Aalto University
School of Electrical
Engineering

Trajectory Generation Using Dynamic Movement Primitives

Learning, Adaptation, and Control

Fares J. Abu-Dakka

Learning goals

- Understand the idea behind robot learning
- Understand the formulation of dynamic movement primitives: its
    - *benefits.*
    - *usability.*
    - *etc.*

Introduction:
Background, motivations and challenges

Robots are expected to assist us in our daily life tasks.

Introduction:
Background, motivations and challenges

Robots are expected to assist us in our daily life tasks.

Hard-coding the environments and related skills is infeasible.

Introduction:
Background, motivations and challenges

Difficulties

Robots are expected to assist us in our daily life tasks.

Hard-coding the environments and related skills is infeasible.

Introduction:
Background, motivations and challenges

Difficulties

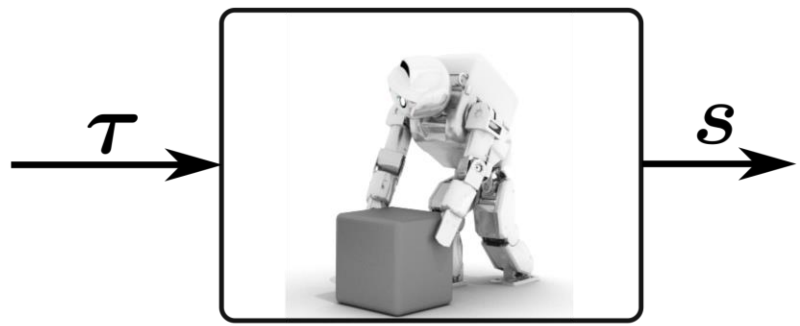
Robots are expected to assist us in our daily life tasks.

Hard-coding the environments and related skills is infeasible.

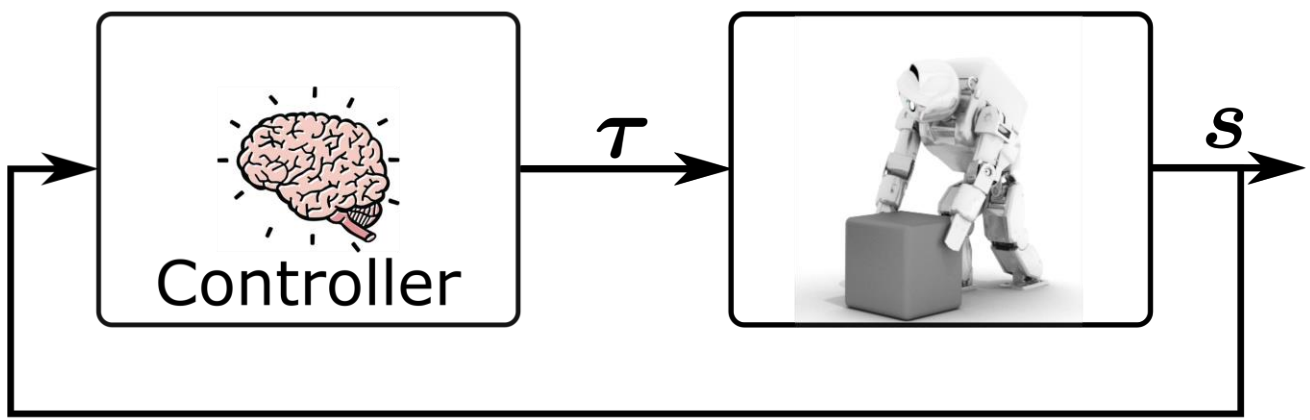
Solution

Learning

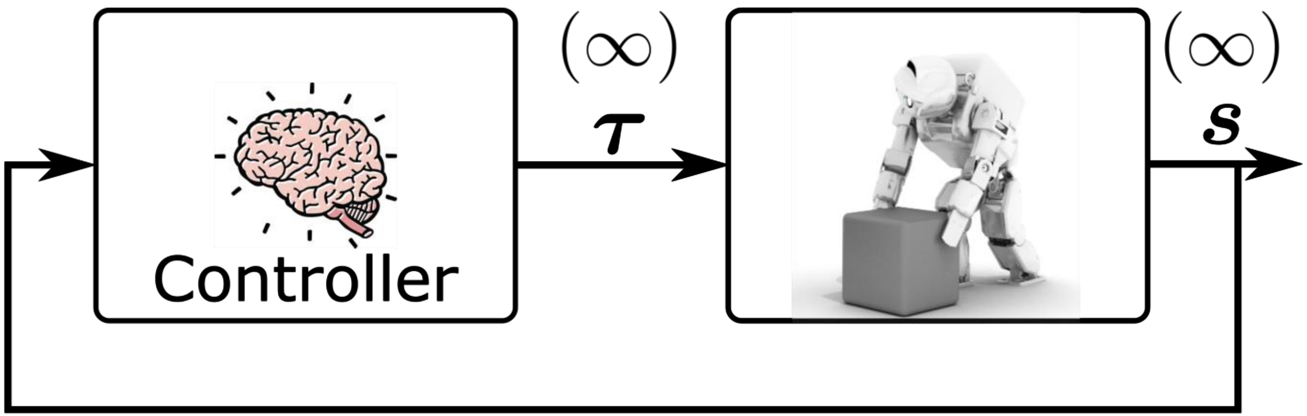
Introduction:
Challenges of robot learning



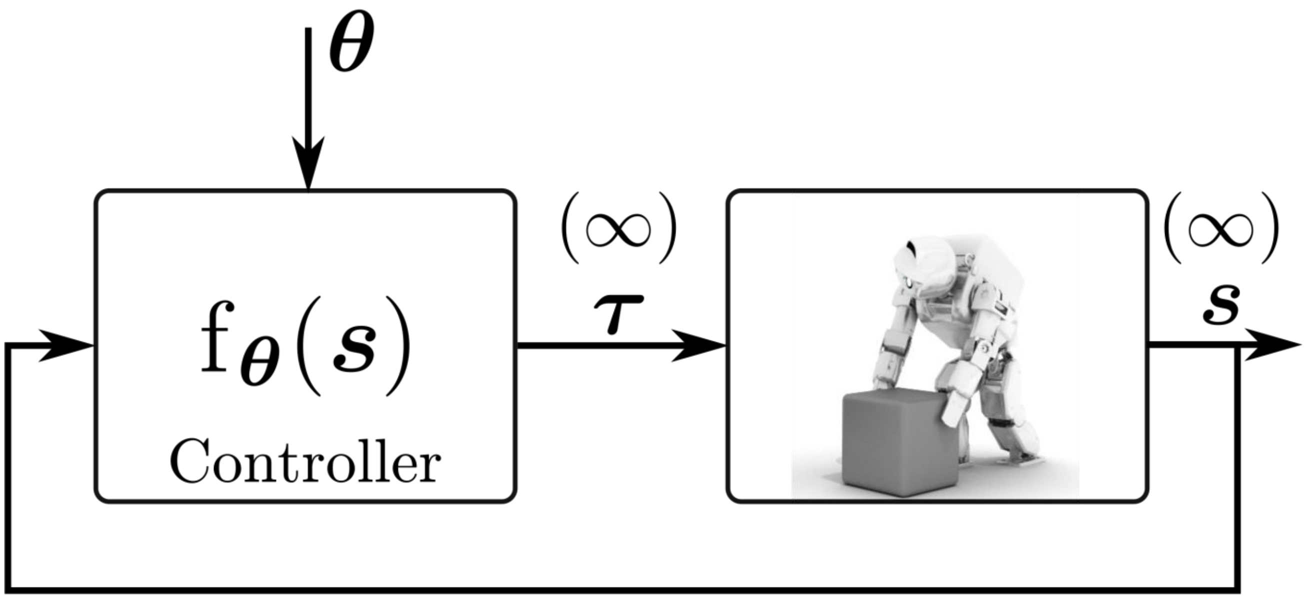
Introduction:
Challenges of robot learning



