Machine Learning Engineer Nanodegree

Capstone Proposal

Omar Castro
June 14th, 2018

Proposal

Domain Background

Robot learning, which according to the definition in Wikipedia is a research field at the intersection of machine learning and robotics [1], has attracted a great deal of attention over the last years. Self-driving cars, self-flying drones, and autonomous robotic vacuum cleaners are becoming familiar objects in our daily lives. However, teaching robots even the most basic skills is still a daunting task, requiring thousands of trials, highly trained specialists and huge computing resources.

For emerging economies, with scarce resources and different needs from those of developed countries, where most of the research is carried out, handling robot learning requires a frugal approach. The domain background of the project I propose concerns frugal robot learning, employing very low-cost hardware. The main problems of cheap robot hardware are fast tear and wear, high levels of noise, and little availability of sensors or sensor feedback in general. Therefore, frugal robot learning requires data efficient learning strategies that allow the robot to learn in as few trials as possible, taking noise into account, and with light algorithms in terms of computational resources.

Fortunately, data efficient robot learning has been an active topic of research. One of the most cited papers in this area is [2], where Deisenroth and Rasmussen teach a low-cost robot manipulator to perform a basic task with very few trials using PILCO (probabilistic inference for learning control), a data-efficient model-based policy search method [3]. However, PILCO requires large computational times to optimize the policy, since it relies on computationally expensive methods [4], [8]. A more promising approach to achieve frugal robot learning seems to be Bayesian Optimization (BO). Calandra used BO to teach a low cost bipedal to walk with a steady gait [5]. BO is a state-of-the-art model-based approach to optimization under uncertainty.

My personal motivation in choosing data efficient robot learning for my project is twofold: first, I have had a passion for robotics all my life, so being able to teach a robot is something I really enjoy; and second, living in the Dominican Republic, a poor country with failing public services, I believe robots can carry out some of the tasks the government neglects, like public cleaning services. You can't walk one meter in Santo Domingo that is not polluted with industrial waste. In that sense, I have in mind the idea of designing a self-driving robot that collects industrial waste in public places. I tried to create a basic prototype some time ago, just to find out the huge complexity of the task.

In this project I will address just one of the tasks this robot would do, which is picking the garbage using a robot arm. I plan to take the Robotic Engineer Nanodegree offered by Udacity to have the tools to finish this endeavor, so this project is the first stone towards the final objective. Obviously, robots won't solve the garbage problem in Santo Domingo, but they can highlight its importance, and contribute to the civic education of the population and the authorities, or at least I hope they would.

Problem Statement

The problem this project addresses is to teach a very low-cost 6 DOF robot manipulator (US$ 86.99 on Ebay) to grab objects at any discrete position on a plane using a data efficient algorithm. The inputs will be the inicial position of the arm and the angles of the servos in this position, the coordinates of the object to grab, and the coordinates of the container. The tasks to be learned would be the optimal trajectory between the inicial and final positions, grab the object, and drop it on a container.

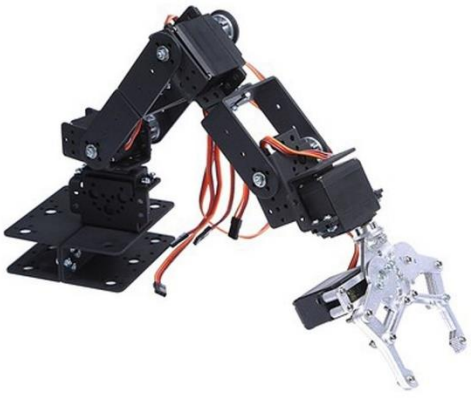

Figure 1: Low cost 6 DOF robot arm by SainSmart

To manipulate the robot arm, I will use the robot controller EZB-V4, which according to their creators, powers more than 20, 000 robots worldwide [5]. The controller costs US$ 79.99 on the web page of ezrobot. The choice of EZB-V4 was a convenience one since I already had one unit which I used to build robots as a hobby. The software that controls EZB-V4, Ez-Builder [6], is a free application provided by the same company to interact with robots. The application allows to set the angles of the servos, the speeds in a scale from 0 to 10 (0 being the fastest), and the time the action will take.

Figure 2: Robot controller EZB-V4 by ezrobot

Datasets and Inputs

This is a Reinforcement Learning project, and therefore, the main dataset will be generated by the movement of the arm itself and its interaction with the environment. The dataset will contain the inicial position of the arm at the beginning of each trial, the angles of the servos and between the parts of the arm in this position, the angles of the servos and between the parts of the arm during the trajectories, the time the trajectory takes, the angles of the servos and between the parts of the arm at the final point, and the position of the clamp at the final point. The goal of the task, whether the object was grabbed or not, will be registered also, and the same set of measures will be taken from the position of the object until it is dropped in the bucket.

