

# NODE and Contraction Methods for Dynamics Learning from Human Expert Demonstrations

Tufail Ahmed<sup>1</sup>, Sangmoon Lee<sup>1</sup><sup>a</sup> and Ju H. Park<sup>2</sup><sup>b</sup>

<sup>1</sup>Department of Electronics and Electrical Engineering, Kyungpook National University, Daegu, Republic of Korea

<sup>2</sup>Department of Electrical Engineering, Yeungnam University, Kyongsan, Republic of Korea


**Keywords:** Neural Ordinary Differential Equations (NODE), Learning from Demonstrations (LfD), Dynamic Systems, Imitation Learning, Initial Value Problem, Contraction Theory.


**Abstract:** In this paper, we propose model-free or learning-from-demonstration methodologies for accurately estimating the complex and nonlinear behaviors of dynamic systems such as mobile robots, robotic arm manipulators, and unmanned aerial vehicles (UAVs). Under learning from demonstration (LfD), this study investigates two different approaches: The first proposed methodology is the contraction theory, in which the assigned task demonstration is practically performed by the human expert, who tries to learn and imitate it. On the other hand, the same task learns and imitates by utilizing the neural ordinary differential equations (NODEs) for dynamic systems. Using the concepts of both approaches, we tried to make it possible for the system to pick up on and imitate the shown behavior or demonstration accurately. In dynamics learning, the proposed contraction method utilizes the conceptual framework of the contraction theory, which ensures the motions of dynamic systems that eventually converge to nominal or desired behavior. At the same time, NODE uses the neural network with different configurations of hidden layers, learning rate, nonlinear activation function, and ODE solver. A spiral trajectory is considered a human expert demonstration that is estimated by both methodologies (i) NODE and (ii) contraction theory. For validation purposes, we compared the results of both approaches.

## 1 INTRODUCTION

Learning by demonstration, or LfD for short, is a useful strategy for rapidly enhancing robotic efficiency. It enables robots to gain capabilities by observing what they want to do. Focusing on allowing the robotic device to program by itself, the human operator demonstrates an action to the robot by demonstrating how the operation ought to be performed. Learning action patterns through as few demos as possible is vital, and the quantity of storage required is reduced when taught skills are concisely represented (Khansari-Zadeh, et. al., 2011; Calinon, S., et. al., 2007). It is possible to represent actions from one point to another to ensure all come to an end at a designated spot in state space (Schaal, S., 1999). Simplifying more complex tasks can yield fundamental components of robot automation surveillance: sequences of one point to motions or

modeling between point actions (Kulic, D., et. al., 2008). When an operator directs an autonomous device throughout an activity, it automatically sees the process from its point of view. LfD: While dynamical actions specify how to emulate, one point to another action includes steps made by human experts to solve the problem (Dautenhahn, K., and Nehaniv, C. L., 2002). Robotic trajectories are shown via kinesthetic training to circumvent the matching issue, whereby human observers passively guide the robot along its ideal motion (B., Akgun, and Subramanian, K., 2011). One of the earliest instances of digital summoning taught via examples is dynamical motion primitive concepts (DMP). DMP is used to combine a linear dynamical system and a nonlinear force factor, which is obtained in one demo (Ijspeert, A., et. al., 2013). Poor replication could occur from implementing restrictive stabilization criteria. If one concentrates too heavily on precise

<sup>a</sup> <https://orcid.org/0000-0001-8252-952X>

<sup>b</sup> <https://orcid.org/0000-0002-0218-2333>

reproduction, one may become less resilient to disruptions, which could eventually cause deviation. There are instances where precision suffers because reliability is given precedence over efficiency. It might not be the most effective solution if the intricate dynamics underneath is fascinating. Finding the right balance between studying the intricacies of kinetics and maintaining stabilization in a system that changes is difficult. Enabling robots and autonomous devices to perform tasks efficiently within dynamic settings and learning from demonstrated actions (LfD) is a critical capability. Highly complex, non-linear trajectories like spirals are a common challenge for conventional LfD techniques. To enhance the prediction and reproduction of these trajectories, this work explores the application of contraction theory and neural networks with ordinary differential equations (neural ODEs). We examine such approaches' theoretical underpinnings, real-world applications, and comparative effectiveness.