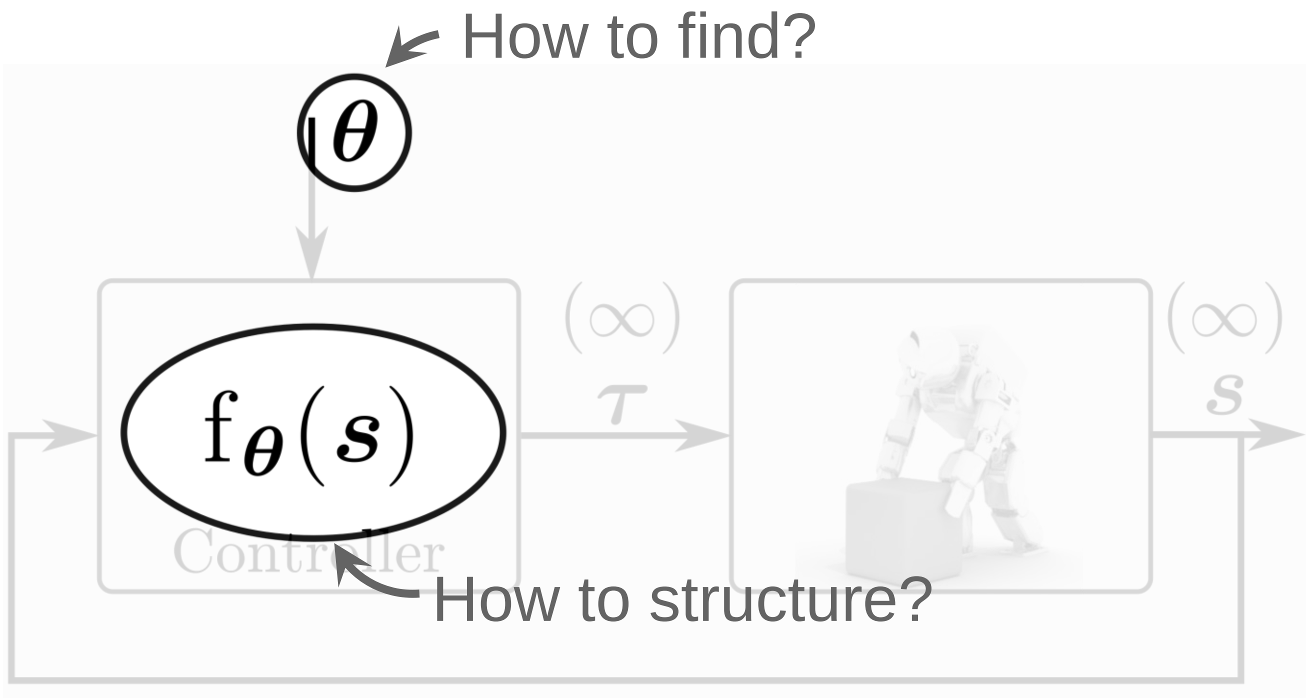
Introduction:
Challenges of robot learning



Introduction:
Challenges of robot learning



Introduction:
Challenges of robot learning

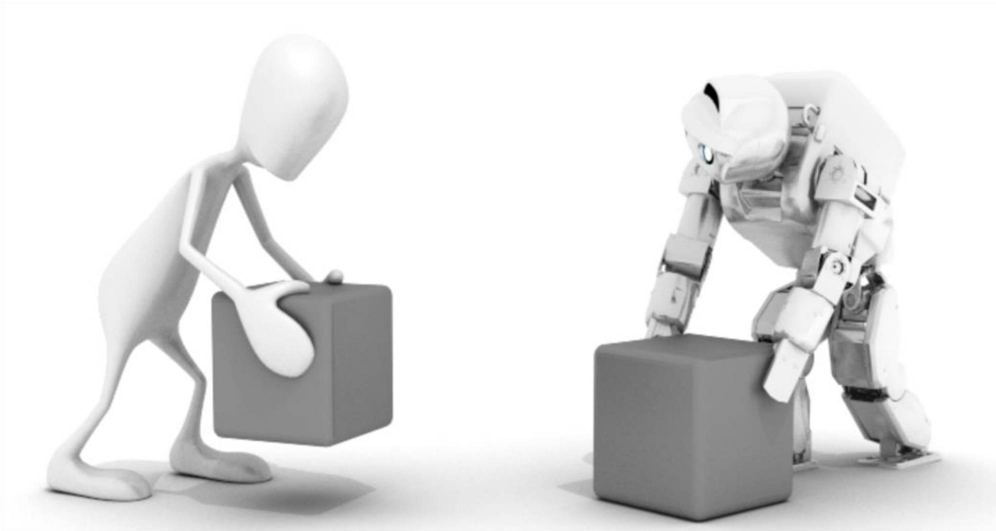


Introduction:
Challenges of robot learning

Introduction:
Challenges of robot learning

Reinforcement Learning

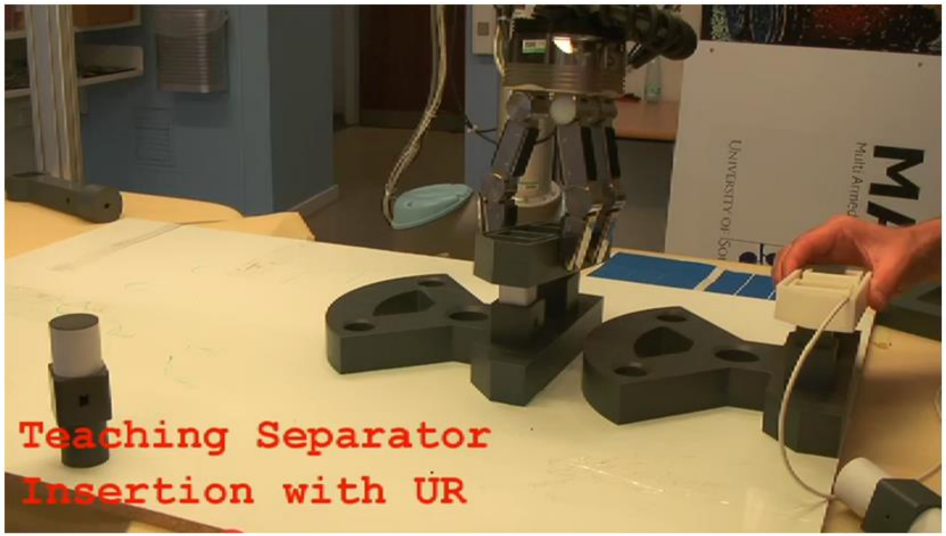
Introduction:
Challenges of robot learning



Learning from Human Demonstration

Introduction:
Learning from Demonstration

Develop new robot behavior through intuitive teaching



Teleoperation uses a magnetic tracker attached to the object held by human demonstrator.

Kinesthetic guiding uses the robot's gravity compensation mode.

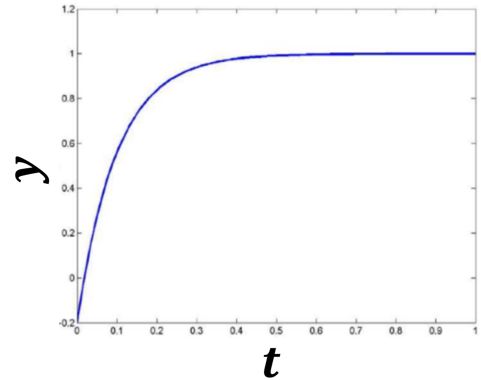
Introduction:
Dynamical Systems as Trajectory Generators

- **Dynamical systems can be used to represent trajectories:**
  - Integrating the dynamical system results in a trajectory. $\dot{y} = f(y)$
  - Mimics physical systems.
  - Build-in Smoothness.
- **Linear differential equations:**
  - well-defined behavior.
  - But: limited class of movements.

