Advanced Robotics Assessment Two - Learning from Demonstration

Eden Attenborough
School of Computer Science
University of Lincoln
Lincoln, United Kingdom
28238148@students.lincoln.ac.uk

***Abstract*—In this assessment we review and evaluate three papers in the field of learning from demonstration of manipulators. We also present mathematical formulation of Dynamic Movement Primitives and Stable Estimator of Dynamical Systems, and test their implementations on previously created datasets.**

***Index Terms*—Learning from Demonstration, Dynamic Movement Primitives, Stable Estimator of Dynamical Systems, Gaussian Mixture Models, Gaussian Mixture Regression, Time-invariant Dynamical Systems**

I. Introduction

Learning from demonstration (LfD) in the context of robotics means robots acquiring new skills by learning from a human. Advantages of this approach include the east of introducing complex skills, and making the programming of robots simpler; that is, they do not need to be programmed by an expert. LfD has often been used for the control of manipulators, which are very important in the field of manufacturing, which increases profitability by relying on humans less; and in the field of healthcare assistance robots, for instance surgical manipulators. Surprisingly, LfD has also been used in the field of mobile robotics- for instance controlling the trajectories of autonomous aerial vehicles; and crucially, for locomotive robots, enabling bipedal robots to walk [1].

In this document, Section II provides a brief review of three learning from demonstration system and compares them. Section III outlines the mathematical formulation of Dynamic Moment Primitives (DMPs) and Stable Estimator of Dynamical Systems (SEDSs). Finally, Section IV describes and implementation of these algorithms on some given datasets, and discusses their hyper-parameters.

II. Literature Review

[3] pertains to a soft-robotic manipulator. The field of animal-based soft robotics has considerable advantages, such as less precision being required and better safety when dealing with fragile objects, but creates additional challenges that are discussed here. The manipulator is made from an ionic polymer-metal composite (IPMC), a flexible and soft manipulator. Notably it has many more degrees of freedom than a typical manipulator. Demonstrations were difficult- the manipulator was controlled through teleoperation of a single joint at a time, an un-ideal movement, with a camera observing the manipulator at certain points, a very slow process. The demonstration was determined from this observation, not from the teleoperation. This requires a mapping task, which seems unclear how they achieved. What is clear, is that the demonstrations were fed into a Savitkzy-Golay Filter pre-processing. The demonstrations were encoded with Gaussian Mixture Models. The parameters from the GMMs were estimated using an expectation maximization algorithm, which was initialized by a k-means clustering algorithm. Using the k-means clustering as a starting point for the expectation maximization algorithm is a good idea, since that algorithm is sensitive to its initialization. Paths were then generalized with Gaussian Mixture Models. Unfortunately, their manipulator could not guarantee a solution due to a lack of generalizability in the Gaussian Mixture Models. Moreover, it only worked in a static environment, in which the initial location is known.

[2] is a learning from demonstration algorithm proposal designed for a manipulator, called an augmented *Joint-space Task-oriented Dynamical System* (JT-DS)): It computes a motion in joint-space, that provably converges to a task-space target; that ‘is formulated in such a way that joint-space motions are learned from demonstrations as synergies; and can transit through kinematic singularities’. These learned movement behaviour synergies are modulated through joint space with a Linear Parameter Varying system, for who’s parameters are determined by reducing the dimensionality of the joint space. They evaluated a number of dimensionality-reducing algorithms to generate these embeddings, and selected Kernel Principal Component Analysis with the Radial Basis Function as the kernel. The number of local synergy regions, and the activation parameters for the weighting of these regions, are determined by fitting a Gaussian Mixture Regression model from the demonstrations.