The paper's remaining structure is as follows. In the next section, the problem formulation of both proposed methods (node and contraction theory) is presented. In Section 3, we present the neural ODE and contraction theory learning framework for learning the dynamics of the dynamic systems. In Section 4, we performed the simulation and demonstrated the effectiveness of both the proposed methodologies. Finally, we concluded the paper with a summary and future research direction.

Table 1: Notations Used in This Paper.

Symbols	Meanings
$x(t)^*$	Nominal trajectory
$\dot{x}$	Rate of change of the state
$f_{\theta}^{\wedge}(x)$	Learn or estimated nonlinear function
$\hat{x}(t_i)$	Estimated current state
$\hat{x}(t_{i+1})$	Estimated future state
$t_0$	initial time
$t_i$	final time
$g(x(t_i))$	Learn nonlinear function in NODE
$f_{true}(x, t)$	Learn nonlinear function in contraction
$\delta x$	virtual displacement in trajectories
$\lambda_{max}$	maximum eigen value of Jacobian

## 2 PROBLEM FORMULATION

We formulate the robotic system's state-to-state motions as an autonomous dynamic system with a nominal unknown trajectory  $x(t)^*$  made up of N demonstration data points. A system's state can be regarded as each demonstration data point. When a trajectory moves with noise or disturbance, the autonomous dynamic system is

$$\dot{x} = f_{\theta}^{\wedge}(x) + \varepsilon \quad (1)$$

Where  $f$  is the learnable nonlinear function, and the additive term represents the noise in the system. The system without noise can be represented by the below-mentioned equation.

$$\dot{x} = f_{\theta}^{\wedge}(x) \quad (2)$$

We considered the single spiral trajectory as the demonstration of the expert and try to estimate it accurately using the neural ode and contraction method. In this work, we use supervised learning method for the given demonstration data, N. The objective is to learn the nonlinear spiral function accurately with the minimum loss value and try to estimate the desired trajectory. The prediction of the nonlinear function can be achieved by using the below mentioned NODE equation.

$$\hat{x}(t_{i+1}) = \hat{x}(t_i) + NODE(f_{\theta}x(t), t, \theta), x(t_0), t_0, t_i), \quad (3)$$

The objective is to minimize the difference between the desired trajectory and the learned nonlinear trajectory to find the optimal parameter values which reduce the loss, we considered the following loss minimization for the NODE.

$$\min_{\theta} \frac{1}{NT} \sum_{i=1}^N \sum_{i=1}^T \|x_i(t_i) - g(x(t_i))\|_2^2 \quad (4)$$

Above mentioned equation expresses the loss function. The parameters  $\theta$ , and values continuously update until the loss values reach the least minimum, or in other words the difference between the predicted and observed state becomes negligible.

### 2.1 Forward Propagation

$$x(t_k) = ODEsolve(f(x(t), t, \theta), x(t_0), t_0, t_k), \quad (5)$$

## 2.2 Back-Propagation

$$\begin{bmatrix} x(t_0) \\ \frac{\partial L}{\partial x(t_0)} \\ \frac{\partial L}{\partial \theta} \end{bmatrix} = \text{ODESolve} \left( \begin{bmatrix} f(x(t), t, \theta) \\ \frac{\partial f(x(t), t, \theta)}{\partial x} \\ \frac{\partial f(x(t), t, \theta)}{\partial \theta} \end{bmatrix}, \begin{bmatrix} x(t_k) \\ \frac{\partial L}{\partial x(t_k)} \\ 0_\theta \end{bmatrix}, t_k, t_0 \right), \quad (6)$$

The implementation of the NODE architecture can be considered with a basic neural network that is fully connected and possesses one hidden layer.

$$\frac{dx(t)}{dt} = \sigma(W_2 \cdot \sigma(W_1 \cdot x(t) + b_1) + b_2), \quad (7)$$

where,  $x(t)$ , is the particular system state,  $W_1$  and  $W_2$  are the matrix of weights,  $b_1$  and  $b_2$  are the bias vectors, and  $\sigma$ , is the nonlinear activation function. The numerical technique Runge-Kutta is used to solve the ODE to determine the system's state at any given moment. we can reconfigure the NODE network architecture by modifying the hidden layers and learning rate of the neural network

$$\begin{aligned} t_{k+1} &= t_k + h \\ s_1 &= f(x(t_k), t_k) \\ s_2 &= f(x(t_k) + \frac{h}{2}s_1, t_k + \frac{h}{2}) \\ s_3 &= f(x(t_k) + \frac{h}{2}s_2, t_k + \frac{h}{2}) \\ s_4 &= f(x(t_k) + hs_3, t_k + h) \\ x(t_{k+1}) &= x(t_k) + \frac{h}{6}(s_1 + 2s_2 + 2s_3 + s_4) \end{aligned} \quad (8)$$

