# Temporal Alignment and Demonstration Selection as Pre-processing Phase for Learning by Demonstration

**J. Donjat[a], A. Legeleux[b], C. Buche[c], D. Duhaut[b]**
[a]Lab-STICC, ENIB, France
[b]Lab-STICC, Univerity of South Brittany, France
[c]IRL CROSSING, ENIB, Australia
donjat@enib.fr; legeleux@enib.fr; buche@enib.fr; dominique.duhaut@univ-ubs.fr

**Abstract**

Robots can benefit from users' demonstrations to learn motions. To be efficient, a pre-processing phase needs to be performed on data recorded from demonstrations. This paper presents pre-processing methods developed for Learning By Demonstration (LbD). The pre-processing phase consists in methods composed of alignment algorithms and algorithms that select the good demonstrations. In this paper we propose six methods and compare them to select the best one.

## Introduction

The field of Learning by Demonstration aims to develop techniques where the movement is programmed by the operator by showing the wanted movement to the robot (Zhu and Hu 2018; Calinon 2009; Argall et al. 2009; Ravichandar et al. 2020; Chernova and Thomaz 2014). It allows a non-expert user to program the robot in an easy practical way. Execution time and speed of demonstrations will not be exactly the same due to the human nature of the demonstrations (Calinon 2009). The operator could eventually make some mistakes and demonstrate movements with noise (Argall et al. 2009; Ravichandar et al. 2020; Chernova and Thomaz 2014). In this paper, we offer alignment algorithms and selection of good demonstrations algorithms. We also propose pre-processing methods to align demonstrations and take the better ones. Demonstrations will be performed by the operator then pre-processed before sending them to the learning phase. The first step is to re-align the demonstrations in duration. Aligned demonstrations are required for a good learning (Calinon 2009). The second step is selecting good demonstrations or only good parts of them to have the best learning phase (Argall et al. 2009; Ravichandar et al. 2020; Chernova and Thomaz 2014). During the experimenting phase, this pre-processing will be tested on three robots, on three different movements which increase in difficulty. For each movement, five data-sets will be created, composed of combinations of good and bad demonstrations to cover a very large panel of cases. At first, we will present the related work. Then we will present pre-processing: alignment, selection and technologies which use both. After, experiments performed and metrics used to get results will be described. Furthermore, these results will be compared to see which method should be used in the context of LbD. As a conclusion, we will explain which technology is the best and present some ideas for future work.



## Related Work

In the followings sections, the word demonstration is used as an equivalent of the term time-series. One of the most used technologies for this kind of application is called Dynamic Time Warping (DTW) (Müller 2007; Berndt and Clifford 1994; Sabbaghi, Bahrami, and Ghidary 2014). It analyses the similarity in two demonstrations which can include speed variations. A lot of variants of DTW exist, they have specific features for a very large panel of uses. Each version of DTW can improve a specific aspect like the speed (Geler et al. 2019; Sakoe and Chiba 1978; Choi et al. 2020) or the precision (Jeong, Jeong, and Omitaomu 2011; Muscillo et al. 2007; Munich and Perona 1999), but the calculation time and the complexity of the algorithm will also increase. It is possible to use combinations of several versions of DTW to combine advantages. DTW will compare two demonstrations and will get a warping path and a final cost. The warping path represents the best match between the points of the two demonstrations. The final cost represents the sum of the absolute difference between every pair of points of the warping path.

The Ramer-Douglas-Peucker also called Douglas-Peucker (Douglas and Peucker 1973; Ramer 1972) algorithm can be used to resize demonstrations to a given number of points. To resize demonstrations, the demonstration with the smallest number of points is taken as reference. The algorithm then deletes the non-important points by using a point-to-edge distance tolerance and preserves the important ones. At the end, all demonstrations have the same number of points and the global aspect of each demonstration is preserved.

Another algorithm was created by (Kyrarini 2019): this method will split the demonstrations following important points and will align them with Douglas-Peucker on the smallest demonstration. Then a cost representing similarity between demonstrations will be calculated with a formula using weight and Manhattan distance. The demonstration with the smaller cost is considered as the reference. Then a K-mean or a threshold is used to select demonstrations considered as good.

