and Autonomous Systems* 75: 340–351.

Gao J, Zhou Y and Asfour T (2019) Projected force-admittance control for compliant bimanual tasks. In: IEEE-RAS international conference on humanoid robots, Beijing, China, 06–09 November 2018, pp. 607–613.

Gašpar T, Nemec B, Morimoto J, et al. (2018) Skill learning and action recognition by arc-length dynamic movement primitives. *Robotics and Autonomous Systems* 100: 225–235.

Ghalamzan EA, Paxton C, Hager G, et al. (2015) An incremental approach to learning generalizable robot tasks from human demonstration. In: IEEE international conference on robotics and automation, Seattle, WA, USA, 26–30 May 2015, pp. 5616–5621.

Ginesi M, Meli D, Nakawala H, et al. (2019) A knowledge-based framework for task automation in surgery. In: International conference on advanced robotics, Belo Horizonte, Brazil, 02–06 December 2019, pp. 37–42.

Ginesi M, Meli D, Roberti A, et al. (2021a) Dynamic movement primitives: volumetric obstacle avoidance using dynamic potential functions. *Journal of Intelligent and Robotic Systems* 101: 1–20.

Ginesi M, Sansonetto N and Fiorini P (2021b) Overcoming some drawbacks of dynamic movement primitives. *Robotics and Autonomous Systems* 144.

Guerin K, Riedel S, Bohren J, et al. (2014) Adjutant: a framework for flexible human-machine collaborative systems. In: IEEE/RSJ international conference on intelligent robots and systems, Chicago, IL, USA, 14–18 September 2014, pp. 1392–1399.

Gutzeit L, Fabisch A, Otto M, et al. (2018) The besman learning platform for automated robot skill learning. *Frontiers Robotics AI* 5.

Haddadin S, Weitschat R, Huber F, et al. (2016) *Optimal control for viscoelastic robots and its generalization in real-time*. Springer Tracts in Advanced Robotics, Vol. 114: 131–148.

Hangl S, Ugur E, Szedmak S, et al. (2015) Reactive, task-specific object manipulation by metric reinforcement learning. In: International conference on advanced robotics, Istanbul, Turkey, 27–31 July 2015, pp. 557–564.

Hazara M and Kyrki V (2016) Reinforcement learning for improving imitated in-contact skills. In: IEEE-RAS international conference on humanoid robots, Cancun, Mexico, 15–17 November 2016, pp. 194–201.

Hazara M and Kyrki V (2017) Model selection for incremental learning of generalizable movement primitives. In: *International conference on advanced robotics*, Hong Kong, China, July 2017, pp. 359–366.

Hazara M and Kyrki V (2018) Speeding up incremental learning using data efficient guided exploration. In: *IEEE international conference on robotics and automation*, Brisbane, QLD, Australia, May 2018, pp. 1–8.

Hazara M and Kyrki V (2019) Transferring generalizable motor primitives from simulation to real world. *IEEE Robotics and Automation Letters* 4(2): 2172–2179.

Hazara M, Li X and Kyrki V (2019) Active incremental learning of a contextual skill model. In: IEEE/RSJ international conference on intelligent robots and systems, Macau, China, 03–08 November 2019, pp. 1834–1839.

Herzog S, Wörgötter F and Kulvicius T (2016) Optimal trajectory generation for generalization of discrete movements with boundary conditions. In: IEEE/RSJ international conference on intelligent robots and systems, Daejeon, Korea, October 9–14, 2016, pp. 3143–3149.

Hoffmann H, Pastor P, Park DH, et al. (2009) Biologically-inspired dynamical systems for movement generation: automatic real-time goal adaptation and obstacle avoidance. In: IEEE international conference on robotics and automation. Kobe, Japan, pp. 2587–2592.

Hogan N (1984) Adaptive control of mechanical impedance by coactivation of antagonist muscles. *IEEE Transactions on Automatic Control* 29(8): 681–690.

Hogan N (1985) Impedance control: an approach to manipulation: Part i–theory, part ii–implementation, and part iii–applications. *Journal of Dynamic Systems, Measurement, and Control* 107(1): 1–7.

Hong Z, Bian S, Xiong P, et al. (2023) Vision-locomotion coordination control for a powered lower-limb prosthesis using fuzzy-based dynamic movement primitives. *IEEE Transactions on Automation Science and Engineering* 28: 1–13.

Hotson G, Smith R, Rouse A, et al. (2016) High precision neural decoding of complex movement trajectories using recursive bayesian estimation with dynamic movement primitives. *IEEE Robotics and Automation Letters* 1(2): 676–683.

Hu Y, Wu X, Geng P, et al. (2018) Evolution strategies learning with variable impedance control for grasping under uncertainty. *IEEE Transactions on Industrial Electronics* 66(10): 7788–7799.

Huang R, Cheng H, Guo H, et al (2016b) Learning cooperative primitives with physical human-robot interaction for a human-powered lower exoskeleton. In: IEEE/RSJ international conference on intelligent robots and systems, Daejeon, Korea, 9-14 October, pp. 5355–5360

Huang R, Cheng H, Guo H, et al (2016a) Hierarchical interactive learning for a human-powered augmentation lower exoskeleton. In: IEEE international conference on robotics and automation, Stockholm, Sweden, 16257-21263 May, pp. –.

Huang Y, Rozo L, Silvério J, et al. (2019) Kernelized movement primitives. *The International Journal of Robotics Research* 38(7): 833–852.

Huang Y, Abu-Dakka FJ, Silvério J, et al. (2021) Toward orientation learning and adaptation in cartesian space. *IEEE Transactions on Robotics* 37(1): 82–98.

Hwang S, Lee S, Shin D, et al. (2019) Intuitive gait pattern generation for an exoskeleton robot. *International Journal of Precision Engineering and Manufacturing* 20(11): 1905–1913.

Hwang SH, Sun DI, Han J, et al. (2021) Gait pattern generation algorithm for lower-extremity rehabilitation–exoskeleton robot considering wearer's condition. *Intelligent Service Robotics* 14(3): 345–355.

Ijspeert AJ, Nakanishi J and Schaal S (2001) Trajectory formation for imitation with nonlinear dynamical systems. In: IEEE/RSJ

international conference on intelligent robots and systems. Maui, HI, USA, pp. 752–757.

Ijspeert A, Nakanishi J and Schaal S (2002a) Learning attractor landscapes for learning motor primitives. In: Advances in neural information processing systems 15. Vancouver, BC, Canada: Cambridge, MA: MIT Press, pp. 1523–1530.

Ijspeert AJ, Nakanishi J and Schaal S (2002b) Learning rhythmic movements by demonstration using nonlinear oscillators. In: IEEE/RSJ international conference on intelligent robots and systems, Volume 1. Lausanne, Switzerland, pp. 958–963.

Ijspeert AJ, Nakanishi J, Hoffmann H, et al. (2013) Dynamical movement primitives: learning attractor models for motor behaviors. *Neural Computation* 25(2): 328–373.

Ijspeert AJ, Nakanishi J and Schaal S (2002c) Movement imitation with nonlinear dynamical systems in humanoid robots. IEEE International Conference on Robotics and Automation, Washington D.C., USA, 11-15 May, pp.1398–1403.

Iori F, Perovic G, Cini F, et al. (2023) DMP-based reactive robot-to-human handover in perturbed scenarios. *International Journal of Social Robotics* 15(2): 233–248.

