Comparison of LQR, MPC and Q-learning for the problem of controlling a Double inverted pendulum

Nagaraj Srinivasa Murthy
Department of Electrical, Computer and Energy engineering
Universoty of Colorado, Boulder
nasr2889@colorado.edu

***Abstract*—In this project, we aim to compare different methods of controlling a double inverted pendulum. The chosen system is complex and exhibits chaotic behavior but is also well studied in the domain of nonlinear controls. There are two main methods of controlling this systems, one involves moving the base or cart and the other is by applying a torque at the pivot point between the two pendulums. We first study a simple version of this system called Pendubot and control it with Linear Quadratic Regulator (LQR). We then study the full swing-up and balancing control of a double inverted pendulum on a cart with Model Predictive Control (MPC). Finally, we learn a controller for balancing the system in simulation.**

***Index Terms*—LQR, MPC, Q-learning**

I. Introduction

Dynamical systems extensively represent natural phenomena occurring anywhere [1]. These systems of ordinary differential equations in terms of the 'state' of the system are modeled by taking in real world data. A model is then built around this data and these models can then be used for prediction [2]. Control theory helps us to manipulate these systems to get the desired response. Modeling and system identification plays a very important role in understanding the behavior of the system before designing control algorithms. In some cases, it might not be feasible to get the model of the system. In such cases, developing a learning algorithm that can learn to control the system without prior knowledge of the system is very promising.

A simple one link inverted pendulum is a nonlinear system that is well studied in control theory. Previous work in designing control algorithms include designing a fuzzy logic control for inverted pendulum [3], a simple PID control for the system [4] and even optimal control approaches like in [5]. In our case, we consider the double inverted pendulum which is more complex and chaotic. A nonlinear quadratic regulator (NQR) has been successfully designed in [7] which is based on stabilizing the pendulum between the two target points with respect to its trajectory.

Reinforcement learning finds its applications in solving control problems. It is the study of stochastic programming methods for Markov Decision Processes (MDPs) in several applications where accurate analytical models of the complete system are difficult to obtain. Several optimal control strategies like LQR control are valid only within a small neighborhood of the target points and fails beyond these neighborhood. A combination of PID and Q-learning has been implemented in [8] for the swing-up problem of the system. Another approach which gives promising results for the considered system can be seen as a model-based reinforcement learning algorithm called Model Predictive Control (MPC). A real time swing-up controller is successfully designed using Nonlinear Model Predictive Control approach in [6].

Reinforcement learning method shows promising results for the control of double inverted pendulum in the sense that it quickly adapts to external disturbances, model uncertainties and internal noises. The goal of the reinforcement learning agent is to maintain the pendulum in the target position while exploring different actions in terms of force applied to the cart.

II. Double inverted pendulum on a cart

The system consists of two joint pendulums attached to a cart that can move horizontally on a track, see figure 1 below.

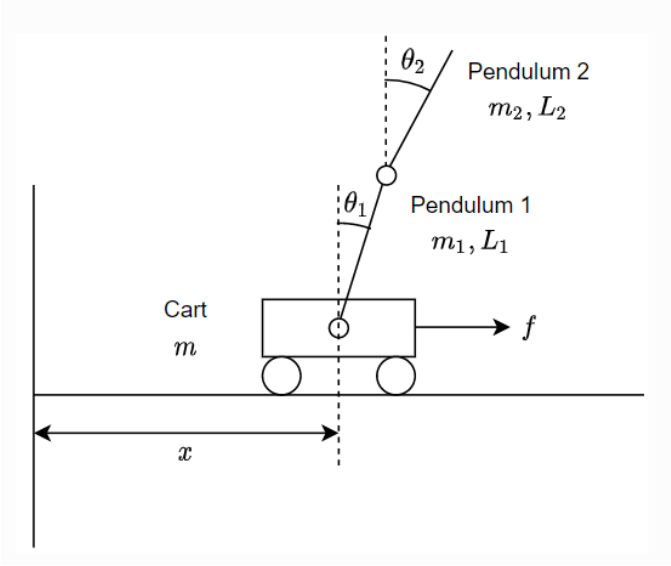

Fig. 1. Double inverted pendulum on a cart.