The DTW is a lot slower compared to the algorithm created by (Kyrarini 2019). The complexity of the time calculation of the DTW follows an increasing parabolic curve as the time of the two demonstrations compared increases. On the contrary, the algorithm created by (Kyrarini 2019) follows a linear evolution as the time of the two compared demonstrations increases. For the Douglas-Peucker algorithm, it is required to have a constant sampling period to be able to use it. Because we are deleting points during the alignment phase, it is only possible to align demonstrations on the smallest one. Furthermore, the Douglas-Peucker algorithm needs to be used carefully, because over suppression may damage the overall shape of the demonstration. Considering the previously mentioned technologies, algorithms using DTW and the work of (Kyrarini 2019) have been implemented.

## Pre-processing for temporal alignment and selection of good demonstrations

In this work, the learning algorithm used after the pre-processing is composed of a Gaussian Mixture Model (GMM) and a Gaussian Mixture Regression (GMR) (Calinon, Guenter, and Billard 2007; Calinon and Billard 2008).

### Temporal Alignment

The first temporal alignment algorithm is called DP-N (Elmar de Koning 2011). We propose a second algorithm which has been developed, called Proportional. The DP-N is a variation of the mentioned Douglas-Peucker (Douglas and Peucker 1973; Ramer 1972) called Douglas-Peucker version N. DP-N will be used for the alignment. It takes the smallest demonstration and re-sizes every other demonstration to the smallest demonstration size. By doing that, every demonstration will have the same amount of points. The Proportional algorithm developed is trivial as shown in Equation 1. The alignment is made on a selected demonstration to align on. From the demonstration to align on, the final time called Alignment Time is taken. For each point of each joint of each demonstration, the algorithm divides the time value of each point by the Final Time of its demonstration and multiplies it by the Alignment Time.

$$Time_{new} = Time * \frac{\text{Alignment Time}}{\text{Final Time}} \quad (1)$$

### Selection of good demonstrations

The first selection of good demonstrations' algorithm that we propose is called Point-to-Point (PTP) selection algorithm and works only if we use the temporal alignment DP-N previously mentioned. The second algorithm is based on DTW (Müller 2007) and it is called DTW selection algorithm. The error score is the sum of the absolute difference between each consecutive point for each joint of a demonstration. The demonstration called reference is the demonstration which has the smallest accumulated error score when compared to other demonstrations (Equation 2). Then with the error scores of every demonstration compared to the reference, a one dimension K-mean with two clusters will be set, and 1000 iterations will be done. Finally, only demonstrations situated in the good cluster where the reference is are selected and kept. If the reference is the only demonstration in its cluster, demonstrations are compared to a threshold and only the ones which are below this threshold are selected. The DTW selection algorithm is very similar to the precedent one. The changes are only how the error score is calculated. Where a point to point difference was calculated for the PTP method, the DTW will obtain an error score from the optimal warping path, called final cost previously. As explained on the DTW part, this assures the best alignment between the reference and the selected demonstration.

$$Ref_D = min_{A \in D}(\sum_{d=1}^{D}(\sum_{j=1}^{J}\sum_{p=1}^{Pt}|Pt_d - Pt_A|)) \quad (2)$$

*Where D=Demo, J=Joint, Pt=Point (Angular Value)*

### Pre-processing Methods

The Figure 1 shows the effects of the pre-processing phase and the movement generation obtained after the learning phase. There are 6 methods. Two of them will be control methods which will use alignment algorithms only (called MA1 and MA2) as the learning phase needs aligned demonstrations to work properly. The 4 others are based on combinations of alignment and selection algorithms previously mentioned. There are two categories in addition to the control methods category. The first will impact the demonstra-

