Probabilistic Movement Primitives applied to obstacle avoidance

Ignacio Alzugaray, Nicola Covallero, Eric Guerrero,

***Abstract*—Several techniques have been developed along the years in order to capture complex movements to be later reproduced by the robots. During the last decade, the movements primitives with especial regard to the Dynamic Movement Primitives (DMP) have successfully achieve the characterization of complex human-like movements which had been used to solve a whole new set of tasks. In the last years, a new approach called Probabilistic Movement Primitive (ProMP) has been developed in which the trajectories are handled with stochastic techniques. This work will review and study the movement primitives emphasizing the ProMP approach. The techniques and algorithm will be later validated in an experimental stage. These experimental applications will cope with obstacle avoidance as the movement will be done in different environment with obstacles.**

I. Introduction

In the last years, robot learning have become one of the most important fields in robotics. Since the robotic systems have increased their presence in real world setups, it has become a must that they can perform their tasks successfully under uncertain conditions or even react to unexpected events. Regarding this matter, the robot learning is a key field which increase the comprehension and improve the reaction behaviors of the robotic systems in a human-like manner.

A significant part of the scientific robotic community has been researching over the last decades in the achievement of human-like task using robots. As this paradigm present complex tasks usually carried on by humans, the movements involved in the success of these task have also a remarkable complexity.

In order to cope with this complexity in the movements, the movement primitive (MP) framework was developed. In this approach, the movements where processed as a set of finite parameters which abstracts and synthesizes an arbitrary movements. This parametrization of the movements imply that several techniques can be applied such as the learning of these movements by means of learning these parameters or the modification of the movements to cope with uncertain environments.

The description of the arbitrary movements through has have been dominated over the last decade by the Dynamic Movement Primitive (DMP) approach. Using DMP a differential equation is defined such that its particular solution and the sum of an additional force term represent the movement. In this case, the force function shapes the natural solution of the differential equation to characterize the movement. Therefore, the parameters that embed the movement are used in this force function.

Recently, a new framework called Probabilistic Movement Primitives (ProMP) has been developed. This new approach represent the points of a trajectory as a weighted sum of finite probabilistic functions. In this case, the weights of these probabilistic functions are the parameters that embeds the movement. However, the inclusion of probabilistic terms to represent MP offers a new set of techniques that will be reviewed in later in this project.

Both approaches applied to MP, these are, the DMP and the ProMP will be briefly reviewed and compared. Nonetheless, the ProMP will be studied in detail as it is the main approach used in the applications developed during this project. The different algorithms of the ProMP will be implemented firstly in MATLAB and later tested in a 7-DOF robotic manipulator. Different experiments will be proposed to highlight the different techniques and operators of the ProMP approach.

The structure of this work starts with the brief explanation of the motivation behind the selected topic in the section II. Then a small survey on the movements primitives with special regard to the DMPs is approached in section III. The ProMPs are studied in detail in section IV and then applied in experiments in section V. Finally, the future works and lines of research are explained in section VI and the conclusion of the project are collected in section VII.

II. Motivation

This project offers an unique opportunity to study in detail some of the current state of the art techniques related to the field of motion primitives. Specifically, in the project the ProMP approach will be used to generate, learn and process human-like trajectories. This approach has been chosen in this project due not only to the mentioned advantages that it offers compared to the classical approaches but also it is in an early stage of development and therefore there still are new methods and applications to be discovered for this technique.

Additionally, this project has been strongly motivated by the fact that a WAM Robot from Barrett Technology, this is, serial 7-DOF arm is provided by the Institut de Robòtica i Informàtica industrial (IRI) to test the ProMP. The sofware that implements the ProMP in the robot is also provided by the group of Perception and Manipulation which has been used in the past to develop new applications with this algorithms. By using these resources, this projects is to be developed close to a real work environment.

Finally, in this project the different techniques used in ProMP will not only be applied in the robotic system but also

Students of the Master's degree on Robotics and Automatic control. ETSEIB (UPC). Cataluyna.

reviewed in detail. This methodology in which the different algorithms will be analyzed and implemented will provide to the authors of this project a solid knowledge of the current state of the art in motion primitives.

III. Movement primitives

This project will be focused in a specific field of robotics regarding the generation, processing and learning of movements. The movements of the robotics system have been traditionally approached as a sequence of configurations, this is, a trajectory which the robot should follow to complete a task. The trajectory using this approach is usually generated by the connection of a initial and final configuration. In controlled scenarios such as industrial environments, this trajectory can be simple computed once and the reproduced. In dynamically changing environments, this can be adapted by solving the trajectory at each instant such that it verify a set of kinematic constraints of the robot. While these approaches have been successfully implemented in controlled industrial-like environments with several techniques, it usually fails when the robot has to perform in a environment with significant uncertainty or when it has to perform complex human-like motions. In order to cope with these limitations, the new framework of motion primitives has been developed for the last years.

The motion primitives are a generalization of movements to represent trajectory with arbitrary complexity. The motions can be described by using set of finite number of parameters, this is, an abstraction which allows the parametrization of the movements. This parameters embedded the trajectory in a structured manner inside the movement primitive algorithms. By means of using structured parameters for the movement representation different learning techniques can be applied to learn them and therefore learn also the movement they represent. Additionally, the representation of the movements in terms of parameters also allows the adaptation of that movements to dynamic and uncertain environments if the parameters are consequentially modified. The motion primitives represent a milestone in the robot learning field as they are able to handle human-like complex trajectories. In this regard, the learning process of motion by means of motion primitives have been intensively developed in the learning by demonstration (LbD) framework.

In the last decade, the motion primitives have been widely approached using the Dynamic Motion Primitives (DMP) since it was initially presented in [9]. Due to its high impact in the robotic community several relevant works were developed such as [10] were the initial learn algorithm is redefined, inspired by the central pattern generation method [11].