Introduction:
Dynamical Systems as Trajectory Generators

- **First order linear dynamical system:**

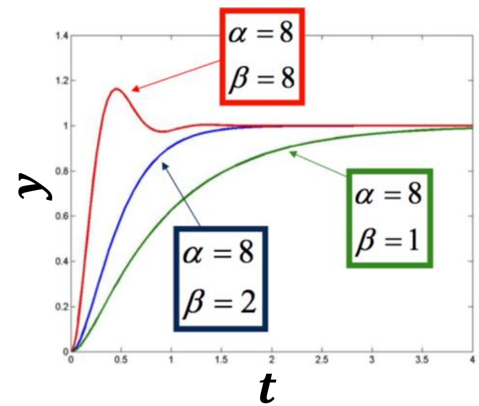
$$\dot{y} = \alpha(g - y)$$



- **Second order linear dynamical system:**

$$\ddot{y} = \alpha(\beta(g - y) - \dot{y})$$

  - g goal attractor
  - $\alpha\beta$ spring constant (stiffness)
  - α damping



Dynamic Movement Primitives (DMPs)

- **Dynamic movement primitives (DMPs):** are non-linear dynamic systems (Stefan Schaal's lab, 2002, updated in 2013 by Auke Ijspeert), and then updated to include **Cartesian space** by Abu-Dakka et al. 2015, then updated to include **Symmetric Positive Definite (SPD)** matrices by Abu-Dakka et al. 2020.

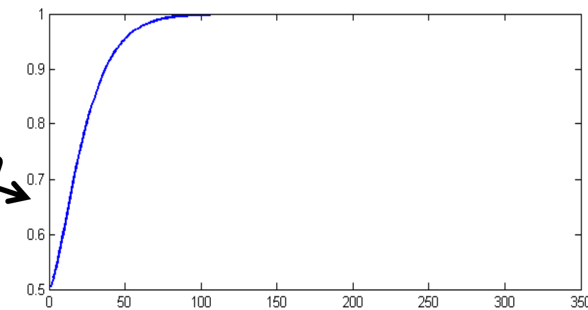
- DMPs provide a comprehensive framework for the effective imitation learning and control of robot movements.

DMPs

- A DMP for a single degree of freedom trajectory y is defined by a set of nonlinear differential equations:

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

2nd order system



x state variable of the system that makes equation (1) a time-independent system.
z is a scaled velocity of y.
τ is the time constant.
α_z and $\beta_z > 0$ define the behavior of the 2nd order system.
$\tau > 0$, $\alpha_z = 4\,\beta_z$ and $\alpha_x > 0$, the convergence of the underlying dynamic system to a unique attractor point at $y = g$, $z = 0$ is ensured.

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

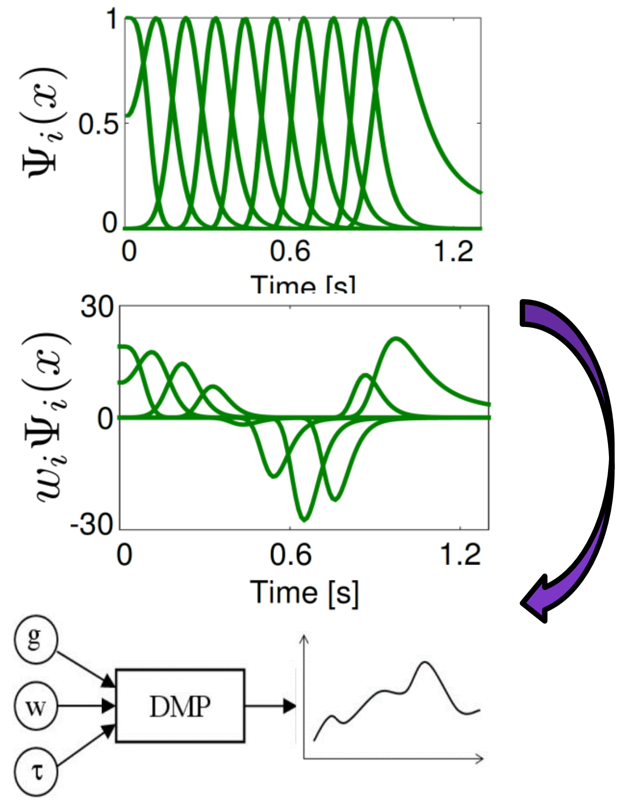
DMPs:
Forcing Term

Trajectory representation

$$f(x) = \frac{\sum_{i=1}^{N} \omega_i \Psi_i(x)}{\sum_{i=1}^{N} \Psi_i(x)} x(g - y_0),$$

$$\Psi_i(x) = \exp(-h_i(x - c_i)^2)\,,$$

RBF →



***f*(*x*) is:**

- a linear combination of N nonlinear radial basis functions,
- encodes the desired additional acceleration profile,
- learnable function,
- enables the robot to follow any smooth trajectory from the initial position y_0 to the final configuration g.

h_i, c_i and N are width, centers and no. of Gaussian functions. ω_i weight parameters adopted to reconstruct the recorded motion.

DMPs:
Learning from Demonstration

- **Given:**
  - A desired trajectory and its derivatives $\{y, \dot{y}, \ddot{y}\}_{t=1}^{T}$
  - A goal attractor g
  - Constant positive parameters $\alpha_z, \beta_z, \alpha_x$
  - Temporal Scaling τ: Adjusted to movement duration.
- **The weights w can be learned by linear regression:**
  - Compute desired values for each time step

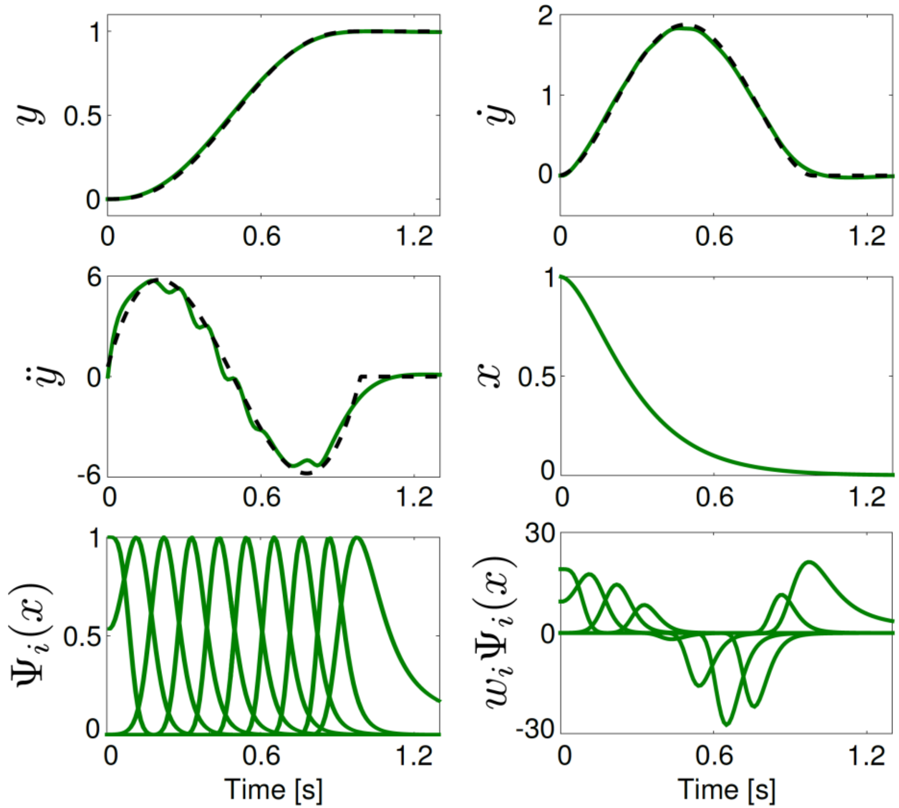
$$f_t^d = \tau^2 \ddot{y}_t - \alpha_z \left(\beta_z \left(g - y_t \right) - \tau \dot{y}_t \right)$$