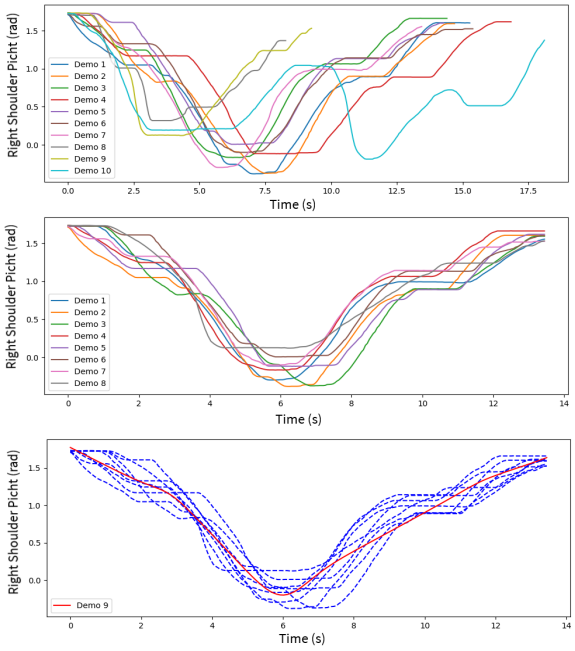


Figure 1: From top to bottom : original demonstrations, aligned and selected demonstrations, generated movement (in red) after learning phase with aligned and selected demonstrations (in dotted blue line).

tions in their entirety. The second will act on part of them: each demonstration will be divided into portions according to specific time markers, provided by the user. This step is called the sub-division step. Method 1 (M1) will combine the DP-N alignment algorithm and the Point-to-Point selection algorithm: the first step is to align demonstrations with the DP-N algorithm; the second step is to select the reference and the third step is to select the good demonstrations. Method 2 (M2) uses the Proportional alignment algorithm and the DTW selection algorithm: at first the reference is selected; alignment between the reference and the other demonstrations is made, and good demonstrations are selected. Method 3 (M3) and method 4 (M4) are the same methods as M1 and M2 respectively but in these two methods, the subdivision aspect is introduced. It allows the selection of a sub-section of a demonstration instead of an entire demonstration.

### Characteristics and constraints of algorithms and methods

The DP-N alignment algorithm aligns all the demonstrations on the smallest one. The Proportional alignment algorithm can align on any demonstration selected to be the reference. The Point-to-point selection algorithm is used with the DP-N alignment because it requires a constant sampling period and the same amount of points for each demonstration to be able to compare them. Furthermore, use of methods with subdivisions means that important points and where they are in the timeline are acknowledged. If not, these methods would not be appropriate and the methods which use entire demonstrations would be better.

## Experiments

### Setup

Experiments will follow this scheme: 3 tasks will be done by 3 robots, Nao, Pepper, YuMi (Figure 2). The data collected are the angular values of the robots joints in time. These values constitute the demonstrations. Nao and Pepper robots have 5 joints per arm and 1 joint per hand to be able to open/close it. YuMi has 7 joints per arm and 1 per gripper. Tasks will be performed with a single arm. There is a difference in frequency between YuMi robot and Nao, Pepper robots. YuMi has a frequency of 250Hz and Nao and Pepper have a frequency of 30Hz. There is no vision system included on robots to help the tasks' realisation. Conditions are similar for every task. The first task is writing the letter D in the air, the second is to give a glass to a person and the third is to take a goblet and put it in a mug. They are respectively called : Letter D (LD), Give Glass (GG) and Goblet Into Mug (GIM). The tasks demonstrated are increasingly difficult due to spatial and material constraints applied. For the tests, 5 data-sets made of 10 demonstrations will be prepared for every task and for each robot. Different combinations will be tested, from a very good data-set to a data-set that includes a lot of bad demonstrations. The following Table 1 resumes all of these combinations.