The system is described in terms of its horizontal position x, and the two angles θ, where $\theta_1 = \theta_2 = 0$ denotes the upright position.

Thus, the state can be defined as $s \in \mathbb{R}^6$ as follows,

$$s = \begin{bmatrix} x & \theta_1 & \theta_2 & \dot{x} & \dot{\theta}_1 & \dot{\theta}_2 \end{bmatrix}, \tag{1}$$

with the input u defined as the horizontal force applied to the cart.

$$u = f \tag{2}$$

The dynamics of the double inverted pendulum can be derived from the Euler-Lagrangian equations, which is assumed to be of the following form,

$$M(x)\dot{x} = A(x)x + Bu \tag{3}$$

where, it can be shown that

$$\det(M) > 0, \forall x \tag{4}$$

such that we can obtain the ODE:

$$\dot{x} = M(x)^{-1}(A(x)x + Bu) \tag{5}$$

Please see [9] for the complete derivation of the equations of motion. With the dynamics defined, we can move on to the optimal control approaches to stabilize the system and MDP formulation for learning based control.

A. LQR control

The Linear Quadratic Regulator (LQR) is a state-feedback controller, which means the control input is a function of the current state of the system. This controller computes the optimal feedback gains based on the system model and a cost function which is usually quadratic. This controller is very effective in controlling linear systems which means the nonlinear system must be converted to a linear model through the process of linearization.

For our problem, we wish to stabilize the pendulum in the upright position where $\theta_1 = \theta_2 = 0$. Considering only the first pendulum actuated, we aim to stabilize the second pendulum in upright position. This is a simple version of the double inverted pendulum considered above but still gives the same insight into the process.

We linearize the model around the upright position by computing the jacobians of system matrices with respect to the states and control input. Once we have the linear model in the form,

$$\dot{x} = Ax + Bu \tag{6}$$

We can use state feedback control u as,

$$u = -Ks \tag{7}$$

where K is the gain matrix obtained by solving the Reccatti equation and minimizing the quadratic cost function given by,

$$J = \int_0^\infty \left(s^T Q s + u^T R u\right) dt \tag{8}$$

where s is the state vector, u is the input vector, Q is the state weighting matrix, and R is the control weighting matrix.

In the optimal control of nonlinear inverted pendulum dynamical system using LQR approach, all the instantaneous states of the pendulum, angle θ, angular velocities, cart position x , and cart velocity $\dot{x}$ have been considered available for measurement which are directly fed to the LQR. The LQR is designed using the linear state-space model of the system.

To verify the performance of the controller in the upright position, a small disturbance is added to the pendulums and the controller can be seen negating the effects of the disturbance, see figure 2 below. The gain matrix obtained for the problem is given by,

$$K = [-164.98126238 - 159.27636722 - 70.22294604 - 40.75935388] \tag{9}$$

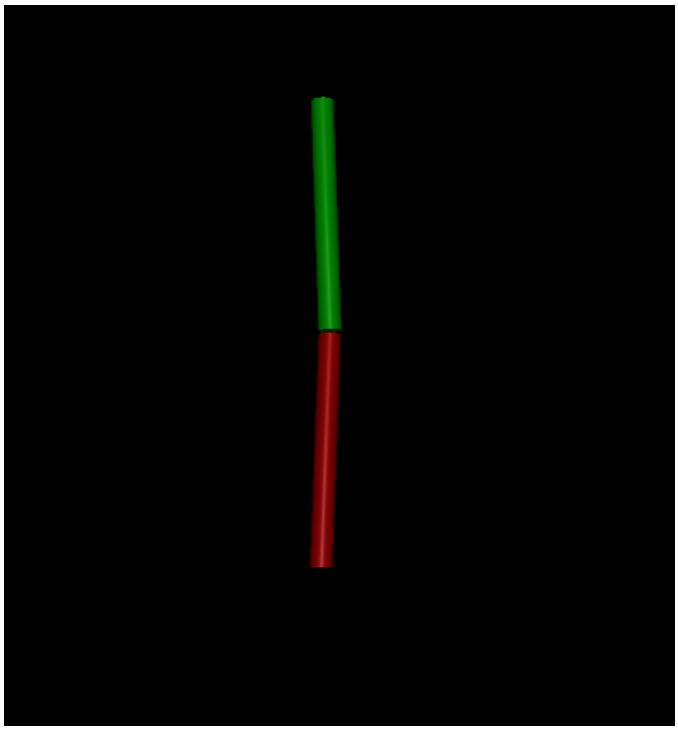

Fig. 2. LQR control of the system.

