INVERTED PENDULUM CONTROL USING BAYESIAN NETWORKS

UNIVERSITY OF MARYLAND

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE COMPLETION OF COURSE ENPM808F-ROBOT LEARNING

BY

GOWTHAM RAJ GUNASEELA UDAYAKUMAR

Abstract

The paper discusses the Bayesian approach for the problem of inverted pendulum control. The method alleviates the planning in action space encountered in Markov Decision Process. The algorithm for the control of inverted pendulum builds upon the Markov Decision Process and models the control parameters with Bayesian Networks. The state space parameters of the pole position are further implemented using a binary tree data structure to reduce the running time complexity of MDP based algorithms. Further the implications of node failure occurring at a given level is discussed with the possible regeneration strategy.

Introduction

Inverted pendulum has always been a classical control problem that has long evaded the control theorists. There has been a strategic improvement in the control procedure for the pendulum model. The evolution of machine learning algorithm has made the inverted pendulum model ever more interesting. There has been wide range of machine learning strategies implemented that range from supervised learning to deep networks to control the cart position. Markov decision Process is one such popular methodology adopted wide over for its simplicity in realizing over real-time and model based approach for controlling. Though the algorithm offers wide range of advantages, it suffers from two big limitations that limits its capability as the most efficient strategy. The first one being the effective trade-off between exploration and exploitation and the algorithm takes a running time in polynomial factor to converge to solution.

The goal of the project is to build a Bayesian belief networks to extend the MDP algorithm such that the tradeoff between exploration and exploitation is managed. Then the algorithm is implemented in a binary tree structure to reduce the complexity of the algorithm.

Literature Review

I. CONTROL SYSTEMS METHODOLOGY

Classical control methods involve PID control algorithms [1] that balance the cart in its upright position. Feedback is a basic concept in system control. In applying a control input to the cart-pole system, the controller usually considers the behavior of the system and bases its control inputs on the measured outputs of the plant (the system under control). Control based on feedback is called closed loop control. A closed loop control as implemented by control theorists for solving the problem considers the cart position at every time step.

The figure below shows a simple inverted pendulum which has pole of length l attached at an angle theta to the cart with mass M. The PID control will consider the frequency response of the problem which is focused on the control of the pendulum's position. Since the assumption here is a single-input, single-output (SISO) systems, the design criteria have been made simpler. The controller design will try to restore the vertically upward position after having an impulse force to the stable position. The design considerations here also include that the stable position be reached within 5

seconds and the deviation of the pole be not more than 0.05 radians from the mean position. The equilibrium can be assumed for either the position θ=0 or θ=pi.

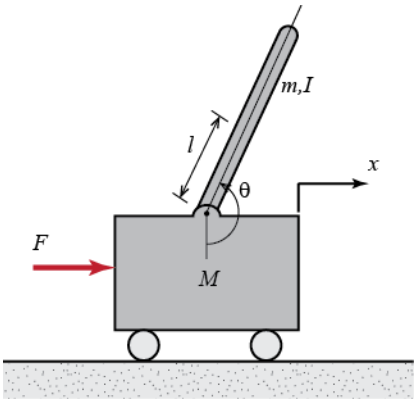

Fig 1. An inverted pendulum

A more complex version of the control algorithm can be implemented considering a single-input, multi-output (SIMO) system. Here the attempt is to control the both the cart's position and the pendulum angle. Force analysis and system of equations will follow the above design considerations which will be generated using Newtonian mechanics. Alternatively, Lagrangian mechanics can be used to find the system of equations using the total energy of the system. The non-linear equations will then be simplified to linearized system equations and represented using Transfer Function or state space format. The system can now be programmed using Matlab or other programming environment a series of command inputs that satisfies the equations and maintains controllability and observability.

II. MACHINE LEARNING METHODS

The dawn of machine learning also paved a new path for the inverted pendulum problem. As a classic problem, it caught the eye of most machine learners who developed an extensive set of algorithms that range from locally-weighted regression, genetic algorithm to deep networks where several layers of pendulum states are modeled.

For Example, Charles W. Anderson [3] in his paper had done learning to control an Inverted Pendulum using Neural Networks. The learning assumes no prior knowledge of the dynamics. He has made a significant change in his approach to the pendulum problem using neural networks in that he has assumed that the feedback is unavailable on each step. It appears only as a failure signal when the pendulum goes out of the boundary of its horizontal track. These assumptions should deal with issues of delayed performance evaluation, learning under uncertainty, and most importantly learning of non-linear functions. The network and the related computations are performed by each unit that specify a function from input to output vectors. The function will then be parametrized by the connection weights between units and on the inputs to the network. The function can be altered by the learning method that changes the values of the weights. If the desired output is not available, the performance of the network must be analyzed and corresponding

weights adjusted taking into consideration the effect of the changed output and its interaction with the outside environment.