Since the algorithm chosen is Bayesian Optimization, which is a model-based approach, I created a mathematical model linking the dimensions of the parts, angles of the servos and corresponding angles between the parts of the arm, with the coordinates of the clamp on the plane for each combination of angles. Using this model, I generated a series of simulations, resulting in a database with 400 data points linking angles of the servos, angles between parts of the arm, and final positions of the clamp corresponding to each combination of angles.

This dataset was then used as input into a third-grade polynomial regression model that allows to predict, given the coordinates of the object to grab, the final angle of the main vertical servo, while the angles of the other servos are calculated using trigonometric formulas. This simple model can be used with any 6 DOF robot arm. It takes as inputs the measurements of the mobile parts of the arm and the inicial servos angles and can predict the final angles of the servos given any point on a plane. This model will be used as input to the Bayesian Optimization algorithm that will optimize the trajectory of the arm given the position of the object.

Solution Statement

The solution proposed to teach the robot arm using a frugal model both in terms of trials and computational resources is to apply a model-based Reinforcement Learning approach. The solution of a Model-based RL can be derived from two alternative method families: value function methods and policy search methods [8]. In this project I will use Bayesian Optimization, which is a policy search approach.

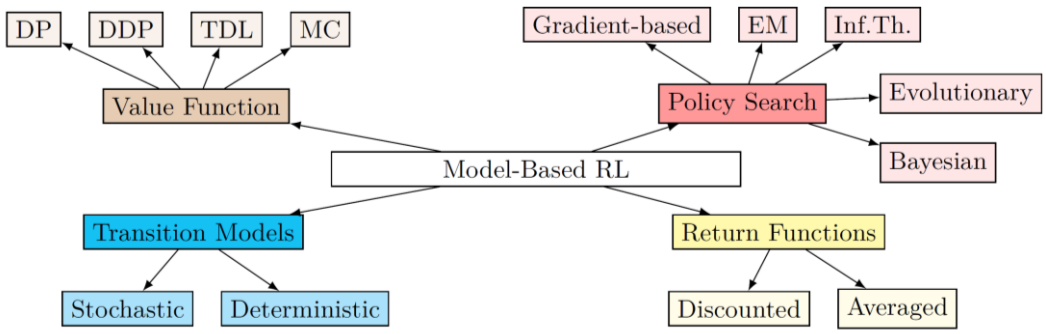

Figure 3: Categorization of approaches for solving a model-based RL problem [8]

Bayesian Optimization is a global optimization method based on response surfaces [5]. Response surface-based optimization methods iteratively create a dataset D = {θ, f(θ)} of parameters θ and the corresponding functions evaluations f(θ) [5]. In Bayesian Optimization a probabilistic model is used to create the response surface that models f. The use of a probabilistic model allows to model noisy observations and to explicitly take the uncertainty about the model itself into account, which makes the probabilistic model more robust to the effect of model errors [5]. The most common probabilistic model used in BO is a Gaussian Process (GO), and it is the one that will be employed in this project.

Benchmark Model

The benchmark models that will be used to compare the results of the study will be pure random search and Grid Search, like the benchmark models used by Calandra in [5]. The measures to compare will be the number of trials until the arm learns the policy, and if the arm learns the policy after a maximum of trials. Calandra used 100 trials as a maximum in [5], and in his study for learning bipedal walk, BO methods learned the policy after 20 to 30 evaluations, while the other methods didn't succeed in finding a right policy for robust walking even after 100 evaluations.

Evaluation Metrics

The main measures of success of this problem are the quantity of objects the arm can grab by number of trials, the quantity of objects dropped inside the container by number of grabbed objects, and the number of trials to learn the behavior. The main objective of the project is to use an algorithm that allows to minimize the number of trials, and that takes noise into account (the arm is very noisy, with fluctuations of more than 1 cm relative to the final aim in a repetitive task using the same parameters).

Project Design

To find an adequate solution for the project, the strategy proposed began with a manual approach to the solution of controlling the robot arm to pick an object on a plane. This first stage was already completed during the previous research phase of the project, in which I tried to find out if I was going to be capable of completing it successfully or not. The manual operation of the arm was extremely difficult. Setting the angles and the velocities for the 6 DOFs to reach an object with no previous knowledge could take dozens of trials. In the process I spent a few days and burned 7 servos, since the arm is very fragile. This gave me the necessary insight to know that using an off-policy RL approach was not feasible for learning an optimal policy, since it would require thousands of trials and errors, and neither the robot hand, not the rest of the hardware (or the experimenter) would support that.

Once the manual exploration stage was completed, the next phase consisted in finding an appropriate model-based RL approach to find a data efficient solution for the problem. Among the methods researched, Bayesian Optimization seemed the most promising one, since it was relatively fast to learn a policy and computationally efficient. However, being a model-based approach, it required a model to be based on. Therefore, the next phase in the search of a solution to the problem was to find an adequate model.