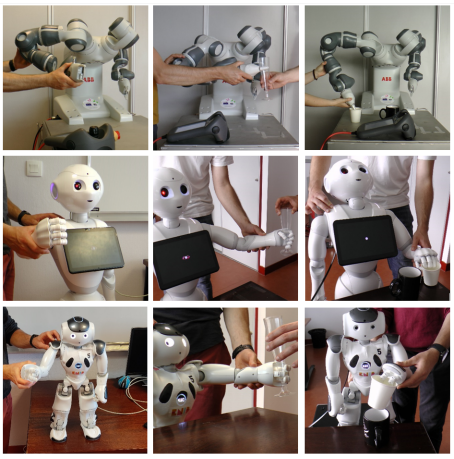

Figure 2: Photos of the 3 robots and the 3 tasks. Robots in columns (top to bottom): YuMi, Pepper, Nao. Tasks in line (left to right): Letter D, Give Glass, Goblet Into Mug.

Table 1: Demonstration combinations for the 5 data-sets. Each data-set is composed of 10 demonstrations. #Demos = Number of demonstrations.

| Data-sets | #good demos | #normal demos | #bad demos |
|---|---|---|---|
| D1 | 10 | - | - |
| D2 | 8 | 2 | - |
| D3 | 7 | 2 | 1 |
| D4 | 6 | 2 | 2 |
| D5 | 4 | 3 | 3 |

### Metrics

The term ideal demonstration will be used to name a demonstration selected subjectively by the user as a demonstration which represents the best wanted movement. Robot joints don't have the same motion amplitude, depending on the movement performed. Before each metric, a normalisation is done on every joint to ensure that each joint has the same impact in the error score. Metrics used will be Computing Time (CT) to know which method is the fastest. Final Mean Absolute Error (FMAE) (Equation 3) and Final Mean Squared Error (FMSE) (Equation 4) which are means based on Mean Absolute Error (MAE) (Chai and Draxler 2014) and Mean Squared Error (MSE) (Wang and Bovik 2009). The last metric is the Success Rate (SR). For the metrics before the learning phase, MAE is used to compare two demonstrations and obtain a similarity score for the compared demonstrations. The MSE is similar to the MAE except that the square difference is calculated instead of the absolute difference. This score shows us the disparity between the two demonstrations. The cross information from MAE and MSE provides information about the similarities of the two demonstrations and how errors are distributed throughout the demonstrations.

$$FMAE = \frac{1}{D}\sum_{d=1}^{D}\frac{1}{J}\sum_{j=1}^{J}\frac{1}{P_t}\sum_{p=1}^{Pt}|Pt_{demo} - Pt_{ideal}| \quad (3)$$

$$FMSE = \frac{1}{D}\sum_{d=1}^{D}\frac{1}{J}\sum_{j=1}^{J}\frac{1}{P_t}\sum_{p=1}^{Pt}(Pt_{ideal} - Pt_{demo})^2 \quad (4)$$

*Where : D=Demo, J=Joint, Pt=Point (Angular value).*

For the metric after the learning phase, the Success Rate (SR) of the task will be evaluated. The success rate is a value between 0 and 100% given by the user, where 0% is an absolute failure and 100% a perfect success of the task. The value is an addition of several categories as follows in Table 2 : *successful task* is to ensure that the main goal is reached; *no blocking movement* means there are no movements which could lead to a task failure (ex: the arm of the robot is blocked under the table); *no additional movement* means there is no noise in the movement; *fluidity* is an small criteria in our case due to our choice of movements and because no liquid was manipulated.

Table 2: Gradation percentage of the Success Rate

| Success Rate | |
|---|---|
| 50% | Successful Task |
| 25% | No Blocking Movement |
| 20% | No Additional Movement |
| 5% | Fluidity |

## Results

A comparison between methods MA1, MA2, M1, M2, M3, M4 will be presented. Results will be presented as follows : General results, Data-sets, Robots, and Tasks.

**General results** In this section, all the data-sets were merged to have the mean of the results from every data-set. Methods which use the DTW algorithm are the ones which have the highest Computing Time. The method which has the lowest FMAE is M1 closely followed by M2 (Table 3, Table 4). For the FMSE, M1 is overall the best and sometimes M2. For the Success rate (Table 5), the better results are obtained with M2, followed by MA2, M4, M1, M3, and at the end MA1. It appears that M2 is the best because it is a good compromise between having good FMAE/FMSE even if it is not the best one, and having good Success Rate. M2 has the highest Computing Time but it remains acceptable for LbD.