Iturrate I, Sloth C, Kramberger A, et al. (2019) Towards reversible dynamic movement primitives. In: IEEE/RSJ international conference on intelligent robots and systems. Macau, China: IEEE, pp. 5063–5070.

Jaques M, Burke M and Hospedales T (2021) Newtonianvae: proportional control and goal identification from pixels via physical latent spaces. In: IEEE computer society conference on computer vision and pattern recognition, Nashville, TN, USA, 20–25 June 2021, pp. 4452–4461.

Joshi RP, Koganti N and Shibata T (2017) Robotic cloth manipulation for clothing assistance task using dynamic movement primitives. In: *Proceedings of the advances in robotics*, New Delhi, India, June 2017, pp. 1–6.

Joshi R, Koganti N and Shibata T (2019) A framework for robotic clothing assistance by imitation learning. *Advanced Robotics* 33(22): 1156–1174.

Kamali K, Akbari AA and Akbarzadeh A (2016) Trajectory generation and control of a knee exoskeleton based on dynamic movement primitives for sit-to-stand assistance. *Advanced Robotics* 30(13): 846–860.

Karlsson M, Carlson FB, Robertsson A, et al. (2017a) Two-degree-of-freedom control for trajectory tracking and perturbation recovery during execution of dynamical movement primitives. *IFAC-World Congress* 50(1): 1923–1930.

Karlsson M, Robertsson A and Johansson R (2017b) Autonomous interpretation of demonstrations for modification of dynamical movement primitives. In: IEEE international conference on robotics and automation, Singapore, 29 May 2017–03 June 2017, pp. 316–321.

Karlsson M, Robertsson A and Johansson R (2018) Convergence of dynamical movement primitives with temporal coupling. In: European control conference, Limassol, Cyprus, 12–15 June 2018. IEEE, pp. 32–39.

Kastritsi T, Dimeas F and Doulgeri Z (2018) Progressive automation with DMP synchronization and variable stiffness control. *IEEE Robotics and Automation Letters* 3(4): 3789–3796.

Khansari-Zadeh SM and Billard A (2011) Learning stable non-linear dynamical systems with Gaussian mixture models. *Transactions on Robotics* 27(5): 943–957.

Khansari-Zadeh SM and Billard A (2014) Learning control Lyapunov function to ensure stability of dynamical system-based robot reaching motions. *Robotics and Autonomous Systems* 62(6): 752–765.

Kim JJ, Park SY and Lee JJ (2015) Adaptability improvement of learning from demonstration with sequential quadratic programming for motion planning. In: IEEE/ASME international conference on advanced intelligent mechatronics, Busan, Korea (South), 07–11 July 2015, pp. 1032–1037.

Kim H, Seo H, Son C, et al. (2018a) Cooperation in the air: a learning-based approach for the efficient motion planning of aerial manipulators. *IEEE Robotics and Automation Magazine* 25(4): 76–85.

Kim W, Lee C and Kim H (2018b) Learning and generalization of dynamic movement primitives by hierarchical deep reinforcement learning from demonstration. In: IEEE/RSJ international conference on intelligent robots and systems, Madrid, Spain, 01–05 October 2018, pp. 3117–3123.

Kober J and Peters J (2010) Imitation and reinforcement learning. *IEEE Robotics and Automation Magazine* 17(2): 55–62.

Kober J and Peters J (2011) Policy search for motor primitives in robotics. *Machine Learning* 84(1-2): 171–203.

Kober J, Mohler B and Peters J (2008) Learning perceptual coupling for motor primitives. In: IEEE/RSJ international conference on intelligent robots and systems, Nice, France, 22–26 September 2008, pp. 834–839.

Kober J, Mohler B and Peters J (2010a) Imitation and reinforcement learning for motor primitives with perceptual coupling. In: O Sigaud and J Peters (eds.) From motor learning to interaction learning in robots. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 209–225.

Kober J, Mülling K, Krömer O, et al. (2010b) Movement templates for learning of hitting and batting. In: IEEE international conference on robotics and automation. Anchorage, AK, USA: IEEE, pp. 853–858.

Kober J, Oztop E and Peters J (2011) Reinforcement learning to adjust robot movements to new situations. In: International joint conference on artificial intelligence, Barcelona, Spain, July 2011, pp. 2650–2655.

Kober J, Wilhelm A, Oztop E, et al. (2012) Reinforcement learning to adjust parametrized motor primitives to new situations. *Autonomous Robots* 33(4): 361–379.

Kober J, Bagnell JA and Peters J (2013) Reinforcement learning in robotics: a survey. *The International Journal of Robotics Research* 32(11): 1238–1274.

Koene A, Endo S, Remazeilles A, et al. (2014) Experimental testing of the coglaboration prototype system for fluent human-robot object handover interactions. In: IEEE international symposium on robot and human interactive communication, Edinburgh, UK, 25–29 August 2014, pp. 249–254.

Kordia AH and Melo FS (2021) Movement recognition and prediction using DMPS. In: IEEE international conference on robotics and automation, Xi'an, China, 30 May 2021–05 June 2021, pp. 8544–8550.

Kormushev P, Calinon S and Caldwell DG (2010) Robot motor skill coordination with em-based reinforcement learning. In: IEEE/RSJ international conference on intelligent robots and systems, Taipei, Taiwan, 18–22 October 2010. pp. 3232–3237.

Kormushev P, Calinon S and Caldwell DG (2011) Imitation learning of positional and force skills demonstrated via kinesthetic teaching and haptic input. *Advanced Robotics* 25(5): 581–603.

Koropouli V, Hirche S and Lee D (2015) Generalization of force control policies from demonstrations for constrained robotic motion tasks. *Journal of Intelligent and Robotic Systems* 80(1): 133–148.

Koutras L and Doulgeri Z (2020a) A correct formulation for the orientation dynamic movement primitives for robot control in the cartesian space. In: Conference on robot learning, PMLR, virtual, 16-18 November, pp. 293–302.

Koutras L and Doulgeri Z (2020b) Dynamic movement primitives for moving goals with temporal scaling adaptation. In: IEEE international conference on robotics and automation. Virtual, 31 May - 31 August, pp. 144–150.

Kramberger A, Gams A, Nemec B, et al. (2016a) Generalization of orientational motion in unit quaternion space. In: IEEE-RAS international conference on humanoid robots. Cancun, Mexico, pp. 808–813.

Kramberger A, Piltaver R, Nemec B et al. (2016b) Learning of assembly constraints by demonstration and active exploration. *Industrial Robot: An International Journal* 43: 524–534.

Kramberger A, Gams A, Nemec B, et al. (2017) Generalization of orientation trajectories and force-torque profiles for robotic assembly. *Robotics and Autonomous Systems* 98: 333–346.

Kramberger A, Shahriari E, Gams A, et al. (2018) Passivity based iterative learning of admittance-coupled dynamic movement primitives for interaction with changing environments. In: IEEE/RSJ international conference on intelligent robots and systems. Madrid, Spain: IEEE, pp. 6023–6028.

Krömer O, Detry R, Piater J, et al. (2010a) Grasping with vision descriptors and motor primitives. In: *International conference on informatics in control, automation and robotics*, Madeira, Portugal, June 2010, Vol 2, pp. 47–54.

Krömer OB, Detry R, Piater J, et al. (2010b) Combining active learning and reactive control for robot grasping. *Robotics and Autonomous Systems* 58(9): 1105–1116.