The DMP approaches uses a differential equation with the following form.

$$\ddot{y} = \alpha(\beta(g - y) - \dot{y}) + f \tag{1}$$

which solution represent the movement. Note that there there exist two different attractors in, this is position y and velocity $\dot{y}$ which can be tuned with the gains α and β respectively. The end position of the movement can be dynamically modified by changing the goal point g. Finally, the solution of the differential equation is shaped by means of the force function f.

The main contribution of the DMP is embedded in the force term. By selecting a adequate forcing function, the resulting solution of the differential equation and therefore the movement can be modified. In this case, a example of suitable forcing function could be defined as

$$f(x, g) = \frac{\sum_i \psi_i \omega_i}{\sum_i \psi_i} x(g - y_0) \tag{2}$$

where ψ_i is an specific the basis function, usually with Gaussian form, and ω_i are the parameters.

The parametrization of the movement, this is, the computation of the weights ω_i can be computed from the shape of the force function f. This force function profile can be obtained by the difference between the natural solution of the differential equation and the movement that is to be to imitated. Then different learning algorithm can be applied to compute the weights and therefore train the movement primitive.

Another key idea developed in this framework is the time representation. The variable x used in the force function and in the basis functions ψ_i is a remapping of the actual time t using the following expression

$$x(t) = e^{-\alpha_x t} \tag{3}$$

With this transformation of time into the so-called canonical system, the time space $t \in [0, \infty)$ is remapped into to $x(t) \in [1, 0)$. By tuning parameter α_t the time of execution of the trajectory can be straightforwardly adapted.

The DMPs have been proven a powerful technique to representation of movements. Moreover, several works have developed many interested applications with DMPs to generate complex movements that can adapt to changing environmental conditions. It has been also developed a extensive literature from the application of DMPs to LbD problems to solve task that required human-like movements.

In the last years, a new technique have been developed in the same spirit as DMPs but using a probabilistic approach. This approach called Probabilistic Movement Primitives (ProMP) will be detailed in the following section as it will be the the main core of this project.

IV. Probabilistic Movement Primitives

The ProMP approach was initially presented in [7] and further developed in the following years in [6] and [8]. The ProMP approach represents the movements as a sum of probabilistic functions at each time instant. In this case, the probabilistic functions are computed as a defined basis function multiplied by an specific probabilistic weight. Since all the basis functions have the same form for a single ProMP, the movement is parametrized in terms of the weights that multiply each of them.

The inclusion of the movement primitives into the probabilistic framework adds a whole new set of options to model and vary the different movements. As the motions are treated in a probabilistic manner the robustness of the trajectories to environmental perturbations can be improved. Additionally, the parameters can be modified to synthesized new motions regarding different initial or final goals or even intermediate via-points. The main features of the ProMP will be discussed briefly in the following sections. ——-

A. Movements with Time-Varying variance

Let τ be a single trajectory defined by a sequence of configurations q_t up to the instant T.

$$\tau = \{q_t\}_{t=0,\ldots,T} \tag{4}$$

Then a configuration of the trajectory at some time instant t defined in terms of position q_t and velocity $\dot{q}_t$ can be represented by the following expression

$$\mathbf{y_t} = \begin{bmatrix} q_t \\ \dot{q}_t \end{bmatrix} = \mathbf{\Phi}_t^T \omega + \epsilon_y, \tag{5}$$

where $\mathbf{\Phi}_t$ is the linear basis function model which are multiplied by the weight vector ω and $\epsilon_y \sim \mathcal{N}\left(0, \Sigma_y\right)$ is zero-mean Gaussian noise.

In this project since only stroke-based movements will be reproduced, the Gaussian basis functions will be used.

$$b_i^G = \exp\left(-\frac{(z_t - c_i)^2}{2_h}\right) \tag{6}$$

$$\phi_i(z_t) = \frac{b_i(z_t)}{\sum_j b_j(z_t)} \tag{7}$$

In other cases in which rhythmic movements are to be represented it may be more suitable to use frequency based functions such as Von-Misses basis functions.

The resulting trajectory can be interpreted as a probabilistic distribution as a results of the joint distribution between all the probabilistic distribution defined at each time instant.

$$p(\tau|\omega) = \prod_t \mathcal{N}\left(\mathbf{y_t}|\mathbf{\Phi}_t^T\omega, \Sigma_y\right) \tag{8}$$

To obtain a compact representation and easy the learning process of the trajectories the vector of parameters $\theta = \{\mu_\omega, \Sigma_\omega\}$ is introduced. This parametrization determines the probabilistic function which represents the weights.

$$p(\omega;\theta) = \mathcal{N}\left(\omega|\mu_\omega, \Sigma_\omega\right) \tag{9}$$

then the trajectory can be obtaining by marginalizing the original expression

$$p(\tau;\theta) = \int p(\tau|\omega)p(\omega;\theta)\left(\omega|\mu_\omega, \Sigma_\omega\right) \tag{10}$$

The probabilistic parametrization of the trajectory allows to apply several mathematical techniques for stochastic functions. In this case, the mean value of the of the weights leads to the computation of the mean trajectory (Figure 1).

However, since the weights can be sampled from the distribution function, different trajectories can be generated from the same trained parameters. This particular aspect is specially interesting during the so-called exploration in learning algorithm. The exploration refers to the autonomous learning process from which a trained robot explores solutions close to the one already obtained. This is usually implemented when using other techniques as DMPs as the addition of a small exploration noise to the computed trajectories. However, using ProMP, the exploration become natural as the weights are sampled from probabilistic functions. The exploration is used in combination with a feedback process that allow the robotic system to reinforce successful movements.

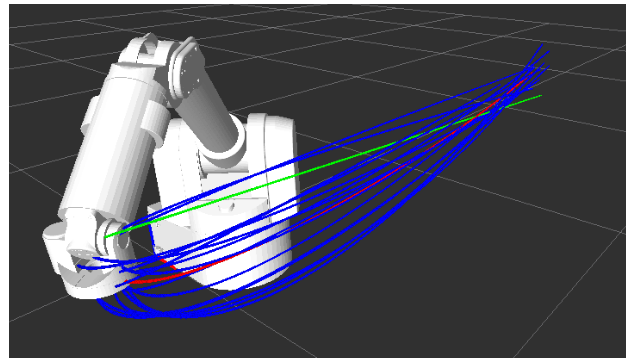


Fig. 1: Training of a ProMP with learning by demonstration: demonstration trajectories (blue), mean of trajectory (red), sampled trajectory for exploration (green).