**Data-sets** Depending on the data-sets (averaged across robots and tasks), the Computing Time decreases when the data-sets deteriorate, but it is probably due to the variable duration of demonstrations and because the bad demonstrations are shorter than the good ones. The FMAE and FMSE (Figure 3) increase according to the deterioration of the demonstrations quality. M1 is the best method closely followed by M2 and M4 even if the gap between them seems more distant for the last data-sets. For the Success Rate (Table 6), M2 is the best and sometimes MA2.

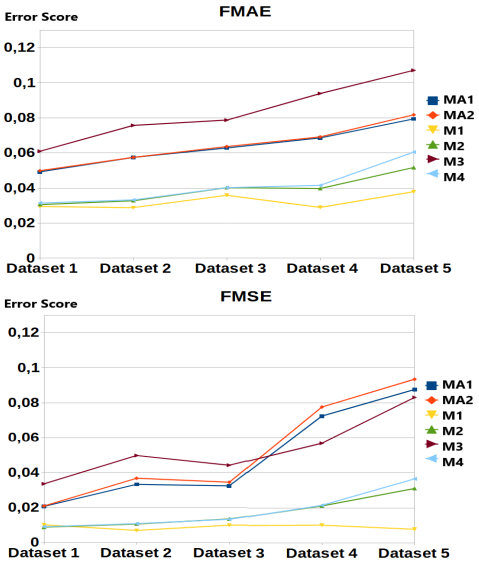


Figure 3: FMAE and FMSE of data-sets

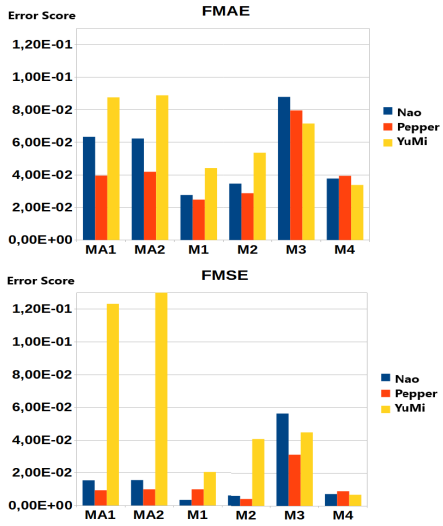


Figure 4: FMAE and FMSE of robots

**Robots** Depending on the robots (averaged across data-sets and tasks), Nao and Pepper are close, though Pepper is faster than Nao. YuMi is the one which has the highest Computing Time due to the previously mentioned superior frequency, thus more data to process. The FMAE (Figure 4) is the best with Pepper followed by Nao and then YuMi, except for M3 and M4 where there are some differences due to the sub-divisional aspect. YuMi has more error probably due to its superior amount of data compared to the other two. The FMSE is bad with MA1 and MA2, this is greatly reduced when the method M1, M2, M3 or M4 is used. Overall M3 is the worst. The Success Rate is better with M2, except for YuMi where MA2 is the best, but this could be a bias in the experiments (Table 7, Robots column).

Table 3: FMAE for each methods

| | Nao | | | Pepper | | | YuMi | | |
|---|---|---|---|---|---|---|---|---|---|
| Methods | LD | GG | GIM | LD | GG | GIM | LD | GG | GIM |
| MA1 | 9.88e-2 | 4.56e-0.2 | 4.57e-2 | 4.94e-2 | 3.07e-2 | 3.86e-2 | 1.35e-1 | 3.56e-2 | 9.21e-2 |
| MA2 | 9.50e-2 | 4.61e-2 | 4.61e-2 | 5.34e-2 | 3.27e-2 | 3.95e-2 | 1.41e-1 | 3.53e-2 | 9.08e-2 |
| M1 | **3.34e-2** | **2.05e-2** | **2.90e-2** | **2.61e-2** | **1.93e-2** | 2.91e-2 | **7.49e-2** | **1.41e-2** | **4.36e-2** |
| M2 | 4.58e-2 | 2.82e-2 | 2.98e-2 | 3.85e-2 | 2.09e-2 | **2.67e-2** | 9.93e-2 | 1.42e-2 | 4.73e-2 |
| M3 | 1.27e-1 | 7.31e-2 | 6.39e-2 | 1.02e-1 | 4.91e-2 | 3.60e-2 | 1.48e-1 | 4.06e-2 | 1.10e-1 |
| M4 | 4.58e-2 | 3.77e-2 | 2.98e-2 | 5.08e-2 | 2.09e-2 | **2.67e-2** | 9.93e-2 | 1.42e-2 | 4.73e-2 |