There were several options of models to use for the project: Open AI was one of them, using other robotic simulators, like V-rep was another. There was also the option of applying demonstration learning and use the demonstrations as initial points for learning a policy. The last option I considered was creating a model from scratch to find the final angles of the DOFs given a point on the horizontal plane. I tried the options of the existing simulators, but I found it was too hard to simulate the real conditions of the arm I was using in them, so I feared the policies learned in these simulators would not work in the physical world. The option of the demonstration was viable, but I found it was difficult to generalize [11]. The model from scratch approach seemed harder than demonstration learning, but easier to generalize and closer to the reality of the arm in question.

Creating a relatively simple model of the arm from scratch was hard, but really rewarding. I first set some restrictions to reduce the complexity of the model. The main restriction was that the clamp would always reach the horizontal plane with an angle of 90 degrees. This reduced the response surface to be explored and allowed me to find rules using trigonometric equations to relate each combination of angles of the 6 DOFs with a single point in the plane. The angle of the main horizontal servo was calculated by the Pythagorean theorem, given the distances X and Y, the distance from the point to the center of the arm was the hypothenuse, whose square was equal to the sum of the squares of X and Y.

The restriction in the movement of the clamp allowed me to set only one angle, the one of the main vertical servo, and from it the angles of the 2 other vertical servos were derived by trigonometric equations. With all the angles I could calculate the horizontal distance of the clamp to the center of the arm. Using different angles of the main horizontal servo, which can move from 30 degrees to 130 degrees, I created a dataset with 400 simulations relating the values of the 3 vertical servos with the distance from the center off the arm.

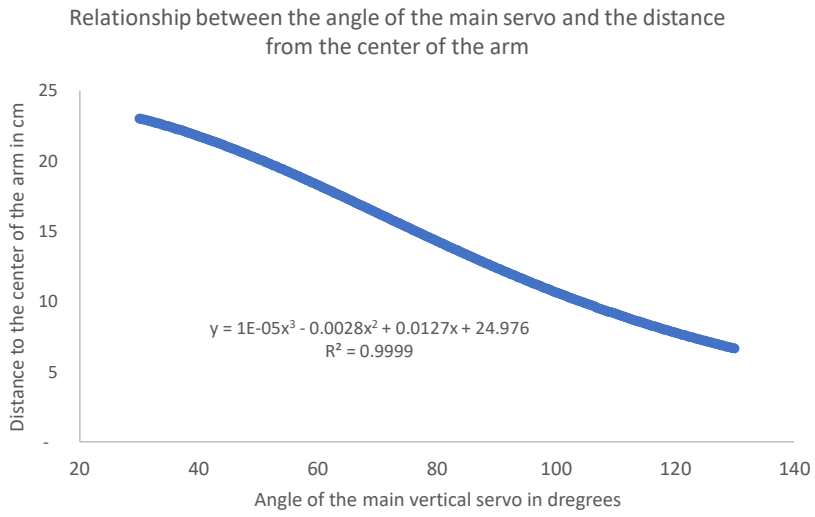

Figure 4: Relationship between the main vertical angle and the distance

At this point I already had the relationship between the angles of the vertical servos and the distance of the clamp from the center of the arm (figure 4), and I also had the relationship between the coordinates of the point and the angle of the main horizontal servo. However, I still couldn't find, given a point in the plane, the corresponding angle of the main vertical servo, but I knew that the relationship between distance and angle could be expressed as a polynomial equation. I therefore used the dataset of simulations with different angles and its corresponding distances, set the angle as the dependent variable, the distance as the explanatory variable, and fitted a polynomial equation of degree 3 (figure 5). With this last step I was able to estimate the angles of all DOFs given a point in the horizontal plane. I am sure there are easier ways to do this, but I created this one from scratch, so I am proud of my accomplishment, and I was also able to apply some of the tools learned in this ML Nano degree.

```
from sklearn.metrics import make_scorer
from sklearn import linear_model
from sklearn import model_selection
from sklearn.preprocessing import PolynomialFeatures

poly = PolynomialFeatures(degree=3)

X_train_ = poly.fit_transform(X_train)
X_test_ = poly.fit_transform(X_test)
```

```
clf = linear_model.LinearRegression()
clf.fit(X_train_, y_train)

best_train_predictions = clf.predict(X_train_)
best_test_predictions = clf.predict(X_test_)
print('The training R2 Score is', r2_score(best_train_predictions, y_train))
print('The testing R2 Score is', r2_score(best_test_predictions, y_test))
```

```
The training R2 Score is 0.999971611326
The testing R2 Score is 0.999963816555
```

```
print(clf.coef_)
print(clf.intercept_)
```

```
[  0.00000000e+00  -1.71385536e+01   7.77943134e-01  -1.64767091e-02]
214.176747118
```

Figure 5: Plolynomial regression to find the angle of the main vertical servo given the distance of a point from the center of the arm

With the initial model estimated, the next step in the project will be to estimate a GO model to optimize the policy of grabbing and dropping an object with the robot arm. Bayesian optimization works by constructing a posterior distribution of functions (gaussian process) that best describes the function you want to optimize [9]. As the number of observations grows, the posterior distribution improves, and the algorithm becomes more certain of which regions in parameter space are worth exploring and which are not (figure 6).

