CS401–Intelligent Robots Spring 2019 Midterm Test

name: 黄玉安 **no:11610303**

P I

1.D

2.B

3.C

4.B

5.D

6.D

7.A

8.A

9.B

10.C

P II

1. we know p(z=open|x1=open) = 1-0.3 =0.7, p(z=closed|x1=closed) = 1-0.2 = 0.8, p(z=open|x2=open) = 0.8, p(z=closed|x2=closed) = 0.8, (1)

$$\begin{aligned}&p(x1=open|z=open)\\=&\frac{p(z=open|x1=open)*p(x1=open)}{(p(z=open|x1=open)*p(x1=open)+p(z=open|x1=close)*p(x1=close))}\\=&0.7*0.5/(0.7*0.5+0.2*0.5)=7/9\end{aligned}$$

$$\begin{aligned}&p(x2=open|z=open)\\=&\frac{p(z=open|x2=open)*p(x2=open)}{(p(z=open|x2=open)*p(x2=open)+p(z=open|x2=close)*p(x2=close))}\\=&0.8*0.5/(0.8*0.5+0.2*0.5)=4/5\end{aligned}$$

(2)

$$\begin{aligned}&p(x1=close,x2=close|z1=open,z2=open)\\=&p(x1=close|z1=open)*p(x2=close|z2=open)\\=&(1-p(x1=open|z1=open))*(1-p(x2=open|z2=open))\\=&(1-7/9)(1-4/5)=2/45\end{aligned}$$

2.

Markov assumption: zn is independent of z1, ..., zn-1 if we know x.

So,

$$
\begin{aligned}
&p(x_t|u1, z1, u2, z2, \ldots, u_t, z_t, m) \\
=&\eta p(z_t|x_t, u_{1:t-1}, z_{1:t-1}, m) * p(x_t|u_{1:t-1}, z_{1:t-1}, m) \\
=&\eta p(z_t|x_t, m) * \int p(x_t|x_{t-1}, u_{1:t-1}, z_{1:t-1}, m) * p(x_{t-1}|u_{1:t-1}, z_{1:t-1}, m) dx_{t-1} \\
=&\eta p(z_t|x_t, m) * \int p(x_t|x_{t-1}, u_{t-1}, m) * p(x_{t-1}|u_{1:t-1}, z_{1:t-1}, m) dx_{t-1} \\
=&\eta p(z_t|x_t, m) * \int p(x_t|x_{t-1}, u_t - 1, m) * Bel(x_{t-1}) dx_{t-1}
\end{aligned}
$$

More, Bel(x1) = p(x1), so we can get all `p(x_t| u1, z1, u2, z2,..., u_t, z_t, m)`

P III

1.

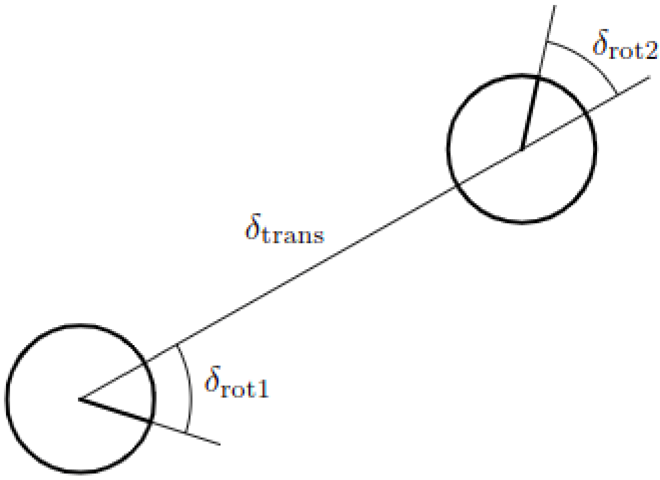
For velocity model, it use this algorithm, here $x_{t-1} = (x, y, \theta)^T, x_t = (x', y', \theta')^T, u_t = (v, \omega)^T$