Markov Decision Process

Markov Decision Process is a standard mathematical framework for modeling decision making situation where the outcomes are stochastic and involves the decision maker's control. MDP is a widely-followed algorithm for optimization problems and hence one of the standard approaches for the inv0erted pendulum control. To be specific in terms of definition, Markov Definition Process is a discrete time stochastic control process. A Markov Decision Process in other words obeys Markov property that states that a process is said to have Markov property if the conditional probability distribution of future states of the process depends only upon the present state and not on the sequence of states that preceded it.

Mathematically, at each time step, the process is in state s and the decision maker can choose an action a that is available from the state s. The process then randomly chooses to move into a new state s' and gives the policy maker a reward R(s, s'). The movement to new state is determined by the probability that will improve the reward for reaching the goal state.

The Markov decision process is treated as a 5-tuple,

(S, A, P(s, s') , R(s, s'), γ)

S is a finite set of states,

A is a finite set of actions

P is the probability that action 'a' in state 's' at time t will lead to state s' at time t+1.

R is the immediate reward after the transition from state 's' to state s' due to action 'a'.

γ is the discount factor that exploits the difference in importance between future rewards and present rewards.

$$\text{Policy, } \pi = S \rightarrow A$$

The approach of MDP is to find a policy for the decision maker that will choose the optimum set of actions for transforming from initial state to goal state. The policy π will calculate cumulative function of the random rewards that maximizes the value. The approaches so far calculate V(s) and π that holds the discounted sum of rewards and the set of actions respectively. Value iteration and policy iteration are among the other notable methods that are used for solving MDP problem. We will focus on the value iteration approach that calculates π(s) instead of π whenever V(s) is to be found. The process is repeated until V converges to satisfy the equation (also Known as Bellman Equation).

$$V(s) := \sum_{s'} P_{\pi(s)}(s, s') \left(R_{\pi(s)}(s, s') + \gamma V(s')\right.$$

MDP has an advantage of converging faster to the solution and can explore more states in each time. The serious limitations of the process are its inefficiency in trade-off between exploration and exploitation and a running time complexity of O(n*n2) (a polynomial running time).

The goal of the project will be to overcome the limitations imposed by Markov Decision Process for learning the control of inverted pendulum.

Bayesian Belief Networks

A Bayesian belief network is a graphical representation of probabilistic dependency model. The model represents variables through nodes which are interconnected and the connecting arcs represent the casual relationships between these variables. a node will represent one of the number of variables. The node may then take one of the number of possible states available to it. The belief also known as certainty or closer to likelihood in each state will be determined from the belief in each state to every other possible state connected to it. The belief in BBN of a node is updated only when the belief in corresponding state of the directly connected node changes.

Here, the problem of inverted pendulum is modeled into belief networks that models cart position and pole angle amongst other states as beliefs.