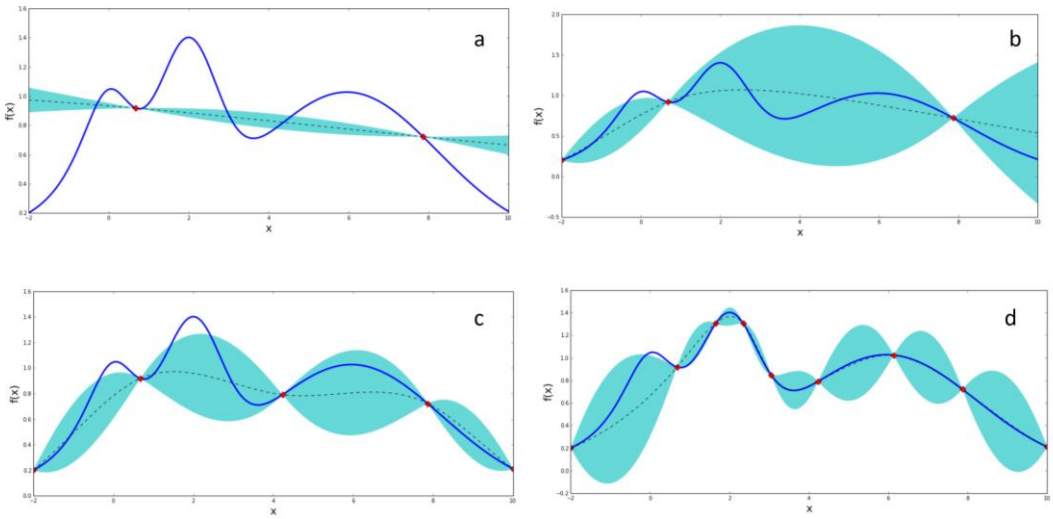

Figure 6: Example of BO. The model is fed at least 2 inicial points, and it explores random points (a) after 2 steps, (b) after 3 steps, (c) after 5 steps, and (d) after 9 steps.

The policies resulting from the GO process will be fed to the arm controller for each trial, and the resulting intermediate and landing points will be fed back to the model. I plan to make a maximum of 30 trials by model, because I also have to try random search and grid search, the benchmark models. I already have a stock of spare parts and servos in case the arm fails before completing the total number of trials.

References:

1. https://en.wikipedia.org/wiki/Robot_learning
2. Deisenroth, M. P., Rasmussen, C. E., & Fox, D. (2012). Learning to control a low-cost manipulator using data-efficient reinforcement learning. In *Robotics: Science and Systems.* (Vol. 7, pp. 57-64). MIT Press Journals.
3. M. P. Deisenroth and C. E. Rasmussen. PILCO: A Model-Based and Data-Efficient Approach to Policy Search. In ICML, 2011.
4. Deisenroth, M.P.; Fox, D.; Rasmussen, C.E. (2014). Gaussian Processes for Data-Efficient Learning in Robotics and Control, IEEE Transactions on Pattern Analysis and Machine Intelligence.
5. Calandra, R., Seyfarth, A., Peters, J. et al. Ann Math Artif Intell (2016) Bayesian optimization for learning gaits under uncertainty. 76: 5. https://doi.org/10.1007/s10472-015-9463-9
6. https://www.ez-robot.com/Shop/Default.aspx?CatId=9
7. https://www.ez-robot.com/EZ-Builder/

8. Polydoros, A.S. & Nalpantidis, L. J Intell Robot Syst (2017) Survey of Model-Based Reinforcement Learning: Applications on Robotics 86: 153. https://doi.org/10.1007/s10846-017-0468-y
9. Jasper Snoek, Hugo Larochelle, Ryan P. Adams. (2012). Practical Bayesian Optimization of Machine Learning Algorithms. arXiv:1206.2944
10. Deisenroth, Marc & Edward Rasmussen, Carl. (2011). PILCO: A Model-Based and Data-Efficient Approach to Policy Search.. 465-472.
11. HUSSEIN, A., GABER, M.M., ELYAN, E. and JAYNE, C. 2017 Imitation learning: a survey of learning methods. ACM computing surveys [online], 50(2), article 21. Available from: https://doi.org/10.1145/3054912