  - Compute shape parameters by linear regression

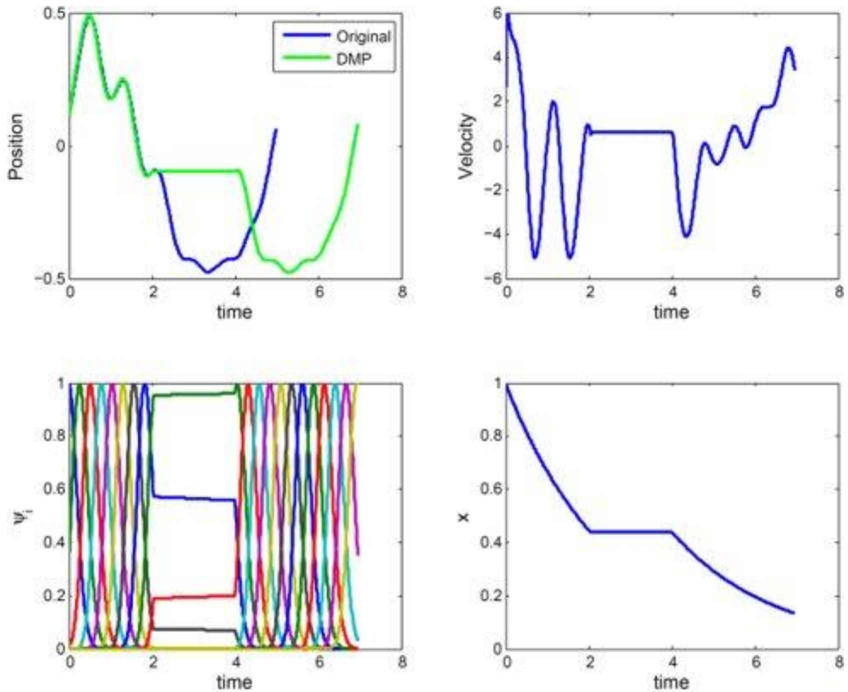
$$\mathbf{w} = (\boldsymbol{\Phi}^T \boldsymbol{\Phi} + \sigma^2 \mathbf{I})^{-1} \boldsymbol{\Phi}^T \mathbf{f}^d$$

$\boldsymbol{\Phi}$ is a matrix of Ψ_i

DMPs



DMPs



Robustness against perturbation:

Phase stopping

- The time evolution of phase can also be modulated online.
- If the robot cannot follow the desired motion, $\alpha_{px}|\bar{y} - y|$ becomes large, which in turn makes the phase change x small.

$$\tau \dot{x} = -\frac{\alpha_x x}{1 + \alpha_{px}\|\bar{y} - y\|}$$

$$\tau \dot{y} = 1 + \alpha_{py}(\bar{y} - y)$$

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

DMPs

Robustness against perturbation:

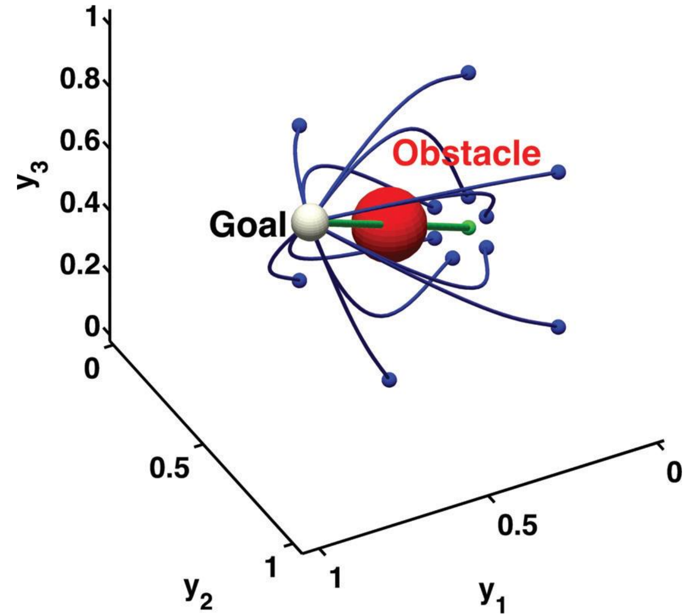
- Obstacle Avoidance: Spatial coupling

$$\tau\dot{z} = \alpha_z(\beta_z(g - y) - z) + f(x) + C_t,$$
$$\tau\dot{y} = z,$$
$$\textit{Spatial Coupling } C_t = \gamma \mathbf{R}\dot{\mathbf{y}}\theta \exp(-\beta\theta)$$

where

$$\theta = \arccos\left(\frac{(\mathbf{o}-\mathbf{y})^{\mathrm{T}}\dot{\mathbf{y}}}{|\mathbf{o}-\mathbf{y}||\dot{\mathbf{y}}|}\right)$$
$$\mathbf{r} = (\mathbf{o}-\mathbf{y}) \times \dot{\mathbf{y}}.$$

θ is the angle between $\dot{\mathbf{y}}$ and $(\mathbf{o}-\mathbf{y})$ (Obstacle position – Current position)

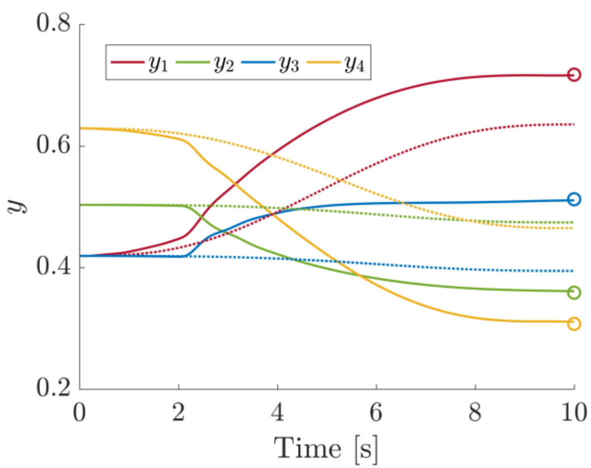


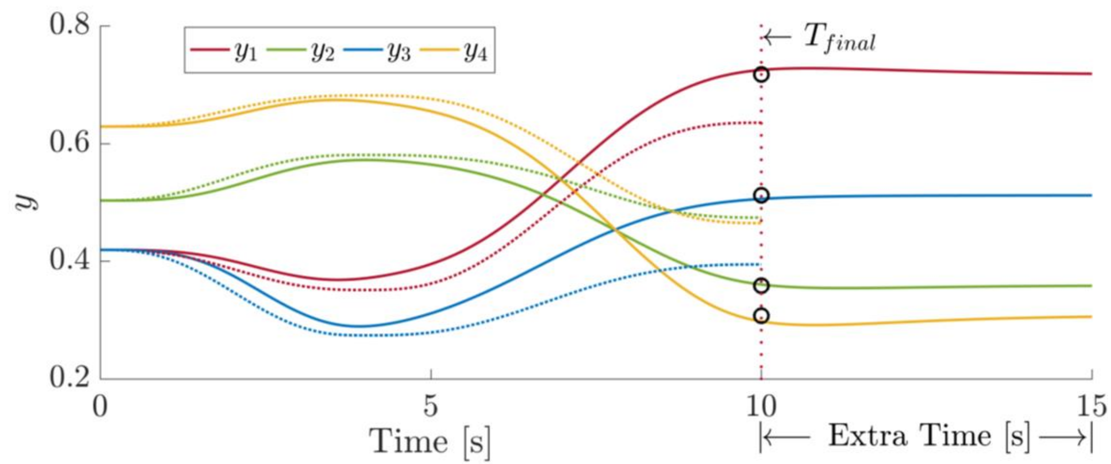
[1] Hoffmann, H., et al (2009). Biologically-inspired dynamical systems for movement generation: Automatic real-time goal adaptation and obstacle avoidance. In International Conference on Robotics and Automation (pp. 2587–2592). Piscataway, NJ.
[2] Ijspeert, A. J., et al (2013). Dynamical movement primitives: Learning attractor models for motor behaviors. *Neural Computations*, *25*(2), 328–373.