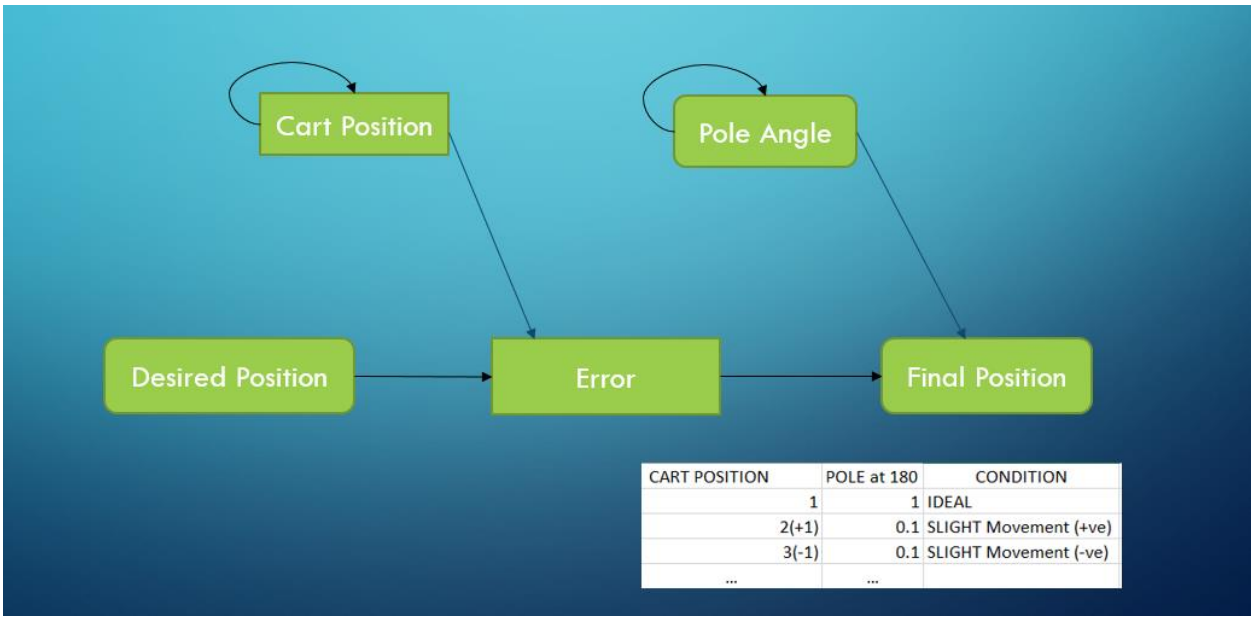

Fig 2. Modeling the Bayesian belief network from two network parameters

As observed from the picture, the cart position, pole angle is modeled as states that take beliefs.

Mathematically,

- A Bayesian analysis involves the estimation of prior probability $Pr(\theta)$ and likelihood $Pr(x|\theta)$ to compute posterior probability,

$$Pr(\theta|x) = Pr(\theta) * Pr(x|\theta)$$

- Likelihood of a set of parameter values, θ, given outcomes x, is equal to the probability of those observed outcomes given those parameter values, calculate posterior probability.
- Posterior probability is the assigned conditional probability after taking the evidence into consideration.

Upon reaching the state, the posterior probability updates its value which replaces the transition probabilities of the MDP. The posterior probability update gives the algorithm certain states not to be repeated for learning where the ideal position is found by moving to the state that has highest probability. This reduces the complexity of the controller and brings the running time to Quadratic measure. The tradeoff between exploration and exploitation has been taken care by the planning of state transition in the belief space instead of state space where the action happens. Hence an effective value of learning rate has been obtained that eliminates the conflict between search for immediate reward and convergence rate.

Binary tree structure

Though BBN eliminates the conflict in the trade-off for planning, the running time of the algorithm is still quadratic in nature which is better than polynomial time but can certainly be improved. The implementation of the algorithm is now made into a binary tree structure. Binary Tree is a special data structure used for data storage purposes. A binary tree has a special condition that each node can have a maximum of two children. The nodes of the tree will be the states of the pendulum. The initial branch will be the direction of the cart movement into positive or negative direction. Further branches will split the states into small sub-states that define the position of the pendulum with respect to the reference position. The pole will have 108 states that will be boxed by the controller.

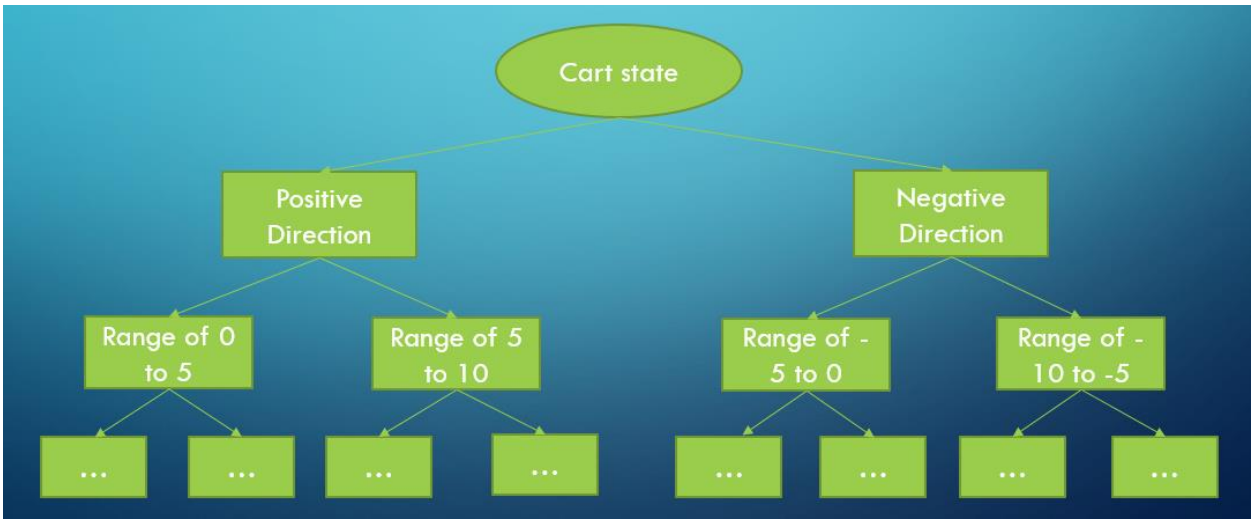

Fig 3. Binary tree structure for the Inverted Pendulum problem

The figure shows the sub states that contain the various cart position as nodes. This implementation improves the running time to O(nlogn) which is efficient than the quadratic running time of the Bayesian network algorithm.