In the contraction theory for the learned models. The problem of system identification can be represented as

$$\dot{x} = f_{true}(x, t) \quad (9)$$

$$\dot{x} = f_{true}(x, t) = f_L(x, t) + (f_{true}(x, t) - f_L(x, t)),$$

The true function is unknown, approximated by the contracting learning model.

## 3 MAIN RESULTS

Dynamic systems are mathematical methods that depict or learn the evolution of a system's dynamics across time and are capable of understanding complex systems behaviours. The equation of an autonomous dynamic system is:

$$\frac{dx(t)}{dt} = f(x(t), t), \quad (10)$$

where,  $x(t) \in \mathbb{R}^n$  are the system state,  $t \in T$ , is time interval T, and  $f: \mathbb{R}^n \times \mathbb{R} \rightarrow \mathbb{R}^n$  a vector field of nonlinear function which defines the dynamics of the system and which needs to be learned. If we look around mobile robots, robotic arm manipulator, UAVs, and many other industrial systems possess the dynamic behaviours which are nonlinear, complex and hard to learn. To tackle the nonlinear and complex behaviours of such systems we considered spiral trajectory for our work as the nonlinear and complex dynamic system which can be express in mathematical form mention below.

$$\begin{aligned} x &= r \cos(\theta) \\ y &= r \sin(\theta) \end{aligned} \quad (11)$$

We learned the considered nonlinear spiral trajectory with the proposed method of Neural ordinary differential equations (NODE) which is

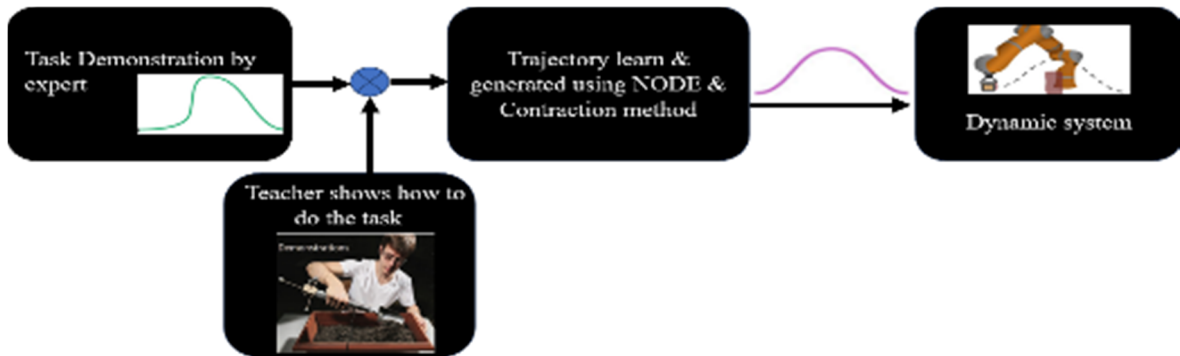


Figure 1: Overall view of proposed system.

efficient in depicting time series data and dynamical systems. To estimate “ $f$ ”, the NODEs employ neural networks described in equation.

$$\frac{dx}{dt} = \dot{x} = f_{\theta}(x(t_0), t) \quad (12)$$

The above-mentioned equation represents the mathematical form of the neural ode. The initial state of learnable nonlinear dynamic system “ $f_{\theta}$ ”, is given by the  $x(t_0)$ . The nonlinear spiral function states estimations acquired by the help of the neural network, which optimized the learned parameters  $\theta$  for all given states. The optimized parameter reduced the loss values between the encoded hidden states and the predicted

$$L(\theta) = \sum_i^T \|x_i - g(x(t_i))\|^2 \quad (13)$$

In forward propagation it solves the ode and in backward propagation it will compute the gradient which updates the weights of neural network to obtained the optimal values of learnable parameters. The neural ode architecture implementation can be represent as

$$\frac{dx(t)}{dt} = \text{relu}(W_2 \cdot \text{relu}(W_1 \cdot x(t) + b_1) + b_2), \quad (14)$$

The visualization of the proposed NODE framework is given below

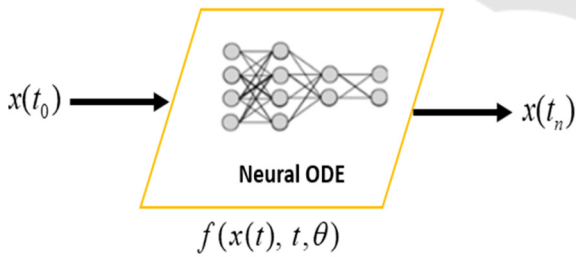


Figure 2: NODE framework.