Table 4: FMSE for each methods

| | Nao | | | Pepper | | | YuMi | | |
|---|---|---|---|---|---|---|---|---|---|
| Methods | LD | GG | GIM | LD | GG | GIM | LD | GG | GIM |
| MA1 | 3.00e-2 | 3.01e-2 | 4.55e-3 | 1.45e-2 | 5.28e-3 | 7.99e-3 | 2.92e-1 | 1.22e-2 | 6.53e-2 |
| MA2 | 3.01e-2 | 7.11e-3 | 9.10e-3 | 1.51e-2 | 5.83e-3 | 8.54e-3 | 3.21e-1 | 1.19e-2 | 6.44e-2 |
| M1 | **4.55e-3** | **1.43e-3** | 4.17e-3 | **3.22e-3** | 2.19e-3 | 4.46e-3 | **4.50e-2** | 1.99e-3 | **1.42e-2** |
| M2 | 9.11e-3 | 5.07e-3 | **4.10e-3** | 6.62e-3 | **1.98e-3** | **4.06e-3** | 1.06e-1 | **1.85e-3** | 1.46e-2 |
| M3 | 1.14e-1 | 2.93e-2 | 2.57e-2 | 3.87e-2 | 7.02e-2 | 6.53e-3 | 1.25e-1 | 1.33e-2 | 5.90e-2 |
| M4 | 9.11e-3 | 8.43e-3 | **4.10e-3** | 1.43e-2 | **1.98e-3** | **4.06e-3** | 1.06e-1 | **1.85e-3** | 1.46e-2 |

Table 5: General Success Rate

| | Nao | | | Pepper | | | YuMi | | | General mean | Ranking |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Methods | LD | GG | GIM | LD | GG | GIM | LD | GG | GIM | | |
| MA1 | 64.6 | 46.8 | 40.4 | 68.2 | 78.8 | 43.6 | 77.2 | 73.2 | 48 | 60.089 | 6 |
| MA2 | **98** | 65.2 | 40.4 | 77.6 | **88.6** | 43.6 | 88.4 | 82.4 | **82.4** | 74.067 | 2 |
| M1 | 70.6 | 56.8 | 56.4 | 75 | 83.8 | 52.4 | **93.4** | 59.8 | 53 | 66.8 | 4 |
| **M2** | 90.4 | **68.2** | **79.6** | 93.6 | 86 | **58.4** | 92.6 | **99.2** | 30 | **77.555** | **1** |
| M3 | 73.4 | 54.8 | 72.8 | 79 | 86.8 | 40.2 | 73.8 | 57.8 | 39.6 | 62.244 | 5 |
| M4 | 88 | 56 | 63 | **96.4** | 87.4 | 52.6 | 72.6 | 59.2 | 59.6 | 70.533 | 3 |

Table 6: Success Rate of data-sets

| | Data-sets | | | | |
|---|---|---|---|---|---|
| Methods | Data-set 1 | Data-set 2 | Data-set 3 | Data-set 4 | Data-set 5 |
| MA1 | 78.667 | 78.889 | 62.889 | 45.333 | 34.667 |
| MA2 | **90** | **88.333** | 75.111 | 67.222 | 49.667 |
| M1 | 75.889 | 70.333 | 73.444 | 58.333 | 56 |
| **M2** | 84.333 | 82.778 | **89.555** | **67.889** | **63.222** |
| M3 | 89.667 | 84.889 | 62.444 | 48.333 | 35.889 |
| M4 | 72.444 | 86 | 77.889 | 67.778 | 48.555 |