Kronander K and Billard A (2015) Passive interaction control with dynamical systems. *IEEE Robotics and Automation Letters* 1(1): 106–113.

Krug R and Dimitrov D (2015) Model predictive motion control based on generalized dynamical movement primitives. *Journal of Intelligent and Robotic Systems* 77(1): 17–35.

Krug R and Dimitrovz D (2013) Representing movement primitives as implicit dynamical systems learned from multiple demonstrations. In: IEEE international conference on advanced robotics, Montevideo, Uruguay, 25–29 November 2013. pp. 1–8.

Krüger N, Ude A, Petersen H, et al. (2014) Technologies for the fast set-up of automated assembly processes. *Kunstliche Intelligenz* 28(4): 305–313.

Kulvicius T, Ning K, Tamosiunaite M, et al. (2011) Modified dynamic movement primitives for joining movement sequences. In: IEEE international conference on robotics and automation, Shanghai, China, 09–13 May 2011. pp. 2275–2280.

Kulvicius T, Ning K, Tamosiunaite M, et al. (2012) Joining movement sequences: modified dynamic movement primitives for robotics applications exemplified on handwriting. *IEEE Transactions on Robotics* 28(1): 145–157.

Kulvicius T, Biehl M, Aein MJ, et al. (2013) Interaction learning for dynamic movement primitives used in cooperative robotic tasks. *Robotics and Autonomous Systems* 61(12): 1450–1459.

Kupcsik A, Deisenroth M, Peters J, et al. (2017) Model-based contextual policy search for data-efficient generalization of robot skills. *Artificial Intelligence* 247: 415–439.

Kuppuswamy N and Alessandro C (2011) Impact of body parameters on dynamic movement primitives for robot control. *Procedia Computer Science* 7: 166–168. 2nd European Future Technologies Conference and Exhibition 2011 (FET 11).

Lafleche JF, Saunderson S and Nejat G (2019) Robot cooperative behavior learning using single-shot learning from demonstration and parallel hidden markov models. *IEEE Robotics and Automation Letters* 4(2): 193–200.

Lanotte F, McKinney Z, Grazi L, et al. (2021) Adaptive control method for dynamic synchronization of wearable robotic assistance to discrete movements: validation for use case of lifting tasks. *IEEE Transactions on Robotics* 37(6): 2193–2209.

Lantz V and Murray-Smith R (2004) Rhythmic interaction with a mobile device. In: *Nordic conference on human–computer interaction*, Vol 82, Tampere, Finland, October 2004, pp. 97–100.

Lauretti C, Cordella F, Guglielmelli E, et al. (2017) Learning by demonstration for planning activities of daily living in rehabilitation and assistive robotics. *IEEE Robotics and Automation Letters* 2(3): 1375–1382.

Lauretti C, Cordella F, Ciancio AL, et al. (2018) Learning by demonstration for motion planning of upper-limb exoskeletons. *Frontiers in Neurorobotics* 12: 5.

LeCun Y, Bengio Y and Hinton G (2015) Deep learning. *Nature* 521(7553): 436–444.

Lee S and Suh I (2013) Skill learning and inference framework for skilligent robot. In: IEEE/RSJ international conference on intelligent robots and systems, Tokyo, Japan, November, pp. 108–115.

Lee H, Kim H, Kim W, et al. (2018) An integrated framework for cooperative aerial manipulators in unknown environments. *IEEE Robotics and Automation Letters* 3(3): 2307–2314.

Lee H, Seo H and Kim HG (2020) Trajectory optimization and replanning framework for a micro air vehicle in cluttered environments. *IEEE Access* 8: 135406–135415.

Lemme A, Reinhart R and Steil J (2014) Self-supervised bootstrapping of a movement primitive library from complex trajectories. In: IEEE-RAS international conference on humanoid robots, Madrid, Spain, November, pp. 726–732.

Lentini G, Grioli G, Catalano M, et al. (2020) Robot programming without coding. In: IEEE international conference on robotics and automation, Paris, France, 31 May 2020–31 August 2020, pp. 7576–7582.

Li W and Fritz M (2015) Teaching robots the use of human tools from demonstration with non-dexterous end-effectors. In:

IEEE-RAS international conference on humanoid robots, Seoul, Korea (South), 03–05 November 2015, pp. 547–553.

Li J, Wang J, Wang S, et al. (2021b) Human–robot skill transmission for mobile robot via learning by demonstration. *Neural Computing and Applications*, September, pp. 1-11.

Li Z, Zhao T, Chen F, et al. (2018) Reinforcement learning of manipulation and grasping using dynamical movement primitives for a humanoidlike mobile manipulator. *IEEE/ASME Transactions on Mechatronics* 23(1): 121–131.

Li C, Fahmy A, Li S, et al. (2020) An enhanced robot massage system in smart homes using force sensing and a dynamic movement primitive. *Frontiers in Neurorobotics* 14.

Li J, Li Z, Li X, et al. (2021a) Skill learning strategy based on dynamic motion primitives for human-robot cooperative manipulation. *IEEE Transactions on Cognitive and Developmental Systems* 13(1): 105–117.

Li G, Jin Z, Volpp M, et al. (2023) ProDMP: a unified perspective on dynamic and probabilistic movement primitives. *IEEE Robotics and Automation Letters* 8(4): 2325–2332.

Liang Y, Li W, Wang Y, et al. (2021) Dynamic movement primitive based motion retargeting for dual-arm sign language motions. In: IEEE international conference on robotics and automation, Xi'an, China, 30 May 2021–05 June 2021, Vol 2021-May, pp. 8195–8201.

Lioutikov R, Kroemer O, Maeda G, et al. (2016) Learning manipulation by sequencing motor primitives with a two-armed robot. In: E Menegatti, N Michael, K Berns, et al. (eds) *Intelligent Autonomous Systems 13*. Cham: Springer International Publishing, 1601–1611.

Liu Z, Hu F, Luo D, et al. (2014) Visual gesture recognition for human robot interaction using dynamic movement primitives. In: IEEE international conference on systems, man and cybernetics, San Diego, CA, USA, 05–08 October 2014, pp. 2094–2100.

Liu C, Geng W, Liu M, et al. (2020) Workspace trajectory generation method for humanoid adaptive walking with dynamic motion primitives. *IEEE Access* 8: 54652–54662.

Lončarević Z, Pahič R, Ude A, et al. (2021) Generalization-based acquisition of training data for motor primitive learning by neural networks. *Applied Sciences* 11(3).

Lu Z, Wang N and Yang C (2021) A constrained DMPS framework for robot skills learning and generalization from human demonstrations. *IEEE/ASME Transactions on Mechatronics* 26(6): 3265–3275.

Lu Z, Wang N and Shi D (2022) DMPS-based skill learning for redundant dual-arm robotic synchronized cooperative manipulation. *Complex and Intelligent Systems* 8(4): 2873–2882.

Lundell J, Hazara M and Kyrki V (2017) Generalizing movement primitives to new situations. In: Y Gao, S Fallah, Y Jin, et al. (eds) *Towards Autonomous Robotic Systems*. Cham: Springer International Publishing, 16–31.

Luo D, Han X, Ding Y, et al. (2015) Learning push recovery for a bipedal humanoid robot with dynamical movement primitives. In: IEEE-RAS international conference on humanoid robots, Seoul, Korea (South), 03–05 November 2015, pp. 1013–1019.