[4] is a learning from demonstration manipulator designed to assist children with cerebral palsy. Physical therapy has been proven to help children with cerebral palsy with their symptoms and improve their quality of life, but is an expensive and repetitive task which is ideal to be conducted by robots. The implementation consists of a manipulator that assists and corrects the child’s movement of an object between two locations, a ‘pick and place’ task. The demonstration is kinaesthetic, but not done on the manipulator itself. Instead, a helper moves a second, ‘slave’ manipulator to initially help move the main, ‘master’ manipulator. Consequently, it is therefore

TABLE I: Table comparing the differences between the reviewed learning from demonstration systems

Implementation	What is learned	Demonstration Type	Methods used	Stability	Convergence
[2]	Low level control inputs	Kinaesthetic teaching	Dimensionality reduction with PCA; GMMs	Guaranteed, with Lyapunov	Guaranteed
[3]	Low level control inputs	Observation of keypoints with a camera. Demonstrations difficult due to the soft manipulator's movements need to be pre-planned by the demonstrator	Gaussian Mixture Models with optimal parameters estimated with Expectation Maximization algorithm, with initialization from K-Means clustering; Gaussian Mixture Regression	Not guaranteed	Not guaranteed
[4]	Low level control inputs	Kinaesthetic teaching with a mapping between two manipulators	Gaussian Mixture Models and Gaussian Mixture Regression, unclear how parameters are selected	Unclear	Unclear

required to have some sort of force-feedback system, to provide haptic feedback about the relative movement intentions of the two manipulators, which the researchers implemented. Their is an additional problem mapping the movements of the two manipulators, since they were not the same model. The researchers used a simple PID control loop, but it is unclear how well this will work with manipulators of different capabilities. The work uses a Gaussian Mixture Model to model movements and a Gaussian Mixture Regression model for movements, however, the work is unclear about the stability and convergence of the system; the main focus of the work seems to me the mapping and haptic feedback system.

In conclusion, we have reviewed three works in the context of learning from demonstration. Two of the works used novel demonstration methods, and 'traditional' learning from demonstration algorithms, and the other proposed a novel learning from demonstration algorithm. A table showing different features of these works is shown in Table I. In the literature, there is a section of works that make adaptions to already existing works. For instance, [5] makes an adaption to the Dynamic Movement Primitive algorithm for encoding motion data, using convolutional neural networks for the forcing term in that algorithm, and using a camera for the demonstration. This is a novel approach, but its usefulness is perhaps limited by the amount of training data required. CNNs require a lot of training data, and in this context the training data is derived from demonstrations, which are real actions in the world; consequently, it may take a lot of demonstrations to achieve anything accurate. The researchers are aware of this, and they suggest that a CNN would only be used as a starting point for some re-enforcement learning system.

Moreover, in our research we noticed some patterns of note. For example, kinaesthetic demonstrations seem to be generally used for manipulators and observation-based demonstrations were generally used by mobile robots. Gaussian Mixture Models seemed to the the most common way of modelling movements.

III. Dynamic Movement Primitives and Stable Estimator of Dynamical Systems

A. Dynamic Movement Primitives

Dynamic Movement Primitives (DMPs) model movements. For each joint, the acceleration $\ddot{q}$ is provided, given the speed $\dot{q}, q, s$:

$$\ddot{q} = I(q, \dot{q}) + f(s) \tag{1}$$

It is a combination of a non-linear and linear system:

$$I(q, \dot{q}) = k_{\text{gain}}(g - d) - d_{\text{gain}}\dot{q} \tag{2}$$

A simple PD controller from control theory, which guarantees convergence to the goal; g. $f(s;\theta)$ is the forcing term, the non-linear element; and s is the time. The forcing term, is represented by:

$$f(s) = \frac{\sum_{i=1}^{N} w_i h_i(s; \mu_h, \sigma_h^2)}{\sum_{i=1}^{N} w_i} s(g - q_0) \tag{3}$$

A weighted sum of radial basis functions, where:

$$h_i(s; \mu_h, \sigma_h^2) = \exp(-\frac{(s-\mu_h)^2}{2\mu_h^2})$$

The vector **w** is a hyper-parameter, derived from a machine learning algorithm. N is simply the number of radial basis functions.

Then, it can be trained, given a teaching trajectory $T, \ddot{q}_d^{1:T}, \dot{q}_d^{1:T}, q_d^{1:T}$, and a target q_0; we first calculate the forcing term:

$$f_d = \ddot{q}_d - I(q_d, \dot{q}_d) = \ddot{q}_d - (k_{\text{gain}}(g - d) - d_{\text{gain}}\dot{q}) \tag{4}$$

Then for each radial basis function, the locally weighted quadratic error criterion is minimized:

$$J_i = \sum_{t=1}^{T} h_i(s)(f_d(s) - w_i q_t(g - q_0))^2 \tag{5}$$

This gives the result:

$$w_i = \frac{r^T \Gamma_i f_d}{r^T \Gamma_i r}$$

The parameter of the number of radial basis functions is critical, there is a trade-off between complexity and accuracy. This is discussed more in Section IV.

