Lund University
Automatic Control

FRTN70 Project in Systems, Control and Learning
Spring 2023

Scientific Machine Learning

Project Plan Group 2

Hashim Siddig[1] Jingmo Bai[2] Julien Moreau[3] Yuyang Jin[4]

Teachers: Emma Tegling, Johan Grönqvist

[1]hashims97@gmail.com
[2]jingmobai@outlook.com
[3]bjorn.duktig@student.lu.se
[4]jinyuyang0518@gmail.com

1 Project Purpose

The goal of this project is to train a reinforcement learning model that swings up the Furuta pendulum and stabilizes it upwards using a physical model of the Furuta Pendulum as a training environment before deploying the model to the real pendulum. Then we will also try to train the model on some tricks (for example flipping the pendulum a number of times).

2 Equipments and material

In this project we will use the Furuta Pendulum from the Automatic Control department at Lund University. The Furuta pendulum consists of a base controlled by a motor and a free falling pendulum arm attached to the base.

3 Modelling and Implementation

We have a physical model for the pendulum that consists of non-linear equations of motion.The report by Magnus Gäfvert [1] gives the dynamics function for the pendulum. It is noticeable that our model can be regarded as a special case of the provided one in the report. For our pendulum, we don't have a ball at the end of the pendulum arm, which means we only need to set the mass of the ball to zero in the dynamics function. Additionally, it is not allowed to apply any external torque on both joints of the pendulum in our project. We can also take friction into account to improve on this model. To achieve this, we can add two terms in the equation of base speed, corresponding to back-EMF and friction in the joints respectively and regularize for the equation of arm speed but this time we neglect the back-EMF. One challenge is to identify the correct parameters for the model. We plan to try the friction parameters given by the report firstly. Chances are that these parameters are not good for the pendulum we use, but that is fine because we only need an approximate simulated model to train the RL model on, then we will continue training it on the real pendulum after we get a good enough result for the simulation environment.

The pendulum can be controlled via a Jupyter notebook written in Julia. The available software also returns the position of the pendulum so it can be used to train Reinforcement Learning algorithms in real time. The angular position (ϕ, θ) can then be differentiated to obtain the speed $(\dot{\phi}, \dot{\theta})$, giving us complete information about the system. We propose to write all the software using Julia. We will use the *Flux* Deep Learning package to handle neural network training. We can also use notebooks from previous classes to kickstart the development. All code will be shared using a Gitlab repository.

4 Exploration Ideas

There is a large number of techniques that we can use to control the pendulum. Here is a list of the ideas we will experiment with. The goal is to implement simpler white-box techniques first then increasing the difficulty level to greyer box.

- Implement a simple controller to swing up the pendulum. Then add a RL component on top of it to handle small movements or friction. The swing up can be achieved using a non-linear controller (e.g. energy control). Then the balancing can be handled with RL.
- Use the friction-free equations in a model-based RL algorithm (like Alpha-zero) and see how it performs with the real pendulum. Try different reward functions to make it more robust.
- Train a black box RL model in a simulation environment (using DDPG for example) then try it on the real pendulum and continue the training in real time using transfer learning.

5 Division of labour

We have decided to split our project into three parts: system identification, RL model design, training in simulation, testing and evaluation.

5.1 System identification

Using the equations for the furuta pendulum we can develop a White or Grey box Automatic Control model. The parameters estimated during this process will help us train reinforcement learning model in the simulation environment.

Julien and Jin will work on the system identification process and get the simulation function ready.

5.2 Reinforcement Learning Template Code

We will explore different Reinforcement Learning models, decide on which models we are going to use, and develop the base code to implement the chosen models.

Hashim and JingMo will work on getting the functions to implement the models ready.

5.3 Training Reinforcement Learning Models

We will experiment with different Reinforcement models using either Model-Free, Model-Based techniques. These models will train on a simulated environment using the equations of motion and parameters we get from the system identification process.

