Solutions for Discrete-bayes

Yunhai Han

March 8, 2020

1 Discrete-bayes filter

In this exercise you will be implementing a discrete Bayes filter accounting for the motion of a robot on a $1-\mathrm{D}$ constrained world. Assume that the robot lives in a world with 20 cells and is positioned on the 10 th cell. The world is bounded, so the robot cannot move to outside of the specified area. Assume further that at each time step the robot can execute either a move forward or a move backward command. Unfortunately, the motion of the robot is subject to error, so if the robot executes an action it will sometimes fail.

Here is the general procedures for bayes filter:

Start with an initial belief, bel($\mathbf{x}_0) = p\left(\mathbf{x}_0\right)$

At every t from 1 to T apply the following loop:

1: for x in $\mathcal{X}$:

2: compute the predicted belief

$$\overline{\mathrm{bel}}(x)=\int_{\mathcal{X}} p\left(x|s,u_t\right)\ \text{bel}\ (s)ds \quad \left(\text{ or } \overline{\mathrm{bel}}(x)=\sum_s p\left(x|s,u_t\right)\mathrm{bel}(s)\right) \tag{1}$$

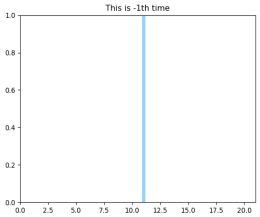
3: correct the predicted belief

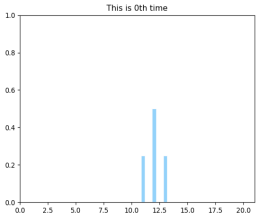
$$\mathrm{bel}(x)=\eta p\left(z_t|x\right)\overline{\mathrm{bel}}(x) \tag{2}$$

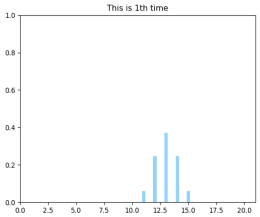
In this case, we are not given the measurement data, so we only have to compute the predicted belief. Since at the very beginning, the sum of probability is one, after each iteration, the sum would always be one. It's not necessary for us to normalize them(no measurement model).

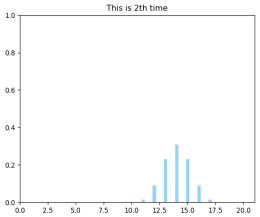
2 Results

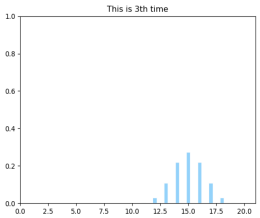
Here I put all the figures after each iteration.

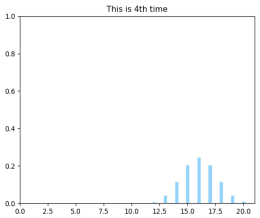


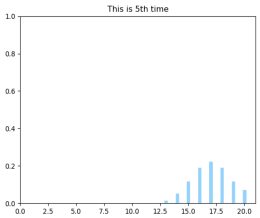


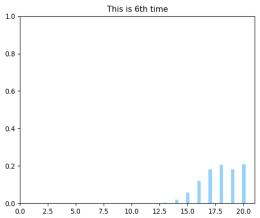


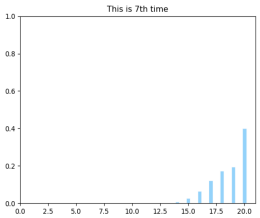


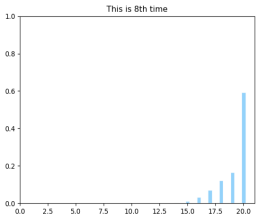


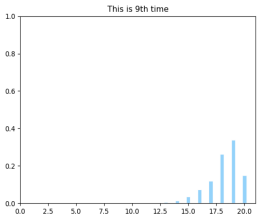


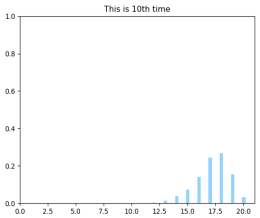


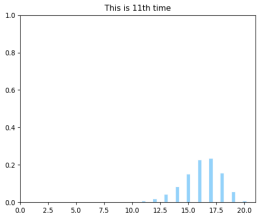












After each iteration, the belief always sums to one.

```
'''
 RESTART: C:/Users/Administrator/Desktop/winter quarter/MAE145/Homework/Extra/di
screte bayes filter/python_program.py
1.0
1.0
1.0
1.0
1.0
1.0
1.0
1.0
1.0
1.0
1.0
1.0
```

Figure 1: Print results

3 Codes

Here I put all the python codes:

```
#A53307224
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats
import intervals as I
import matplotlib.pyplot as plt

def motion_model(length, ut):
    A = np.zeros([length, length])
    for i in range(length):
        next_p = i + ut
        if next_p >= length:
            next_p = length - 1
        elif next_p <= 0:
            next_p = 0
        A[next_p, i] += 0.5
        next_p = i + 2 * ut
        if next_p >= length:
            next_p = length - 1
        elif next_p <= 0:
            next_p = 0
        A[next_p, i] += 0.25
        A[i,i] += 0.25
    return A
def draw_particles(index, particles):
    plt.title("This is " + str(index) + "th time")
    X = np.arange(len(particles))+1
    plt.bar(X, particles, alpha=0.9, width = 0.35, facecolor =
        'lightskyblue', edgecolor = 'white', label='one', lw=1)
    plt.axis([0, 21, 0, 1])
    plt.savefig(str(index)+".png")
    plt.pause(0.4)

if __name__ == "__main__":
    plt.ion()
    u = [1]*9 + [-1]*3
    bel = np.hstack ((np.zeros(10), 1, np.zeros(9)))
    draw_particles(-1, bel)
    for i in range(0, len(u)):
        bel = motion_model(len(bel), u[i]) @ bel.T
        bel = bel.T
        sum_cr = sum(bel)
        plt.cla()
        draw_particles(i, bel)
```

```
    print(sum(bel))
plt.ioff()
plt.show()
```