B. Stable Estimator of Dynamic Systems

At the core of Stable Estimator of Dynamic Systems (SEDSs) are Gaussian Mixture Models (GMMs) and Gaussian Mixture Regression (GMR). Gaussian Mixture Models are an alternative method of modelling joint movements. This document does not provide the mathematical basis of GMMs and GMR, since there is not enough space. The mathematical basis for SEDS can be described the following way. Given a demonstration $\mathbb{D} = \{\xi_1, \dot{\xi}_1, \xi_2, \dot{\xi}_2, ..., \xi_n, \dot{\xi}_n\}$, the objective is to 'learn' a mapping $f()$ like $\dot{\xi} = f(\xi)$. Using Gaussian Mixture Regression:

$$p(\dot{\xi}|\xi) = \sum_{n=1}^{N} o_n(\xi) \mathcal{N}(\dot{\xi}|\xi; m^n_{\dot{\xi}|\xi} \Sigma^n_{\dot{\xi}|\xi}) \tag{6}$$

$$\dot{\xi} = f(\xi) = \sum_{n=1}^{N} h_n(\xi) A^n \xi + b^n$$

Where:

$$A^n = \Sigma^n_{\dot{\xi}\xi} \frac{1}{\Sigma^n_{\dot{\xi}\xi}}$$

$$b^n = m^n_{\dot{\xi}} - A^n m^n_{\xi}$$

The parameters of the Gaussian Mixture Model can be estimated by solving a minimization problem, minimizing the mean squared error:

$$min_\theta J(\theta) = \frac{1}{\tau} \sum_{n=1}^{N} \sum_{t=0}^{T^n} \|\dot{\xi}^{t,n}_{\text{pred}} - \dot{\xi}^{t,n}_{\text{demo}}\|^2 \tag{7}$$

Alternatively the negative log-likelihood can be used. Either way, they are subject to the SEDS constraints:

$$b_k = -A_k \xi^\star$$

$$A_k + (A_k)^T \prec 0$$

$$\Sigma_k \succ 0$$

$$0 \le o_k \le 1$$

$$\sum_{k=1}^{K} o_k = 1$$

Where k is the number of gaussian components, $\xi^\star$ is the desired stability point, and o_k is the mixture coefficient.

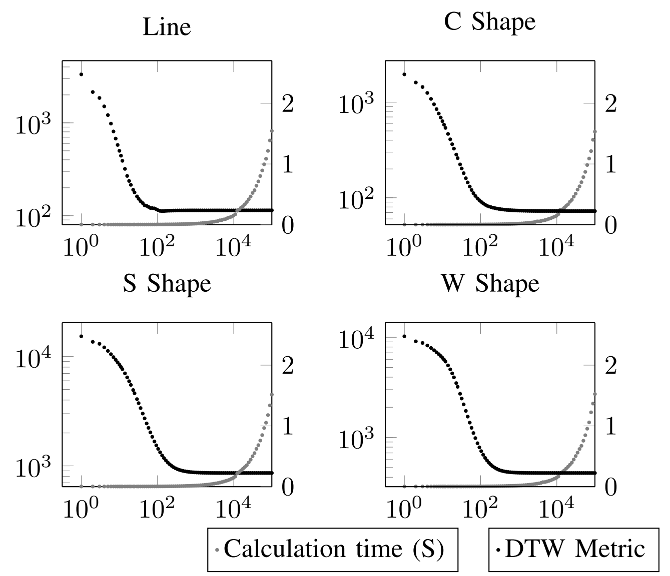

Fig. 1: Plots showing how DMP distance to the original converges with the number of radial basis functions used. Since there was a different number of data points for the estimated path and the original path, Dynamic Time Warping (DTW) [6] distance was used (left y axes). Time taken for the DMP to roll out was used as a metric of complexity (right y axes). All axes (besides time taken) are logarithmic.