B. Temporal Modulation

In order to decouple the trajectories from the time a new phase variable z is introduced. This new variable normalizes the time between $z_{t=0} = 0$ and $z_{t=T} = 1$. Then, the basis functions are transformed with the following expression

$$\phi_t = \phi(z_t) \tag{11}$$

$$\dot{\phi}_t = \phi'(z_t)\dot{z}_t \tag{12}$$

By selecting a function of the phase variable the execution time of trajectory can be controlled.

The temporal modulation is a key factor when numerous trajectories are used to learn the parameters. Since the time elapsed to execute the trajectories will be different for each one the normalization is necessary.

Additionally to the normalization, the trajectory that will be used in the learning process would have to be processed such that the different movements resemble to each others at each time instant. This process is called time wrapping and applies an optimization of the whole batch of trajectories that will be used in the ProMP. This optimization modulates the time of each trajectory independently in order to obtain highly similar trajectories which will be used in the learning of the movement.

C. Encoding Coupling between Joints

Up to this point the ProMPs have been developed for a single joint. However, the model can be straightforwardly extended to a multiple joint system. However, this robotic system will usually have coupling between the different joints as in a serial arm. This effect will be considered in order to obtain an accurate representation of the trajectory.

The new trajectory obtained with the multiple joint robot is extended up to N dimensions. Consequently, the basis function ϕ_t are grouped into a diagonal matrix Φ_t in order to be multiplied by a new vector of probabilistic weights $\omega = \left[\omega_1^T, \ldots, \omega_N^T\right]^T$. Then the expression of the equation (10) is transformed into

$$p(\tau|\omega) = \prod_t \mathcal{N}\left(\begin{bmatrix}\mathbf{y_{1,t}}\\ \vdots \\ \mathbf{y_{N,t}}\end{bmatrix} \middle| \begin{bmatrix}\mathbf{\Phi}_t^T & \cdots & 0\\ \vdots & \ddots & \vdots\\ 0 & \cdots & \mathbf{\Phi}_t^T\end{bmatrix}\omega, \Sigma_y\right) \tag{13}$$
$$= \prod_t \mathcal{N}\left(\mathbf{y}_t|\mathbf{\Psi}_t^T\omega, \Sigma_y\right)$$

D. Learning from demonstrations

The ProMP poses a framework to represent in a compact manner different complex movements. However, as the DMPs, the main benefits of this framework is the simplicity in learning the different movements. In this case, it will be emphasized the learning from demonstrations, this is, trajectories that are registered by the robot joints. This registered trajectories can be obtained, for instance, from an exploration process or when the robot is guided by an human agent imitating a movement.

Provided a new set of demonstration trajectories, the parameters $\theta = \{\mu_\omega, \Sigma_\omega\}$ that encode the movements can be computed. This parameters can be computed by fitting a distribution that can be assumed Gaussian-like $p(\omega;\theta) = \mathcal{N}\left(\omega|\mu_\omega, \mathbf{\Sigma}_\omega\right)$. Then the new ProMP can be computed by included the information that codify the new demonstrations in the following form.

$$p(\mathbf{y}_t;\theta) = \int \mathcal{N}\left(\mathbf{y}_t|\mathbf{\Psi}_t^T\omega, \mathbf{\Sigma}_y\right)\mathcal{N}\left(\omega|\mu_\omega, \mathbf{\Sigma}_\omega\right) \tag{14}$$
$$= \mathcal{N}\left(\mathbf{y}_t|\mathbf{\Psi}_t^T\mu_\omega, \mathbf{\Psi}_t^T\mathbf{\Sigma}_\omega\mathbf{\Psi}_t + \mathbf{\Sigma}_y\right) \tag{15}$$

The computation of the parameters $\theta = \left(\mu_\omega, \mathbf{\Sigma}_\omega\right)$ can be visualized as a fitting of the distribution probabilistic function of the weights. Under the assumption that distribution is Gaussian-like, several techniques can be used to obtain the parameters θ.

One first approach to compute θ is the least square fitting to merge the information from N different demonstration trajectories stacked in $\mathbf{y}$. Since the same basis functions Φ set is used in a single ProMP, the parameters θ can be obtained with the Moore-Penrose pseudoinverse.

$$\mu_\omega = \left(\mathbf{\Psi}^T\right)^\dagger \mathbf{y} \tag{16}$$
$$\mathbf{\Sigma}_\omega = \mathbf{E}\left[(\mathbf{y} - \mathbf{\Psi}^T\mu_\omega)^2\right] \tag{17}$$

Although the least square fitting yield an overall good performance, when the number of demonstration are scarce or ill conditioned the resulting θ parameters may not represent correctly the movements. However, there exist other possibilities to the computation of the parameters exploiting their probabilistic properties. For instance, the parameters can be learned from multiple demonstrations by maximum likelihood estimation applying a expectation-maximization algorithm, that iteratively maximizes a likelihood function. It is based in two steps: an expectation step that produces an expectation of the log-likelihood based in the current estimated parameters, and a maximization step in which the parameters that maximize the log-likelihood found in the expectation step are computed.

E. Modulation of the trajectories

The ProMP framework does not only reproduce similar trajectories that the ones used for demonstration. Additionally, it can perform several operations over the already learned trajectories. By modifying the weight distribution, this is, the θ parameters, the resulting trajectories of the ProMP can be designed to achieve specific tasks.