Algorithm motion_model_velocity(x_t, u_t, x_{t-1}):

$$
\begin{aligned}
&\mu = \frac{1}{2} \frac{(x - x')\cos\theta + (y - y')\sin\theta}{(y - y')\cos\theta - (x - x')\sin\theta} \\
&x^* = \frac{x + x'}{2} + \mu(y - y') \\
&y^* = \frac{y + y'}{2} + \mu(x' - x) \\
&r^* = \sqrt{(x - x^*)^2 + (y - y^*)^2} \\
&\Delta\theta = \operatorname{atan2}(y' - y^*, x' - x^*) - \operatorname{atan2}(y - y^*, x - x^*) \\
&\hat{v} = \frac{\Delta\theta}{\Delta t} r^* \\
&\hat{\omega} = \frac{\Delta\theta}{\Delta t} \\
&\hat{\gamma} = \frac{\theta' - \theta}{\Delta t} - \hat{\omega} \\
&\textit{return } \mathbf{prob}(v - \hat{v}, \alpha_1 v^2 + \alpha_2 \omega^2) \cdot \mathbf{prob}(\omega - \hat{\omega}, \alpha_3 v^2 + \alpha_4 \omega^2) \\
&\qquad\qquad \cdot \mathbf{prob}(\hat{\gamma}, \alpha_5 v^2 + \alpha_6 \omega^2)
\end{aligned}
$$

The return value is the value of $p(x_{t+1}|x_t, u_t)$.

For odometry model



Algorithm motion_model_odometry(x_t, u_t, x_{t-1}):

$$\delta_{\text{rot1}} = \text{atan2}(\bar{y}' - \bar{y}, \bar{x}' - \bar{x}) - \bar{\theta}$$
$$\delta_{\text{trans}} = \sqrt{(\bar{x} - \bar{x}')^2 + (\bar{y} - \bar{y}')^2}$$
$$\delta_{\text{rot2}} = \bar{\theta}' - \bar{\theta} - \delta_{\text{rot1}}$$

$$\hat{\delta}_{\text{rot1}} = \text{atan2}(y' - y, x' - x) - \theta$$
$$\hat{\delta}_{\text{trans}} = \sqrt{(x - x')^2 + (y - y')^2}$$
$$\hat{\delta}_{\text{rot2}} = \theta' - \theta - \hat{\delta}_{\text{rot1}}$$

$$p_1 = \textbf{prob}(\delta_{\text{rot1}} - \hat{\delta}_{\text{rot1}}, \alpha_1 \hat{\delta}^2_{\text{rot1}} + \alpha_2 \hat{\delta}^2_{\text{trans}})$$
$$p_2 = \textbf{prob}(\delta_{\text{trans}} - \hat{\delta}_{\text{trans}}, \alpha_3 \hat{\delta}^2_{\text{trans}} + \alpha_4 \hat{\delta}^2_{\text{rot1}} + \alpha_4 \hat{\delta}^2_{\text{rot2}})$$
$$p_3 = \textbf{prob}(\delta_{\text{rot2}} - \hat{\delta}_{\text{rot2}}, \alpha_1 \hat{\delta}^2_{\text{rot2}} + \alpha_2 \hat{\delta}^2_{\text{trans}})$$

Here, control $u_t, = (\bar{x}_{t-1}, \bar{x}_t)^T, \bar{x}_{t-1} = (\bar{x}, \bar{y}, \bar{\theta})^T, u_t = (v, \omega)^T$.

the value of $p(x_{t+1}|x_t, u_t)$ is p1 * p2 * p3.

Similarity: for both motion models, we presented two types of implementations, one in which thr $p(x_t|u_t, x_{t-1})$ is calculated in closed form, and one that enables us to generate samples from $p(x_t|u_t, x_{t-1})$.

Difference: 1. velocity model add a third moise paremeter, expressed as a noisy "final votation".

2. Odemetry: the readings are technically not controls, use them just like controls.

2.

