HW 3. Hidden Markov Model

MinKu Kang (tominku@kaist.ac.kr)

October 31, 2017

1 Introduction

In this assignment, you will design inference algorithms for HMM (Hidden Markov Model). Specifically, you will implement an exact filtering algorithm (Exact Inference) and an approximate filtering algorithm (Particle Filter).

2 Driverless Car

In this assignment, we refer to `http://stanford.edu/~cpiech/cs221/homework/prog/driverlessCar/driverlessCar.html` from Stanford University. Please download the codes on the website. Since the whole code is based on **Python 2.x**, please use **Python 2.x**.

3 Project Information

The task is to infer the positions of the other cars given the observations. An observation at time t is a noisy measurement of the distance between the agent(me) and the other car. As time step goes, the observations are accumulated resulting in more and more accurate posterior distributions over the positions of the other cars. You should visually check whether your posterior distribution (filtering result) gets improved as time step goes (as observations are accumulated). The filtering algorithm mainly consists of two parts. The first part updates a probability distribution over the positions of a car given an observation. At the second part, the probability distribution over the positions of the car is updated based on the (leanred) transition probabilities with the time step increased by 1.

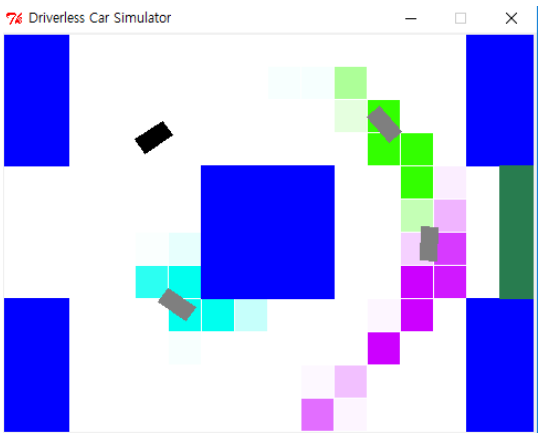

Figure 1: A simulation result with 3 cars and -d option on (Exact Inference)

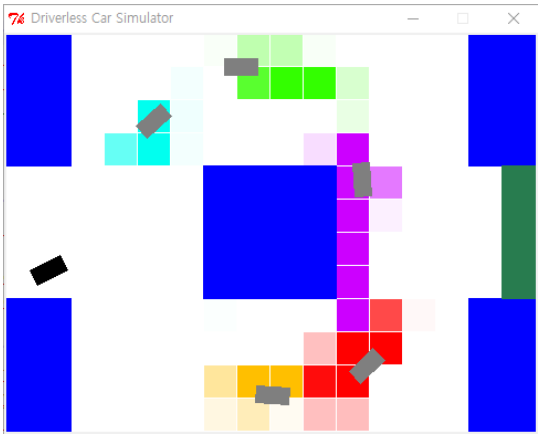

Figure 2: A simulation result with 5 cars and -d option on (Exact Inference)

4 What to Do

- Implement `learner.py` , where you learn the transition probabilities of the cars.
- Implement `exactInference.py` , where you compute the probability distribution over the positions of a car given an observation. You may also need to use the learned transition probabilities. Refer to filtering algorithms for HMM.
- Implement `particleFilter.py` , where you compute the probability distribution over the positions of a car with finite number of particles given an observation. You may also need to use the learned transition probabilities. Refer to particle filter algorithms.

5 What to Submit

- Submit `learner.py` , with a topmost comment line for your studnet number (e.g., # 20161234)
- Submit `exactInference.py` , with a topmost comment line for your studnet number (e.g., # 20161234)
- Submit `particleFilter.py` , with a topmost comment line for your studnet number (e.g., # 20161234)
- Do not make a .zip file. Upload the three files on KLMS with the file names untouched (learner.py, exactInference.py, particleFilter.py).

Figure 3: Please write down your student number as a comment line (e.g., # 20161234) at the first line of each file (.py), **so that a grading script can recognize your id.**