We will work on two pairs each pair will train its own model and try to come up with ideas to improve on the model and get as creative as possible with it.

6 Time Plan

6.1 Subtasks

- Identify system parameters for the friction-free and friction model. **Estimated deadline**: 30/3
- Develop the code for the simulation environment. **Estimated deadline**: 3/4
- Develop the code for the Reinforcement learning models. **Estimated deadline**: 5/4
- Explore different models, choose a suitable model design and report on your results for each model. **Estimated deadline**: 19/4
- Tuning the chosen model until good results are achieved in the simulated environment. **Estimated deadline**: 25/4
- Train, test and improve the model on the real pendulum at the lab. **Estimated deadline**: 29/4
- Writing the report for the feedback . **Estimated deadline**: 3/5
- Training the model on performing tricks and be creative. **Estimated deadline**: 17/5
- A complete project report should be pushed to git and the supervisor notified. **Estimated deadline**: 24/5
- Prepare for the presentation. **Estimated deadline**: 1/6
- Revision of **Estimated deadline**: 7/6

6.2 Important dates

- 27/3 - Hand in project plan.
- 31/3 - Git repo in shape.
- 4/4 - First self evaluation.
- 5/4 - Feedback seminar 1 on the modeling and design.
- 3/5 - Preliminary report should be pushed to git to allow peer review by other groups.
- 9/5 - Feedback seminar 2 on the design and implementation.

- 24/5 - A complete project report should be pushed to git and the supervisor notified.
- 30/5 - Second self evaluation.
- 1/6 - Final presentation and demonstration.
- 7/6 - A final revised report should be pushed to git and the supervisor notified.

6.3 Gantt Chart

We have formalized the time plan as a Gantt chart. The major tasks can be seen plotted in Figure 1.

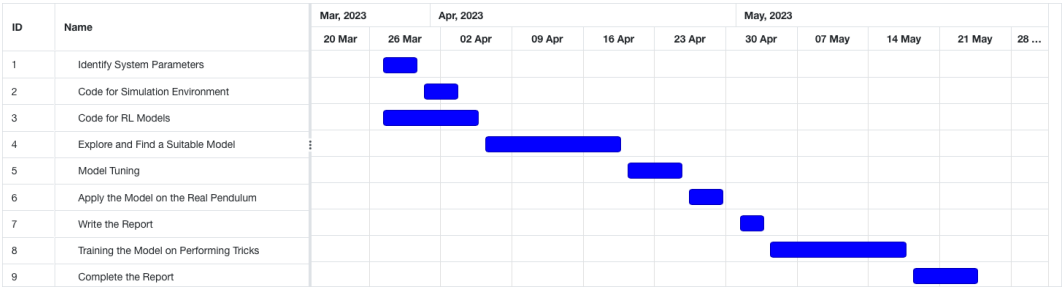

Figure 1: The Gantt chart describing the work flow of our project.

7 Risk Assessment

- The Reinforcement Learning model design is too ambitious and needs to be reconsidered. **Likelihood**: 80%. **Risk**: Wasting one or two weeks of development. **Contingency plan**: Don't spend more than one week trying to make it converge, switch to a simpler method.
- The model trained on the simulation doesn't generalize to the real world. **Likelihood**: 50%. **Risk**: Having to go back to the simulation and make the model more robust. **Contingency plan**: Add a control theory solution on top of RL.
- The training is working but taking too much time. **Likelihood**: 50%. **Risk**: Spending days waiting for the results. **Contingency plan**: Use a second pendulum or ask for a better server to handle heavy computing.
- The hardware breaks. **Likelihood**: 10%. **Risk**: Having to wait a couple of days (maybe a week) for repair or replacement. **Contingency plan**: Work on the simulation or on the report in the meantime.

References

[1] M. Gäfvert, *Modelling the Furuta Pendulum*. Technical Reports TFRT-7574, Department of Automatic Control, Lund Institute of Technology (LTH), 1998.