B. Model Predictive Control

Model predictive control (MPC) is a control scheme where a model is used for predicting the future behavior of the system over finite time window, the horizon. Based on these predictions and the current measured/estimated state of the system, the optimal control inputs with respect to a defined control objective and subject to system constraints is computed. After a certain time interval, the measurement, estimation and computation process is repeated with a shifted horizon. This is the reason why this method is also called receding horizon control (RHC).

The system model plays a central role in MPC. For the case of continuous systems, trying to solve the optimal control problem directly is in the general case computationally intractable because it is an infinite-dimensional problem. a full discretization method, namely orthogonal collocation, to discretize the problem gives promising results.

The control objective is to erect the double pendulum and to stabilize it in the up-up position. It is not straight-forward to formulate an objective which yields this result. Classical set-point tracking, e.g. with the set-point:

$$\theta_s = [0, 0, 0] \tag{10}$$

and a quadratic cost function like the one defined in (8) is known to work very poorly. Instead we will use an energy-based formulation for the objective. If we think of energy in terms of potential and kinetic energy it is clear that we want to maximize the potential energy (up-up position) and minimize the kinetic energy (stabilization). For the kinetic energy, we have:

$$E_{\text{kin.,cart}} = \frac{1}{2} m \dot{x}^2 \tag{11}$$

$$E_{\text{kin.},p_1} = \frac{1}{2} m_1((\dot{x} + l_1 \dot{\theta}_1 \cos(\theta_1))^2 + (l_1 \dot{\theta}_1 \sin(\theta_1))^2) + \frac{1}{2} J_1 \dot{\theta}_1^2 \tag{12}$$

$$\begin{aligned} E_{\text{kin},p_2} = \frac{1}{2} m_2 ((\dot{x} + L_1 \dot{\theta}_1 \cos(\theta_1) \\ + l_2 \dot{\theta}_2 \cos(\theta_2))^2 + (L_1 \dot{\theta}_1 \sin(\theta_1) + l_2 \dot{\theta}_2 \sin(\theta_2))^2) \\ + \frac{1}{2} J_2 \dot{\theta}_1^2 \end{aligned} \tag{13}$$

and for the potential energy:

$$E_{\text{pot.}} = m_1 g l_1 \cos(\theta_1) + m_2 g (L_1 \cos(\theta_1) + l_2 \cos(\theta_2)) \tag{14}$$

The closed loop system is simulated for about 100 steps with an initial state of the system as the bottom stable position. The result can be seen in figure 3 where solid lines are the recorded trajectories and dashed lines are the predictions of the scenarios.

III. MDP FORMULATION

The state of the MDP is simply the state $s \in \mathbb{R}^6$ of the system as given by (1) above. The state space is continuous. The agent's actions correspond to choosing the right amount of force to be applied to the cart so that the pendulum is stabilized in the upright position. The state transition probabilities are unknown to the agent, but intrinsically given by (5) above. To specify the rewards, we specify two different types of costs and then set the reward to equal the negative total cost. The first type of cost is associated with every failed episode. For swing-up, we define an episode as failed once the pendulum system moves out of bound. For balancing, an episode fails once the pendulum falls over, that is, $\theta_1 > \frac{\pi}{2}$ The second type of cost is associated with each state and penalizes any position of the pendulum system such that,

$$[x \quad \theta_1 \quad \theta_2] \quad \neq \quad [0 \quad n2\pi \quad m2\pi] \tag{15}$$

where $n, m \in \mathbb{Z}$. We also define the discount factor as $\gamma = 0.99$.