One of the most important operations in the ProMP framework is the conditioning. By conditioning a ProMP, the resulting probabilistic trajectory is modified to pass through a point with a specific accuracy at a given time instant. If there are several joints, this conditioning of the trajectory can be imposed any number of the available degrees of freedom which are encoded by a probabilistic function of weights. Usually, this conditioning is done in the Cartesian space and its effect is limited to the position of the robot. However, it can be extended to the orientation of the robot or even it can be used in joint space.

Given a conditioning such that the trajectories are modified to meet at the instant t the state y_t with a desired accuracy $\mathbf{\Sigma_y}$, it can be defined a desired observation $\mathbf{x}_t = \left[y_t, \Sigma_y\right]$. Then applying the Bayes theorem

$$p(\omega\|\mathbf{x}_t) \propto \mathcal{N}\left(\mathbf{y}_t \mid \mathbf{\Psi}_t^T\omega, \mathbf{\Sigma_y}\right)p\left(\omega\right) \tag{18}$$

Finally, the probabilistic trajectory can be modified in order to satisfy the conditioning defined by the probability $p(\omega|\mathbf{x}_t^*)$ using the following expression.

$$\mu_w^{new} = \mu_w + \mathbf{\Sigma}_w\mathbf{\Psi}_t\left(\mathbf{\Sigma}_y^* + \mathbf{\Psi}_t^T\mathbf{\Sigma}_w\mathbf{\Sigma}_t\right)^{-1}\left(\mathbf{y}_t^* - \mathbf{\Psi}_t^T\mu_w\right) \tag{19}$$
$$\mathbf{\Sigma}_w^{new} = \mathbf{\Sigma}_w - \mathbf{\Sigma}_w\mathbf{\Psi}_t\left(\mathbf{\Sigma}_y^* + \mathbf{\Psi}_t^T\mathbf{\Sigma}_w\mathbf{\Sigma}_t\right)^{-1}\mathbf{\Psi}_t^T\mathbf{\Sigma}_w \tag{20}$$

There are also other possible operations which are a direct consequence of the probabilistic approach used in the representation of the movements. For example, using a similar approach as the the conditioning, different movement primitives defined in the ProMP framework can be combined or blended in a given range of time. However, the utilities of this operations are limited to some specific situations that are out of the scope of this project.

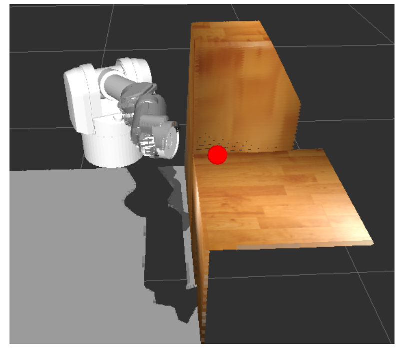
V. Experiments

The experimental part of this project can be subdivided in two main parts. The first experimental part was devoted to the understanding of the algorithms involved in the ProMP whereas the second experimental part regards the implementation in a real robotic system.

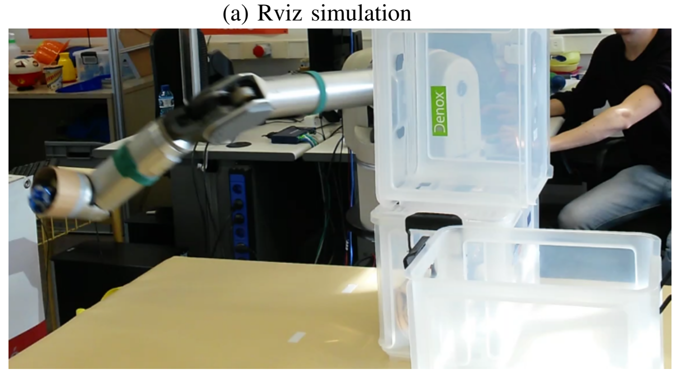
Initially, the algorithm that defines the ProMP framework were implemented in MATLAB. These are the probabilistic trajectory definition of the movements, learning of ProMP by demonstration based on least squared and conditioning. By implementing all the different codes the authors of this project could establish a solid base of knowledge not only of the ProMPs but of the general motion primitive framework. At to this point, the obtained results are limited application and validation of the algorithm applied to synthetic simulated data.

In the second part of the experimentation, a WAM robot from Barret Inc. have been used. This serial arm has 7 degree of freedom and uses an application interface implemented in C++ with several and useful functionalities. In addition to the code provided by the manufactures, for the implementation of the algorithms it has been also used the code of the researcher of the group of Manipulation and Perception of the IRI. Since this source code implements all the different functions required for the ProMPs only small changes have been committed to adapt the code to the different experiments. Beside the C++ code applied directly on the robot, MATLAB code has been also used for processing of the trajectories such as the implementation of the expectation-maximization algorithm for training. It has been also used in order to visualize the different training and resulting trajectories. In this case other ROS packages such as Gazebo and Rviz have been also used to visualize the trajectories and robot in a more simulated environment similar to the experiment setup.

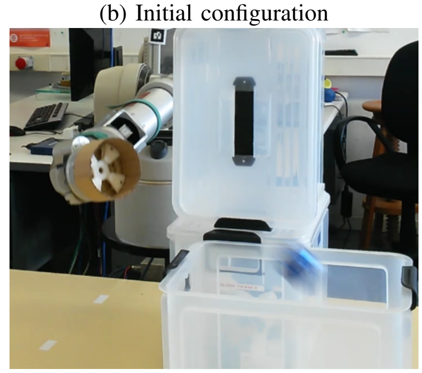
Two different experiments have been defined in which different mock-up of industrial setting have been designed. In the first experiment the robot is trained with demonstrations in order to throw an object to an specific area. In this experiment the basic concepts of ProMP such as the training and the probabilistic sampling of trajectories will be reviewed and validate. In the second experiment an industrial application have been developed in which an operator teaches the robot how to push an object towards a classifier bin. In this second experiment, it is emphasized the conditioning of the trajectories. In both experiments the robot movements avoids the obstacles of the environment. This obstacle avoidance is done directly in the training stage or indirectly by using of the conditioning of the trajectory.