Table 7: Success Rate of Robots and Tasks

| | Robots | | | Tasks | | |
|---|---|---|---|---|---|---|
| Methods | Nao | Pepper | YuMi | LD | GG | GIM |
| MA1 | 50.6 | 63.533 | 66.133 | 70 | 66.267 | 44 |
| MA2 | 67.867 | 69.933 | **84.4** | 79.667 | 66.8 | 53.933 |
| M1 | 61.267 | 70.4 | 68.733 | 79.667 | 66.8 | 53.933 |
| **M2** | **79.4** | **79.333** | 73.933 | **92.2** | **84.467** | 56 |
| M3 | 67 | 68.667 | 57.067 | 75.4 | 66.467 | 50.867 |
| M4 | 69 | 78.8 | 63.8 | 85.667 | 67.533 | **58.4** |

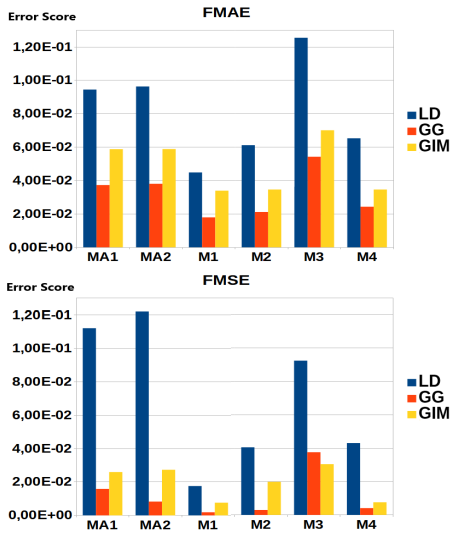


Figure 5: FMAE and FMSE of tasks

**Tasks** Depending on the tasks (averaged across data-sets and robots), computing time for each task is directly linked to their duration which is irrelevant to analyse. The FMAE (Figure 5) is the best for GG followed by GIM and then LD. This is a little surprising because LD should be better than the two others due to the lower difficulty of the task. A possible explanation could be that LD has no spatial constraint and so the movement traced in the air to write the letter D could be very different from one movement to another. This is correlated by the FMSE aspect which is huge for LD. Overall M1 has the best error score closely followed by M2. For the Success Rate (Table 7, Tasks column), the best method is M2.

## Discussion

There are two methods which stand out from the others, these are methods M1 and M2. M1 is good because it reduces the error the most and, after the selection of good demonstrations, it is with this method that we obtain the smallest error. M2 is the one which has the better results after the learning phase in terms of success rate. But there are some parameters to take into account before choosing which method is the best. First, M1 is based on the DP-N algorithm which has the particularity of reducing the amount of points

by selecting the best ones. This explains the smaller amount of errors but also deteriorates the shape of the movement. On the contrary, M2 does not delete any points which can induce a higher error score, even if M1 and M2 are pretty close in terms of error score. M2 keeps the exact shape and this is why we obtain a better movement generation with this method. For the methods with subdivisions, M4 is viable but not optimal and M3 is the worst method. For the LD task, the highest amount of error with this task is probably due to the non-physical aspect of the movement. Indeed, GG and GIM include movements with physical objects, where as LD is based on a visual aspect which is more prone to error and more subjective to evaluate. So it appears that M2 is the better compromise between FMAE/FMSE and Success Rate, furthermore it has a reasonable Computing Time.

## Conclusion

Several algorithms and methods where developed and evaluated with precise metrics on 3 robots, 3 movements and with 5 data-sets from good to worst. It seems that M2 which uses the Proportional alignment algorithm and DTW selection algorithm gives us the best results after the learning phase for movement generation. Globally, it appears that M2 is the overall best method to use. Furthermore improvement could be added to this present work, if the important points for the subdivision methods are selected with better criteria, this could give us better results. The previous results could also be analysed by splitting them in two kind of data-sets, with and without bad demonstrations, to see a more precise trend. Other learning algorithms could be tested to see if they lead to another method that is more adapted.