DMPs

Goal switching:

- Adaptation to new goal attractor g $\qquad \tau \dot{g} = \alpha_g(g_{new} - g)$

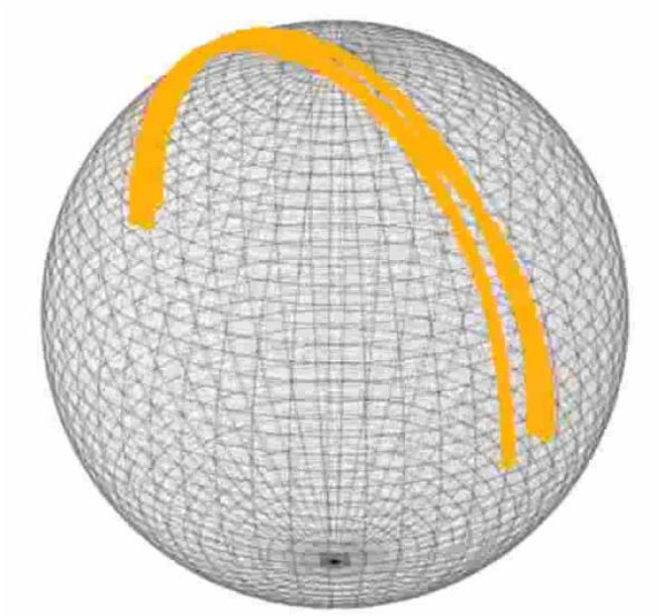


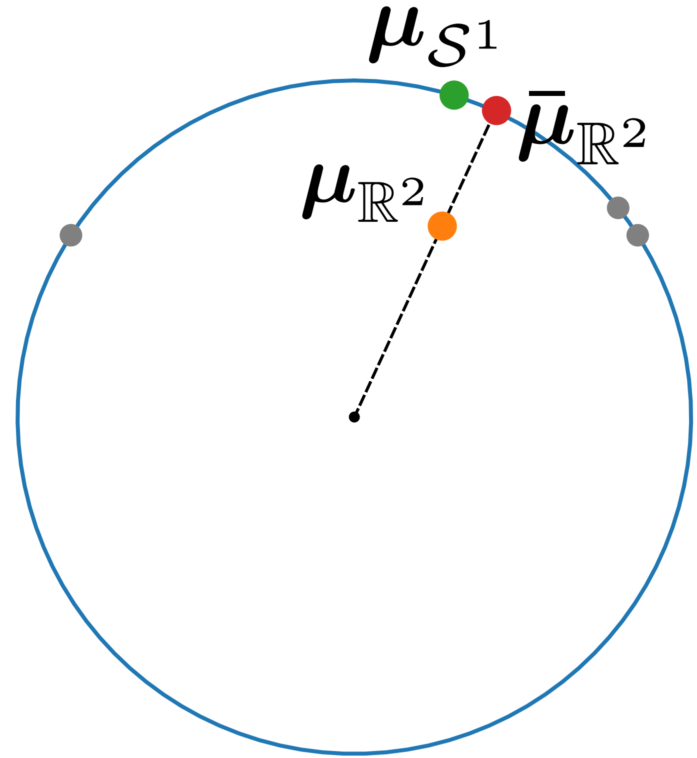


DMPs

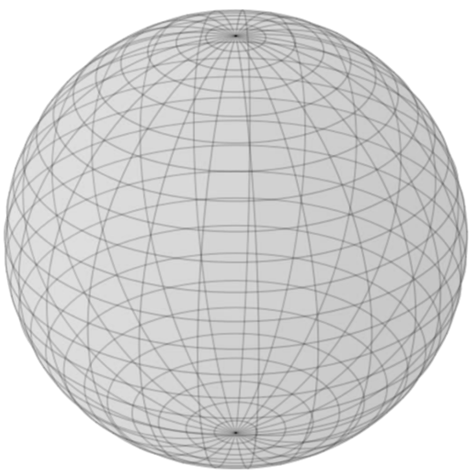
Movement sequencing

Geometry-aware DMPs:
Non-Euclidean Data

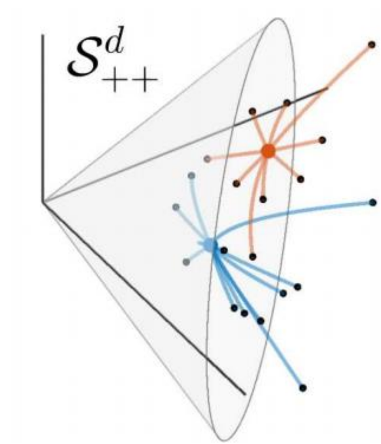




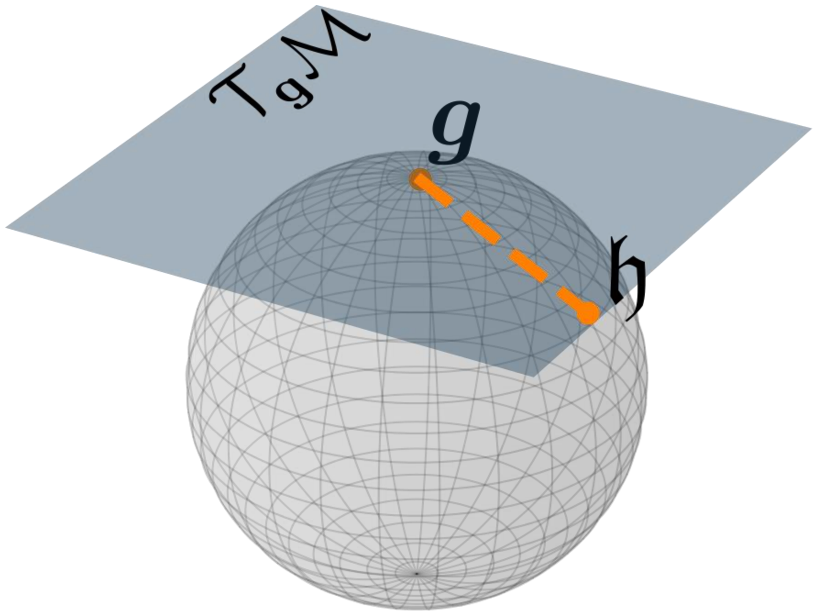
Geometry-aware DMPs:
Riemannian Manifolds: Definition



"A smooth topological space that locally resembles a Euclidean space (e.g. $\mathbb{R}^d$, Sym^d)."



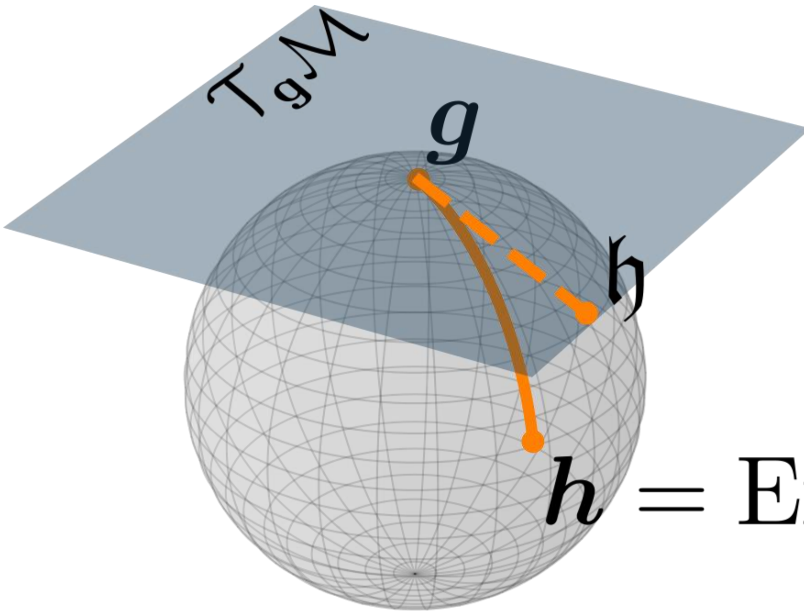
Geometry-aware DMPs:
Riemannian Manifolds: Tangent space



The metric in the tangent space is flat, which allows the use of classical arithmetic tools.

