HW5 Report

109550206 陳品劭 Self link

Part I. Implementation (20%):

Part 1

1. Compute the distance observation d_t and your position a_t.
2. Use util.pdf to computes the probability density function for a Gaussian where mean is that distance.
3. Set belief probabilities.
4. Normalize self.belief

```
# BEGIN_YOUR_CODE (our solution is 9 lines of code, but don't worry if you deviate from this)
#raise Exception("Not implemented yet")
for row in range(self.belief.numRows):
    for col in range(self.belief.numCols):
        dist = math.dist( ( util.colToX(col), util.rowToY(row) ), (agentX, agentY))
        self.belief.setProb(row, col, self.belief.getProb(row, col) * util.pdf(dist, 
Const.SONAR_STD, observedDist))
self.belief.normalize()
# END_YOUR_CODE
```

Part 2

1. Create newBrief.
2. For every (oldTile, newTile), transProb, compute new probability and add into newBrief.
3. Update self.belief with newBrief.
4. Normalize self.belief.

```
# BEGIN_YOUR_CODE (our solution is 10 lines of code, but don't worry if you deviate from this)
#raise Exception("Not implemented yet")
newBeilf = util.Belief(self.belief.getNumRows(), self.belief.getNumCols(), value=0)
for (oldTile, newTile), transProb in self.transProb.items():
    newBeilf.addProb(newTile[0], newTile[1], self.belief.getProb(oldTile[0], oldTile[1]) * 
transProb)
self.belief = newBeilf
self.belief.normalize()
# END_YOUR_CODE
```

Part 3-1

1. Create reweight dict.
2. Compute probability as part 1 and store in reweight with key (row, col)
3. Create resample dict.
4. Resample in |self.NUM_PARTICLES| times and load those particles in resample.
5. Update self.particles by resample.

```
# BEGIN_YOUR_CODE (our solution is 12 lines of code, but don't worry if you deviate from this)
#raise Exception("Not implemented yet")
reweight = dict()
for (row, col), num in self.particles.items():
    dist = math.dist( ( util.colToX(col), util.rowToY(row) ), (agentX, agentY))
    reweight[(row,col)] = self.particles[(row, col)] * util.pdf(dist, Const.SONAR_STD, observedDist)
```

```
resample = dict()
for i in range(self.NUM_PARTICLES):
    particle = util.weightedRandomChoice(reweight)
    if particle in resample:
        resample[particle] += 1
    else:
        resample[particle] = 1

self.particles = resample
# END_YOUR_CODE
```

Part 3-2

1. Create proposal.
2. For every particle, resample in num times and load in proposal.
3. Update self.particles by proposal.

```
# BEGIN_YOUR_CODE (our solution is 6 lines of code, but don't worry if you deviate from this)
#raise Exception("Not implemented yet")
proposal = collections.defaultdict(int)
for particle, num in self.particles.items():
    for i in range(num):
        x = util.weightedRandomChoice(self.transProbDict[particle])
        if x in proposal:
            proposal[x] += 1
        else:
            proposal[x] = 1
self.particles = proposal
# END_YOUR_CODE
```