Luo L, Foo MJ, Ramanathan M, et al. (2022) Trajectory generation and control of a lower limb exoskeleton for gait assistance. *Journal of Intelligent and Robotic Systems* 106(3): 64.

Luo J, Liu W, Qi W, et al. (2023) A vision-based virtual fixture with robot learning for teleoperation. *Robotics and Autonomous Systems* 164.

Maeda G (2022) Blending primitive policies in shared control for assisted teleoperation. In: IEEE international conference on robotics and automation, Philadelphia, PA, USA, 23–27 May 2022, pp. 9332–9338.

Mandery C, Borrás J, Jöchner M, et al. (2016) Using language models to generate whole-body multi-contact motions. In: IEEE/RSJ international conference on intelligent robots and systems, Daejeon, Korea (South), 09–14 October 2016, pp. 5411–5418.

Mao R, Yang Y, Fermüller C, et al. (2015) Learning hand movements from markerless demonstrations for humanoid tasks. In: IEEE-RAS international conference on humanoid robots, Madrid, Spain, 18–20 November 2014, pp. 938–943.

Matsubara T, Hyon SH and Morimoto J (2010) Learning stylistic dynamic movement primitives from multiple demonstrations. In: IEEE/RSJ international conference on intelligent robots and systems, Taipei, Taiwan, 18–22 October 2010, pp. 1277–1283.

Matsubara T, Hyon SH and Morimoto J (2011) Learning parametric dynamic movement primitives from multiple demonstrations. *Neural Networks* 24(5): 493–500.

Mavsar M, Ridge B, Pahic R, et al. (2022) Simulation-aided handover prediction from video using recurrent image-to-motion networks. *IEEE Transactions on Neural Networks and Learning Systems*: 1–13.

Meier F, Theodorou E, Stulp F, et al. (2011) Movement segmentation using a primitive library. In: IEEE/RSJ international conference on intelligent robots and systems, San Francisco, CA, USA, 25–30 September 2011, pp. 3407–3412.

Montebelli A, Steinmetz F and Kyrki V (2015) On handing down our tools to robots: single-phase kinesthetic teaching for dynamic in-contact tasks. In: IEEE international conference on robotics and automation, Seattle, WA, USA, 26–30 May 2015, pp. 5628–5634.

Mülling K, Kober J and Peters J (2010) Learning table tennis with a mixture of motor primitives. In: IEEE-RAS international conference on humanoid robots. Nashville, TN, USA, pp. 411–416.

Mülling K, Kober J, Krömer O, et al. (2013) Learning to select and generalize striking movements in robot table tennis. *The International Journal of Robotics Research* 32(3): 263–279.

Mussa-Ivaldi FA (1999) Modular features of motor control and learning. *Current Opinion in Neurobiology* 9(6): 713–717.

Nah M, Krotov A, Russo M, et al. (2020) Dynamic primitives facilitate manipulating a whip. In: IEEE RAS and EMBS international conference on biomedical robotics and biomechatronics, New York, NY, USA, 29 November 2020–01 December 2020, pp. 685–691.

Nakanishi J, Morimoto J, Endo G, et al. (2004) Learning from demonstration and adaptation of biped locomotion. *Robotics and Autonomous Systems* 47(2-3): 79–91.

Nakanishi J, Rawlik K and Vijayakumar S (2011) Stiffness and temporal optimization in periodic movements: an optimal control approach. In: IEEE/RSJ international conference on intelligent robots and systems, San Francisco, CA, USA, September, pp. 718–724.

Nemec B and Ude A (2012) Action sequencing using dynamic movement primitives. *Robotica* 30(5): 837.

Nemec B, Vuga R and Ude A (2011) Exploiting previous experience to constrain robot sensorimotor learning. In: IEEE--RAS international conference on humanoid robots. Bled, Slovenia, pp. 727–732.

Nemec B, Forte D, Vuga R, et al. (2012) Applying statistical generalization to determine search direction for reinforcement learning of movement primitives. In: IEEE-RAS international conference on humanoid robots, Osaka, Japan, 29 November 2012–01 December 2012. pp. 65–70.

Nemec B, Gams A and Ude A (2013a) Velocity adaptation for self-improvement of skills learned from user demonstrations. In: IEEE-RAS international conference on humanoid robots, Atlanta, GA, USA, pp. 423–428.

Nemec B, Vuga R and Ude A (2013b) Efficient sensorimotor learning from multiple demonstrations. *Advanced Robotics* 27(13): 1023–1031.

Nemec B, Likar N, Gams A, et al. (2016) Bimanual human robot cooperation with adaptive stiffness control. In: IEEE-RAS international conference on humanoid robots, Cancun, Mexico, 15–17 November 2016, pp. 607–613.

Nemec B, Žlajpah L, Šlajpa S, et al. (2018) An efficient pbd framework for fast deployment of bi-manual assembly tasks. In: IEEE-RAS international conference on humanoid robots, Beijing, China, pp. 166–173.

Nemec B, Simonic M and Ude A (2020) Learning of exception strategies in assembly tasks. In: IEEE international conference on robotics and automation, Paris, France, 31 May 2020–31 August 2020, pp. 6521–6527.

Neumann K and Steil JJ (2015) Learning robot motions with stable dynamical systems under diffeomorphic transformations. *Robotics and Autonomous Systems* 70: 1–15.

Niekum S, Osentoski S, Konidaris G, et al. (2012) Learning and generalization of complex tasks from unstructured demonstrations. In: IEEE/RSJ international conference on intelligent robots and systems, Vilamoura-Algarve, Portugal, 07–12 October 2012, pp. 5239–5246.

Niekum S, Osentoski S, Konidaris G, et al. (2015) Learning grounded finite-state representations from unstructured demonstrations. *The International Journal of Robotics Research* 34(2): 131–157.

Ning K, Kulvicius T, Tamosiunaite M, et al. (2011) Accurate position and velocity control for trajectories based on dynamic movement primitives. In: IEEE international conference on robotics and automation, Shanghai, China, 09–13 May 2011, pp. 5006–5011.

Ning K, Kulvicius T, Tamosiunaite M, et al. (2012) A novel trajectory generation method for robot control. *Journal of Intelligent and Robotic Systems* 68(2): 165–184.

Norrlöf M (1991) An adaptive iterative learning control algorithm with experiments on an industrial robot. *IEEE Transaction on Robotics and Automation* 18(2): 188–197.

Oikonomidis I, Kyriazis N and Argyros A (2011) Efficient model-based 3d tracking of hand articulations using kinect. In: *British machine vision conference*, Dundee, UK, August - September 2011, pp. 101.1–101.11.

Ojer De Andres M, Mahdi Ghazaei Ardakani M and Robertsson A (2018) Reinforcement learning for 4-finger-gripper manipulation. In: IEEE international conference on robotics and automation, Brisbane, QLD, Australia, 21–25 May 2018, pp. 4257–4262.

Pahic R, Gams A, Ude A, et al. (2018) Deep encoder-decoder networks for mapping raw images to dynamic movement primitives. In: IEEE international conference on robotics and automation, Brisbane, QLD, Australia, 21–25 May 2018, pp. 5863–5868.

Pahič R, Ridge B, Gams A, et al. (2020) Training of deep neural networks for the generation of dynamic movement primitives. *Neural Networks* 127: 121–131.

Pahic R, Gams A and Ude A (2021) Reconstructing spatial aspects of motion by image-to-path deep neural networks. *IEEE Robotics and Automation Letters* 6(1): 255–262.