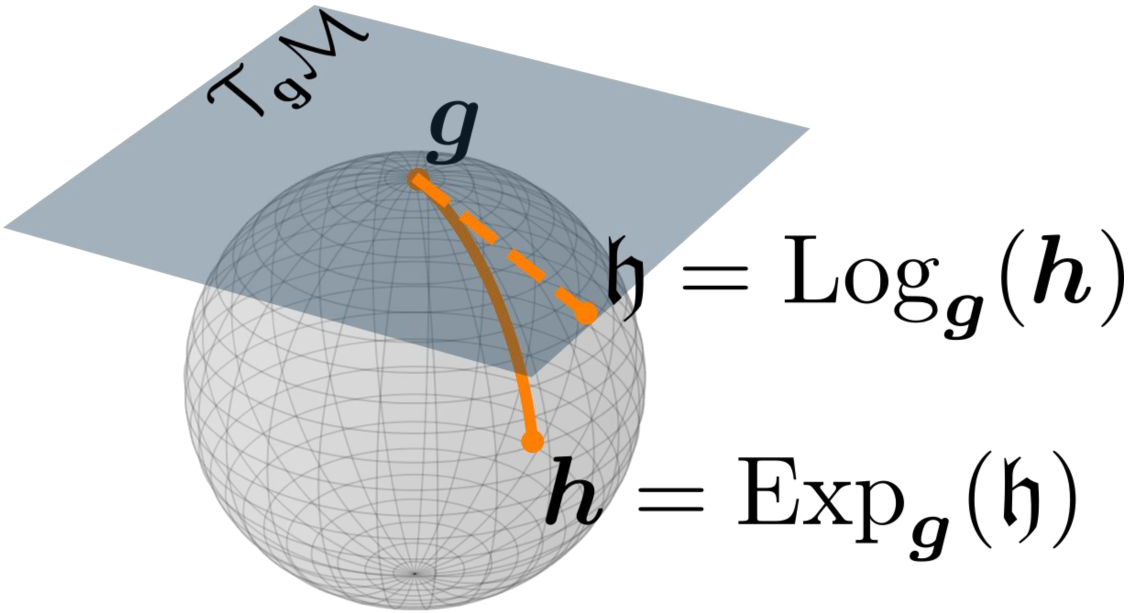
To operate on tangent spaces, a mapping system is required to switch between $\mathcal{T}_g\mathcal{M}$ and $\mathcal{M}$.

Geometry-aware DMPs:
Riemannian Manifolds: Exponential map



$$\boldsymbol{h} = \mathrm{Exp}_{\boldsymbol{g}}(\mathfrak{h})$$

Geometry-aware DMPs:
Riemannian Manifolds: Logarithmic map



Geometry-aware DMPs:
Riemannian Manifolds

Re-interpretation of basic standard operations in a Riemannian manifold

	Euclidean space	Riemannian manifold
Subtraction	$\overrightarrow{\mathbf{ab}} = \mathbf{b} - \mathbf{a}$	$\overrightarrow{\mathbf{AB}} = \mathrm{Log}_{\mathbf{A}}(\mathbf{B})$
Addition	$\mathbf{b} = \mathbf{a} + \overrightarrow{\mathbf{ab}}$	$\mathbf{B} = \mathrm{Exp}_{\mathbf{A}}(\overrightarrow{\mathbf{AB}})$
Distance	$\mathrm{dist}(\mathbf{a}, \mathbf{b}) = \lVert \mathbf{b} - \mathbf{a} \rVert$	$\mathrm{dist}(\mathbf{A}, \mathbf{B}) = \lVert \overrightarrow{\mathbf{AB}} \rVert_{\mathbf{A}}$
Interpolation	$\mathbf{a}(t) = \mathbf{a}_1 + t\overrightarrow{\mathbf{a}_1\mathbf{a}_2}$	$\mathbf{A}(t) = \mathrm{Exp}_{\mathbf{A}_1}(t\overrightarrow{\mathbf{A}_1\mathbf{A}_2})$

X. Pennec, P. Fillard, and N. Ayache, "A riemannian framework for tensor computing," International Journal of Computer Vision, vol. 66, no. 1, pp. 41–66, 2006.

Geometry-aware DMPs:

Sphere manifold S^d: Unit quaternion S^3

- **Cartesian Space DMPs:** in basic DMP equations, direct integration of unit quaternions (used to represent 3-D orientation) does not ensure that the normal of quaternions stays equal 1.

$$\tau\dot{\boldsymbol{\eta}} = \alpha_z(\beta_z 2\log(\mathbf{g}_o * \overline{\mathbf{q}}) - \boldsymbol{\eta}) + f_o(x),$$

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

$$\tau\dot{x} = -\alpha_x x,$$

$\mathbf{g}_o \in \mathbf{S}^3$ denotes the goal orientation.
$\overline{\mathbf{q}} = \overline{\upsilon + \mathbf{u}} = \upsilon - \mathbf{u}$ denotes the quaternion conjugation.
$\mathbf{q}_1 * \mathbf{q}_2 = (\upsilon_1 + \mathbf{u}_1) * (\upsilon_2 + \mathbf{u}_2)$
$= \left(\upsilon_1\upsilon_2 - \mathbf{u}_1^{\mathrm{T}}\mathbf{u}_2\right) + (\upsilon_1\mathbf{u}_2 + \upsilon_2\mathbf{u}_1 + \mathbf{u}_1 \times \mathbf{u}_2)$
$\boldsymbol{\eta} \in \mathbb{R}^3$ is treated as quaternion with zero scalar.

The quaternion logarithm log: $\mathbf{S}^3 \rightarrow \mathbb{R}^3$, $\log(\mathbf{q}) = \log(\upsilon + \mathbf{u}) = \begin{cases} \arccos(\upsilon)\frac{\mathbf{u}}{\|\mathbf{u}\|}, & \mathbf{u} \neq 0 \\ [0,0,0]^{\mathrm{T}}, & \text{otherwise} \end{cases}$

Abu-Dakka, F. J., Nemec, B., Jørgensen, J. A., Savarimuthu, T. R., Krüger, N., & Ude, A. (2015). Adaptation of manipulation skills in physical contact with the environment to reference force profiles. Autonomous Robots, 39(2), 199-217.