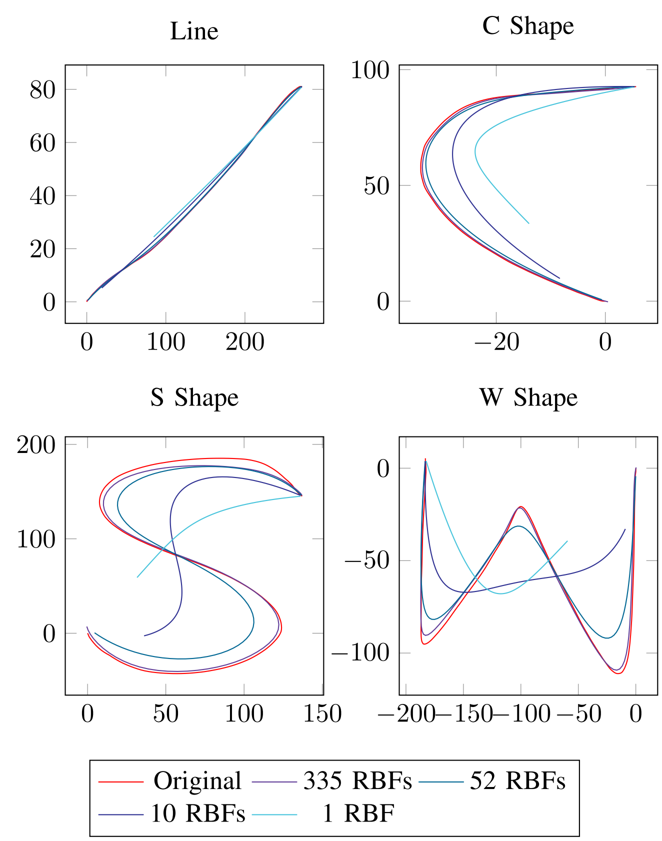

Fig. 2: Plots showing how DMP trajectories are affected by the number of Radial Basis Functions

IV. Implementation and Evaluation

A. Dynamic Movement Primitives

A script was written to apply Dynamic Movement Primitives (DMPs) to the provided datasets. An experiment was run to see how the number of Radial Basis Functions (RBFs) affected the trajectories compared to the original. A script was created that executed DMP rollouts with a varying number of RBFs, from 1 to 100000, with a logarithmic range between them. As a metric of complexity, the time taken to compute the DMP rollout was recorded. It is difficult to compare the original trajectory with the computed trajectory. If there was the same number of points for both, Root Mean Squared Error could be used. However there was not an equal number of points generated; consequently another method had to be used. We selected the *Dynamic Time Warping* algorithm [6], which is an ideal metric for this task. The euclidean distance was used for the distance function. The results for this experiment are shown in Figure 1. Plots showing how the number of RBFs affects the trajectories is shown in Figure 2. The speed at which the DTW distance converges depends on the complexity of the trajectory. A greedy algorithm to find an optimal number of RBFs could be to iteratively keep adding more until the distance difference between iterations reached a certain value.

B. Stable Estimator of Dynamical Systems

A MATLAB implementation of SEDS was preformed upon the given dataset. Experiments were run to see how the number of gaussians parameter affects the similarity from the original. As before, DTWs [6] were used as a metric of distance from the original. The results of these experiments are shown in Figure 3. Unlike with DMPs, there is no linear or exponential relationship between the hyper-parameter and its efficacy. Consequently, it is not possible, as it is with DMPs, to use an elbow point analysis algorithm to estimate the ideal parameter value. Instead, a method similar to [3] can be used, which use the result of a clustering algorithm as the starting point. [3] used K-Means, we experimented with using DBScan.