Pan Z and Manocha D (2018) Realtime planning for high-dof deformable bodies using two-stage learning. In: IEEE international conference on robotics and automation, Brisbane, QLD, Australia, 21–25 May 2018, pp. 5582–5589.

Papageorgiou D, Dimeas F, Kastritsi T, et al. (2020a) Kinesthetic guidance utilizing DMP synchronization and assistive virtual fixtures for progressive automation. *Robotica* 38(10): 1824–1841.

Papageorgiou D, Kastritsi T and Doulgeri Z (2020b) A passive robot controller aiding human coaching for kinematic behavior modifications. *Robotics and Computer-Integrated Manufacturing* 61: 101824.

Paraschos A, Daniel C, Peters J, et al. (2013) Probabilistic movement primitives. In: C Burges, L Bottou, M Welling, et al. (eds) *Advances in Neural Information Processing Systems 26*. Lake Tahoe, Nevada, US: Curran Associates, Inc., pp. 2616–2624.

Park DH, Hoffmann H, Pastor P, et al. (2008) Movement reproduction and obstacle avoidance with dynamic movement primitives and potential fields. In: IEEE-RAS international conference on humanoid robots. Daejeon, South Korea, December, pp. 91–98.

Pastor P, Hoffmann H, Asfour T, et al. (2009) Learning and generalization of motor skills by learning from demonstration. In: IEEE international conference on robotics and automation. Kobe, Japan, pp. 763–768.

Pastor P, Kalakrishnan M, Chitta S, et al. (2011) Skill learning and task outcome prediction for manipulation. In: IEEE international conference on robotics and automation. Shanghai, China: IEEE, pp. 3828–3834.

Pastor P, Righetti L, Kalakrishnan M, et al. (2011a) Online movement adaptation based on previous sensor experiences. In: IEEE/RSJ international conference on intelligent robots and systems, San Francisco, CA, USA, 25–30 September 2011, pp. 365–371.

Pastor P, Kalakrishnan M, Righetti L, et al. (2012) Towards associative skill memories. In: IEEE-RAS international conference on humanoid robots, Osaka, Japan, 29 November 2012–01 December 2012, pp. 309–315.

Pastor P, Kalakrishnan M, Meier F, et al. (2013) From dynamic movement primitives to associative skill memories. *Robotics and Autonomous Systems* 61(4): 351–361.

Paxton C, Jonathan F, Kobilarov M, et al. (2016) Do what i want, not what i did: imitation of skills by planning sequences of actions. In: IEEE/RSJ international conference on intelligent robots and systems, Daejeon, Korea (South), 09–14 October 2016, pp. 3778–3785.

Perk BE and Slotine JJE (2006) Motion primitives for robotic flight control. *arXiv preprint cs/0609140*.

Perrin N and Schlehuber-Caissier P (2016) Fast diffeomorphic matching to learn globally asymptotically stable nonlinear dynamical systems. *Systems & Control Letters* 96: 51–59.

Pervez A and Lee D (2018) Learning task-parameterized dynamic movement primitives using mixture of gmms. *Intelligent Service Robotics* 11(1): 61–78.

Pervez A, Ali A, Ryu JH, et al. (2017a) Novel learning from demonstration approach for repetitive teleoperation tasks. In: IEEE world haptics conference, Munich, Germany, 06–09 June 2017, pp. 60–65.

Pervez A, Mao Y and Lee D (2017b) Learning deep movement primitives using convolutional neural networks. In: IEEE--RAS international conference on humanoid robots, Birmingham, UK, 15–17 November 2017, pp. 191–197.

Pervez A, Latifee H, Ryu JH, et al. (2019) Motion encoding with asynchronous trajectories of repetitive teleoperation tasks and its extension to human-agent shared teleoperation. *Autonomous Robots* 43(8): 2055–2069.

Peternel L and Ajoudani A (2017) Robots learning from robots: a proof of concept study for co-manipulation tasks. In: IEEE-RAS international conference on humanoid robots. Birmingham, UK: IEEE, pp. 484–490.

Peternel L and Ajoudani A (2022) After a decade of teleimpedance: a survey. *IEEE Transactions on Human-Machine Systems* 53(2): 401–416.

Peternel L, Petrič T, Oztop E, et al. (2014) 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.

Peternel L, Noda T, Petrič T, et al. (2016) Adaptive control of exoskeleton robots for periodic assistive behaviours based on emg feedback minimisation. *PLoS One* 11(2): e0148942.

Peternel L, Rozo L, Caldwell D, et al. (2017a) A method for derivation of robot task-frame control authority from repeated sensory observations. *IEEE Robotics and Automation Letters* 2(2): 719–726.

Peternel L, Tsagarakis N and Ajoudani A (2017b) A human–robot co-manipulation approach based on human sensorimotor information. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 25(7): 811–822.

Peternel L, Petrič T and Babič J (2018a) Robotic assembly solution by human-in-the-loop teaching method based on real-time stiffness modulation. *Autonomous Robots* 42(1): 1–17.

Peternel L, Tsagarakis N, Caldwell D, et al. (2018b) Robot adaptation to human physical fatigue in human–robot co-manipulation. *Autonomous Robots* 42(5): 1011–1021.

Peters J and Schaal S (2008a) Policy learning for motor skills. *International Conference on Neural Information Processing*. Berlin Heidelberg: Springer, 233–242.

Peters J and Schaal S (2008b) Reinforcement learning of motor skills with policy gradients. *Neural Networks* 21(4): 682–697.

Petrič T, Gams A, Ijspeert AJ, et al. (2011) On-line frequency adaptation and movement imitation for rhythmic robotic tasks. *The International Journal of Robotics Research* 30(14): 1775–1788.

Petrič T, Gams A, Žlajpah L, et al. (2014a) Online learning of task-specific dynamics for periodic tasks. In: IEEE/RSJ international conference on intelligent robots and systems, Chicago, IL, USA, pp. 1790–1795.

Petrič T, Gams A, Žlajpah L, et al. (2014b) Online approach for altering robot behaviors based on human in the loop coaching gestures. In: IEEE international conference on robotics and automation. Hong Kong, China, pp. 4770–4776.

Petric T, Colasanto L, Gams A, et al. (2015) Bio-inspired learning and database expansion of compliant movement primitives. In: IEEE-RAS international conference on humanoid robots, Seoul, Korea (South), 03–05 November 2015, pp. 346–351.

Petrič T, Goljat R and Babič J (2016) Cooperative human-robot control based on fitts' law. In: IEEE-RAS international conference on humanoid robots, Cancun, Mexico, 15–17 November 2016, pp. 345–350.

Petric T, Gams A, Colasanto L, et al. (2018) Accelerated sensorimotor learning of compliant movement primitives. *IEEE Transactions on Robotics* 34(6): 1636–1642.

Pfeiffer S and Angulo C (2015) Gesture learning and execution in a humanoid robot via dynamic movement primitives. *Pattern Recognition Letters* 67: 100–107.

Prada M, Remazeilles A, Koene A, et al. (2013) Dynamic movement primitives for human-robot interaction: comparison with human behavioral observation. In: IEEE/RSJ international conference on intelligent robots and systems, Tokyo, Japan, pp. 1168–1175.

Prada M, Remazeilles A, Koene A, et al. (2014) Implementation and experimental validation of dynamic movement primitives for object handover. In: IEEE/RSJ international conference on intelligent robots and systems, Chicago, IL, USA, 14–18 September 2014, pp. 2146–2153.

