Dynamic Movement Primitives

Emre Ugur, BM-33

emre.ugur@boun.edu.tr

https://cmpe.boun.edu.tr/~emre/courses/cmpe58y/

cmpe58y@listeci.cmpe.boun.edu.tr

1.1. Authors

2009 IEEE International Conference on Robotics and Automation
Kobe International Conference Center
Kobe, Japan, May 12-17, 2009

Peter Pastor

Learning and Generalization of Motor Skills by Learning from Demonstration

Peter Pastor, Heiko Hoffmann, Tamim Asfour, and Stefan Schaal

Stefan Schaal

Tamim Asfour

1.2. Challenges Adressed

Learning a trajectory from demonstration :

- **Robustness**
  - Movements represented by a set of differential equations
- **Generalization**
  - Changing a goal parameter ensures adaptation to a new goal
- **Correspondence**
  - Trajectory recorded in end-effector space, then mapped to joint space

2. DMP Framework

2.1. DMP Formulations

2.2. How to learn?

2.3. Reproduce the movement

2.4. Advantages and Drawbacks

2.5. Adding terms to the equation

2.5.1. Obstacle Avoidance
2.5.2. Sensory Feedback

2.1. DMP Formulations

- A trajectory is represented by the following set of differential equations:

$$\tau \dot{v} = K(g - x) - Dv + (g - x_0) f(s)$$

$$\tau \dot{x} = v$$

where

x and v are the position and velocity of the system

x_0 and g are the start and goal position

K proportional (spring) constant

D derivative (damping) constant

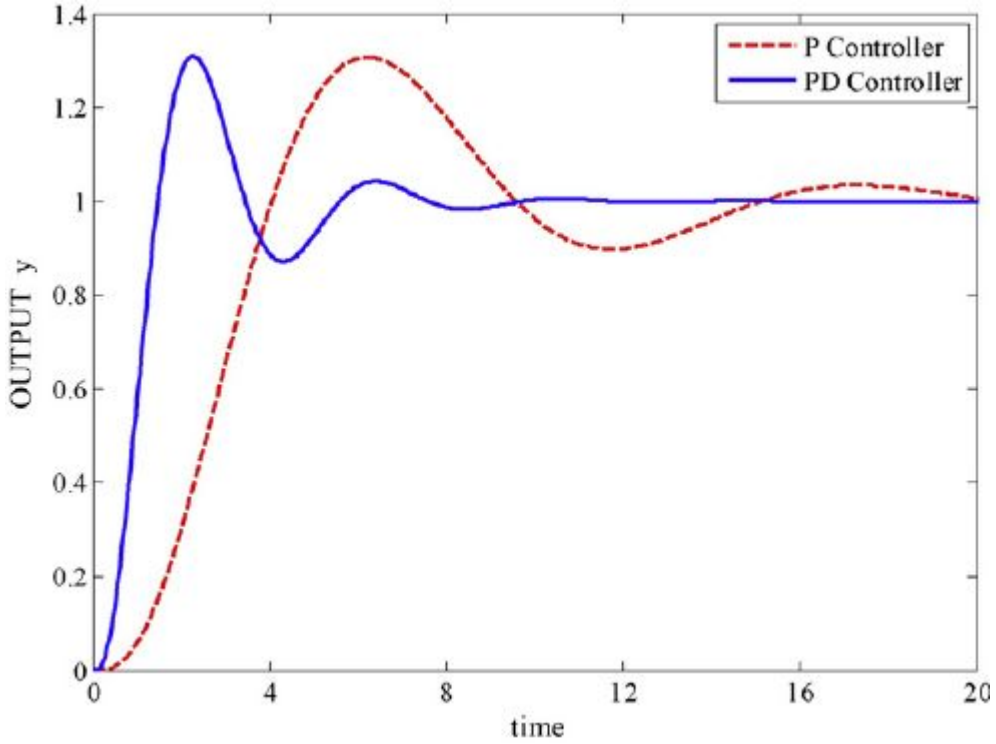
τ is a temporal scaling factor

f is a non-linear function

2.1. DMP Formulations

- For now, think of x as the angular position of a joint.
  - v is the angular velocity, $\dot{v}$ is the angular acceleration of this joint

- A **PD controller** would suffice for this joint to reach the goal angular position.



Shamsuzzoha, M. and S. Skogestad (2010). The setpoint overshoot method: A simple and fast method for closed-loop PID tuning. J. Process Control 20, 1220–1234.

2.1. DMP Formulations

- PD controller + non-linear perturbation:

$$\tau \dot{v} = K\,(g - x) - Dv + (g - x_0)\,f\,(s)$$
$$\tau \dot{x} = v$$

→ Transformation system

- The non-linear function is defined as:

$$f(s) = \frac{\sum_i w_i \psi_i(s)\, s}{\sum_i \psi_i(s)} \quad \text{with} \quad \psi_i = \exp\left(-h_i\,(s - c_i)^2\right)$$

→ Non-linear function

- Phase variable s is defined as:

$$\tau \dot{s} = -\alpha\, s \longrightarrow s(t) = \exp\left(\frac{-\alpha}{\tau}\, t\right)$$

→ Canonical system

2.2. How to learn?

1. Record a movement *x(t)* where time *t* = 1,…,*T* (*T*=duration)

- Take derivative of $x(t) = \dot{x}(t)$, scale by τ to get v(t)
- Take derivative of $v(t) = \dot{v}(t)$

2. Integrate the canonical system

- Choose α and integrate the canonical system for all time steps t=1,…,T

3. Calculate $f_{target}(s)$ by:

$$f_{target}(s) = \frac{-K(g-x) + Dv + \tau\dot{v}}{g - x_0}$$

where x_0 is set to $x(0)$ and g is set to $x(T)$

2.2. How to learn?

4. Minimize the squared error to find weights w_i

$$J = \sum_s (f_{target}(s) - f(s))^2$$

- Create $s(t)$ for all time steps $t=1,\ldots,T$
- Select a number of Gaussian functions, as well as each Gaussian's center and variance to derive $\varphi_i(s)$ from
- Find $f(s)$ in terms of unknown weights w_i
- Finding w_i is a linear regression problem

$$\psi_i = \exp(-h_i (s - c_i)^2)$$

$$X = \begin{bmatrix} \psi_1(s_0) & \psi_2(s_0) & \cdots & \psi_n(s_0) \\ \psi_1(s_1) & & & \psi_n(s_1) \\ \vdots & & & \vdots \\ \psi_1(s_T) & \psi_2(s_T) & \cdots & \psi_n(s_T) \end{bmatrix}$$

$$w = (w_1, w_2, \ldots, w_n)$$

$$y = \frac{f_{target}(s) \sum_i \psi_i(s)}{s}$$

$$\hat{w} = (X^T X)^{-1} (X^T y)$$

2.3. Reproduce the movement

- A movement plan is generated by :
  - Reusing weights w_i
  - Specifying a desired start and goal states, x_0 and g
  -