## Acknowledgment

This article benefited from the support of the project Prog4Yu ANR-18-CE10-0008 of the French National Research Agency (ANR).

## References

Argall, B. D.; Chernova, S.; Veloso, M.; and Browning, B. 2009. A survey of robot learning from demonstration. *Robotics and Autonomous Systems* 57(5):469–483.

Berndt, D. J., and Clifford, J. 1994. Using dynamic time warping to find patterns in time series. In *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, AAAIWS'94, 359–370. AAAI Press.

Calinon, S., and Billard, A. 2008. A probabilistic programming by demonstration framework handling constraints in joint space and task space. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 367–372.

Calinon, S.; Guenter, F.; and Billard, A. 2007. On learning, representing, and generalizing a task in a humanoid robot. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 37(2):286–298.

Calinon, S. 2009. *Robot Programming by Demonstration: A Probabilistic Approach*. EPFL/CRC Press.

Chai, T., and Draxler, R. R. 2014. Root mean square error (rmse) or mean absolute error (mae). *Geoscientific Model Development Discussions* 7(1):1525–1534.

Chernova, S., and Thomaz, A. L. 2014. Robot Learning from Human Teachers. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 8(3):1–121. Publisher: Morgan & Claypool Publishers.

Choi, W.; Cho, J.; Lee, S.; and Jung, Y. 2020. Fast constrained dynamic time warping for similarity measure of time series data. *IEEE Access* 8:222841–222858.

Douglas, D. H., and Peucker, T. K. 1973. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization* 10(2):112–122.

Elmar de Koning. 2011. psimpl - Douglas-Peucker simplification. http://psimpl.sourceforge.net/douglas-peucker.html.

Geler, Z.; Kurbalija, V.; Ivanović, M.; Radovanović, M.; and Dai, W. 2019. Dynamic time warping: Itakura vs sakoe-chiba. In *2019 IEEE International Symposium on INnovations in Intelligent SysTems and Applications (INISTA)*, 1–6.

Jeong, Y.-S.; Jeong, M. K.; and Omitaomu, O. A. 2011. Weighted dynamic time warping for time series classification. *Pattern Recognition* 44(9):2231 – 2240. Computer Analysis of Images and Patterns.

Kyrarini, M. 2019. *Robot Learning from Human Demonstrations for Human-Robot Synergy*. Doctoral dissertation, Universität Bremen.

Munich, M. E., and Perona, P. 1999. Continuous dynamic time warping for translation-invariant curve alignment with applications to signature verification. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 1, 108–115 vol.1.

Muscillo, R.; Conforto, S.; Schmid, M.; Caselli, P.; and D'Alessio, T. 2007. Classification of motor activities through derivative dynamic time warping applied on accelerometer data. In *2007 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 4930–4933.

Müller, M. 2007. Dynamic Time Warping. In *Information Retrieval for Music and Motion*. Berlin, Heidelberg: Springer. 69–84.

Ramer, U. 1972. An iterative procedure for the polygonal approximation of plane curves. *Computer Graphics and Image Processing* 1(3):244–256.

Ravichandar, H.; Polydoros, A. S.; Chernova, S.; and Billard, A. 2020. Recent Advances in Robot Learning from Demonstration. *Annual Review of Control, Robotics, and Autonomous Systems* 3(1):297–330.

Sabbaghi, E.; Bahrami, M.; and Ghidary, S. S. 2014. Learning of gestures by imitation using a monocular vision system on a humanoid robot. In *2014 Second RSI/ISM International Conference on Robotics and Mechatronics (ICRoM)*, 588–594.

Sakoe, H., and Chiba, S. 1978. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing* 26(1):43–49.

Wang, Z., and Bovik, A. C. 2009. Mean squared error: Love it or leave it? a new look at signal fidelity measures. *IEEE signal processing magazine* 26(1):98–117.

Zhu, Z., and Hu, H. 2018. Robot Learning from Demonstration in Robotic Assembly: A Survey. *Robotics* 7(2):17. Number: 2 Publisher: Multidisciplinary Digital Publishing Institute.