Prakash R, Behera L, Mohan S, et al. (2020) Dynamic trajectory generation and a robust controller to intercept a moving ball in a game setting. *IEEE Transactions on Control Systems Technology* 28(4): 1418–1432.

Queißer J and Steil J (2018) Bootstrapping of parameterized skills through hybrid optimization in task and policy spaces. *Frontiers Robotics AI* 5(JUN).

Queißer J, Reinhart R and Steil J (2016) Incremental bootstrapping of parameterized motor skills. In: IEEE-RAS international conference on humanoid robots, Cancun, Mexico, 15–17 November 2016, pp. 223–229.

Rai A, Sutanto G, Schaal S, et al. (2017) Learning feedback terms for reactive planning and control. In: IEEE international conference on robotics and automation, Singapore, 29 May 2017–03 June 2017, pp. 2184–2191.

Ramirez-Amaro K, Beetz M and Cheng G (2015) Understanding the intention of human activities through semantic perception: observation, understanding and execution on a humanoid robot. *Advanced Robotics* 29(5): 345–362.

Rasmussen CE and Williams CKI (2006) *Gaussian Processes for Machine Learning*. Cambridge, Massachusetts: The MIT Press.

Ravichandar H and Dani A (2015) Learning contracting nonlinear dynamics from human demonstration for robot motion planning. In: ASME dynamic systems and control conference, Columbus, Ohio, USA, October 28–30, 2015.

Reinhart R and Steil J (2014) Efficient policy search with a parameterized skill memory. In: IEEE/RSJ international conference on intelligent robots and systems, Chicago, IL, USA, 06 November 2014, pp. 1400–1407.

Reinhart R and Steil J (2015) Efficient policy search in low-dimensional embedding spaces by generalizing motion primitives with a parameterized skill memory. *Autonomous Robots* 38(4): 331–348.

Rouse NA and Daltorio KA (2021) Visualization of stable heteroclinic channel-based movement primitives. *IEEE Robotics and Automation Letters* 6(2): 2343–2348.

Rückert E and d'Avella A (2013) Learned parametrized dynamic movement primitives with shared synergies for controlling robotic and musculoskeletal systems. *Frontiers in Computational Neuroscience* 7.

Salvador S and Chan P (2007) Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis* 11(5): 561–580.

Samant R, Behera L and Pandey G (2016) Adaptive learning of dynamic movement primitives through demonstration. In: International joint conference on neural networks, Vancouver, BC, Canada, 24–29 July 2016, pp. 1068–1075.

Saveriano M, Franzel F and Lee D (2019) Merging position and orientation motion primitives. In: IEEE international conference on robotics and automation, Montreal, QC, Canada, pp. 7041–7047.

Schaal S (1999) Is imitation learning the route to humanoid robots? *Trends in Cognitive Sciences* 3(6): 233–242.

Schaal S (2006a) Dynamic movement primitives -a framework for motor control in humans and humanoid robotics. In: H Kimura, K Tsuchiya, A Ishiguro, et al. (eds.) *Adaptive Motion of Animals and Machines*. Tokyo: Springer Tokyo, 261–280.

Schaal S (2006b) *Dynamic Systems: Brain, Body, and Imitation*, Chapter 1. Cambridge University Press, 177–214.

Schaal S and Atkeson CG (1998) Constructive incremental learning from only local information. *Neural Computation* 10(8): 2047–2084.

Schaal S, Mohajerian P and Ijspeert A (2007) Dynamics systems vs. optimal control–a unifying view. *Progress in Brain Research* 165: 425–445.

Schindlbeck C and Haddadin S (2015) Unified passivity-based cartesian force/impedance control for rigid and flexible joint robots via task-energy tanks. In: IEEE international conference on robotics and automation, Seattle, WA, USA, pp. 440–447.

Schroecker Y, Amor H and Thomaz A (2016) Directing policy search with interactively taught via-points. In: Proceedings of the international joint conference on autonomous agents and multiagent systems, Singapore, May. pp. 1052–1059.

Schwaner KL, Dall'alba D, Jensen PT, et al. (2021) Autonomous needle manipulation for robotic surgical suturing based on skills learned from demonstration. In: IEEE international conference on automation science and engineering, Lyon, France, 23–27 August 2021, pp. 235–241.

Seleem IA, El-Hussieny H and Ishii H (2023) Imitation-based motion planning and control of a multi-section continuum robot interacting with the environment. *IEEE Robotics and Automation Letters* 8(3): 1351–1358.

Shahriari E, Kramberger A, Gams A, et al. (2017) Adapting to contacts: energy tanks and task energy for passivity-based dynamic movement primitives. In: IEEE-RAS international conference on humanoid robots, Birmingham, UK, November, pp. 136–142.

Si W, Wang N and Yang C (2021) Composite dynamic movement primitives based on neural networks for human–robot skill transfer. *Neural Computing and Applications* 1–11.

Sidiropoulos A, Karayiannidis Y and Doulgeri Z (2019) Human-robot collaborative object transfer using human motion prediction based on dynamic movement primitives. In: European control conference. Naples, Italy, pp. 2583–2588.

Sidiropoulos A, Karayiannidis Y and Doulgeri Z (2021) Human-robot collaborative object transfer using human motion prediction based on cartesian pose dynamic movement primitives. In: IEEE international conference on robotics and automation, Xi'an, China, 30 May 2021–05 June 2021, pp. 3758–3764.

Silvério J, Huang Y, Abu-Dakka FJ, et al. (2019) Uncertainty-aware imitation learning using kernelized movement primitives. In: IEEE/RSJ international conference on intelligent robots and systems, Macau, China, November, pp. 90–97.

Simonič M, Petrič T, Ude A, et al. (2021) Analysis of methods for incremental policy refinement by kinesthetic guidance. *Journal of Intelligent and Robotic Systems: Theory and Applications* 102(1).

Sloth C, Kramberger A and Iturrate I (2020) Towards easy setup of robotic assembly tasks. *Advanced Robotics* 34(7–8): 499–513.

Solak G and Jamone L (2019) Learning by demonstration and robust control of dexterous in-hand robotic manipulation skills. In: IEEE/RSJ international conference on intelligent robots and systems, Macau, China, 03–08 November 2019, pp. 8246–8251.

Song C, Liu G, Zhang X, et al. (2020) Robot complex motion learning based on unsupervised trajectory segmentation and movement primitives. *ISA Transactions* 97: 325–335.

Spector O and Zacksenhouse M (2021) Learning contact-rich assembly skills using residual admittance policy. In: IEEE international conference on intelligent robots and systems, Prague, Czech Republic, 27 September 2021–01 October 2021, pp. 6023–6030.

Stein S, Wörgötter F, Schoeler M, et al. (2014) Convexity based object partitioning for robot applications. In: IEEE international conference on robotics and automation, Hong Kong, China, 31 May 2014–07 June 2014, pp. 3213–3220.

Steinmetz F, Montebelli A and Kyrki V (2015) Simultaneous kinesthetic teaching of positional and force requirements for sequential in-contact tasks. In: IEEE-RAS international conference on humanoid robots, Seoul, Korea (South), 03–05 November 2015, pp. 202–209.

Strachan S, Murray-Smith R, Oakley I, et al. (2004) Dynamic primitives for gestural interaction. In: S Brewster and M

Dunlop (eds) *Mobile Human–Computer Interaction*. Springer Berlin Heidelberg, pp. 325–330.

