Machine Learning 2024/25

Homework 1: Robot kinematics

Master in Artificial Intelligence and Robotics

SAPIENZA
UNIVERSITÀ DI ROMA

Professor Luca Iocchi

Homeworks

Each homework will assign up to 2 points that will be added to the final score of the exam in any session within this academic year.

Homework points will remain valid independently of acceptance/failure in exam sessions.

Homeworks are not mandatory.

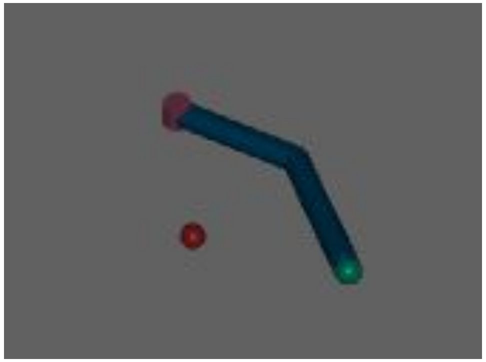
It is not possible to deliver homeworks outside the deadline given during the course.

Homework 1 : Robot kinematics

Deadline: **8/12/2024 23:59 CET** (STRICT DEADLINE!!!)

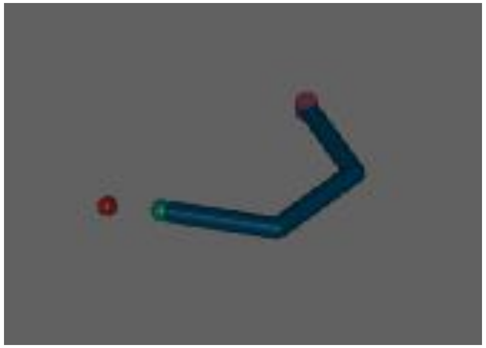
Problems

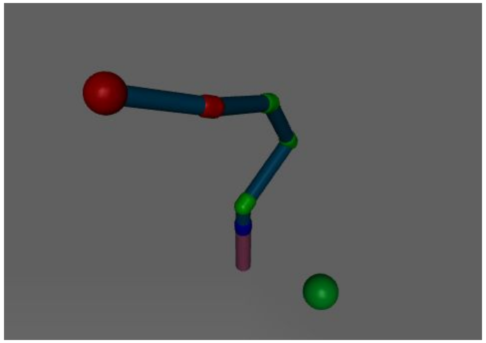
- **Robot forward kinematics**
- Robot inverse kinematics
- Robot control
- Reinforcement learning



Robot models (simulated in Mujoco)

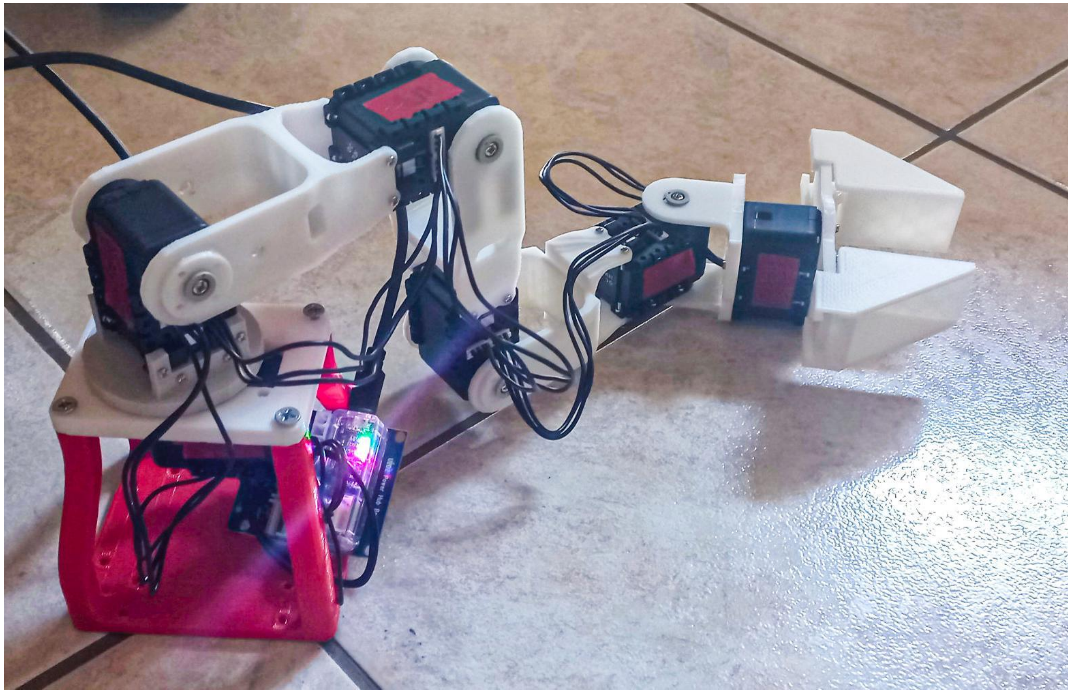
- 2D with 2 joints
- 2D with 3 joints
- 3D with 5 joints