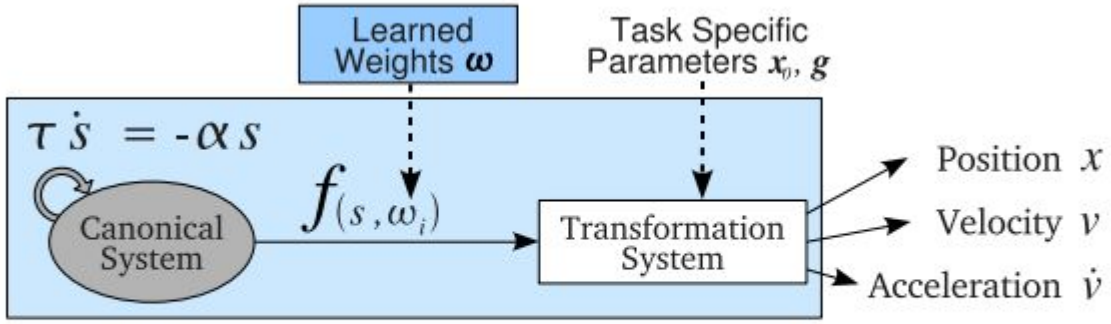


Fig. 1. Sketch of a one dimensional DMP: the canonical system drives the nonlinear function f which perturbs the transformation system.

2.4. Advantages and Drawbacks

Advantages
Convergence to goal is guaranteed
Robust to perturbations
Spatial and temporal invariant
Weights can be learned to generate any trajectory

Drawbacks
If x_0 and g are close, the system will remain at x_0
If g-x_0 is small, change in g may result in huge accelerations
When g-x_0 changes sign, the resulting generalization is mirrored

$$\tau \dot{v} = K\,(g - x) - Dv + (g - x_0)\, f\,(s)$$

$$f(s) = \frac{\sum_i w_i \psi_i(s) s}{\sum_i \psi_i(s)}$$

2.4. Advantages and Drawbacks

Advantages
Convergence to goal is guaranteed
Robust to perturbations
Spatial and temporal invariant
Weights can be learned to generate any trajectory

Drawbacks
If x_0 and g are close, the system will remain at x_0
If g-x_0 is small, change in g may result in huge accelerations
When g-x_0 changes sign, the resulting generalization is mirrored