(a) Rviz simulation



(b) Initial configuration



(c) Final configuration

Fig. 2: First experiment: Throwing of a ball.

A. Experiment 1: Throwing of objects

In this experiment, the basic concepts of the ProMPs will be reviewed applied to a simple motion. A human-like movement will be reproduced in which the robot throws an object.

In this case the robot is to place a ball in a bin. Since the bin is not accessible by the robot due to obstacles in the path, the robot is taught to throw the ball into the bin instead as shown in the Figure 2. The obstacles avoidance is done in this case explicitly as the demonstration trajectories will actually avoid the collision with the environment. In this application it can be remarked the benefits of the movements primitives applied to learning by demonstration. The explicit definition of a dynamic human-like trajectory is avoided since the robot can learn from the movements of the human agent [2].

The demonstration trajectories are processed to compute the parameters of the ProMP using the two different techniques introduced in the theoretical background: least squares and expectation-maximization. The results of the learning with both methods can be observed in Figure 3. In this case, it can be observed how the expectation-maximization algorithm outperforms the least squares in terms of variance and, consequently, in the confidence interval. Such a significant difference is related to the limited number of demonstrations, in this case, 10 of them. Although in general both algorithm can perform equivalently, the least squares rely on a vast amount and good conditioned data in order to work properly.

Once the ProMP is learned, a exploration process is applied to the robot system. The results of the exploration trajectories, these are, sampled trajectories can be observed in Figure 4. Note that all the exploration trajectories lie within the confidence interval computed during the learning of the ProMP. Since the exploration trajectories are similar to the demonstration ones, the obstacle avoidance of the objects in the environment is implicit in the learning phase as it can be observed in [1].

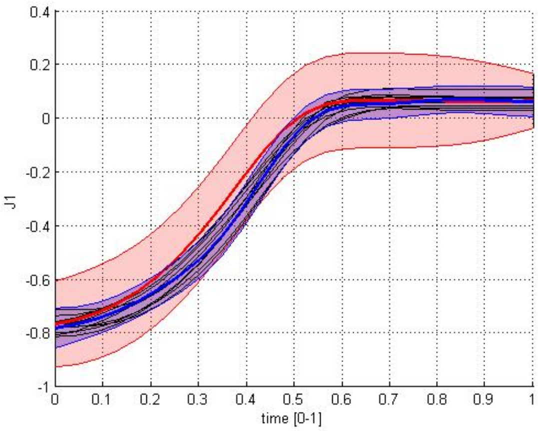


Fig. 3: Expectation maximization and least square fitting for the first joint. Black lines represent the demonstration trajectories, the red line and the red area, represent respectively the mean and the scaled confidence interval with the least square method (scale 1:4) while the blue ones represent them with the expectation maximization algorithm.

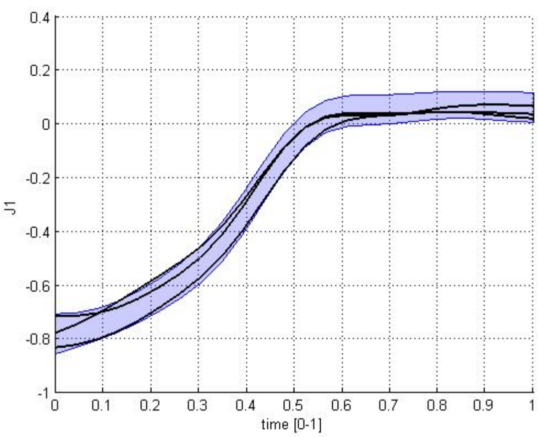


Fig. 4: Exploration of trajectories in the first joint.

B. Experiment 2: Pushing of objects

This second experiment represent a common industrial application in which different pieces are transported on a conveyor belt. At some point, the different pieces transported by the conveyor belt are inspected. After this inspection, some of the pieces are faulty and therefore have to be removed from the conveyor belt. In this case, the pieces are removed by pushing them out of the conveyor belt carefully to avoid the collision with other non faulty pieces at both sides. The piece is pushed up to classification bin depending on the detected fault.

In this application learning by demonstration comes to be really useful because the robot has to move avoiding collisions with only some parts of the workspace, changing the pose of the end effector in order to push the object as long it executes the trajectory. In particular the ProMPs are useful in such application because with just one movement primitive the robot can push and throw the work piece in a bin by simply conditioning its final point.

The movement primitive is firstly trained by moving manually the robot performing repeatedly the desired trajectory [5]. These demonstration trajectories starts at an arbitrary point on one side of the conveyor, pass through the piece location in the conveyor belt and end in any of the classification bins. Note that in order to do not damage the other pieces of the conveyor, the trajectory passes through a narrow passage avoiding the other obstacles. All these trajectories are registered by the robot encoders and later converted to the Cartesian space with Euler angles, this is, a 6 dimensional trajectory. Although this representation is easier to visualize than the joint space, a major drawback appear a trajectory is to be reproduce since they might exist different configuration that reach the same pose in the Cartesian space. The set of demonstration trajectories used for training is processed using least squares to compute the parameters that characterize the ProMP (Figure 7).

The demonstration trajectories as well as the computed ProMP is presented in Figure 5, and shown in [2]. Since the application has been simplified, only the XYZ-axis will be considered. Nonetheless, since the movement is mainly limited to the plane, the information is mainly provided by the translation in X and Y as it can be observed in the graphs. The mean and the confidence interval set to one standard deviation can be also visualized.

The selection of the ending point to which the faulty object will be pushed, this is, the classification bin is done by means of conditioning of the ProMP. Since the different demonstrations trajectories end indistinctly to any of the classification bins, the conditioning of the end point to one of these locations would not substantially modify the whole trajectory. For this experiment only a case of two different bins is considered. It can be easily observed how the probabilistic trajectory is modified in such a way that the trajectory is finishing in the selected classification bin. The actual modification of the ProMP can be observed in Figure 6. The classification in each of the bin can be simulated as shown in figures 8a and 8b. The result of the execution of such trajectories with the real robot is shown in figures 8c [3] and 8b [4].

