Documentation

Author: Nikhil Podila
nikhil.podila@mail.mcgill.ca
McGill ID: 260905414

Assignment 2 ECSE 683

1 Robot

In this assignment, KUKA IIWA robot arm is chosen as the testing platform. The configuration space is the degree-of-freedom joints $\boldsymbol{q} \in \mathbb{R}^7$, and the task-space is the end-effector position $\boldsymbol{x} \in \mathbb{R}^3$.

The state is the position of the configuration space $x = [\boldsymbol{q}]$ and the action is the joint velocity, $u = \dot{\boldsymbol{q}}$. The trajectory is controlled through an inverse kinematic controller using Velocity Control method.

Since the task requirements in this assignment is in 2 dimensions, the value on x-axis for the end-effector position were fixed to $x_x = -0.4$

2 Problem statement

The goal of this assignment is to:

- Generate a trajectory of the end-effector that can reproduce the shape of 3 characters.
- Ensure that the trajectory is within the kinematics and dynamic constraints of the robot

3 Approach

The approach used to perform the motion planning is **Imitation Learning**. Specifically, a handwritten dataset is used in learning a Dynamical System. The Dynamical system is a Gaussian Mixture Model based Lyapunov Dynamical System with Quadratic Lyapunov Function [1].

In this approach, it is described that first a Gaussian Mixture Model is learned to determine the number of kernels $\mathcal{K}$ which best fits the data.

$$\mu_k, \Sigma_k, \pi_k = GMM(\xi)$$

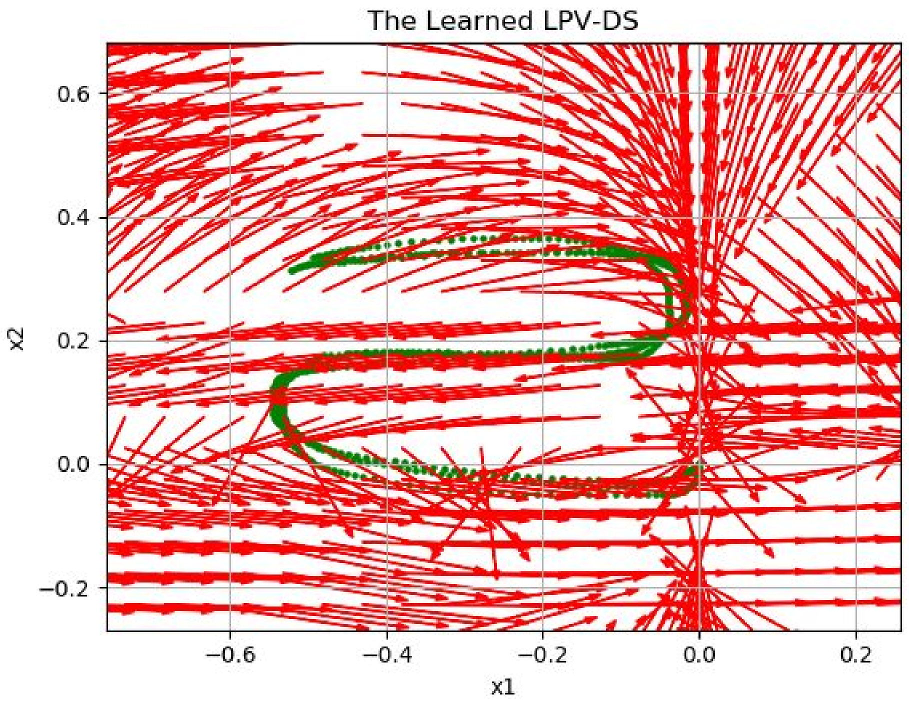

Figure 1: Letter "S" (mirrored) learned using the GMM based LPV-DS with QLF method

Then, the mean squared error between the model and the actual time derivate of the state is minimized with constraints that ensure that the model is a set of linear dynamical systems.

$$\underset{\boldsymbol{A}_k,\boldsymbol{b}_k}{min}\, \Sigma_{n=1}^{\mathcal{N}} \|\dot{\xi}_n - \mathcal{F}(\xi_n)\|^2$$
$$\boldsymbol{A}_k^T + \boldsymbol{A}_k \prec 0$$
$$\boldsymbol{b}_k = -\boldsymbol{A}_k \xi^*$$

An example Dynamical system learned is shown in Figure 1. In this case, a mirrored version of the figure is learned because the robot's arm is behind the letter, when it is being drawn. As a result, the robot must draw the mirrored figure so that the viewer can see the actual letter on the simulation.