$$\tau \dot{v} = K\,(g - x) - Dv + (g - x_0)\, f\,(s)$$

$$\tau \dot{v} = K\,(g - x) - Dv - K\,(g - x_0)\, s + K\, f\,(s)$$

$$f(s) = \frac{\sum_i w_i \psi_i(s) s}{\sum_i \psi_i(s)}$$

2.4. Advantages and Drawbacks

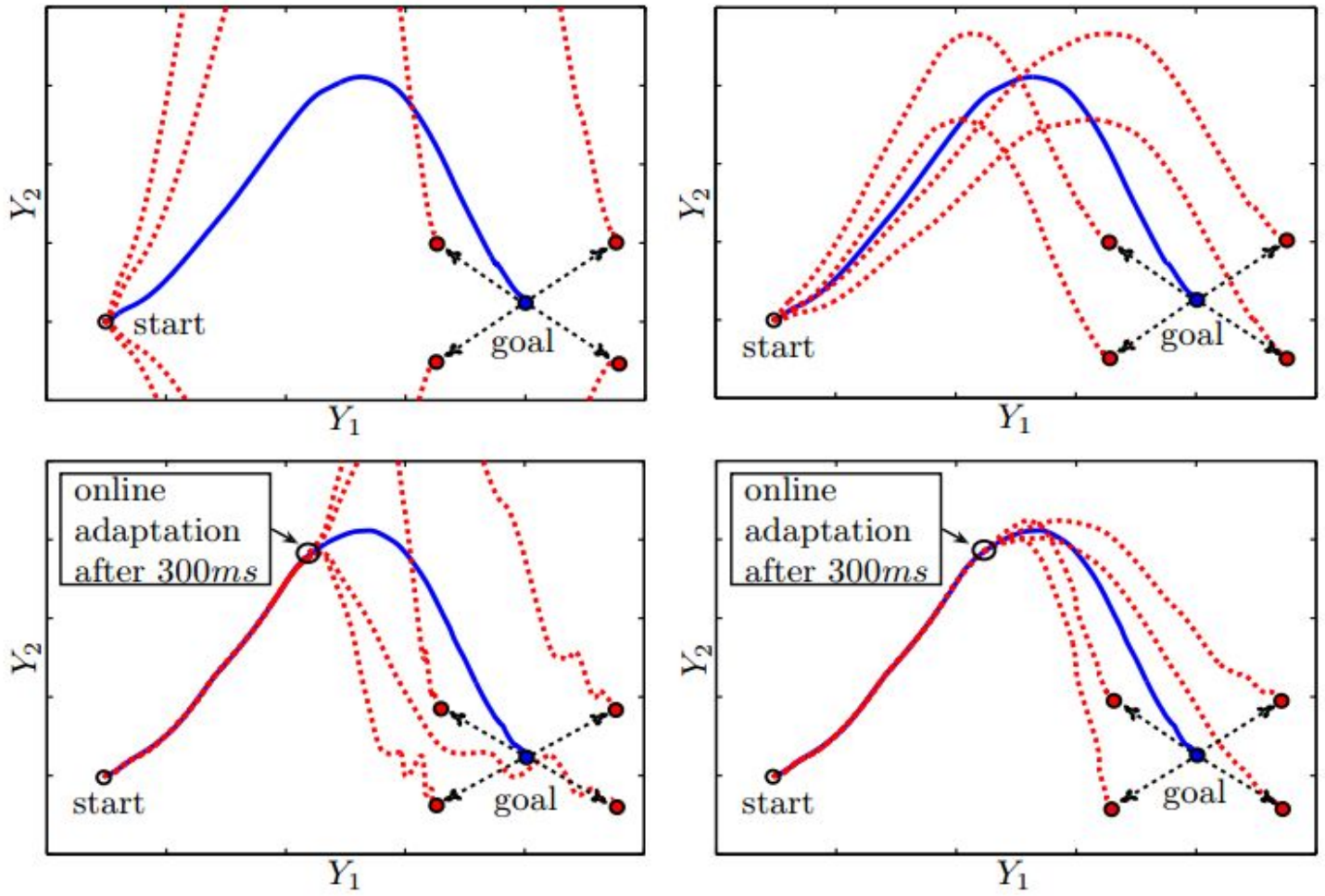


Fig. 2. Comparison of goal-changing results between old (Left) and new (Right) DMP formulation in operational space (Y_1, Y_2) with one transformation system for each dimension. The same original movement (solid line) and goals are used for both formulations. The dashed lines show the result of changing the goal before movement onset (Top) and during the movement (Bottom).

2.4. Advantages and Drawbacks

- New transformation system:

$$\tau \dot{v} = K\,(g - x) - Dv - K\,(g - x_0)\,s + K\,f(s)$$

$$\tau \dot{x} = v$$

→ Transformation system

- The non-linear function is defined as:

$$f(s) = \frac{\sum_i w_i \psi_i(s)\, s}{\sum_i \psi_i(s)} \quad \text{with} \quad \psi_i = \exp\left(-h_i\,(s - c_i)^2\right)$$

→ Non-linear function

- Phase variable s is defined as:

$$\tau \dot{s} = -\alpha\, s \longrightarrow s(t) = \exp\left(\frac{-\alpha}{\tau}\, t\right)$$

→ Canonical system

2.5. Adding terms to the equation

2.5.1. Obstacle Avoidance

- In 3D end-effector space, the scalars x, v and $\dot{v}$ turn into vectors $\boldsymbol{x}$, $\boldsymbol{v}$ and $\dot{\boldsymbol{v}}$ and scalars K, D turn into positive definite matrices $\boldsymbol{K}$, $\boldsymbol{D}$.

- Add only the coupling term $p(x,v)$ to the transformation system :

$$\tau \dot{\boldsymbol{v}} = \boldsymbol{K}(\boldsymbol{g} - \boldsymbol{x}) - \boldsymbol{D}\boldsymbol{v} - \boldsymbol{K}(\boldsymbol{g} - \boldsymbol{x}_0)\boldsymbol{s} + \boldsymbol{K}\boldsymbol{f}(\boldsymbol{s}) + \boldsymbol{p}(\boldsymbol{x}, \boldsymbol{v})$$

$$\boldsymbol{p}(\boldsymbol{x}, \boldsymbol{v}) = \gamma \boldsymbol{R}\boldsymbol{v} \exp(-\beta \phi)$$

where

- $\boldsymbol{R}$ is a rotational matrix with axis $\boldsymbol{r} = (\boldsymbol{x} - \boldsymbol{o}) \times \boldsymbol{v}$ and angle of rotation of $\pi/2$
- $\boldsymbol{o}$ is the position of the obstacle
- γ and β are constants
- φ is the angle between the direction of the end-effector towards the obstacle and the end-effector's velocity vector $\boldsymbol{v}$ relative to the obstacle.

2.5. Adding terms to the equation

2.5.2. Sensory Feedback

- In 3D end-effector space, the scalars *x, v* and turn into vectors $\boldsymbol{x}, \dot{\boldsymbol{w}}$ and and scalars *K, D* turn into positive definite matrices $\boldsymbol{K}$, $\boldsymbol{D}$.

- Add only the coupling term ζ to the transformation system :

$$\tau \dot{\boldsymbol{v}} = \boldsymbol{K}(\boldsymbol{g} - \boldsymbol{x}) - \boldsymbol{D}\boldsymbol{v} - \boldsymbol{K}(\boldsymbol{g} - \boldsymbol{x}_0)\boldsymbol{s} + \boldsymbol{K}\boldsymbol{f}(\boldsymbol{s}) + \boldsymbol{\zeta}$$

$$\boldsymbol{\zeta} = \boldsymbol{K}_1 \boldsymbol{J}_{sensor}^T \boldsymbol{K}_2 (\boldsymbol{F} - \boldsymbol{F}_{des})$$

where

- $\boldsymbol{J}_{sensor}$ is the Jacobian of the task controlled by the movement primitives wrt sensors
- $\boldsymbol{F}$ are the generalized forces read from sensors (wrench = torque + force –> 6D)
- $\boldsymbol{F}_{des}$ is the desired forces acquired from demonstrations
- $\boldsymbol{K}_1$ and $\boldsymbol{K}_2$ are positive definite gain matrices