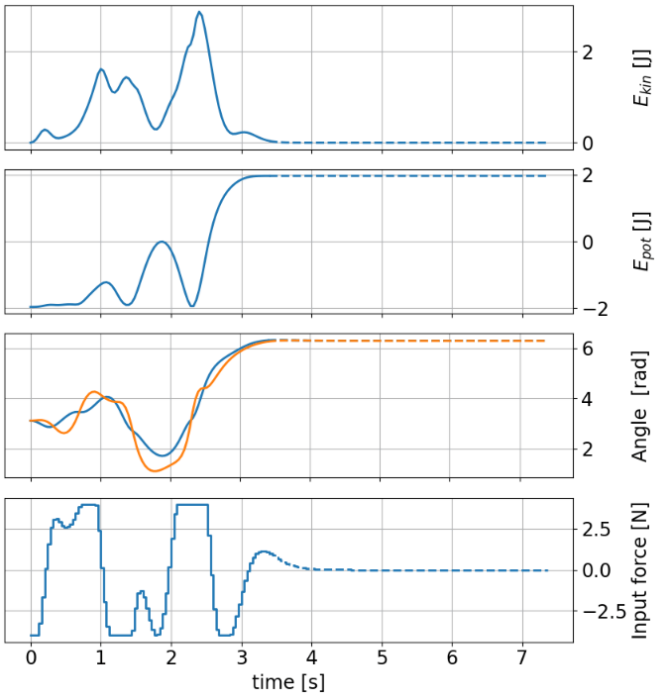

Fig. 3. Model Predictive Control of the system.

A. Linear Q-learning

Linear Q-learning is a reinforcement learning algorithm used to find an optimal policy for a given environment. In contrast to traditional Q-learning, which uses a table to store the expected rewards for each state-action pair, linear Q-learning uses a linear function approximation to estimate the Q-values. The linear function approximation takes the form of a weighted sum of features, where the weights are learned through an iterative process that updates the Q-values based on the observed rewards and transitions. The features used in the approximation are typically derived from the state and action space of the environment.

During each iteration, the agent selects an action based on an exploration-exploitation policy, updates the estimate of the optimal Q-value for the current state-action pair, and updates the weights of the linear function approximation. That is, we approximate the true optimal action-value function q_π by a linear combination of n state-action features. This is given by,

$$q_\pi(s, a) \approx \sum_{j=1}^{n} w_j x_j(s, a) = x(s, a)^T w = \hat{q}(s, a, w), \tag{16}$$

where $x(s, a) \in \mathbb{R}^n$ is the feature vector. Given the features x, the goal is to choose the weights $w \in \mathbb{R}^n$ such that the error between $\hat{q}$ and q_π is minimised. A stochastic gradient descent update rule yields,

$$w \leftarrow w + a(q_\pi(s, a) - \hat{q}(s, a, w)) x(s, a), \tag{17}$$

where $\alpha \in \mathbb{R}$ is the step size for the optimization. Since $q_\pi(s,a)$ is not known, we replace it by the Q-learning target and obtain the actual update rule for w:

$$\triangle w = \alpha(r' + \gamma \max_{a'} \hat{q}(s',a',w) - \hat{q}(s,a,w))x(s,a) \quad (18)$$

$$w \leftarrow w + \triangle w \quad (19)$$

where r' is the immediate reward associated with taking action a in state s, and s' is the successor state. Note that the action space A in this case is discrete, thus turning the $\max_{a'}$ operation in (18) into a simple comparison.

If in each time step we improve the policy by acting ϵ-greedily with respect to $\hat{q}$, meaning that we choose $a = \text{argmax}_{a'} \hat{q}(s,a',w)$ with probability $1-\epsilon$ and a random action in A with probability ϵ, and apply the above update rule, we obtain the linear Q-learning algorithm described in Algorithm 1 below.

initialize parameters $\mathbf{w}_0$;
repeat
  start new episode;
  initialize state $\mathbf{s}_0$;
  $t = 0$;
  repeat
    choose action a_t from $\mathbf{s}_t$ by ε-greedy policy w.r.t to $\hat{q}(\mathbf{s}_t, a', \mathbf{w}_t)$;
    take action a_t;
    observe r_{t+1} and $\mathbf{s}_{t+1}$;
    $a = \arg\max_{a'} \hat{q}(\mathbf{s}_{t+1}, a', \mathbf{w}_t)$;
    $\delta = r_{t+1} + \gamma \hat{q}(\mathbf{s}_{t+1}, a, \mathbf{w}_t) - \hat{q}(\mathbf{s}_t, a_t, \mathbf{w}_t)$;
    $\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha \delta \mathbf{x}(\mathbf{s}_t, a_t)$;
    $t \leftarrow t + 1$;
  until *failed episode or maximum number of time steps reached*;
until *convergence*;

Algorithm 1: Linear Q-learning.