VI. Future work and research lines

In this section it will be presented some of the possible future research lines in the ProMP state of the art. The possible future algorithm described in this section have inferred as during the development of this work.

In this project it has been analyzed the conditioning of the trajectory by means of the inclusion of via-points, these are, the points that the trajectory has to meet with a specific accuracy at a given time. However, in some cases it would be desirable to avoid a point instead, for instance, and obstacle. Since the effect is the contrary, this variation of the algorithm would be called called negative conditioning.

Lets assume a two dimensional robot which has been trained with the represented ProMP of Figure 9a. In the same figure, it is represented the position coordinates of a point $\mathbf{X}_o$. The

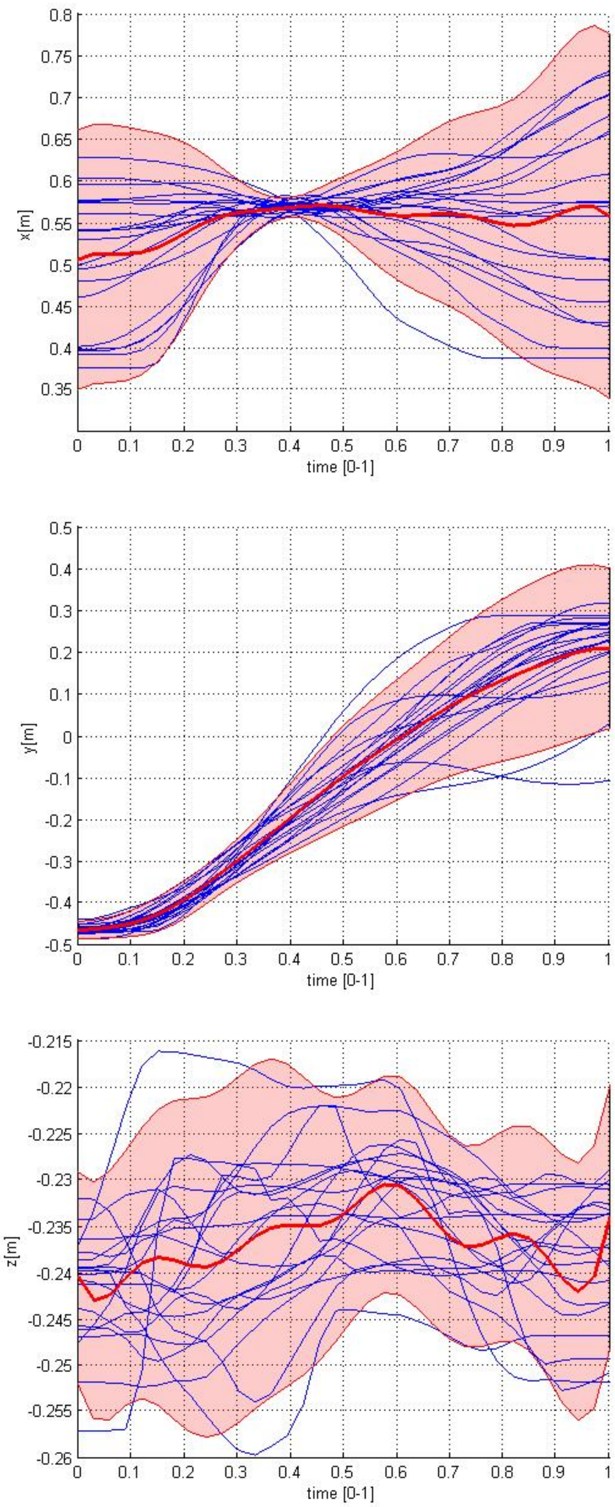


Fig. 5: Least square trajectories' fitting of the Cartesian position. Blue trajectories represent the demonstrations, the red one represents the mean and the red area represents the confidence interval.