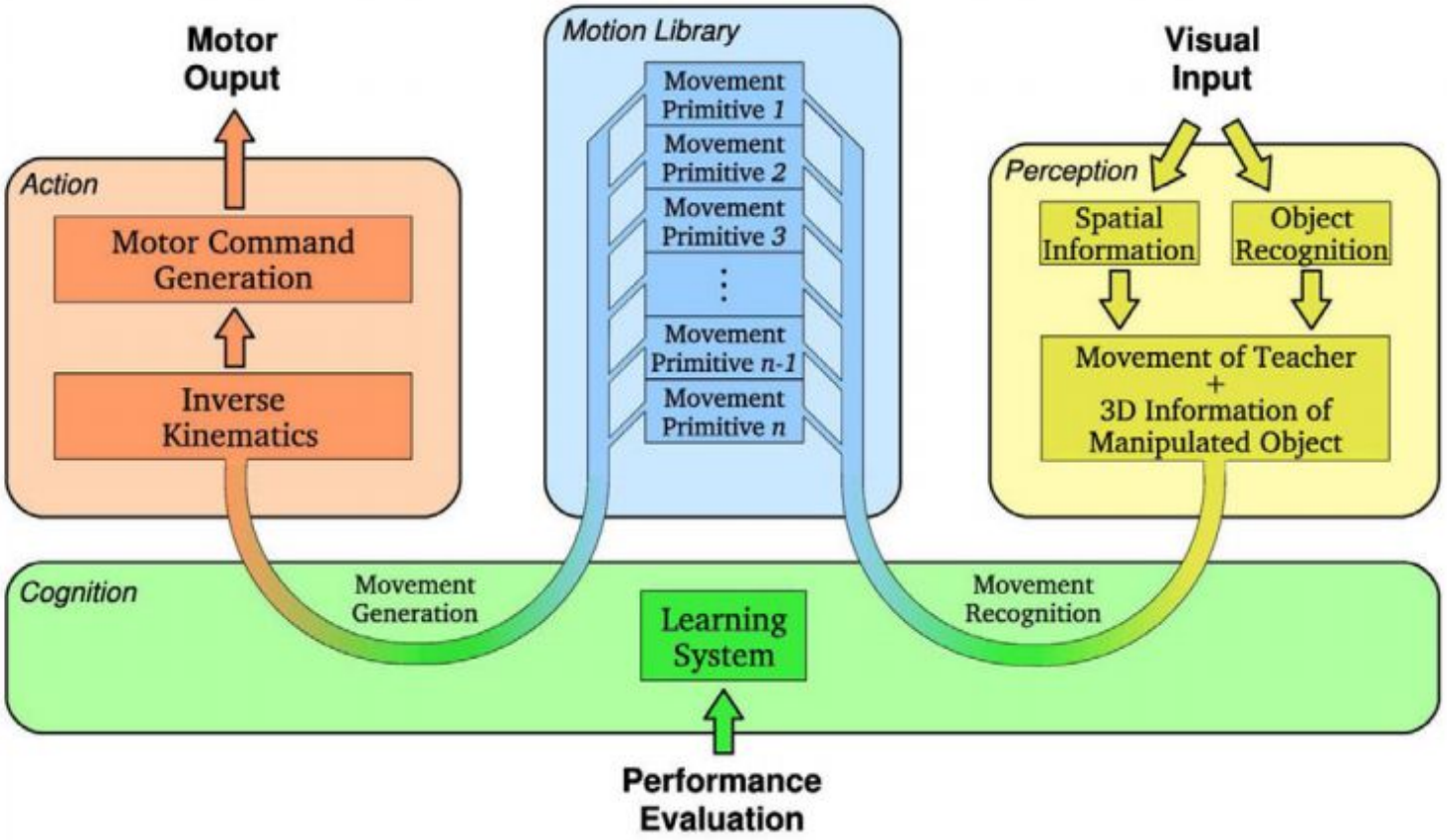
3. Library of movement primitives

3.1. Motion Library

3.2. ASM Framework

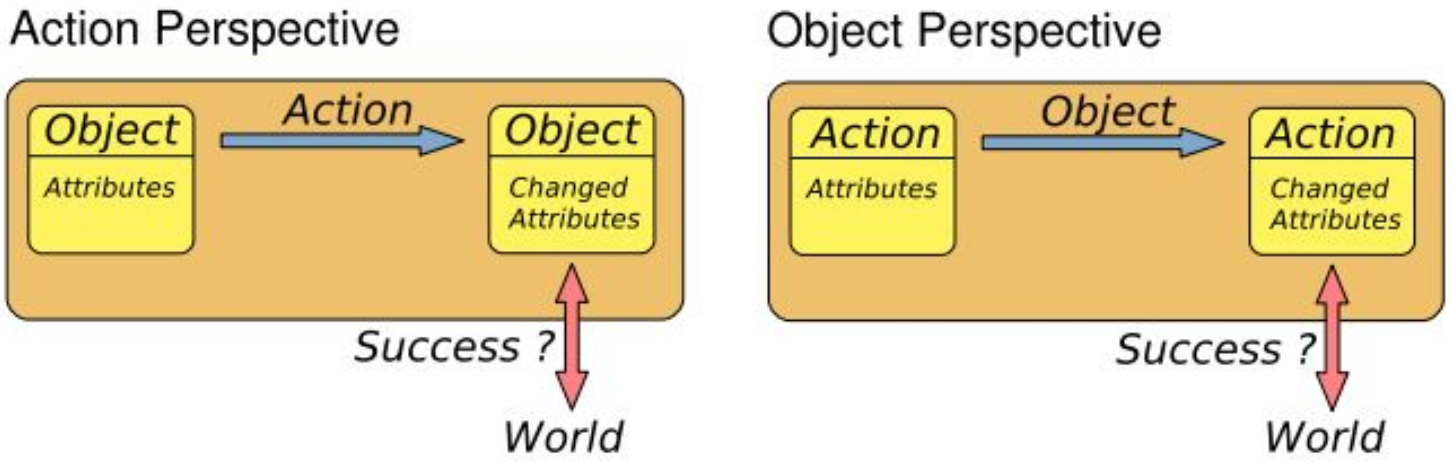
3.3. Combination of Movement Primitives

3.1. Motion Library



3.2. ASM Framework

- Need to relate movements to context (i.e. to objects, the environment etc.) - object affordances → **Object-Action Complexes**
- All sensory events (e.g. joint angle sensors, IMUs, force sensors, touch sensors, etc.) are collected as large sensory feature vector
- Mean and variances from multiple trials recorded and approximated by a function approximator as the non-linear function *f(s)*
- This additional information is called **Associative Skill Memory (ASM).**