B. Simulation environment

The physical system is simulated using a modified version of the CartPole-v0 environment in OpenAI Gym [10]. In each time step, the agent gets to choose an action $a = u$ which is applied to the system in its current state s_t . The system is then propagated to the next time step using the dynamic equations in (5) above and numerical integration using the Runge-Kutta method. The agent now receives an observation of this new state s_{t+1} of the system, as well as the immediate reward r_{t+1} associated with taking the chosen action a in the original state s_t. The current state is also rendered in each time step, see figure 4.

C. Results with Linear Q-learning

In each new episode, the state was initialized by setting,

$$s_0 = \begin{bmatrix} 0 & \theta_{1_0} & \theta_{2_0} & 0 & 0 & 0 \end{bmatrix}, \quad (20)$$

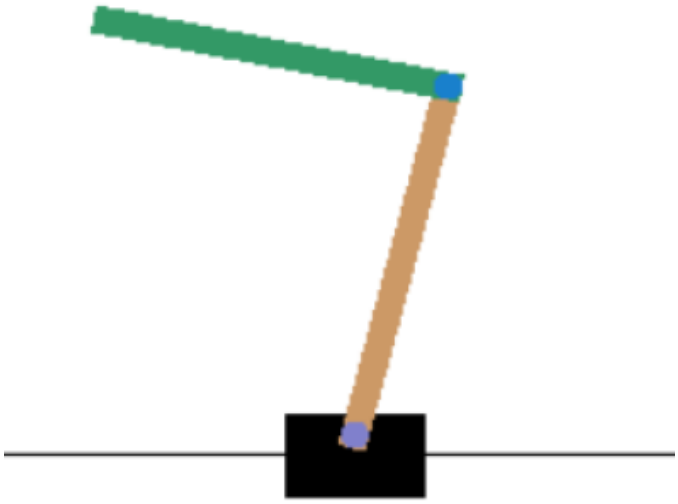

Fig. 4. The simulation environment

where $\theta_{1_0}, \theta_{2_0} \sim \mathbb{U}[-\lambda, \lambda]$. During training, $\lambda = 0.1$. A maximum limit to the number of time steps in a single episode was set to 300. If this limit was reached, the episode was considered successful.

The following results were obtained with exploration probability $\epsilon = 0.01$, step size $\alpha = 0.0001$, action space $A = \{0, \pm 5, \pm 10, \pm 20, \pm 40\}$ and the parameters w initialized to the zero vector. The state-action features used to approximate the action-value function were,

$$x(s,a) = \begin{bmatrix} f_1 & \ldots\ldots\ldots\ldots & f_{18} \end{bmatrix}, \quad (21)$$

where the features f_1 to f_{18} are given in table 1 below. In table 1, $normalAngle(\phi)$ gives the difference between ϕ and its closest multiple of 2π.

Table 1: Features for linear Q-learning.

$f_1 = \theta_0^2$	$f_2 = \theta_0$
$f_3 = normalAngle(\theta_1)^2$	$f_4 = normalAngle(\theta_1)$
$f_5 = normalAngle(\theta_2)^2$	$f_6 = normalAngle(\theta_2)$
$f_7 = \dot{\theta_0}^2$	$f_8 = \dot{\theta_0}$
$f_9 = \dot{\theta_1}^2$	$f_{10} = \dot{\theta_1}$
$f_{11} = \dot{\theta_2}^2$	$f_{12} = \dot{\theta_2}$
$f_{13} = a\theta_0$	$f_{14} = a\dot{\theta_0}$
$f_{15} = a\theta_1$	$f_{16} = a\dot{\theta_1}$
$f_{17} = a\theta_2$	$f_{18} = a\dot{\theta_2}$

With the above parameters, the algorithm converged in 69% of the training attempts. A typical learning curve is given below. All the codes and results are available at GitHub repo.