For velocity:

Algorithm sample_motion_model_velocity(u_t, x_{t-1}):

$$\hat{v} = v + \textbf{sample}(\alpha_1 v^2 + \alpha_2 \omega^2)$$
$$\hat{\omega} = \omega + \textbf{sample}(\alpha_3 v^2 + \alpha_4 \omega^2)$$
$$\hat{\gamma} = \textbf{sample}(\alpha_5 v^2 + \alpha_6 \omega^2)$$
$$x' = x - \frac{\hat{v}}{\hat{\omega}} \sin\theta + \frac{\hat{v}}{\hat{\omega}}\ sin(\theta + \hat{\omega}\Delta t)$$
$$y' = y + \frac{\hat{v}}{\hat{\omega}} \cos\theta - \frac{\hat{v}}{\hat{\omega}}\ cos(\theta + \hat{\omega}\Delta t)$$
$$\theta' = \theta + \hat{\omega}\Delta t + \hat{\gamma}\Delta t$$
$$\text{return } x_t = (x', y', \theta')^T$$

For odometry:

Algorithm sample_motion_model_odometry(u_t, x_{t-1}):

$$\delta_{\text{rot1}} = \text{atan2}(\bar{y}' - \bar{y}, \bar{x}' - \bar{x}) - \bar{\theta}$$
$$\delta_{\text{trans}} = \sqrt{(\bar{x} - \bar{x}')^2 + (\bar{y} - \bar{y}')^2}$$
$$\delta_{\text{rot2}} = \bar{\theta}' - \bar{\theta} - \delta_{\text{rot1}}$$

$$\hat{\delta}_{\text{rot1}} = \delta_{\text{rot1}} - \textbf{sample}(\alpha_1 \delta^2_{\text{rot1}} + \alpha_2 \delta^2_{\text{trans}})$$
$$\hat{\delta}_{\text{trans}} = \delta_{\text{trans}} - \textbf{sample}(\alpha_3 \delta^2_{\text{trans}} + \alpha_4\ \delta^2_{\text{rot1}} + \alpha_4\ \delta^2_{\text{rot2}})$$
$$\hat{\delta}_{\text{rot2}} = \delta_{\text{rot2}} - \textbf{sample}(\alpha_1 \delta^2_{\text{rot2}} + \alpha_2 \delta^2_{\text{trans}})$$

$$x' = x + \hat{\delta}_{\text{trans}}\ \cos(\theta + \hat{\delta}_{\text{rot1}})$$
$$y' = y + \hat{\delta}_{\text{trans}}\ \sin(\theta + \hat{\delta}_{\text{rot1}})$$
$$\theta' = \theta + \hat{\delta}_{\text{rot1}} + \hat{\delta}_{\text{rot2}}$$

$$\text{return } x_t = (x', y', \theta')^T$$

Similarity: Both of them are easy to implement.

Differenece: velocity reuire to calculate the inverse of the physical motion model; Odemetry side-steps the need for an inverse model.

P IV

1. Bean-based sensors:

Algorithm beam_range_finder_model(z_t, x_t, m):

$$
\begin{aligned}
& q = 1 \\
& \textit{for } k = 1 \textit{ to } K \textit{ do} \\
& \quad \textit{compute } z_t^{k*} \textit{ for the measurement } z_t^k \textit{ using ray casting} \\
& \quad p = z_{\text{hit}} \cdot p_{\text{hit}}(z_t^k \mid x_t, m) + z_{\text{short}} \cdot p_{\text{short}}(z_t^k \mid x_t, m) \\
& \qquad + z_{\max} \cdot p_{\max}(z_t^k \mid x_t, m) + z_{\text{rand}} \cdot p_{\text{rand}}(z_t^k \mid x_t, m) \\
& \quad q = q \cdot p \\
& \textit{return } q
\end{aligned}
$$

Scan-based sensors:

Algorithm likelihood_field_range_finder_model(z_t, x_t, m):

$$
\begin{aligned}
& q = 1 \\
& \textit{for all } k \textit{ do} \\
& \quad \textit{if } z_t^k \neq z_{\max} \\
& \qquad x_{z_t^k} = x + x_{k,\text{sens}} \cos\theta - y_{k,\text{sens}} \sin\theta + z_t^k \cos(\theta + \theta_{k,\text{sens}}) \\
& \qquad y_{z_t^k} = y + y_{k,\text{sens}} \cos\theta + x_{k,\text{sens}} \sin\theta + z_t^k \sin(\theta + \theta_{k,\text{sens}}) \\
& \qquad dist = \min_{x',y'} \left\{ \sqrt{(x_{z_t^k} - x')^2 + (y_{z_t^k} - y')^2} \,\middle|\, \langle x', y' \rangle \text{ occupied in } m \right\} \\
& \qquad q = q \cdot \left(z_{\text{hit}} \cdot \mathbf{prob}(dist, \sigma_{\text{hit}}) + \frac{z_{\text{random}}}{z_{\max}} \right) \\
& \textit{return } q
\end{aligned}
$$

Bean-bases sensors suffers two major drawbacks. (1) lack of smoothness. (2) computational involved. Scan-based sensors overcomes these limitations. It does not compute a conditional probability relative to any meaningful generative model of the physics of sensors.

2.

Detection probability:

Algorithm landmark_model_known_correspondence(f_t^i, c_t^i, x_t, m):

$$
\begin{aligned}
& j = c_t^i \\
& \hat{r} = \sqrt{(m_{j,x} - x)^2 + (m_{j,y} - y)^2} \\
& \hat{\phi} = \text{atan2}(m_{j,y} - y, m_{j,x} - x) \\
& q = \mathbf{prob}(r_t^i - \hat{r}, \sigma_r) \cdot \mathbf{prob}(\phi_t^i - \hat{\phi}, \sigma_\phi) \cdot \mathbf{prob}(s_t^i - s_j, \sigma_s) \\
& \textit{return } q
\end{aligned}
$$

generate samples:

Algorithm sample_landmark_model_known_correspondence(f_t^i, c_t^i, m):

$$
\begin{aligned}
&j = c_t^i \\
&\hat{\gamma} = \mathrm{rand}(0, 2\pi) \\
&\hat{r} = r_t^i + \textbf{sample}(\sigma_r) \\
&\hat{\phi} = \phi_t^i + \textbf{sample}(\sigma_\phi) \\
&x = m_{j,x} + \hat{r}\cos\hat{\gamma} \\
&y = m_{j,y} + \hat{r}\sin\hat{\gamma} \\
&\theta = \hat{\gamma} - \pi - \hat{\phi} \\
&\textit{return}\ (x\ \ y\ \ \theta)^T
\end{aligned}
$$

P V

1. $X_{t+1}|x_t \in G(Ax_t + u_t, R)$,

   thus,

$$
\begin{aligned}
\mu_{x_{t+1}|x_t} &= A\mu_t + u_t \\
\Sigma_{x_{t+1}|x_t} &= A\Sigma_t A^T + R
\end{aligned}
$$

2.

multiply by C^TQ^{-1}, then $C^TQ^{-1}z = C^TQ^{-1}Cx - C^TQ^{-1}m + C^TQ^{-1}v$

then $(C^TQ^{-1}C)^{-1}C^TQ^{-1}z = x - (C^TQ^{-1}C)^{-1}C^TQ^{-1}m + (C^TQ^{-1}C)^{-1}C^TQ^{-1}v$

let $\bar{C} = (C^TQ^{-1}C)^{-1}C^TQ^{-1}$, so $x^* = \bar{C}z + \bar{C}m - \bar{C}v$

thus

$$
\begin{aligned}
\mu_{x|z} &= (C^TQ^{-1}C)^{-1}C^TQ^{-1}z + (C^TQ^{-1}C)^{-1}C^TQ^{-1}m \\
\Sigma_{x|z} &= \bar{C}Q\bar{C}^T = (C^TQ^{-1}C)^{-1}
\end{aligned}
$$

3.

$$
\begin{aligned}
\Sigma_{t+1}^{-1} &= (C^TQ^{-1}C) + (A\Sigma_t A^T + R)^{-1} \\
\Sigma_{t+1} &= \frac{1}{(C^TQ^{-1}C) + (A\Sigma_t A^T + R)^{-1}}
\end{aligned}
$$

And for mean value of it:

$$
\begin{aligned}
\mu_{t+1} =& (\Sigma_1 u_1 + \Sigma_2 u_2)\Sigma_{t+1} \\
=& \frac{(A\mu_t + u_t)(A\Sigma_t A^T + R) + ((C^TQ^{-1}C)^{-1}C^TQ^{-1}z + (C^TQ^{-1}C)^{-1}C^TQ^{-1}m)(C^TQ^{-1}C)^{-1}}{(C^TQ^{-1}C) + (A\Sigma_t A^T + R)^{-1}}
\end{aligned}
$$

P VI

- 1.