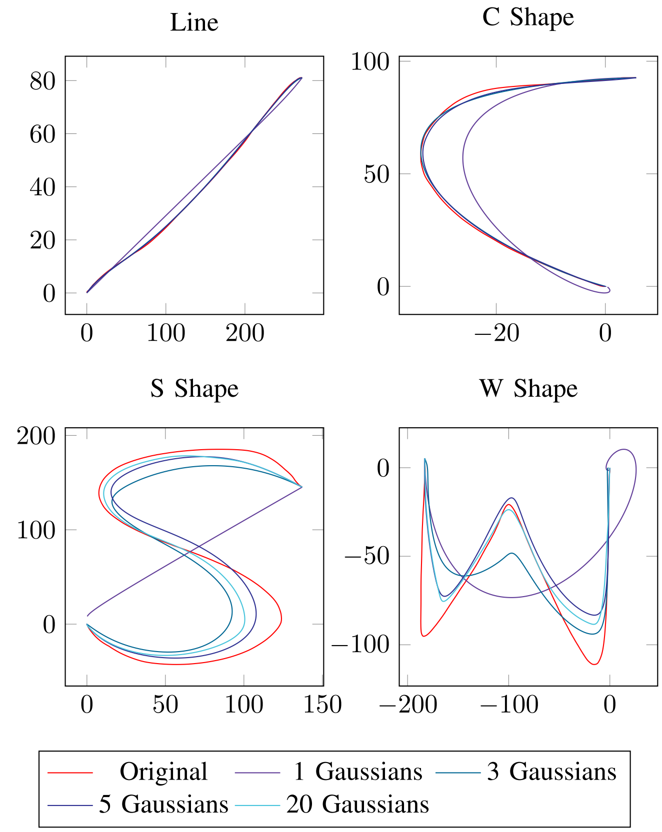

Fig. 3: Plots showing how the number of gaussians affects the result from SEDSs. For the Line and C Shape, higher gaussians were not plotted as their simple shape lead to an overfitting-type error.

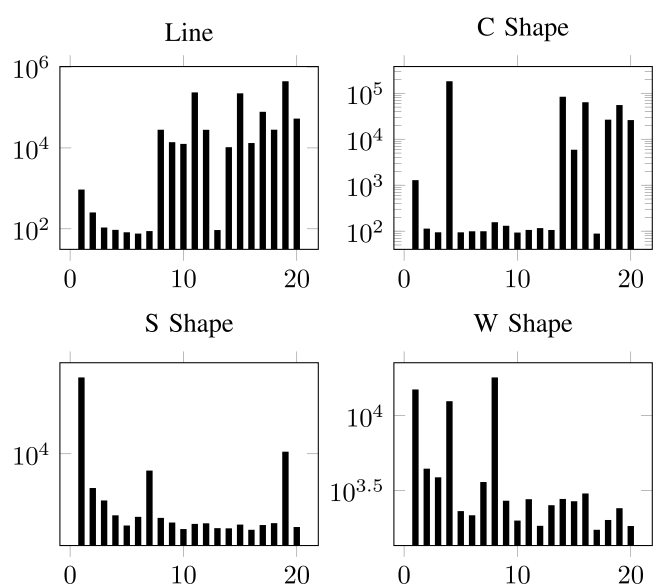

Fig. 4: Figure showing how the number of gaussians affects the results from SEDSs, as before, DTW [6] is used as a metric of similarity. The y axis is logarithmic, to make it easier to compare with Figure 1. Moreover, it also emphasizes the difference between different number of gaussians.

References

[1] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, "Recent advances in robot learning from demonstration," *Annual review of control, robotics, and autonomous systems*, vol. 3, pp. 297–330, 2020.

[2] Y. Shavit, N. Figueroa, S. S. M. Salehian, and A. Billard, "Learning augmented joint-space task-oriented dynamical systems: A linear parameter varying and synergetic control approach," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2718–2725, 2018.

[3] H. Wang, J. Chen, H. Y. Lau, and H. Ren, "Motion planning based on learning from demonstration for multiple-segment flexible soft robots actuated by electroactive polymers," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 391–398, 2016.

[4] M. Najafi, M. Sharifi, K. Adams, and M. Tavakoli, "Robotic assistance for children with cerebral palsy based on learning from tele-cooperative demonstration," *International Journal of Intelligent Robotics and Applications*, vol. 1, pp. 43–54, 2017.

[5] A. Pervez, Y. Mao, and D. Lee, "Learning deep movement primitives using convolutional neural networks," in *2017 IEEE-RAS 17th international conference on humanoid robotics (Humanoids)*. IEEE, 2017, pp. 191–197.

[6] S. Salvador and P. Chan, "Toward accurate dynamic time warping in linear time and space," *Intelligent Data Analysis*, vol. 11, no. 5, pp. 561–580, 2007.