3.2. ASM Framework

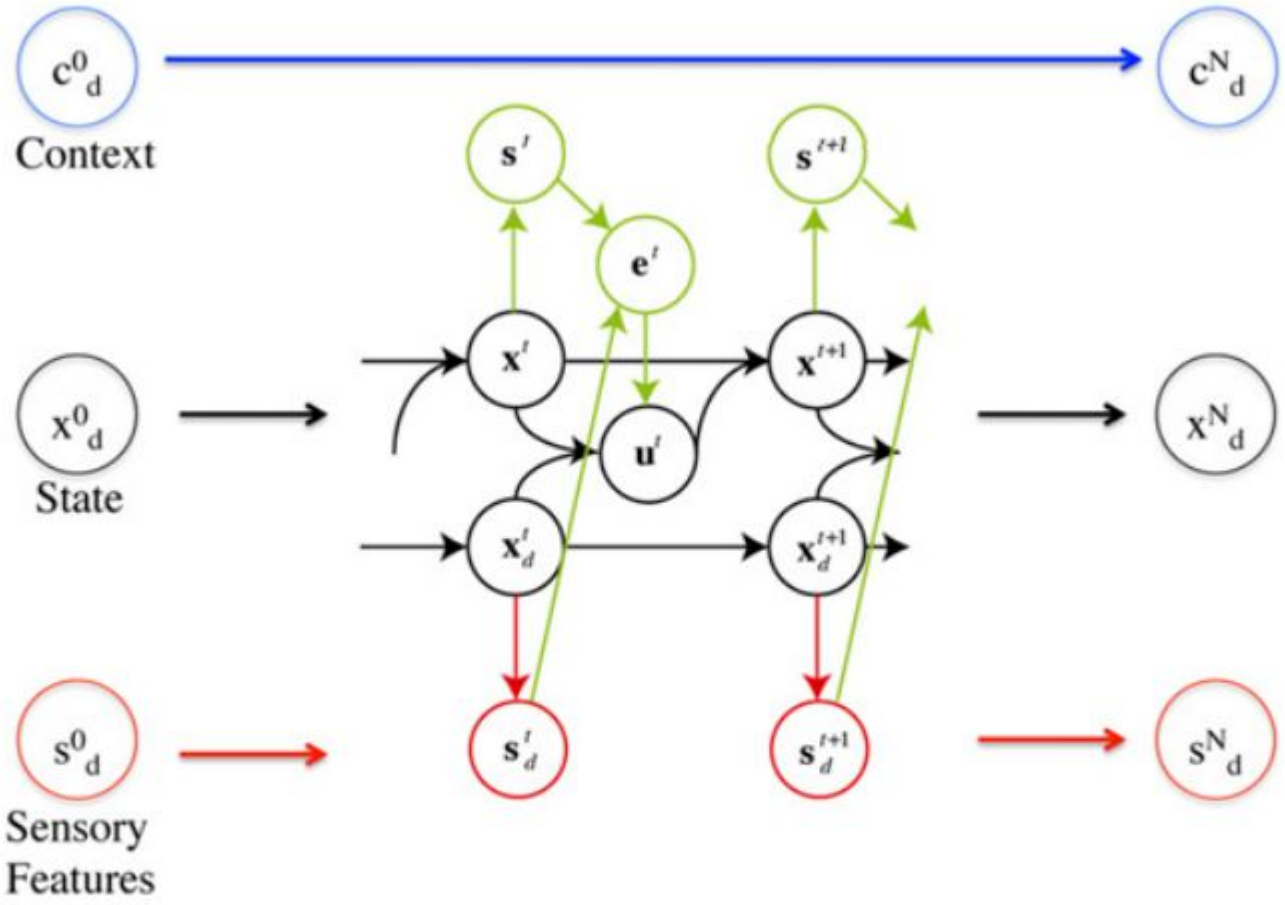
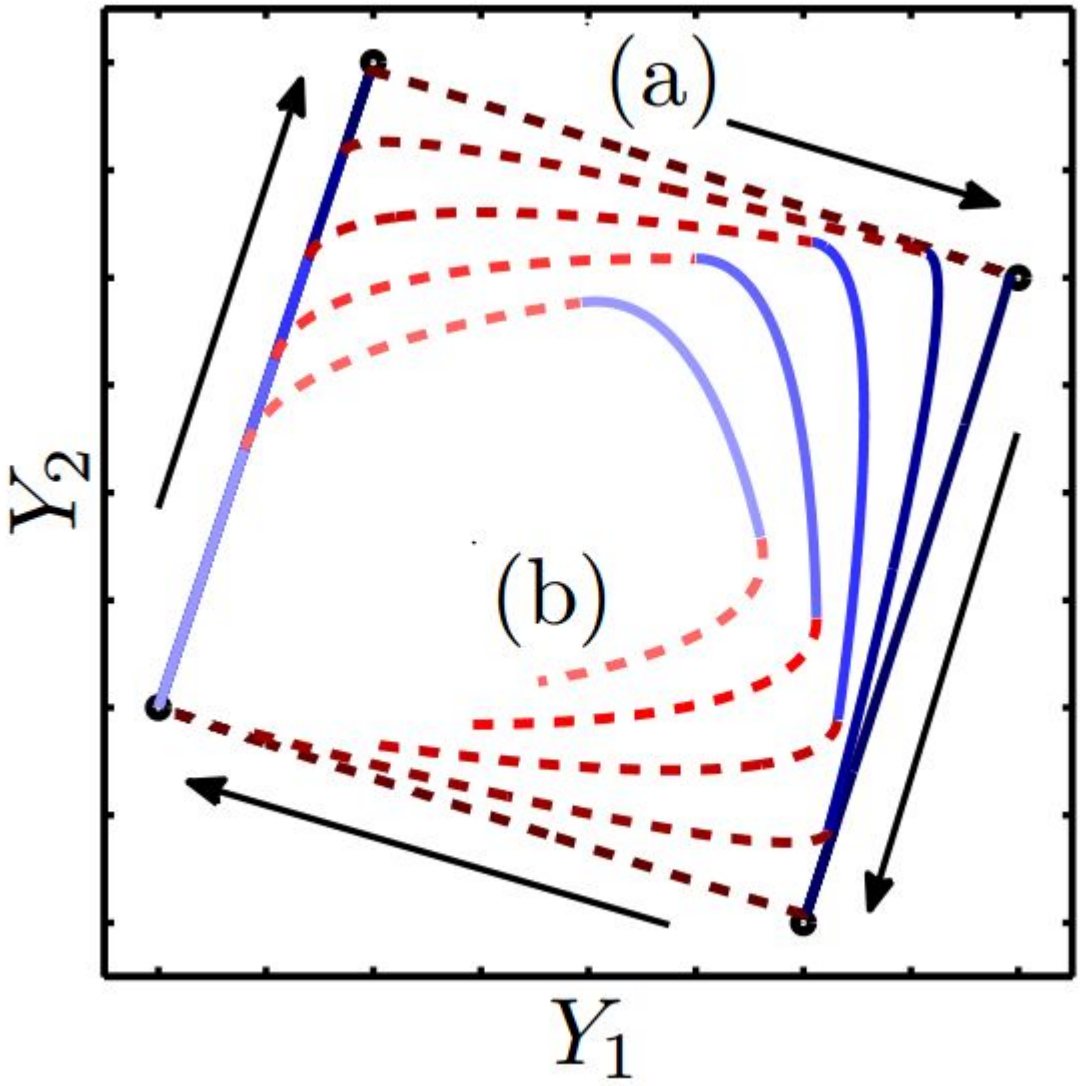


Fig. 8. A sketch of a concept of an associative skill memory as graphical model.