Straizys A, Burke M and Ramamoorthy S (2020) Surfing on an uncertain edge: precision cutting of soft tissue using torque-based medium classification. In: IEEE international conference on robotics and automation, Paris, France, 31 May 2020–31 August 2020, pp. 4623–4629.

Stramigioli S (2001) *Modeling and IPC Control of Interactive Mechanical Systems – A Coordinate-Free Approach, Lecture Notes in Control and Information Sciences*. Springer-Verlag London, Vol. 266.

Stulp F and Schaal S (2011) Hierarchical reinforcement learning with movement primitives. In: IEEE-RAS international conference on humanoid robots. Bled, Slovenia, pp. 231–238.

Stulp F, Oztop E, Pastor P, et al. (2009) Compact models of motor primitive variations for predictable reaching and obstacle avoidance. In: IEEE-RAS international conference on humanoid robots, Paris, France, 07–10 December 2009, pp. 589–595.

Stulp F, Theodorou E, Buchli J, et al. (2011a) Learning to grasp under uncertainty. In: IEEE international conference on robotics and automation, Shanghai, China, 09–13 May 2011, pp. 5703–5708.

Stulp F, Theodorou E, Kalakrishnan M, et al. (2011b) Learning motion primitive goals for robust manipulation. In: IEEE/RSJ international conference on intelligent robots and systems, San Francisco, CA, USA, 25–30 September 2011, pp. 325–331.

Stulp F, Buchli J, Ellmer A, et al. (2012a) Model-free reinforcement learning of impedance control in stochastic environments. *IEEE Transactions on Autonomous Mental Development* 4(4): 330–341.

Stulp F, Theodorou E and Schaal S (2012b) Reinforcement learning with sequences of motion primitives for robust manipulation. *IEEE Transactions on Robotics* 28(6): 1360–1370.

Stulp F, Raiola G, Hoarau A, et al. (2013) Learning compact parameterized skills with a single regression. In: IEEE-RAS international conference on humanoid robots, Atlanta, GA, USA, pp. 417–422.

Su H, Hu Y, Li Z, et al. (2020) Reinforcement learning based manipulation skill transferring for robot-assisted minimally invasive surgery. In: IEEE international conference on robotics and automation, Paris, France, 31 May 2020–31 August 2020, pp. 2203–2208.

Su H, Mariani A, Ovur SE, et al. (2021) Toward teaching by demonstration for robot-assisted minimally invasive surgery. *IEEE Transactions on Automation Science and Engineering* 18(2): 484–494.

Sun X, Li J, Kovalenko AV, et al. (2022) Integrating reinforcement learning and learning from demonstrations to learn nonprehensile manipulation. *IEEE Transactions on Automation Science and Engineering* 20: 1–10. DOI: 10.1109/TASE.2022.3185071.

Sutanto G, Su Z, Schaal S, et al. (2018) Learning sensor feedback models from demonstrations via phase-modulated neural networks. In: IEEE international conference on robotics and automation, Brisbane, QLD, Australia, 21–25 May 2018, pp. 1142–1149.

Tamosiunaite M, Nemec B, Ude A, et al. (2011) Learning to pour with a robot arm combining goal and shape learning for dynamic movement primitives. *Robotics and Autonomous Systems* 59(11): 910–922.

Tan H and Kawamura K (2011) A computational framework for integrating robotic exploration and human demonstration in imitation learning. In: IEEE international conference on systems, man and cybernetics, Anchorage, AK, USA, 09–12 October 2011, pp. 2501–2506.

Tan H, Erdemir E, Kawamura K, et al. (2011) A potential field method-based extension of the dynamic movement primitive algorithm for imitation learning with obstacle avoidance. In: IEEE international conference on mechatronics and automation, Beijing, China, 07–10 August 2011, pp. 525–530.

Tan H, Zhao Y and Kannan B (2016) Applying adaptive control in modeling human motion behaviors in reinforcement robotic learning from demonstrations. In: AAAI Fall symposium - technical report, Palo Alto, California USA, volume FS-16-01–FS-16-05, pp. 79–85.

Tayebi A (2004) Adaptive iterative learning control for robot manipulators. *Automatica* 40(7): 1195–1203.

Theodorou E, Buchli J and Schaal S (2010) A generalized path integral control approach to reinforcement learning. *The Journal of Machine Learning Research* 11: 3137–3181.

Thota P, Ravichandar H and Dani A (2016) Learning and synchronization of movement primitives for bimanual manipulation tasks. In: IEEE 55th conference on decision and control, Las Vegas, NV, USA, 12–14 December 2016, pp. 945–950.

Thrun S (1996) Is learning the n-th thing any easier than learning the first? In: *Advances in Neural Information Processing Systems*, pp. 640–646.

Tomić T, Maier M and Haddadin S (2014) Learning quadrotor maneuvers from optimal control and generalizing in real-time. In: IEEE international conference on robotics and automation, Hong Kong, China, 31 May 2014–07 June 2014, pp. 1747–1754.

Travers M, Whitman J, Schiebel P, et al. (2016) Shape-based compliance in locomotion. In: *Robotics: Science and Systems*, Vol. 12.

Travers M, Whitman J and Choset H (2018) Shape-based coordination in locomotion control. *The International Journal of Robotics Research* 37(10): 1253–1268.

Tsagarakis NG, Caldwell DG, Negrello F, et al. (2017) Walk-man: a high-performance humanoid platform for realistic environments. *Journal of Field Robotics* 34(7): 1225–1259.

Ude A, Gams A, Asfour T, et al. (2010) Task-specific generalization of discrete and periodic dynamic movement primitives. *IEEE Transactions on Robotics* 26(5): 800–815.

Ude A, Nemec B, Petric T, et al. (2014) Orientation in cartesian space dynamic movement primitives. In: IEEE international conference on robotics and automation, Hong Kong, China, May, pp. 2997–3004.

Ugur E and Girgin H (2020) Compliant parametric dynamic movement primitives. *Robotica* 38(3): 457–474.

Umlauft J, Sieber D and Hirche S (2014) Dynamic movement primitives for cooperative manipulation and synchronized

motions. In: IEEE international conference on robotics and automation. Hong Kong, China, pp. 766–771.

Umlauft J, Fanger Y and Hirche S (2017) Bayesian uncertainty modeling for programming by demonstration. In: IEEE international conference on robotics and automation, Singapore, 29 May 2017–03 June 2017, pp. 6428–6434.

Villani L and De Schutter J (2008) Force control. In: B Siciliano and O Khatib (eds.) *Springer Handbook of Robotics*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 161–185.

Vuga R, Nemec B and Ude A (2015a) Enhanced policy adaptation through directed explorative learning. *International Journal of Humanoid Robotics* 12(3).

Vuga R, Nemec B and Ude A (2015b) Speed profile optimization through directed explorative learning. In: IEEE-RAS international conference on humanoid robots, Madrid, Spain, 18–20 November 2014, pp. 547–553.

Vuga R, Nemec B and Ude A (2016) Speed adaptation for self-improvement of skills learned from user demonstrations. *Robotica* 34(12): 2806–2822.

Wahrburg A, Guida S, Enayati N, et al. (2021) FlexDMP extending dynamic movement primitives towards flexible joint robots. In: IEEE international conference on robotics and automation, Xi'an, China, 30 May 2021–05 June 2021, volume 2021-May, pp. 7592–7598.