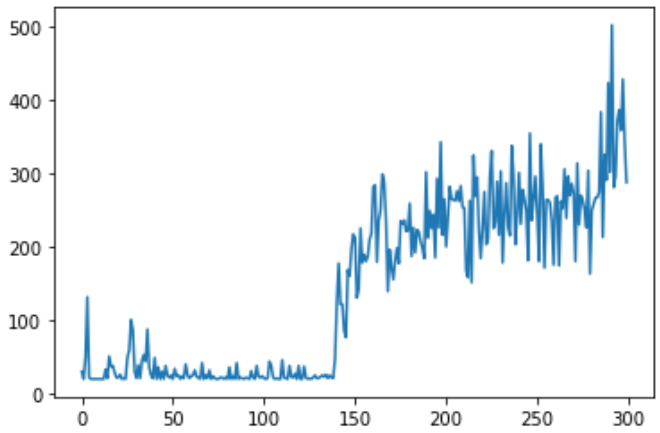

Fig. 5. Learning curve

IV. Conclusion

We have applied linear function approximators to solve the simple problem of balancing the double inverted pendulum in the upright unstable position. Though other approaches like nonlinear model predictive control gives better results, they are highly model dependent. This learning algorithm gives an optimal response without knowing the internal dynamics. Another approach which gives better results for both swing-up and balancing control of this system would be Soft Actor-Critic (SAC) algorithm. In SAC, the policy is represented by a neural network that takes the current state of the system as input and outputs the action to be taken. The policy is learned using a combination of gradient descent and maximum entropy reinforcement learning. The maximum entropy reinforcement learning objective encourages exploration and helps to prevent the policy from becoming overly deterministic. It also uses a critic network to estimate the value of the current state and action, and a target network to estimate the value of the next state. The critic network is used to update the policy network using the soft Q-learning algorithm, which is a variant of the Q-learning algorithm that uses a soft Bellman update to avoid overestimation of the Q-values. Implementing this algorithm would be the next steps.

References

[1] D. Luenberger. Introduction to Dynamic Systems-Theory,Models and Applications. JohnWiley Sons, NewYork, 1979.

[2] T.Duriez, S.Brunton and B.Noack .Machine Learning Control-Taming Nonlinear Dynamics and Turbulence. Springer International Publishing, Switzerland, 2017.

[3] Ahmad Ilyas Roose, Samer Yahya, Hussain Al-Rizzo, Fuzzy-logic control of an inverted pendulum on a cart, Computers Electrical Engineering, Volume 61, 2017, Pages 31-47, ISSN 0045-7906, https://doi.org/10.1016/j.compeleceng.2017.05.016.

[4] Mohammad Ali Eizadian, Majid Naseriyan, CONTROL OF INVERTED PENDULUM CART SYSTEM BY USE OF PID CONTROLLER, Department of Electrical engineering, ISSN 1013-5316.

[5] Jacobe Harris, Mojtaba Yazdani, Development and Control of Double inverted pendulum, Mechanical engineering, University of Utah.

[6] Pathompong Jaiwat, Toshiyuki Ohtsuka, Real-time swing-up of Double inverted pendulum by Nonlinear Model Predictive Controller, ADCONIP, Hiroshima, 2014.

[7] Neusser, Z., Valášek, M. (2013). CONTROL OF THE DOUBLE INVERTED PENDULUM ON A CART USING THE NATURAL MOTION. Acta Polytechnica, 53(6), 883–889. https://doi.org/10.14311/AP.2013.53.0883

[8] A. Zeynivand and H. Moodi, "Swing-up Control of a Double Inverted Pendulum by Combination of Q-Learning and PID Algorithms," 2022 8th International Conference on Control, Instrumentation and Automation (ICCIA), Tehran, Iran, Islamic Republic of, 2022, pp. 1-5, doi: 10.1109/ICCIA54998.2022.9737201.

[9] Bogdanov, Alexander. (2004). Optimal Control of a Double Inverted Pendulum on a Cart.

[10] OpenAI Gym. https://gym.openai.com/