The other proposed method based on contraction theory use to find the differential dynamics of the spiral nonlinear and complex systems by considering the contraction metric with uniform positive definite matrix. It provides the incremental exponential stability for the different trajectories started from the different points. The mathematical expression is express as

$$\dot{x} = f(x, t) \quad (15)$$

can be rewrite in the differential form

$$\delta \dot{x} = \frac{\partial f}{\partial x}(x, t) \delta x \quad (16)$$

Where,  $\delta x$  is the virtual displacement between two neighbouring trajectories as the virtual displacement decreases between two trajectories and desired trajectory if they become one single trajectory we can say that the system accurately estimated the demonstrated trajectory and system exponentially converge to the stability. If there exist a uniformly positive definite metric  $M(x)$ , and satisfy the following conditions, we can say system is contractive.

$$\frac{d}{dt}(\delta x^T M \delta x) \leq -\lambda_{\max} \delta x^T M \delta x \quad (17)$$

$$\|\delta x(t)\| \leq \|\delta x(0)\| e^{-\lambda t} \quad (18)$$

## 4 SIMULATION RESULTS

In this section, we discussed the performed simulation and results obtained by using the both methodologies of NODE and contraction.

### 4.1 Contraction Method

The results in Figure 3 show the difference between the true or nominal trajectory and learned trajectories which started from two different initial points and tried to converge to the desired trajectory but due to less stronger contraction term, the learned could not reach the ideal trajectory accurately.

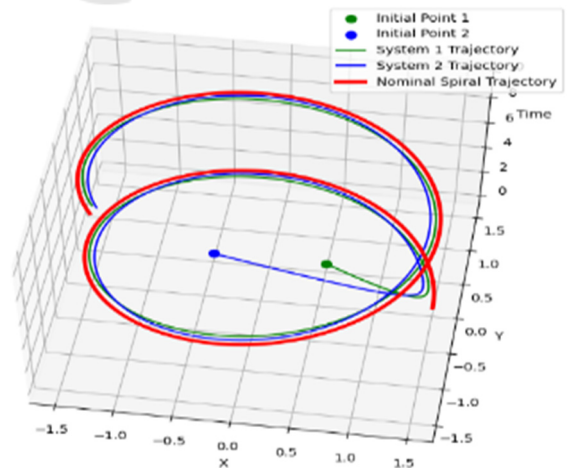


Figure 3: Less contraction system.

In Figure 4 we can see the learned trajectory approximate the true trajectory quite accurately and the difference between the true and learned trajectories are negligible and show the strong contractive behaviour with the strong contractive term considered during the learning.

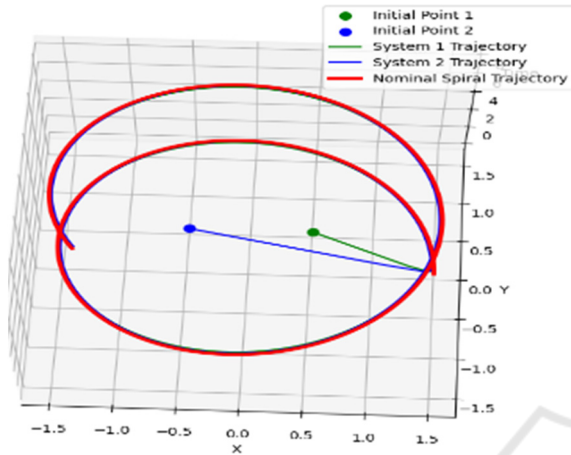


Figure 4: Strong contraction system.

the true and learned trajectories minimize this happened due to strong contraction term considered during the learning.