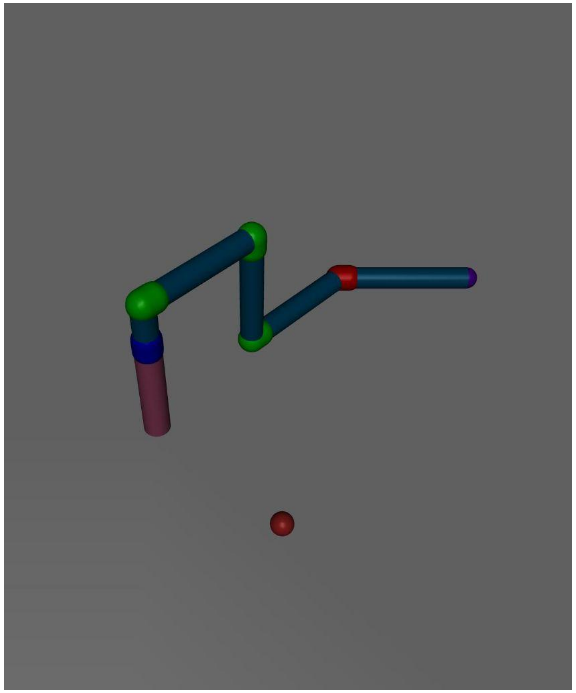




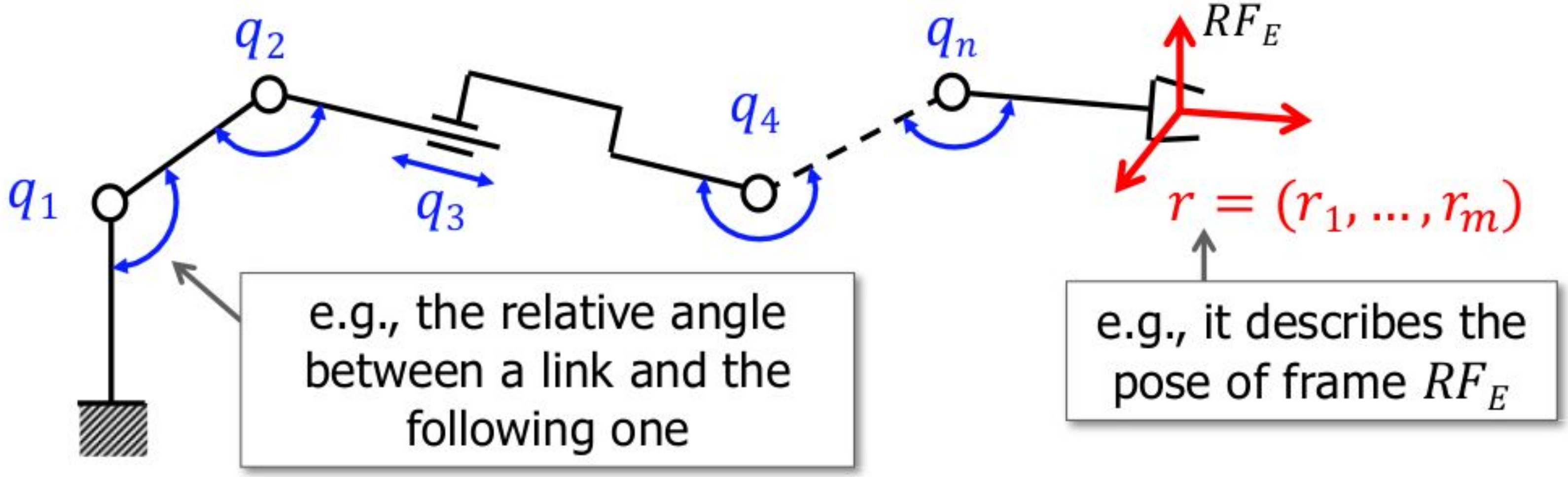
https://github.com/iocchi/MLHW1_robot_kinematics

MARRtino arm





Forward kinematics



Robotics 1 - Prof. Alessandro De Luca

Datasets

Generate 3 datasets using the three robot simulators

Data:
- joint angles
- fingertip position
- fingertip orientation

Build 3 datasets for forward kinematics (as regression)

- joint angles -> fingertip position (orientation)

Choose proper representations for angles and orientations

Preprocessing (if needed)

Datasets

Generated datasets are available in this folder
(use them only if you cannot run the simulator)

https://drive.google.com/drive/folders/1zLanaj-KU5J8UNK37ugSwtJt-fVGZUXw

Each file contains the log of a run with following structure of the filename

<model>_<seed>_<nr. of samples>.csv

E.g., `r2_20_100k.csv` is the file for robot environment `r2` (2 DOF robot) generated with seed `20` and containing `100k` samples

Implement a proper procedure to define training and test data from these files.

Note: consider to select a subset of the samples provided in the datasets (e.g., only 1k samples for the simpler robots) for a more challenging task.