Wang J and Payandeh S (2015) A study of hand motion/posture recognition in two-camera views. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 9475: 314–323.

Wang R, Wu Y, Chan W, et al. (2016) Dynamic movement primitives plus: for enhanced reproduction quality and efficient trajectory modification using truncated kernels and local biases. In: IEEE/RSJ international conference on intelligent robots and systems, Daejeon, Korea, October, pp. 3765–3771.

Wang B, Gong J, Zhang R, et al. (2018) Learning to segment and represent motion primitives from driving data for motion planning applications. In: IEEE conference on intelligent transportation systems, Maui, HI, USA, 04–07 November 2018, pp. 1408–1414.

Wang B, Gong J and Chen H (2019) Motion primitives representation, extraction and connection for automated vehicle motion planning applications. *IEEE Transactions on Intelligent Transportation Systems* 21: 3931–3945.

Wang N, Chen C and Yang C (2020) A robot learning framework based on adaptive admittance control and generalizable motion modeling with neural network controller. *Neurocomputing* 390: 260–267.

Wang N, Chen C and Nuovo AD (2021) A framework of hybrid force/motion skills learning for robots. *IEEE Transactions on Cognitive and Developmental Systems* 13(1): 162–170.

Wang H, He H, Shang W, et al. (2022) Temporal logic guided motion primitives for complex manipulation tasks with user preferences. In: IEEE international conference on robotics and automation, Philadelphia, PA, USA, 23–27 May 2022, pp. 4305–4311.

Weitschat R and Aschemann H (2018) Safe and efficient human–robot collaboration part ii: optimal generalized human-in-the-loop real-time motion generation. *IEEE Robotics and Automation Letters* 3(4): 3781–3788.

Weitschat R, Haddadin S, Huber F, et al. (2013) Dynamic optimality in real-time: a learning framework for near-optimal robot motions. In: IEEE/RSJ international conference on intelligent robots and systems. Tokyo, Japan, November, pp. 5636–5643.

Wensing PM and Slotine JJ (2017) Sparse control for dynamic movement primitives. *IFAC-World Congress* 50(1): 10114–10121.

Wörgötter F, Geib C, Tamosiunaite M, et al. (2015) Structural bootstrapping-a novel, generative mechanism for faster and more efficient acquisition of action-knowledge. *IEEE Transactions on Autonomous Mental Development* 7(2): 140–154.

Wu Y, Wang R, D'Haro L, et al. (2018) Multi-modal robot apprenticeship: imitation learning using linearly decayed DMP+ in a human-robot dialogue system. In: IEEE/RSJ international conference on intelligent robots and systems, Madrid, Spain, 01–05 October 2018, pp. 8582–8588.

Wu M, Taetz B, He Y, et al. (2022) An adaptive learning and control framework based on dynamic movement primitives with application to human–robot handovers. *Robotics and Autonomous Systems* 148.

Xu JX and Wang W (2004) A multiple internal model approach to movement planning. In: IEEE international symposium on intelligent control, Taipei, 4 September 2004, pp. 186–191.

Xu JX, Wang W, Goh J, et al. (2005) Internal model approach for gait modeling and classification. In: IEEE international conference of the ieee engineering in medicine and biology, Shanghai, 17–18 January 2006, pp. 7688–7691.

Xu F, Huang R, Cheng H, et al. (2020) Stair-ascent strategies and performance evaluation for a lower limb exoskeleton. *International Journal of Intelligent Robotics and Applications* 4(3): 278–293.

Xu J, Xu L, Ji A, et al. (2023) A DMP-based motion generation scheme for robotic mirror therapy. *IEEE/ASME Transactions on Mechatronics*: 1–12.

Yang C, Ganesh G, Haddadin S, et al. (2011) Human-like adaptation of force and impedance in stable and unstable interactions. *IEEE Transactions on Robotics* 27(5): 918–930.

Yang PC, Sasaki K, Suzuki K, et al. (2016) Repeatable folding task by humanoid robot worker using deep learning. *IEEE Robotics and Automation Letters* 2(2): 397–403.

Yang C, Zeng C, Fang C, et al. (2018) A DMPS-based framework for robot learning and generalization of humanlike variable impedance skills. *IEEE/ASME Transactions on Mechatronics* 23(3): 1193–1203.

Yang C, Zeng C, Cong Y, et al. (2019) A learning framework of adaptive manipulative skills from human to robot. *IEEE Transactions on Industrial Informatics* 15(2): 1153–1161. DOI:10.1109/TII.2018.2826064.

Yang J, Zhang J, Settle C, et al. (2022) Learning periodic tasks from human demonstrations. In: IEEE international conference on robotics and automation, Philadelphia, Pennsylvania, USA, 23–27 May 2022, pp. 8658–8665.

Yin X and Chen Q (2014) Learning nonlinear dynamical system for movement primitives. In: IEEE international conference on systems, man and cybernetics, San Diego, CA, USA, 05–08 October 2014, pp. 3761–3766.

Yuan Y, Li Z, Zhao T, et al. (2020) DMP-based motion generation for a walking exoskeleton robot using reinforcement learning. *IEEE Transactions on Industrial Electronics* 67(5): 3830–3839.

Zeng C, Chen X, Wang N, et al. (2021) Learning compliant robotic movements based on biomimetic motor adaptation. *Robotics and Autonomous Systems* 135: 103668.

Zhai DH, Xia Z, Wu H, et al. (2022) A motion planning method for robots based on DMPS and modified obstacle-avoiding algorithm. *IEEE Transactions on Automation Science and Engineering*: 1–11.

Zhang H, Fu M, Luo H, et al. (2017) Robust human action recognition using dynamic movement features. In: *Intelligent Robotics and Applications*. Springer International Publishing, 474–484.

Zhang R, Chen J, Wang Z, et al. (2023) A step towards conditional autonomy - robotic appendectomy. *IEEE Robotics and Automation Letters* 8(5): 2429–2436.

Zhao Y, Xiong R, Fang L, et al. (2014) Generating a style-adaptive trajectory from multiple demonstrations. *International Journal of Advanced Robotic Systems* 11(1).

Zhao T, Deng M, Li Z, et al. (2020) Cooperative manipulation for a mobile dual-arm robot using sequences of dynamic movement primitives. *IEEE Transactions on Cognitive and Developmental Systems* 12(1): 18–29.

Zhou Y and Asfour T (2017) Task-oriented generalization of dynamic movement primitive. In: IEEE/RSJ international conference on intelligent robots and systems, Vancouver, BC, Canada, 24–28 September 2017, pp. 3202–3209.

Zhou Y, Do M and Asfour T (2016a) Coordinate change dynamic movement primitives—a leader-follower approach. In: IEEE/RSJ international conference on intelligent robots and systems. Daejeon, South Korea, pp. 5481–5488.

Zhou Y, Do M and Asfour T (2016b) Learning and force adaptation for interactive actions. In: IEEE-RAS international conference on humanoid robots, Cancun, Mexico, 15–17 November 2016, pp. 1129–1134.

Zhou Y, Gao J and Asfour T (2019) Learning via-point movement primitives with inter- and extrapolation capabilities. In: IEEE/RSJ international conference on intelligent robots and systems. Macau, China, pp. 4301–4308.

Zou C, Huang R, Qiu J, et al. (2021) Slope gradient adaptive gait planning for walking assistance lower limb exoskeletons. *IEEE Transactions on Automation Science and Engineering* 18(2): 405–413.