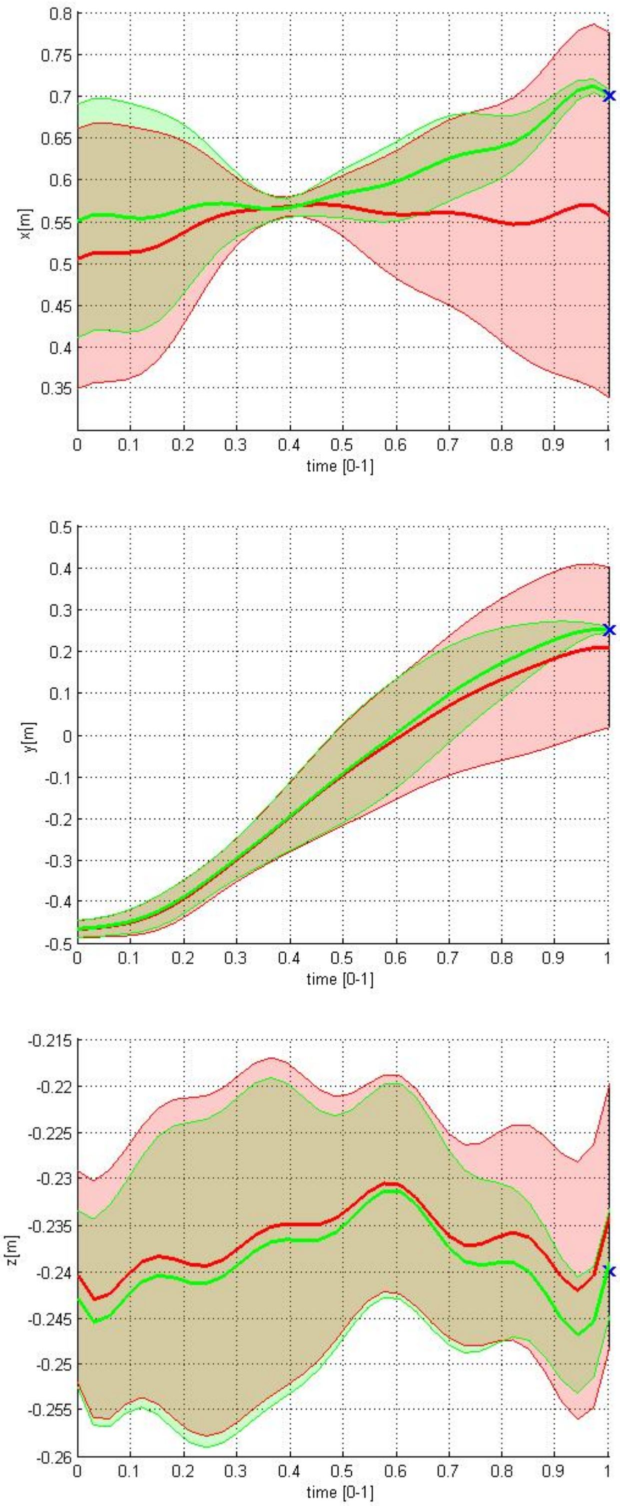


Fig. 6: Conditioning of the final position of the cartesian trajectory. The red trajectory and the red area represent the movement primitive associated to the demonstrations and the green ones represent the new movement primitive with the conditioned final position.

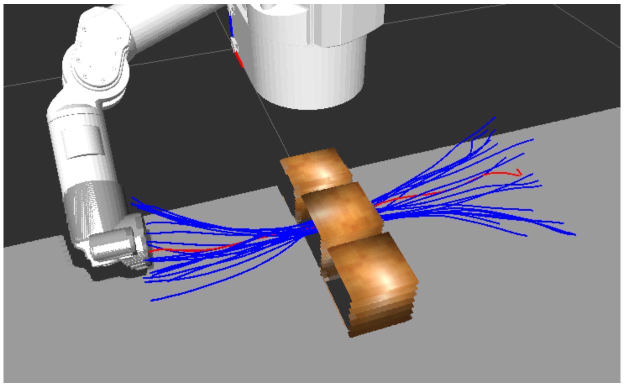


Fig. 7: An initial configuration of a demonstration trajectory of the second experiment. In the simulation with Rviz the blue lines represent the demonstration trajectories, while the red one represents the mean one.

so-called negative conditioning is to be applied in order to avoid this point in the space during the whole trajectory such that the resulting ProMP is deviated as shown in Figure 9b.

However, there are several differences compared to the normal conditioning of points that have to be handle for the implementation of this new algorithm. Firstly, the point to be avoided may not be defined at a given instant since, for instance, it represent a permanent obstacle in the space. Moreover, since this point may represent a spacial location, all the joints have to coincide at the same instant in these coordinates. To cope with this uncertainty, the ProMP will be processed and the time in which the trajectory is most probable to meet the point will be computed.

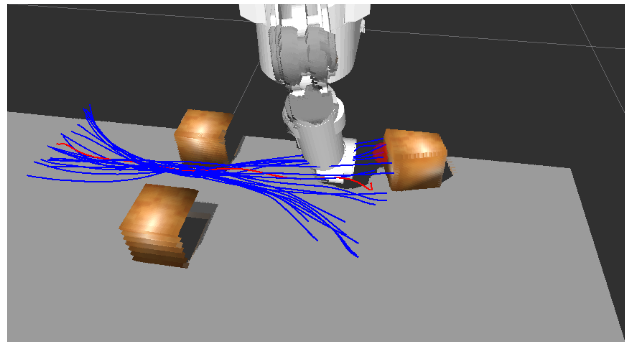
Then different approaches can be used to avoid the point once the time instant is computed. The trivial solution is to define a conditioning such that the trajectory is modified to pass by one side of the point. This is, a conditioning is defined in the neighborhood of the points such that the probability to meet the points is reduced.

A more interesting approach can be used with an extension of the conditioning algorithm. In the original conditioning, the joint probability of the probabilistic function of the trajectory and another Gaussian-like distribution at the specified time is computed. However, instead of defining a Gaussian function to be merged with the trajectory, other probabilistic functions could be defined such that the resulting joint probability of the trajectory is unable to pass through the specified point. In an early stage of development, this could be implemented as a conditioning of the trajectory with multiple Gaussian distribution covering the whole space but the specific location in the same fashion as a filter.

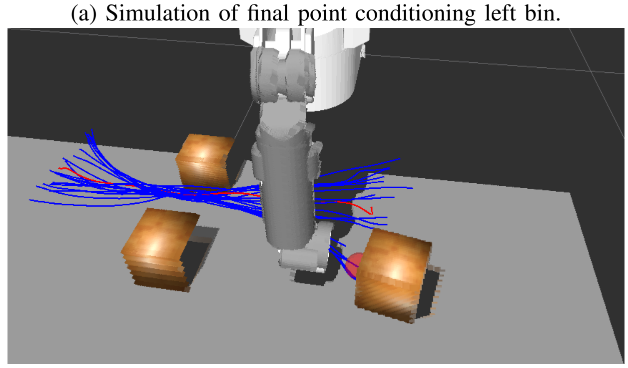
This algorithm could be easily extended to cope with dynamic obstacle avoidance. In this case, the time and location of the point to be avoided would be computed online by means of the dynamics of the object.

VII. Conclusions

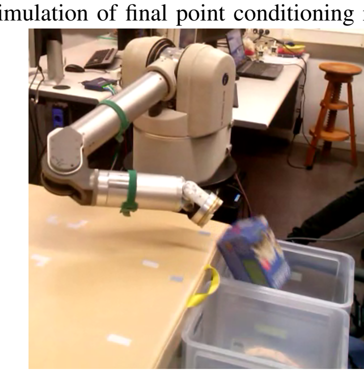
Initially, this project was intended to be an introduction to the robot learning field with the practical application of the



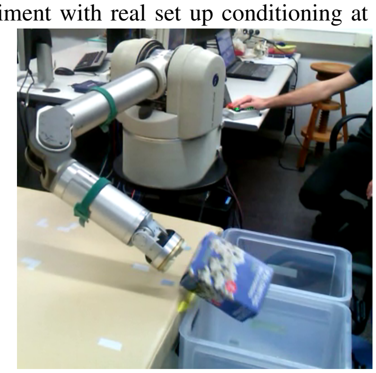
(a) Simulation of final point conditioning left bin.



