PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
Instituto de Ciências Exatas e Informática
Curso de Ciência da Computação - Coração Eucarístico
Profa.: Camila Laranjeira - mila.laranjeira@gmail.com
Disciplina: Inteligência Artificial / 1o Semestre de 2022
Aluna(o):

Exercício Prático 04 - Carro autônomo

Instruções:

- Consulte os slides da disciplina para maiores detalhes sobre a implementação
- O código fonte base está no Canvas sob o título `car.zip`
- **Você deve entregar seu código em um .zip**

Esse projeto é uma variação da atividade driverless car, também ofertada no curso CS221 da Universidade de Stanford.

Primeiros passos: Teste o código. Você deve executar o script drive.py de sua forma mais básica, sem nenhuma inteligência, com o comando a seguir. Para dirigir, use as setas do teclado ou as teclas wasd. Tente dirigir sem bater em nenhum carro (é muito difícil). Note que os carros não podem ser visualizados a menos que o código seja executado em modo debug (veja instruções a seguir).

```
python drive.py -l lombard -i none
```

Este script aceita os seguintes parâmetros:

- -a: Direção autônoma (não-manual)
- -i <método de inferência>: Use none ou exactInference depois de implementado.
- -l <mapa>: Use este mapa (small ou lombard).
- -d: Modo debug. Exibe os carros no mapa
- -p: Todos os carros ficam “estacionados”, parados no mapa.
- -k <número de carros>. Número de carros no mapa.

1. Para essa atividade, você modificará apenas o arquivo **`submission.py`**, especificamente o método `observe(agentX, agentY, observedDist)` na classe `ExactInference`.

Estamos assumindo que o mundo é um grid `n x m` de possíveis posições de obstáculo k **carros estacionados**. A cada ciclo de execução sua função “`observe`” será chamada, recebendo como parâmetro a posição `(x,y)` do seu agente, além da distância `d` até o outro carro, capturada pelo sensor.

Assumimos que a incerteza do sensor de distância tem um desvio padrão definido na constante `Const.SONAR_STD` do código (cerca de ⅔ do tamanho de um carro) e pode ser modelada como uma distribuição gaussiana. O código também fornece a função `util.pdf(mean, std, value)` para calcular a probabilidade de `value` pertencer à distribuição gaussiana `(mean, std)`.

Seu trabalho é atualizar a crença posteriori self.belief da classe ExactInference. Os atributos e métodos mais importantes são:

- `self.belief.numRows`
- `self.belief.numCols`
- `self.belief.getProb(row, col)`
- **`self.belief.setProb(row, col)`**

Você também vai precisar converter linhas e colunas para posições (x, y). Use:

- `util.rowToY(row)`
- `util.colToX(col)`

Após implementada, você pode testar a sua função executando:

```
python drive.py -p -d -a -k 1 -i exactInference
```

Nota 1: Você pode desativar o modo direção autônoma e dirigir o carro manualmente
Nota 2: Altere o número de carros em campo e se certifique que seu código ainda funciona corretamente.

———

2. Para que os carros se movam em campo, é preciso implementar o método `elapseTime`, também em `ExactInference`. Ele define a probabilidade condicional de transição entre a posição atual de um carro c_t e posições futuras c_{t+1}

$$p(c_{t+1} \mid c_t).$$

Felizmente, essa informação já é dada no dicionário `self.transProb`. Mais especificamente, `self.transProb[(oldTile, newTile)]` define a probabilidade de um carro que está em `oldTile` no tempo `t` transicionar para `newTile` em `t+1`.

Seu trabalho é atualizar a crença posteriori `self.belief`, para que a crença de uma nova posição c_{t+1} seja definida pela crença da posição c_t atualizada de acordo com o modelo de transição dado:

$$p(c_{t+1} \mid d1, d2, \ldots, dt) = p(c_t \mid d1, d2, \ldots, dt) * p(c_{t+1} \mid c_t)$$

Você precisará criar um novo objeto `util.Belief` (veja util.py) para garantir que usará os valores originais de `self.belief` ao longo de todo o processo. Apenas ao final, copie os valores de seu novo objeto para `self.belief`. Se preocupe apenas com as posições contidas no dicionário `self.transProb`, quaisquer outras posições tem probabilidade zero de conter um outro carro.

Após implementada, você pode testar a sua função executando:

```
python drive.py -d -a -k 1 -i exactInference -l lombard
```

Nota 1: Você pode desativar o modo direção autônoma e dirigir o carro manualmente
Nota 2: Altere o número de carros em campo e se certifique que seu código ainda funciona corretamente.