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

3.3. Combination of Movement Primitives



(a) zero velocity and acceleration boundary condition

(b) Velocity and acceleration is equal to the that of the preceding DMP

4. Applications of DMP

4.1. Learning DMPs from Demonstration

4.2. Executing DMPs on the Robot

4.3. Robot Experiment

4.1. Learning DMPs from Demonstration

Sarcos Slave Arm

(1)

(2)

(1) Heiko Hoffmann, URL: http://www.heikohoffmann.de/robots.htm, last access: 19/03/2017
(2) Computational Learning&Motor Control Lab, URL: http://www-clmc.usc.edu/Research/ExperimentalEquipment ,last access: 19/03/2017

4.1. Learning DMPs from Demonstration

Sarcos Master Arm

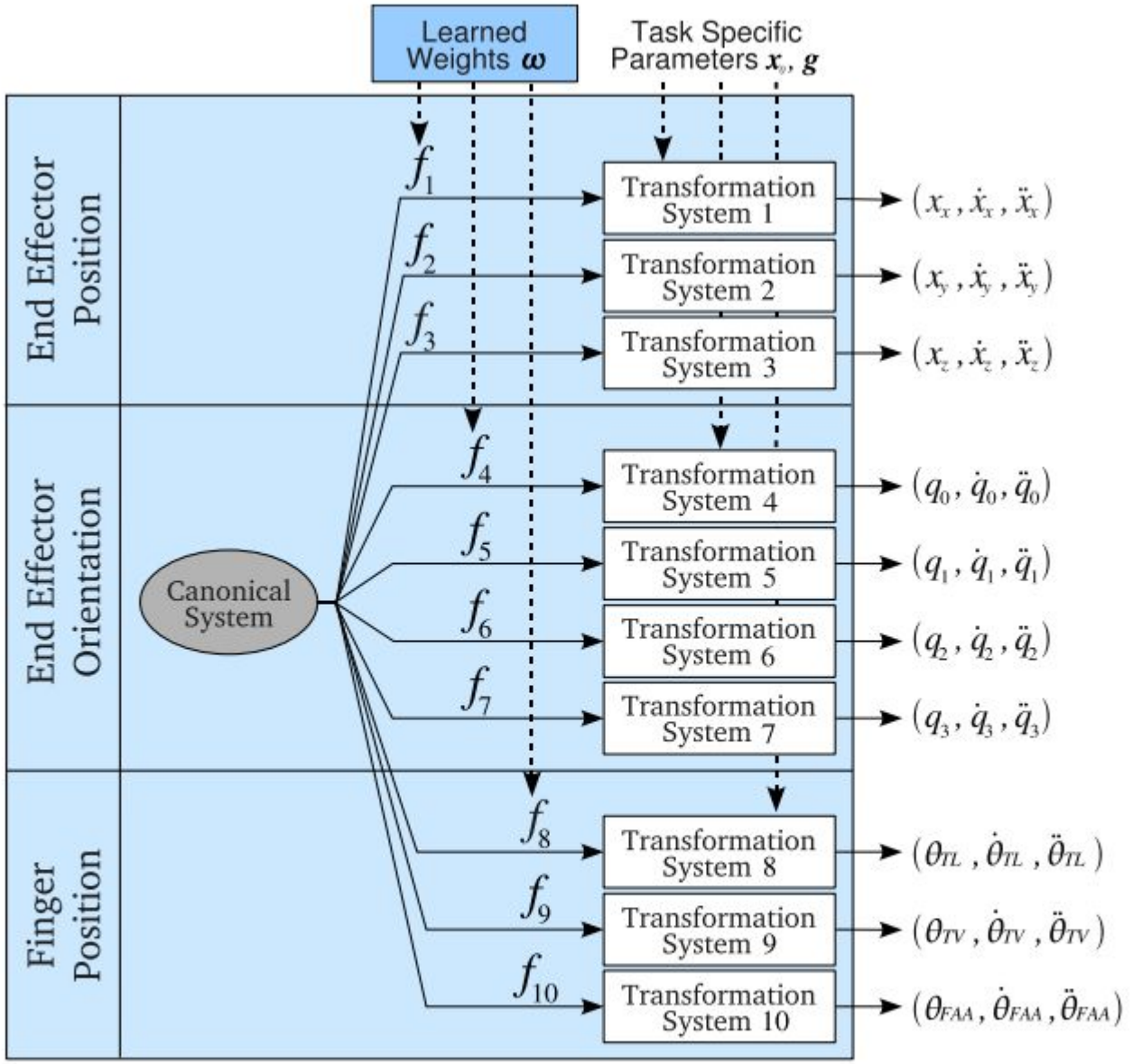
(1) Max Planck Institute for Intelligent Systems, Autonomous Motion , URL: https://am.is.tuebingen.mpg.de/pages/robots last access: 19/03/2017