Conclusion & Future work

The goals of the project have been to overcome the limitations of the Markov Decision Process for designing the controller for the inverted pendulum that is optimized. Modeling the pendulum as belief networks exploited the states in belief space thus eliminating the trade-off limitation. The improved algorithm then uses binary tree structure to improve the complexity of the algorithm by reducing the running time to logarithmic scale. The simulation has been run through Python and an animation has been generated that shows the control of inverted pendulum over a course of period after the learning has been done. The future work will be focused on tree generation when one random node in the tree fails. Regeneration of complete tree will increase the complexity and hence moving forward, the work will be focused on generating only the specific branch where the node has been corrupted to keep the regeneration time in linear scale.

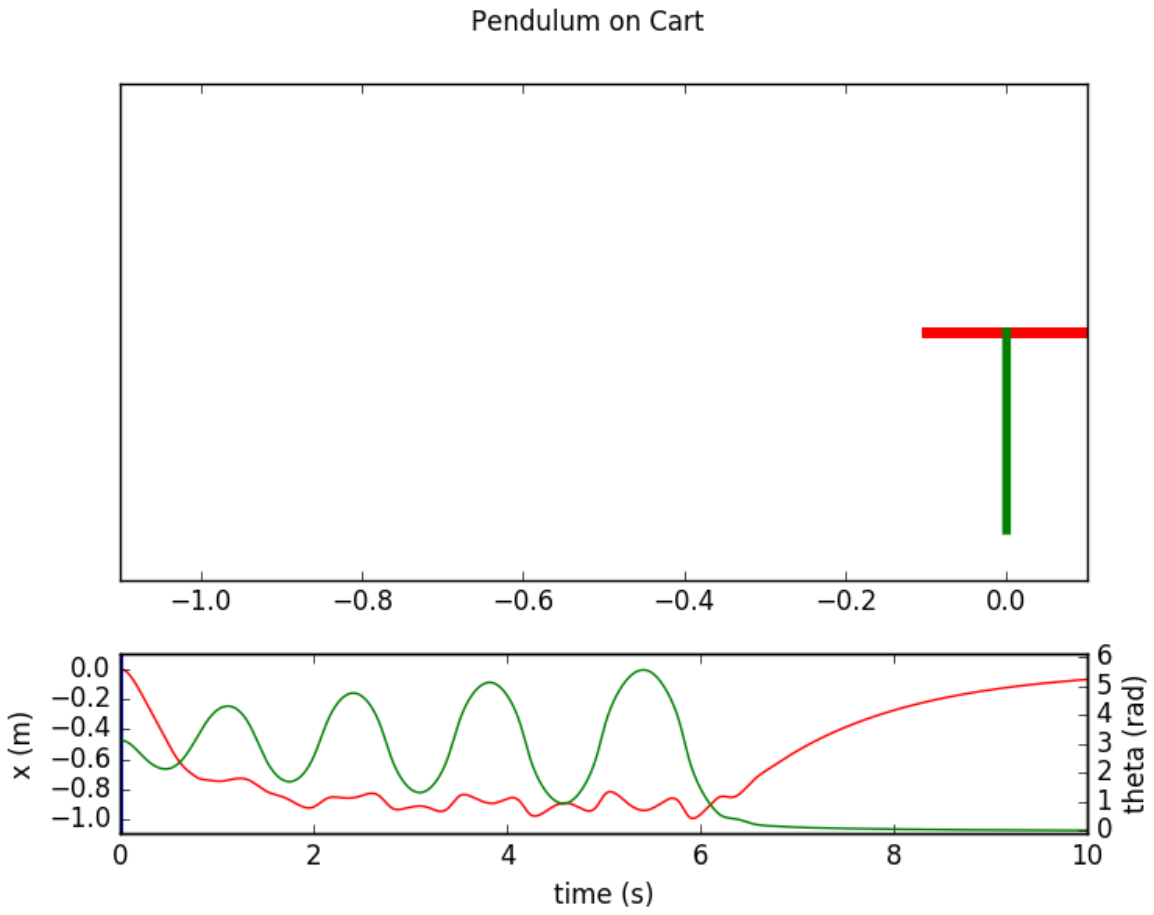

Fig 4 a. Initial cart position

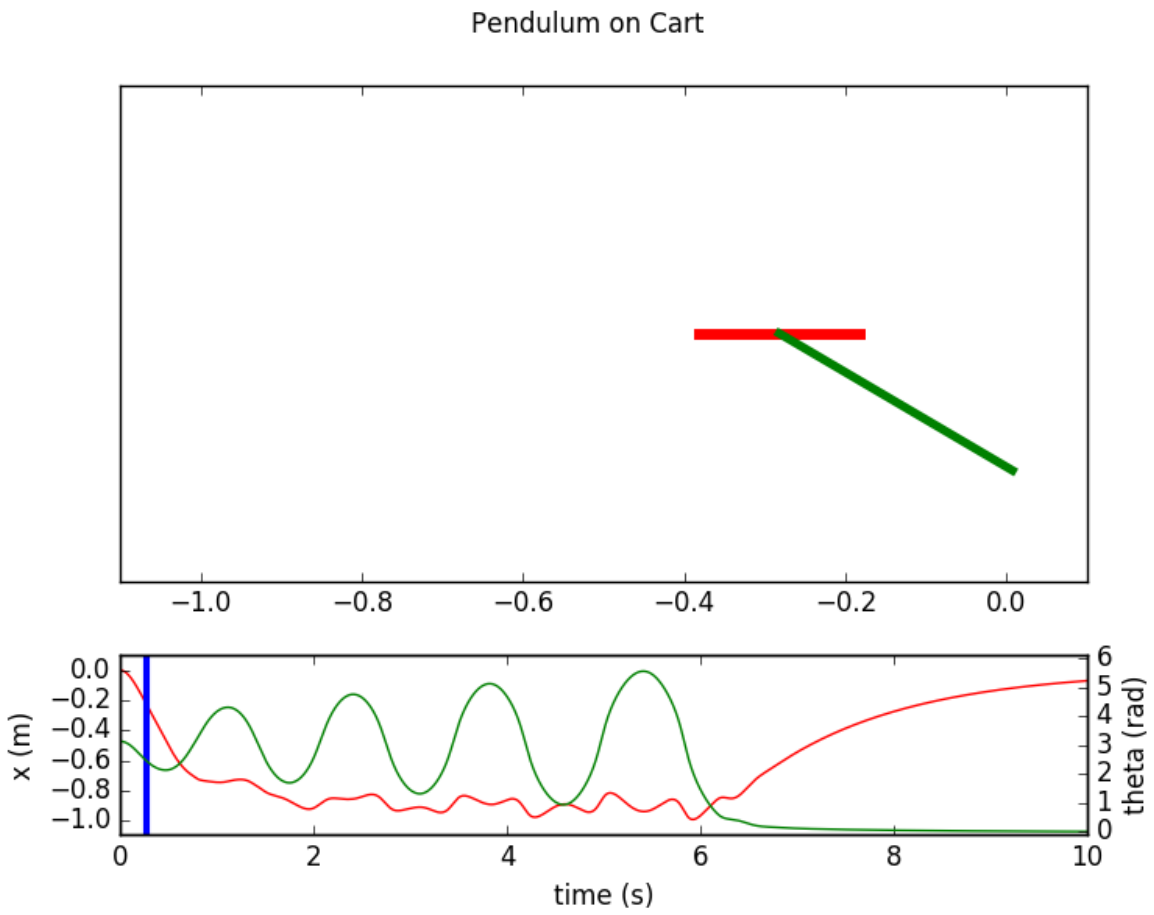

Fig 4 b. Cart position at time $t=t_1$

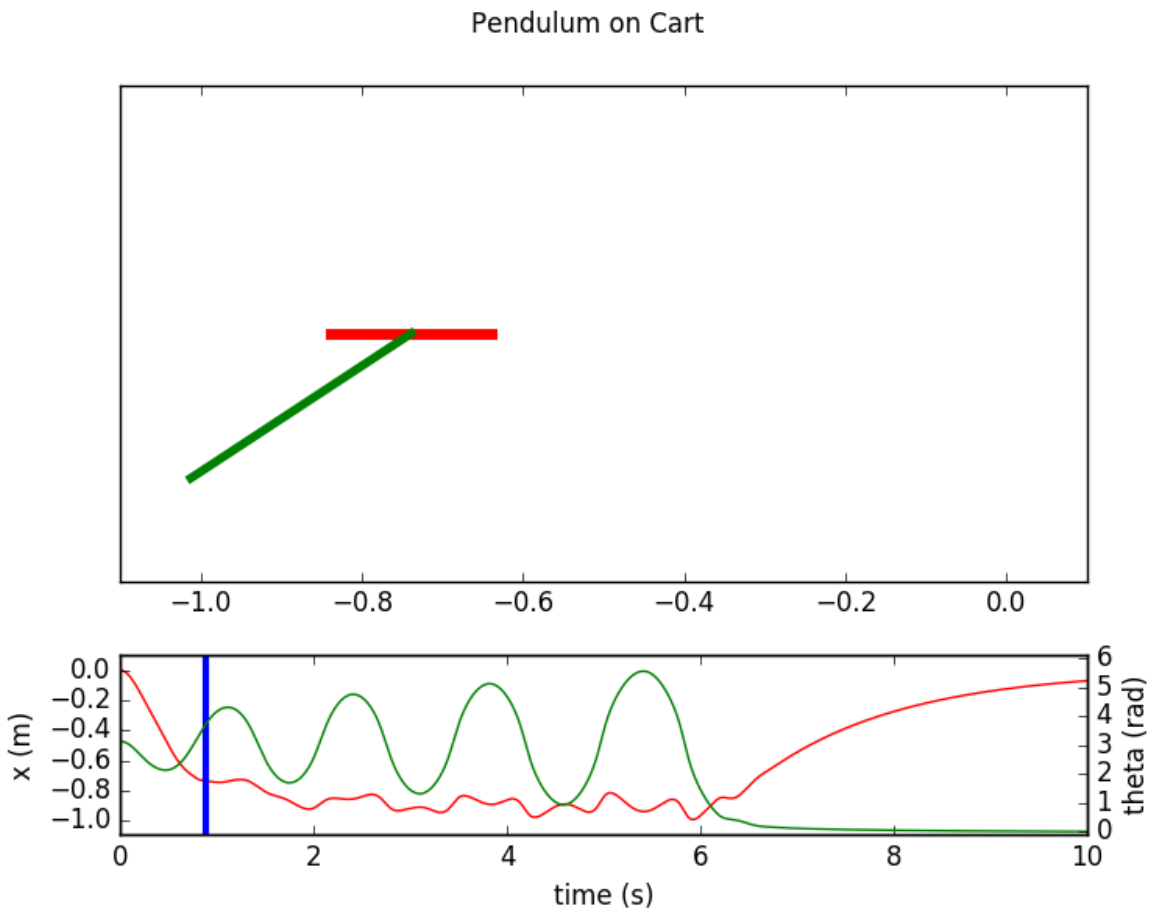

Fig 4 c. Cart position at time $t=t_2$

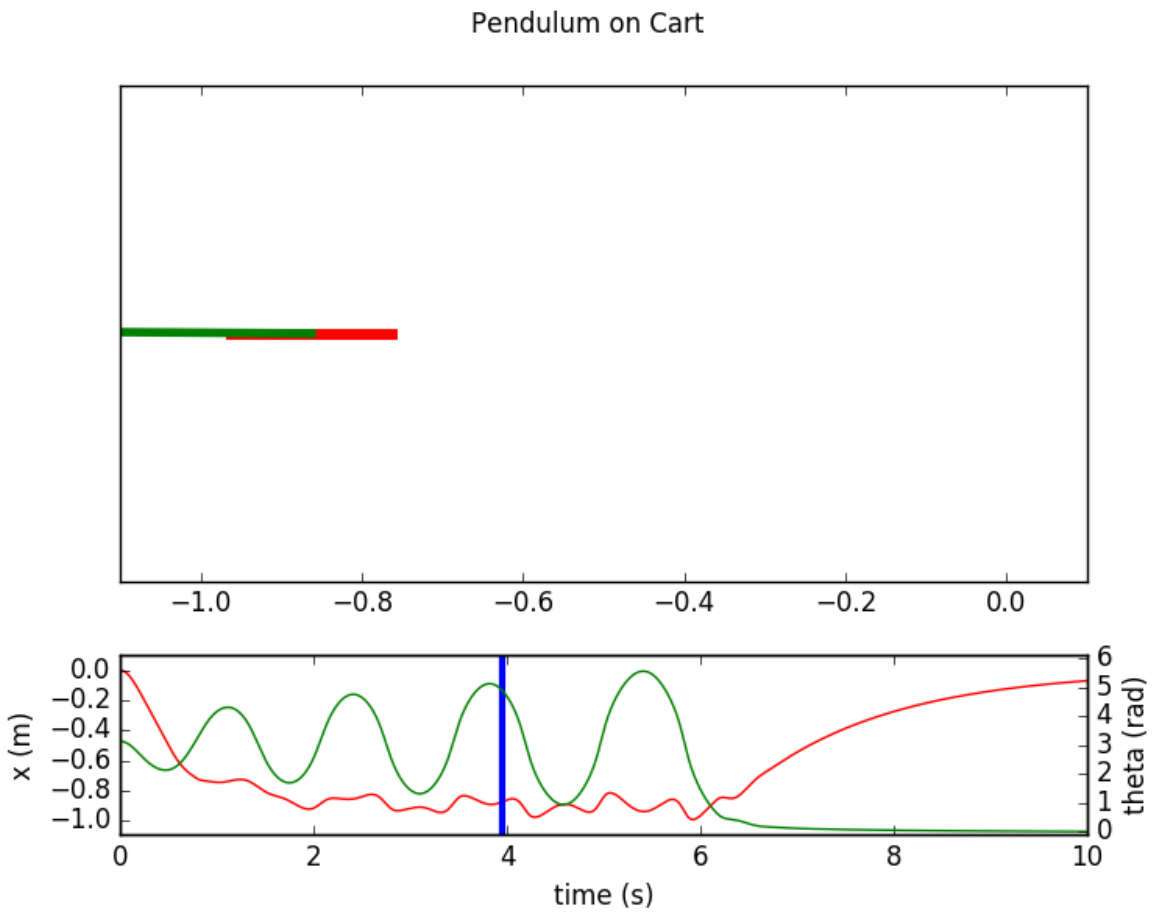

Fig 4 d. Cart position at time t=4s

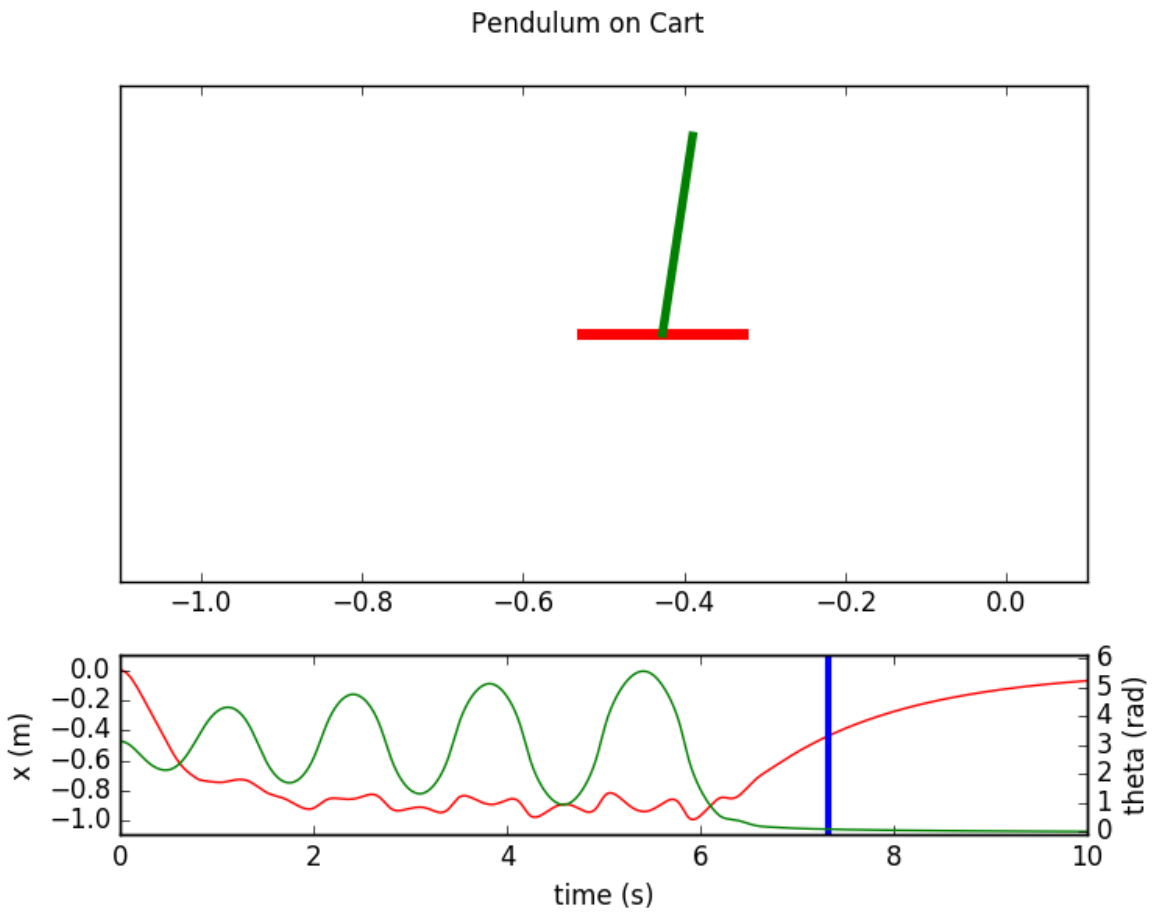

Fig 4 e. Cart position at time t=7s

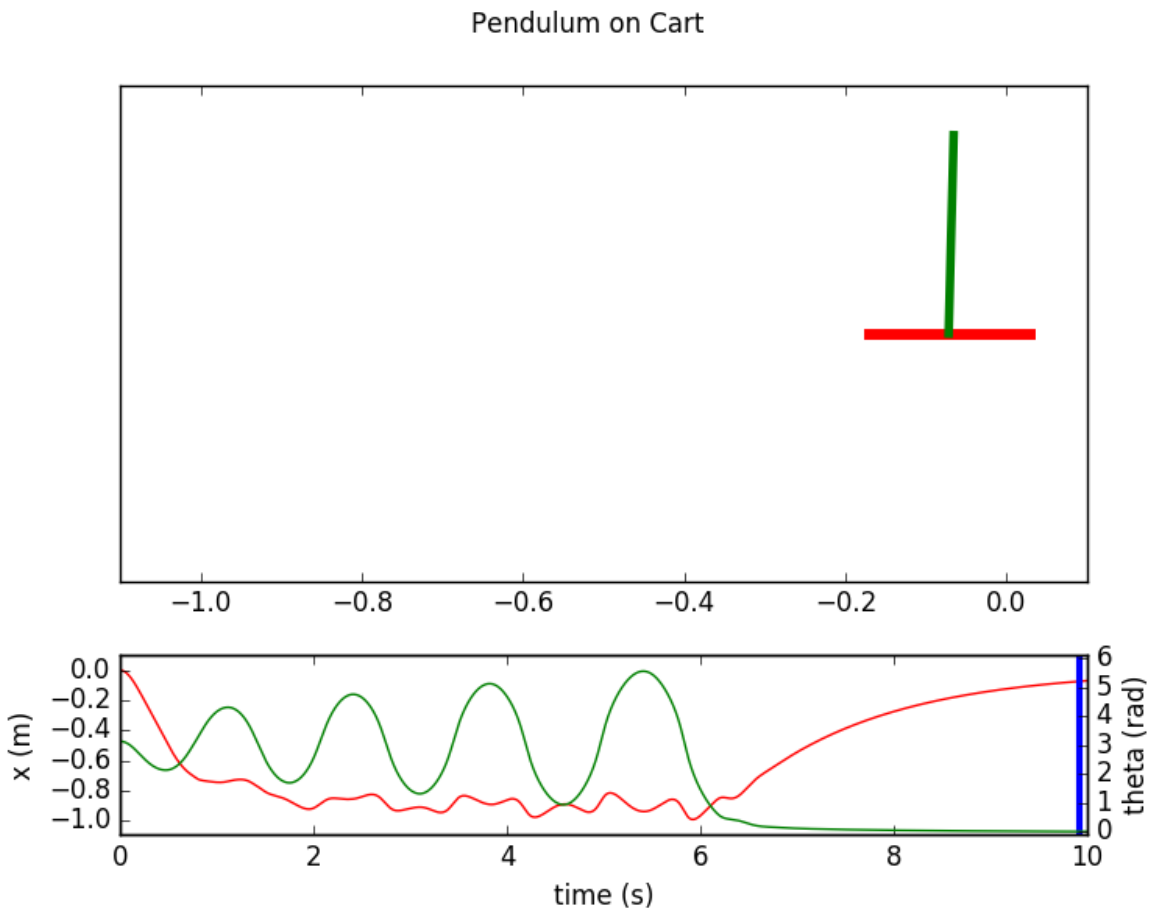

Fig 4 f. Cart position at time t=10s

Bibliography

[1]http://ctms.engin.umich.edu/CTMS/index.php?example=InvertedPendulum&section=ControlStateSpace

[2] https://en.wikipedia.org/wiki/Markov_decision_process

[3] Anderson, Charles W. "Learning to control an inverted pendulum using neural networks." *IEEE Control Systems Magazine* 9.3 (1989): 31-37.

[4] https://pdfs.semanticscholar.org/ed54/3eedde24edd50e39f5567f036b632499240a.pdf

[5] http://mlg.eng.cam.ac.uk/rowan/files/BayesianReinforcementLearning.pdf

[6] Ghavamzadeh, Mohammad, et al. *Bayesian reinforcement learning: a survey*. World Scientific, 2015.

[7] http://courses.cs.washington.edu/courses/cse515/09sp/slides/bnets.pdf

[8] https://datascisuthee.wordpress.com/2015/05/23/play-with-value-iteration-algorithm/

[9] Montemerlo, Michael, et al. "FastSLAM: A factored solution to the simultaneous localization and mapping problem." *Aaai/iaai*. 2002.

[10] Sutton, Richard S., and Andrew G. Barto. *Reinforcement learning: An introduction*. Vol. 1. No. 1. Cambridge: MIT press, 1998.