Artificial Intelligence

Jason

Monday 4th December, 2023; 22:09

Assignment 0

Question 1. Exact Inference Observation: Write down the equation of the inference problem you are trying to solve.

new_belief[p] = $getObservationProb(observation, gameState.getPacmanPosition(), p, self.getJailPosition()) * old\_belief[p]$
getObservationProb(observation, pacmanPosition, position, jailPosition) =
P(noisydistance |pacman position, ghost position)

Question 2. Exact Inference with Time Elapse: Write down the equation of the inference problem you are trying to solve.

new_belief[p1] = $\sum_{\substack{p \in allPositions \\ old\_belief[p] \neq 0}} old\_belief[p] \cdot getPositionDistribution(gameState, p)[p1]$
getPositionDistribution(gameState, ghostPosition) = P(ghost position at time t+1 |ghost position at time t)

Question 3. Alternative answer for GreedyBusterAgent

Correct answer for GreedyBusterAgent:
Average score: 754.7
Alternative answer for GreedyBusterAgent:
Average score: 758.7
Alternative answer 2 for GreedyBusterAgent:
Average score: 754.3
The first alternative answer is better than the correct answer and there is also a variable k that could be played with to change the score.
The first alternative answer is still a bit greedy but added with some randomness so most.
The second alternative answer is a bit less greedy, the strategy is considered less greedy because it doesn't always choose the action that brings Pacman closest to the nearest ghost. Instead, it uses a probability distribution to decide which ghost to chase. This means that while the closest ghost has the highest probability of being chosen, there's still a chance that a farther ghost might be selected. This introduces an element of randomness and exploration into the decision-making process, which can potentially lead to better overall strategies.

Question 4. Joint Particle Filter Time Elapse and Full Test: As you run the autograder note that q14/1-JointParticlePredict and q14/2-JointParticlePredict test your elapseTime implementations only, and q14/3-JointParticleFulltests both your observeUpdate and elapse-Time implementations. Notice the dif-ference between test 1 and test 3. In both tests, pacman knows that the ghostswill move to the sides of the gameboard. What is different between the tests,and why?

In the first test we predict the ghost position, so the longer the ghost stays in a certain position, the more likely it is to be in that position, so pacman will assume more and more that the ghost is in that position.
For the second test, we update based on the last elapse time. So we look more to recent estimates of the ghost position, instead of looking at the whole history of the ghost position. The second test is more accurate because it is based on the most recent information, while the first test is based on the whole history of the ghost position, making use of predictions.