4.1. Learning DMPs from Demonstration

Demonstration

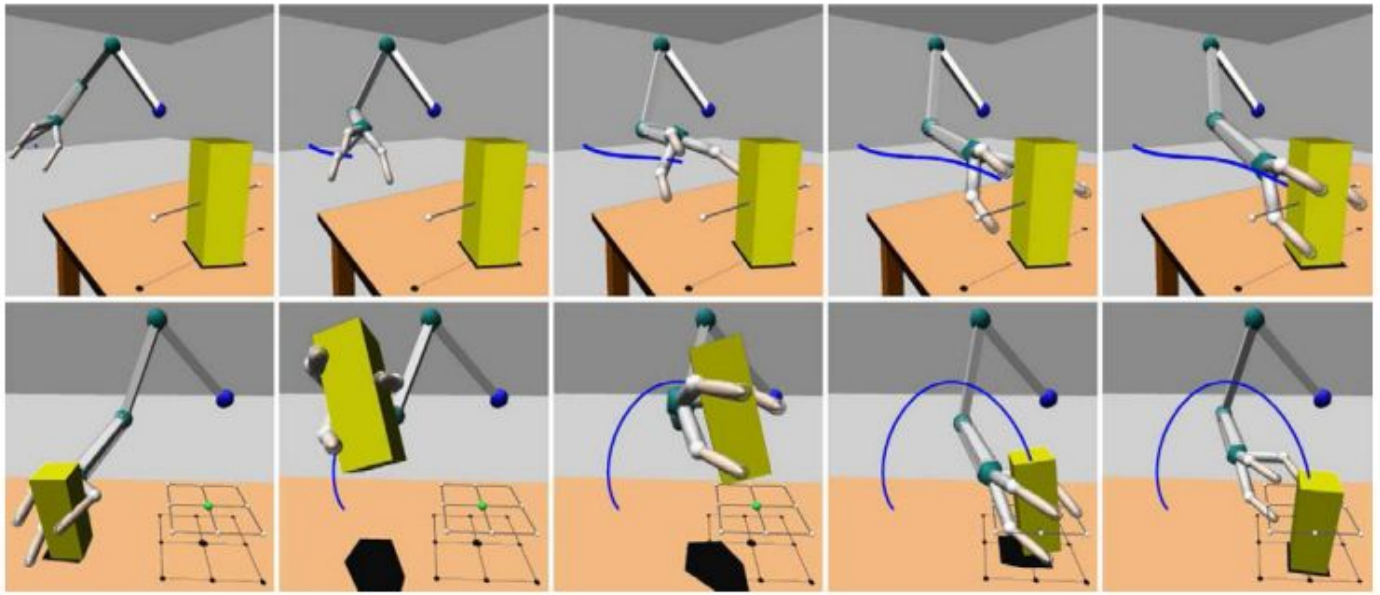
Fig. 8. Sarcos Master arm used to record a human trajectory in end-effector space. Here, the subject demonstrates a pouring movement which after learning the DMP enabled a robot to pour water into several cups (Fig. 12).

4.1. Learning DMPs from Demonstration



4.1. Learning DMPs from Demonstration

Simulation Results



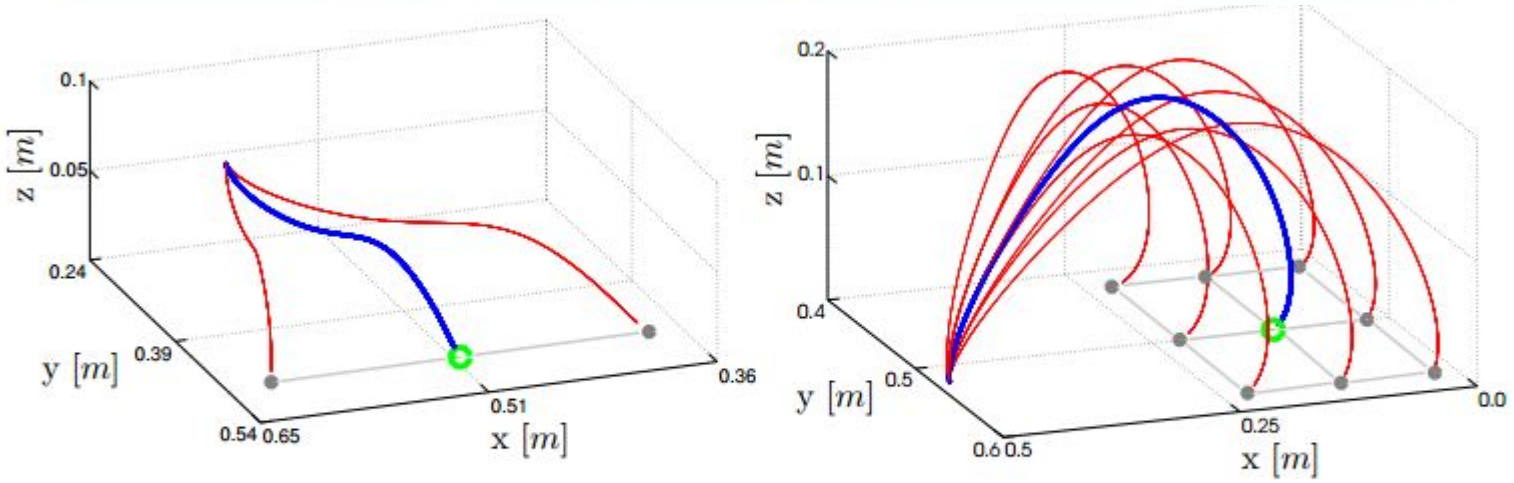


Fig. 10. The desired trajectories (blue lines) from the movements shown in Fig. 9 adapted to new goals (red lines) indicated by the grid.