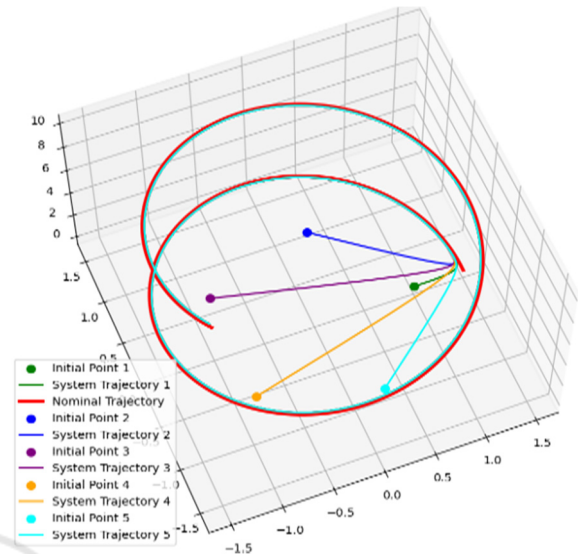


Figure 6: Strong contraction system.

In Figure 5 we considered the five different Trajectories which started from the different initial points, in the first case we considered the minimal contractive term and analysed whether the learned trajectories reached the target trajectory accurately but unfortunately, could not make it and showed the huge difference between the learned and true trajectories this is because the less strong contraction term used during the learning.

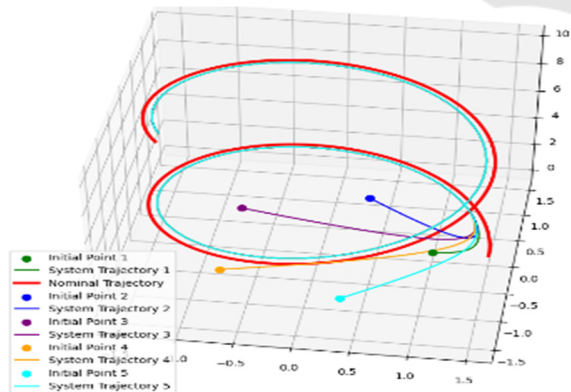


Figure 5: Less contraction system.

In the above Figure 6 five different trajectories started from the five different initial points, the results show that all five trajectories converge into one single trajectory as time goes on which proves the system is highly contractive and stable. The difference between

## 4.2 Neural Ode's

In the Figure 7 the Spiral trajectory is given as the demonstration of the neural ordinary differential network. The red dot line is the spiral demonstration and a blue solid line in the estimation of the observed values. we can see from the results the NODE based architecture provide the accurate estimation but when the initial value changes its performance, degrade due to initial value problem.

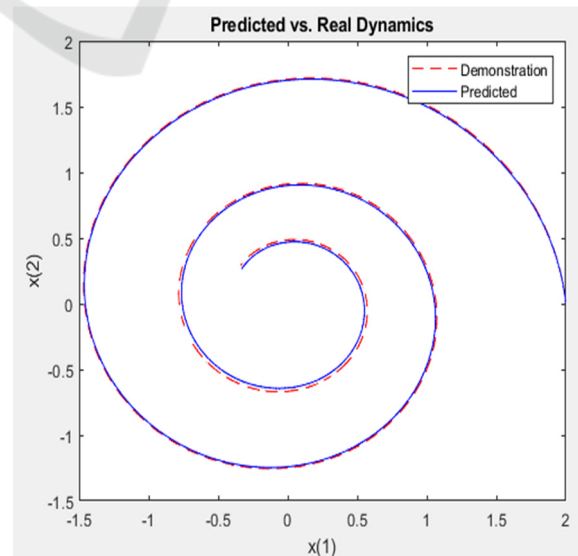


Figure 7: NODE based dynamics learning.

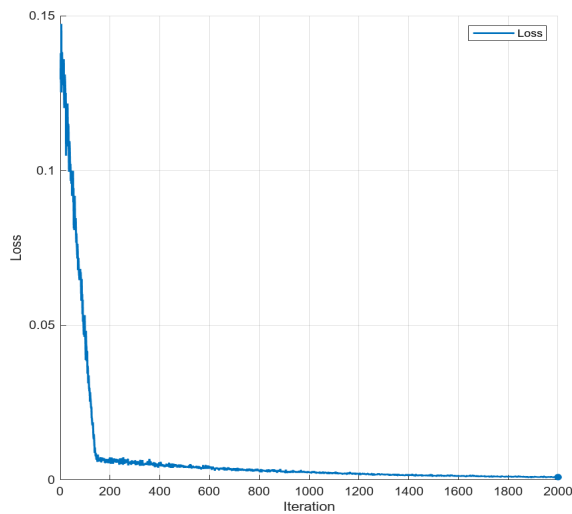


Figure 8: Learning Loss.