Geometry-aware DMPs:
Sphere manifold S^d: Unit quaternion S^3

- Quaternion logarithm can be used to specify the distance metric on the space of unit quaternion $\mathbf{S}^3$ (Ude 1999)

$$\mathrm{d}(\mathbf{q}_1, \mathbf{q}_2) = \begin{cases} \|\log(\mathbf{q}_1 * \overline{\mathbf{q}}_2)\|, & \mathbf{q}_1 * \overline{\mathbf{q}}_2 \neq -1 + [0,0,0]^{\mathrm{T}} \\ \pi, & \text{otherwise} \end{cases}$$

- Quaternion angular velocity: rotates quaternion $\mathbf{q}$ into $\mathbf{g}_o$ within unit sampling time. Thus only the application of the logarithmic map provides a proper mapping of the quaternion difference $\mathbf{g}_o * \mathbf{q}$ onto the angular velocity.

$$\boldsymbol{\omega} = 2\log(\mathbf{g}_o - \overline{\mathbf{q}})$$

- The logarithmic map becomes one-to-one and continuously differentiable if we limit its domain to $\mathbf{S}^3/(-1 + [0,0,0]^{\mathrm{T}}$. Thus, we can define its inverse, i.e. the exponential map $\mathbb{R}^3 \rightarrow \mathbf{S}^3$, as

$$\exp(\mathbf{r}) = \begin{cases} \cos(\|\mathbf{r}\|) + \sin(\|\mathbf{r}\|)\dfrac{\mathbf{r}}{\|\mathbf{r}\|}, & \mathbf{r} \neq 0 \\ 1 + [0,0,0]^{\mathrm{T}}, & \text{otherwise} \end{cases}$$

[1] Ude, A. (1999). Filtering in a unit quaternion space for model-based object tracking. Robotics and Autonomous Systems, 28(2–3), 163–172.
[2] Abu-Dakka, F. J. et al. (2015). Adaptation of manipulation skills in physical contact with the environment to reference force profiles. Autonomous Robots, 39(2), 199-217.

Geometry-aware DMPs:
Sphere manifold S^d: Unit quaternion S^3

- **Phase Stopping:**
  - *In the context of Cartesian space.*

$$\tau\dot{\mathbf{q}} = \frac{1}{2}\left(\boldsymbol{\eta} + \alpha_{pq} 2\log(\tilde{\mathbf{q}} - \bar{\mathbf{q}})\right) * \mathbf{q}$$

  - *In the context of force feed back.*

$$\tau\dot{\mathbf{q}} = \frac{1}{2}\left(\boldsymbol{\eta} - \alpha_{pq}\mathbf{K}_q\mathbf{e}_q(\boldsymbol{x})\right) * \mathbf{q}$$

Abu-Dakka, F. J., Nemec, B., Jørgensen, J. A., Savarimuthu, T. R., Krüger, N., & Ude, A. (2015). Adaptation of manipulation skills in physical contact with the environment to reference force profiles. Autonomous Robots, 39(2), 199-217.

Geometry-aware DMPs:
Special orthogonal manifold $\mathcal{SO}(d)$: Rotation matrix $\mathcal{SO}(3)$

Original formulation

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

$$\tau\dot{\boldsymbol{\eta}} = \alpha_z\left(\beta_z \log\left(\mathbf{R}_g\mathbf{R}^{\mathrm{T}}\right)-\boldsymbol{\eta}\right)+\mathbf{f}_o(x)$$
$$\tau\dot{\mathbf{R}} = [\boldsymbol{\eta}]_\times \mathbf{R}$$

$$\mathbf{f}_o(x) = \frac{\sum_{i=1}^{N}\mathbf{w}_i^o\Psi_i(x_j)}{\sum_{i=1}^{N}\Psi_i(x_j)}x_j =$$
$$\mathbf{D}_o^{-1}(\tau\dot{\boldsymbol{\eta}}_j+\alpha_z\boldsymbol{\eta}_j-\alpha_z\beta_z(\log(\mathbf{R}_g\mathbf{R}_j^{\mathrm{T}})))$$

$$\mathbf{R}(t+\Delta t) = \exp\left(\Delta t\frac{[\boldsymbol{\eta}]_\times}{\tau}\right)\mathbf{R}(t)$$

[1] Ales Ude, Bojan Nemec, Tadej Petric, and Jun Morimoto (2014). Orientation in Cartesian Space Dynamic Movement Primitives. ICRA, 2997–3004, Hong Kong, China.

Applications:
Peg-in-Hole

- A classical assembly problem.
- Requires position and force control
- Solutions:
  - *Hard-coding.*
  - *Learning.*

Applications:
Peg-in-Hole: Learning procedure with DMPs

- Data Acquisition.
- Encode data using Cartesian DMPs for orientation, and original DMPs for position.
- Adapt to a new situation and overcome errors coming from inaccurate pose estimation and other uncertainties.
- Integrate *Iterative Learning Control* to help in a successful peg insertion iteratively.
- Triger phase stopping mechanism to slow down the robot whenever it sense high forces.

Applications:
Peg-in-Hole: Learning procedure with DMPs

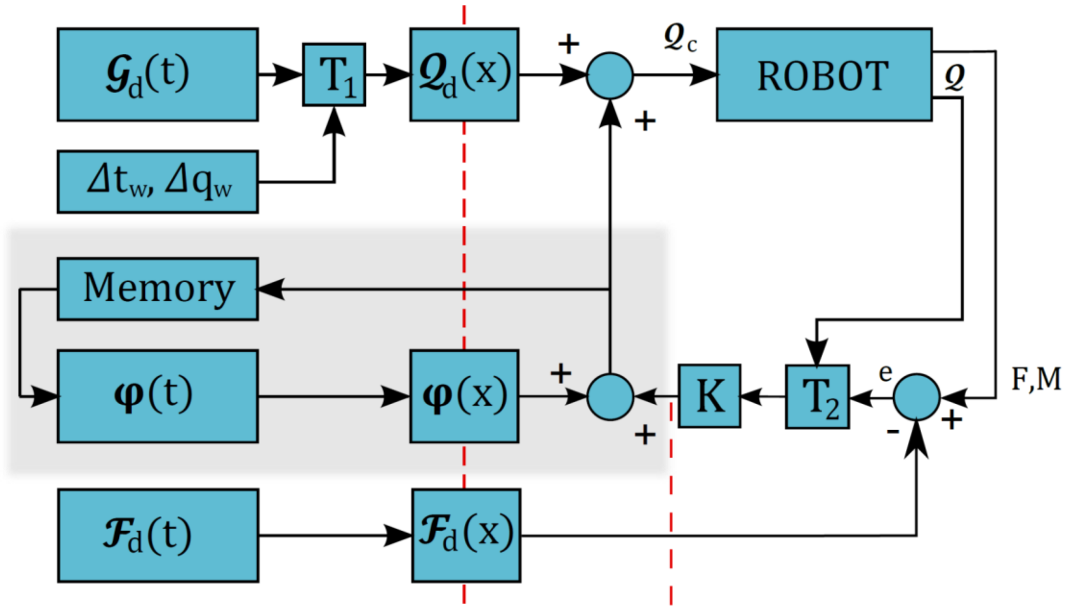
- **Slowing Down**
  - The proposed controller tracks simultaneously the desired position/orientations and forces/torques.
  - Force/torque adaptations requires low gains for stable and robust operation.
  - Thus, force adaptation is usually slow.
  - Slowing down the trajectory execution using DMP slow-down feedback, whenever the force/torque error is above the predefined limit.