4.2. Executing DMPs on the Robot

Summary of Robot Movement Execution from DMPs

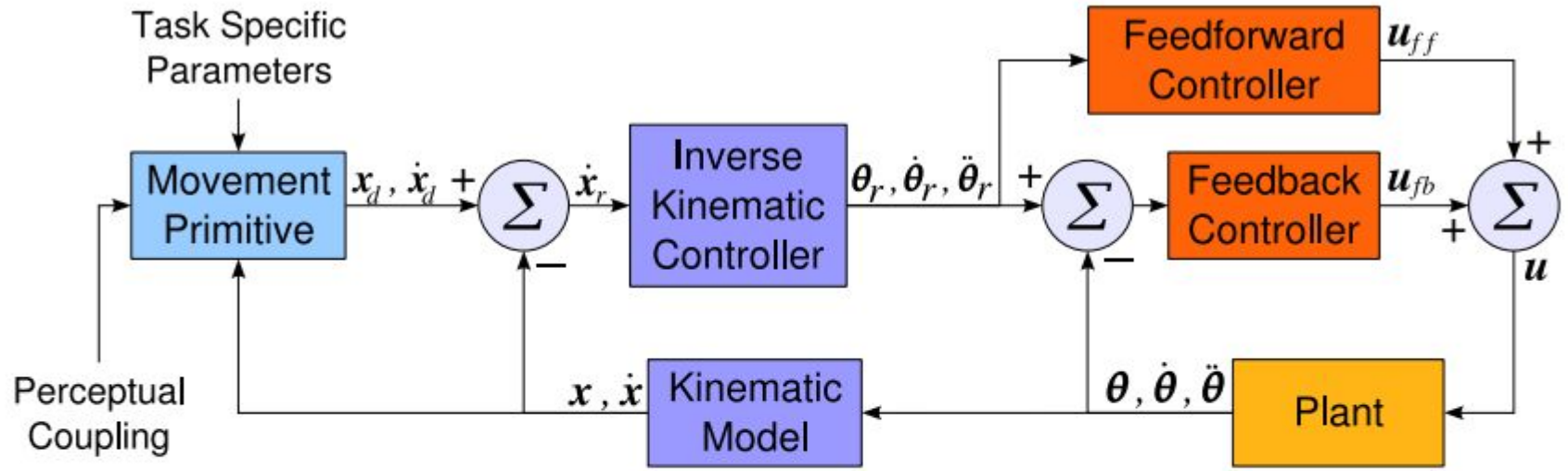
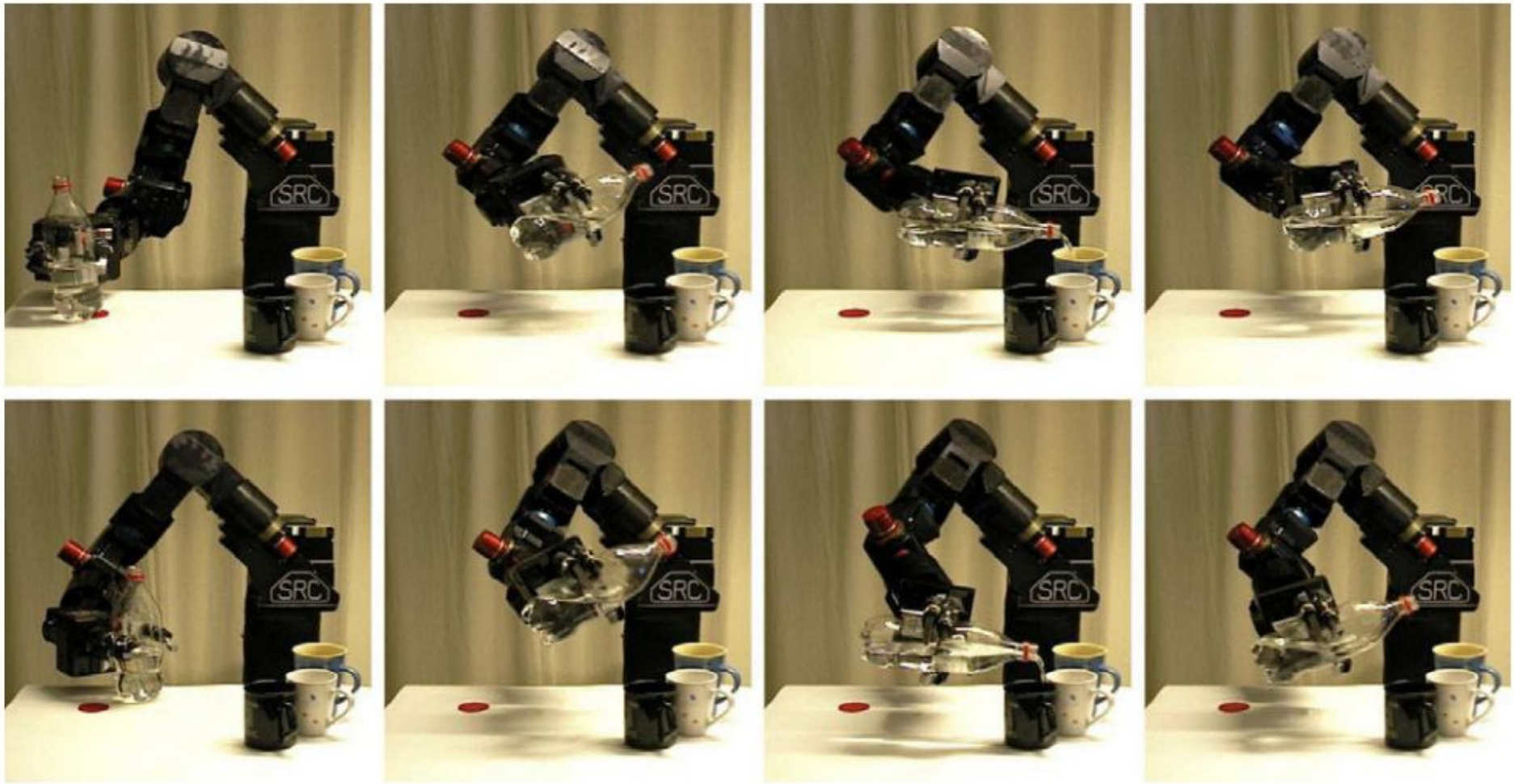


Fig. 11. DMP control diagram: the desired task space positions and velocities are $\boldsymbol{x_d}$, $\dot{\boldsymbol{x}}_{\boldsymbol{d}}$, the reference task space velocity commands are $\dot{\boldsymbol{x}}_{\boldsymbol{r}}$, the reference joint positions, joint velocities, and joint accelerations are $\boldsymbol{\theta}_r$, $\dot{\boldsymbol{\theta}}_r$, and $\ddot{\boldsymbol{\theta}}_r$.

4.1. Learning DMPs from Demonstration

Changing Goal Variable



4.1. Learning DMPs from Demonstration

Changing Goal Variable

4.3. Robot Experiment

Online Adaptation to New Goals

4.3. Robot Experiment

Online Adaptation to New Goals

5. Conclusions

- Robust generalization to new goals
- Human like adaptation
- Automatic obstacle avoidance
- Incorporating sensory feedback
- Associative Skill Memories

The End

QUESTIONS?

URL : http://www.yeniisfikirleri.net/dunyanin-en-ilginc-10-robotu/ , last access: 19/03/2017