Learning forward kinematics

Train a model for learning forward kinematics (FK) for each collected dataset

- define a model (e.g., feedforward NN) - possibly different for each robot
- choose a loss function
- choose a solver
- fit the model and test performance on validation data
  - random train-test split on the same log
  - train on logs with some seeds, test on logs with other seeds
- do some hyper-parameter search (at least 2 hyper-parameters and 4 combinations)

Compute Jacobian matrix J of the learned forward kinematics function

Compare learned J with analytical J (using robot model details)
at least for 2 DOF robot arm

Analytical forward kinematics and Jacobian

Forward kinematics for 2 DOF robot arm

$$x = L_1\, cos\theta_1 + L_2\, cos(\theta_1 + \theta_2)$$

$$y = L_1\, sin\theta_1 + L_2\, sin(\theta_1 + \theta_2)$$

Analytical Jacobian matrix $J = \begin{bmatrix} \frac{\partial x}{\partial \theta_1} & \frac{\partial x}{\partial \theta_2} \\ \frac{\partial y}{\partial \theta_1} & \frac{\partial y}{\partial \theta_2} \end{bmatrix}$ for 2 DOF robot arm

$$J = \begin{bmatrix} -L_1 \sin(\theta_1) - L_2 \sin(\theta_1 + \theta_2) & -L_2 \sin(\theta_1 + \theta_2) \\ L_1 \cos(\theta_1) + L_2 \cos(\theta_1 + \theta_2) & L_2 \cos(\theta_1 + \theta_2) \end{bmatrix}$$

In the simulated model $L_1 = 0.1, L_2 = 0.1$

Inverse kinematics (optional)

Solve inverse kinematics problem with the learned FK and J

current joints, target pos (orientation) -> target joints

Use the learned Jacobian matrix to implement an inverse kinematics algorithm

- Newton-Raphson
- Levenberg-Marquardt

Start from 2 DOF robot arm

Robot Control (optional)

Implement a controller to reach a specific joint configuration

current joints, target joints ->   control actions

Use the learned IK to implement robot control function

- PID controller

Integrate with inverse kinematics to reach a specific target position.

Use Mujoco simulation and target position to validate the controller

Start from 2 DOF robot arm

Deep Reinforcement learning (maybe later)

Use DRL to learn policies to reach a specific target position

current joints, target joints -> control actions

Start from 2 DOF robot arm

Simulator run with random action control

```
run.py [-h] [-env ENV] [-steps STEPS] [-seed SEED] [--render] [--log]

    optional arguments:
      -h, --help    show this help message and exit
      -env ENV      environment [r2,r3,r5] (default: r2)
      -steps STEPS  Execution steps (default: 10,000)
      -seed SEED    Random seed (default: 1000)
      --render      Enable rendering
      --log         Enable data log
```

File format

CSV file (separator ;)

- **joint angles**
- cos joint angles
- sin joint angles
- joint velocities
- **fingertip position**
- fingertip orientation
- target position

Read data

```
import pandas as pd

df = pd.read_csv('...csv',sep=';',header=0)

X = df[['j0', 'j1']].values # features
Y = df[['ft_x', 'ft_y']].values # target
```

Hints

Save trained model for next steps

```
# Save the learned model
model.save('fk.h5')
```

Load saved model

```
# Load saved model
model = tf.keras.models.load_model('fk.h5')
```

Hints

FK prediction (use tensors as input)

```
def FK(model,theta):
    # reshape to batch size 1
    t = tf.reshape(theta, shape=(1,2))
    out = model(t)
    # reshape to 1d vector
    out = tf.reshape(out, shape=(2,))
    return out
```

Jacobian

```
@tf.function
def FK_Jacobian(model,x):
  with tf.GradientTape(persistent=True) as tape:
    tape.watch(x)
    y = FK(model,x)
  return tape.jacobian(y, x)
```

Assignment through Classroom

Deliver through the assignment:

1) A report (PDF file)

2) A ZIP file with the code you used in the project.

3) [OPTIONAL] Videos of robot control

Assignment through Classroom

Report

- PDF file of about 10 pages excluding code, with your name and matricola code
- implemented solutions (for each robot)
  - how data have been generated/preprocessed
  - which methods/algorithms have been used
  - which configurations of the methods have been tried
  - performance metrics (e.g., plot results over training steps)
- hyper-parameter search
- results for different hyper-parameters
- computational training time
- conclusions and future work

Note: GenAI must be properly acknowledged!

Assignment through Classroom

Submit the files (PDF report, ZIPped code) through this assignment, make sure to turn the assignment in.

NOTES:

1) do ***NOT*** put the PDF report into the ZIP file!

2) no other submission mode will be considered (e.g. do ***NOT*** send submissions by email).

This assignment must be **individual** (i.e., one submission for each student) and **original** (i.e., not equal or too similar to other works either from other students in this class or from other sources).

Evaluation will be based on the appropriateness and correctness of the described solution, regardless of the numeric results (as long as they are reasonable).