The code for this approach was adapted from [2] and my Assignment 1 submission for ECSE 683 Fall 2020. The Inverse Kinematics based control applied on the KUKA IIWA Robot was adapted from the original code here [3].

4 Validation of Limits

One of the main objectives and the challenges in this assignment was to ensure that the robot is within its joint limits and validate the same.

To ensure that the planned method is within the velocity limits of the joints, first, the maximum joint velocity is converted to end-effector velocity by using the Jacobian of the Kinematic system. The planned reference end-effector velocity is then compared with this maximum end-effector velocity. If the planned velocity is higher, the previous reference velocity value is used instead. This prevents the joint velocity limits from being reached. This is implemented in the "limitReference()" function in the code.

To ensure that the planned method is within the position/angle limits of the joints, redundancy resolution method is used with the inverse kinematics controller, given by [4]:

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{\dagger}\boldsymbol{x} + \boldsymbol{N}\dot{\boldsymbol{q}}^{0} \text{ where,}$$
$$\boldsymbol{N} = \boldsymbol{I} - \boldsymbol{J}^{\dagger}\boldsymbol{J} \text{ is the nullspace projection matrix of } \boldsymbol{J}$$

This is implemented as a part of the "setRobotTaskReference()" function of the code.

The validation of the joint velocity and angle limits is performed using the "checkKinematicLimits()" function in the code, which returns True only when both the joint velocity limit and the joint angle limit are met.

5 Experiments and Results

The experiments are carried out on the Python language library - PyBullet [5]. The experiments were performed using the dataset provided in [2]. Three letters from the dataset were used to represent and show the learned trajectories of three letters in the dataset - "S", "L" and "V". In the case of letter "V", the dataset of letter "A" was chosen and inverted to obtain "V" as the dataset trajectories of letter "A" did not accurately represent the entire English letter "A".

Since the letters are 2-dimensional, the letters were drawn by the robot on the plane $x = -0.4$.

In all three letters, the trajectories were learned with enough accuracy such that the predicted trajectory of the robot can be legibly read as their corresponding letters, as shown in the videos in the assignment submission. In all three cases, the joint angle and velocity limits were obeyed. This can be seen by running the experiment and not noticing the message "Robot is not within its Kinematics limits" on the console output during the simulation.

6 Limitations

Although the algorithm satisfies the objectives of the assignment, it does not work all the time for all datasets. The algorithm still has a few limitations which could be improved in the future:

- The Gaussian Mixture model is not always accurate in determining the best number of kernels for the given dataset. This is especially true for letters where many different lines are close by. For example, this occurs at the center of letter R.
- The Dynamical system is a set of linear dynamical systems. Thus, the algorithm can be inaccurate in convergence in the state space outside the area covered by the handwritten dataset points.
- The Dynamical system is innaccurate in representing letters with edges, because of the sudden change in direction. In this case, at a single point, two possible directions exist.
- The Dynamical system is inaccurate in representing letters which have to be written by raising a human's hand. For example, The letter 'H' cannot be learned accurately by the system. Such letters have not been learned because of this method.
- Since the method relies on kinematic control, the robot dynamics and inertia are ignored. Any external forces or changes in robot inertia are unaccounted. This could be improved by performing dynamic control instead of kinematic based control.
- The letter sizes from the dataset are manually adjusted to suit the robot size and limits. If the letter sizes are not adjusted, the starting or target state of the robot may be beyond the reach of the robot. In such scenarios, the robot will fail to follow the accurate trajectory.

References

[1] S. M. Khansari-Zadeh and A. Billard, "Learning stable nonlinear dynamical systems with gaussian mixture models," *IEEE Transactions on Robotics*, vol. 27, no. 5, pp. 943–957, 2011.

[2] H.-C. Lin, "Learning dynamical systems example," 2020. [Online]. Available: https://github.com/McGIll-COMP-766-ECSE-683-Fall-2020/python-examples/tree/master/examples/07-dynamical-system

[3] E. Coumins, "Inverse kinematics example," 2020. [Online]. Available: https://github.com/erwincoumans/bullet3/blob/master/examples/pybullet/examples/inverse_kinematics.py

[4] H.-C. Lin, “02-kinematics-2,” 2020. [Online]. Available: https://drive.google.com/file/d/11SPBWiR_Bm4k2zOKCmIsYePWl1RFlYYt/view

[5] E. Coumans and Y. Bai, “Pybullet, a python module for physics simulation for games, robotics and machine learning,” http://pybullet.org, 2016–2019.