(b) Simulation of final point conditioning right bin.

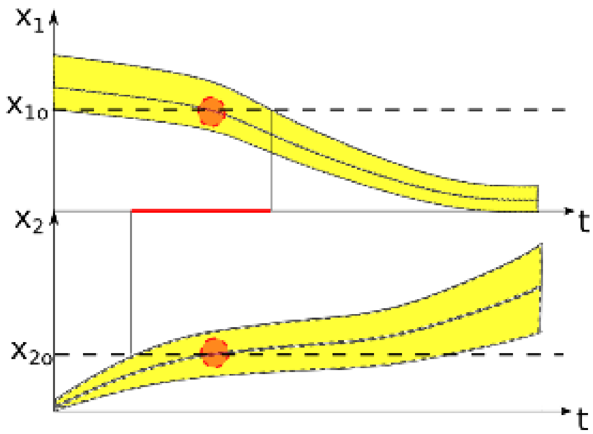


(c) Experiment with real set up conditioning at the left bin

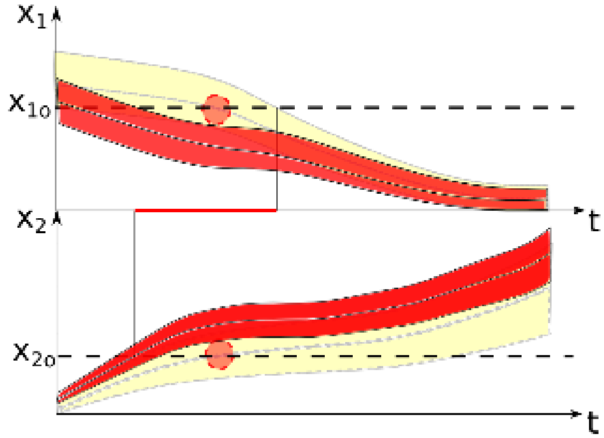


(d) Experiment with real set up conditioning at the right bin

Fig. 8: Second experiment: simulation and execution with the real robot. In the simulation in Rviz the red spherical area represents the selected via point and its qualitative covariance.



(a) Original ProMP and localization of the collision



(b) Resulting ProMP with obstacle avoidance

Fig. 9: Representation of the negative conditioning applied to obstacle avoidance

motion primitives and specifically the ProMPs. Nonetheless, as the project has progressed several other areas of knowledge have been required in order to understand, implement and apply the different algorithms.

Firstly, a brief survey on movements primitives was performed. In this survey, the main differences between DMPs and ProMP were highlighted. Later, the ProMP framework was studied in detail. The formulation of the probabilistic trajectories were presented as well as the time modulation used in the time wrapping of trajectories and the extension to multiple degrees of freedom robots. The main features to successfully compute ProMP from demonstration trajectories and different learning algorithm were introduced. Finally, some useful techniques to modulate the ProMP such as the conditioning were explained.

Then the ProMPs were tested in a 7-DOF serial arm. Two experiments representing a different applications were taken in order to validate the different algorithms studied during the survey. The first experiment was focused on the imitation of a human-like movement whereas the second one represented proper industrial application. In both cases, the movements were not developed in a totally free space environment and therefore some obstacle avoidance strategy was used, in some cases in the training stage and in others in the conditioning.

Furthermore, a novel algorithm has been briefly introduced such that it could serve as a future improvement of the current state of the art in the ProMPs. This algorithm so-called negative conditioning is related to the explicit representation of obstacle avoidance. The formulation of a this algorithm may lead to a major change in the ProMPs since it might be easily extended to dynamical obstacle avoidance.

The ProMP has been proven as a powerful tool to be applied to movement primitives. Its probabilistic approach allows the application of different techniques which could not be directly applied to the DMPs. However, due to its early stage of development the number of these techniques and algorithm is still limited compared to the DMPs. Nonetheless, this is just a motivation for further developments in such a promising approach.

VIII. Acknowledgment

The authors would like to thank Guillem Allenyà for providing the WAM robot and the confidence to work in his lab, Sergi Foix for his help and patience during the experiments with the robot and Adrià Colomé whose help in the understanding of the tshudyeortical background of the ProMPs has been invaluable.

Bibliography

[1] First Experiment: Exploration. https://youtu.be/GsOVhZBxGF0.
[2] First Experiment: Learning by demonstration. https://youtu.be/5UbZHGLYwz8.
[3] Second Experiment: Conditioning - Left bin. https://youtu.be/C6seAf5ijrE.
[4] Second Experiment: Conditioning - Right bin. https://youtu.be/5B2LLkUHAjw.
[5] Second Experiment: Learning by demonstration. https://youtu.be/W9nNyYOwKOU.
[6] G.J. Maeda, M. Ewerton, R. Lioutikov, H.B. Amor, J. Peters, and G. Neumann. Learning interaction for collaborative tasks with probabilistic movement primitives. pages 527–534, 2014.
[7] A. Paraschos, G Neumann, and J. Peters. A probabilistic approach to robot trajectory generation. 2013.
[8] A. Paraschos, E. Rueckert, J Peters, and G. Neumann. Model-free probabilistic movement primitives for physical interaction. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems*, 2015.
[9] S. Schaal. Dynamic movement primitives - a framework for motor control in humans and humanoid robots. In *The International Symposium on Adaptive Motion of Animals and Machines*, 2003.
[10] Stefan Schaal, Jan Peters, Jun Nakanishi, and Auke Ijspeert. Learning movement primitives. In *International Symposium on Robotics Research (ISRR2003)*. Springer, 2004.
[11] Allen I. Selverston. Are central pattern generators understandable? *Behavioral and Brain Sciences*, 3:535–540, 12 1980.