$$\|\mathbf{e}\| = \begin{cases} 0 & \text{if } \|\mathbf{e}_p\| < \max_p \ \wedge \ \|\mathbf{e}_q\| < \max_q \\ \|[\mathbf{e}_p^T, \mathbf{e}_q^T]\| & . \end{cases}$$

$$\tau \dot{x} = -\frac{\alpha_x x}{1 + \alpha_{px}\|\mathbf{e}\|},$$

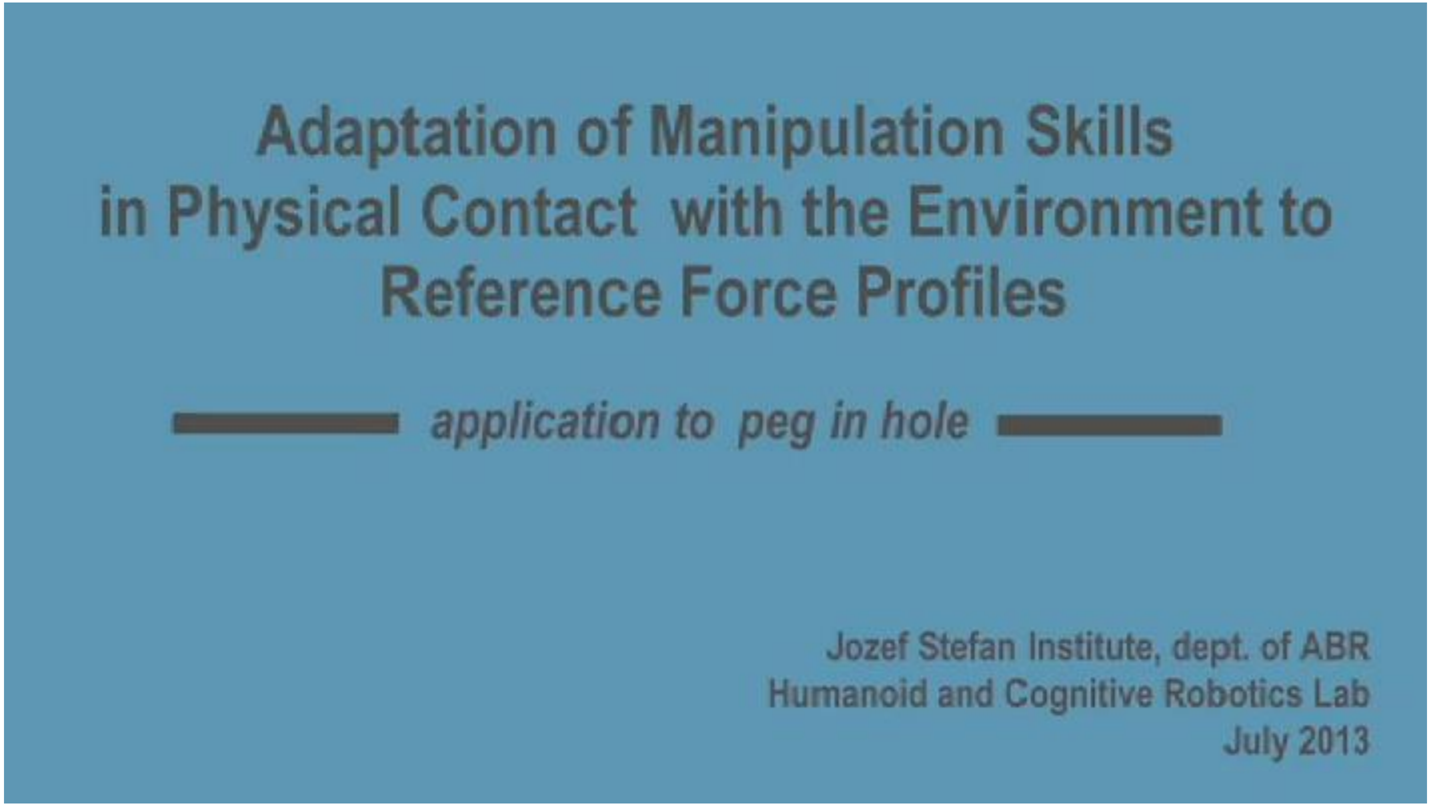
Applications:
Peg-in-Hole: Learning procedure with DMPs

- **Control scheme**

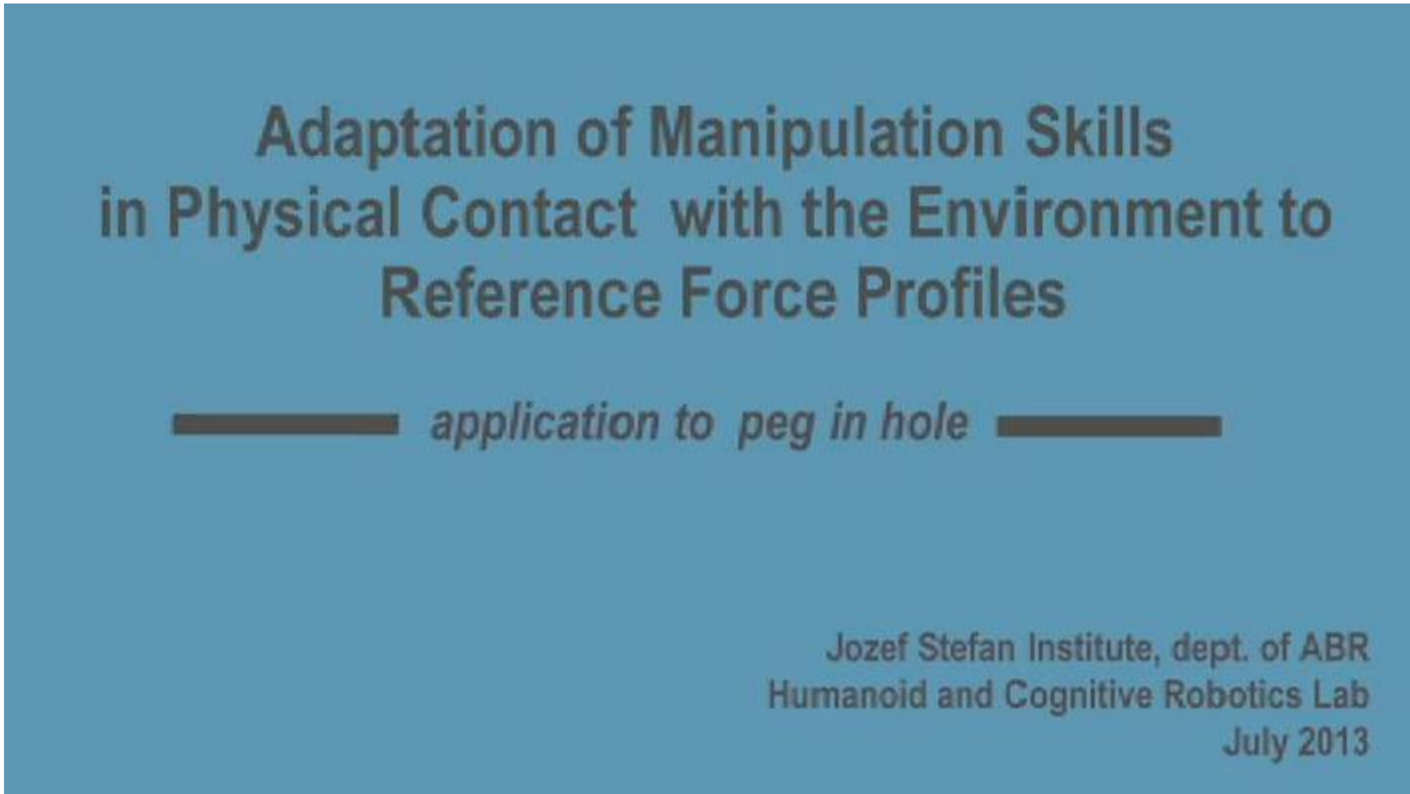


Abu-Dakka, F. J., Nemec, B., Jørgensen, J. A., Savarimuthu, T. R., Krüger, N., & Ude, A. (2015). Adaptation of manipulation skills in physical contact with the environment to reference force profiles. Autonomous Robots, 39(2), 199-217.

Applications:
Peg-in-Hole: Learning procedure with DMPs



Applications:
Peg-in-Hole: Learning procedure with DMPs



Summary

- Robot learning is essential in order to make robots to execute new tasks and avoid hard-coding.
- Learning from demonstration provides a friendly way to teach robots from human.
- Dynamic movement primitive is one of the imitation learning techniques that can be used to teach robots manipulation skills from single demonstration.
- Peg-in-Hole problem: application example.

Readings:

- Saveriano, Matteo, Fares J. Abu-Dakka, Aljaz Kramberger, and Luka Peternel. "***Dynamic Movement Primitives in Robotics: A Tutorial Survey***" arXiv preprint arXiv:2102.03861 (2021).