Figure 8 shows the learning of the NODE framework over the iteration of 2000.

## 5 DISCUSSIONS

In this work, a comparative analysis conducted of two different methods for the task of desired trajectories estimation of dynamic systems given by the demonstrations. Simulation performed on the spiral trajectory of dynamic systems and evaluated based on the performance metrics mentioned below in table 2.

The NODE method shows better accuracy for trajectory estimation, in an unperturbed environment but faces stability issues in a perturbed environment.

Table 2: Performance metric of NODE and contraction methods.

Metric	Neural ODE	Contraction method
Accuracy	More accurate	Accurate
Robustness	Moderate	High
Convergence Rate	Fast	Faster
Computational Efficiency	Efficient but expensive	Efficient
Stability	Sensitive to perturbation	Robust against perturbation
Real Time Application	Good after training	Excellent

On the other hand, contraction method is less accurate within an acceptable margin of accuracy but robust and stable in trajectory estimation both in perturbed and unperturbed conditions. Furthermore, the contraction method can provide fast exponential convergence as compared to NODE even in the external disturbance. Contraction method is more efficient both in computational efficiency and real-time application implementations. These findings show that both methods supersede each other in different performance metrics. The selection of these methods highly based on the type of applications.

## 6 CONCLUSIONS

The presented work tried to cover the dynamic learning of dynamic systems by incorporating the learning from the demonstration method. We considered the spiral trajectory as the expert demonstration data for any particular actions of the dynamic system and used two different methodologies NODE and Contraction theory to learned these demonstration actions, NODE based learning provides better accuracy and flexibility but on the other hand, it is sensitive to the initial value and demand long training time, require high computational cost and lack of robustness under the perturbed conditions for higher dimensional systems. While the contraction theory provides the higher stability, robust to the perturbation, produce good response on the initial value problem, computationally efficient. These two methods show the trade-off between robustness and flexibility, selection of these methodologies based on the demand of the applications. In the future, the proposed work will extent for the practical implementation on robotic arm manipulator or mobile robot dynamic systems with the insertion of obstacles and perturbation. In this work correction term or control term was not considered for the unseen data and spurious attractor problems in future we considered solving such problems with the implementation of an appropriate control method, stability is also not considered we plan to incorporate the stability and safety constraints in the future work. Furthermore, we planned to expand the proposed work by considering the higher-dimension problem.

## ACKNOWLEDGEMENTS

This work was supported in part by the National Research Foundation of Korea (NRF) through

the Korea government, Ministry of Science and ICT under the grant, RS-2024-00350118, 2019R1A5A8080290 and supported by Institute of Information and Communication Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (IITP-2024-RS-2022-00156389).

## REFERENCES

- Khansari-Zadeh, S. M., and Billard, A. (2011). Learning stable nonlinear dynamical systems with gaussian mixture models. In *IEEE Transactions on Robotics*, 27(5):943–957.
- Calinon, S., Guenter, F., and Billard, A. (2007). On learning, representing and generalizing a task in a humanoid robot. *Transactions on Systems, Man, and Cybernetics*, 37(2):286–298.
- Schaal, S. (1999). Is imitation learning the route to humanoid robots? *Trends in Cognitive Sciences*, 3(6): 233–242.
- Kulic, D., Takano, W., and Nakamura, Y. (2008). Incremental learning, clustering and hierarchy formation of whole-body motion patterns using adaptive hidden Markov chains. *The Int. Journal of Robotics Research*, 27(7): 761–784.
- Dautenhahn, K., and Nehaniv, C. L. (2002). *The agent-based perspective on imitation*. Cambridge, MA, USA: MIT Press, pp. 1–40.
- B., Akgun, and Subramanian, K. (2011). Robot learning from demonstration: Kinaesthetic teaching vs. teleoperation.
- Ijspeert, A., Nakanishi, J., Pastor, P., Hoffmann, H., and Schaal, S. (2013). Dynamical Movement Primitives: learning attractor models for motor behaviours, *Neural Computation*, 25(2):328–373.