For $Bel(m_t^{[i]}) = p(m_i | z_{1:t}, x_{1:t})$, It can been clculated below:

$$
\begin{aligned}
p(m_i \mid z_{1:t}, x_{1:t}) \quad & \overset{\text{Bayes rule}}{=} \quad \frac{p(z_t \mid m_i, z_{1:t-1}, x_{1:t})\; p(m_i \mid z_{1:t-1}, x_{1:t})}{p(z_t \mid z_{1:t-1}, x_{1:t})} \\
& \overset{\text{Markov}}{=} \quad \frac{p(z_t \mid m_i, x_t)\; p(m_i \mid z_{1:t-1}, x_{1:t-1})}{p(z_t \mid z_{1:t-1}, x_{1:t})} \\
& \overset{\text{Bayes rule}}{=} \quad \frac{p(m_i \mid z_t, x_t)\; p(z_t \mid x_t)\; p(m_i \mid z_{1:t-1}, x_{1:t-1})}{p(m_i \mid x_t)\; p(z_t \mid z_{1:t-1}, x_{1:t})} \\
& \overset{\text{Markov}}{=} \quad \frac{p(m_i \mid z_t, x_t)\; p(z_t \mid x_t)\; p(m_i \mid z_{1:t-1}, x_{1:t-1})}{p(m_i)\; p(z_t \mid z_{1:t-1}, x_{1:t})}
\end{aligned}
$$

And the same as the opposite

$$
p(\neg m_i \mid z_{1:t}, x_{1:t}) \quad \overset{\text{the same}}{=} \quad \frac{p(\neg m_i \mid z_t, x_t)\; p(z_t \mid x_t)\; p(\neg m_i \mid z_{1:t-1}, x_{1:t-1})}{p(\neg m_i)\; p(z_t \mid z_{1:t-1}, x_{1:t})}
$$

Thus:

$$
\begin{aligned}
& \frac{p(m_i \mid z_{1:t}, x_{1:t})}{1 - p(m_i \mid z_{1:t}, x_{1:t})} \\
& = \quad \frac{p(m_i \mid z_t, x_t)}{1 - p(m_i \mid z_t, x_t)} \; \frac{p(m_i \mid z_{1:t-1}, x_{1:t-1})}{1 - p(m_i \mid z_{1:t-1}, x_{1:t-1})} \; \frac{1 - p(m_i)}{p(m_i)}
\end{aligned}
$$

And for log-odds representation:

$$
l(x) = log \, \frac{p(x)}{1 - p(x)}
$$

using this formula to it, the result is:

$$
\begin{aligned}
& l(m_i \mid z_{1:t}, x_{1:t}) \\
& = \quad l(m_i \mid z_t, x_t) \quad + l(m_i \mid z_{1:t-1}, x_{1:t-1}) - l(m_i)
\end{aligned}
$$

- 2.

$$\frac{0.55}{1-0.55}^{1000*40\%} * \frac{0.4}{1-0.4}^{1000*60\%} \to 0$$

So the value of the occupancy map of this cell is 0.

the value of the reflection map of this cell is $\frac{40\%}{60\%+40\%} = 0.4$

- 3 For landmark 1:

$$\hat{r}_1 = \sqrt{(x_1 - x)^2 + (y_1 - y)^2}$$
$$\hat{\theta} = atan2(y_1 - y, x_1 - x)$$
$$p = prob(\hat{r}_1 - r_1, \epsilon_r) * prob(\hat{\theta} - \theta, \epsilon_\theta)$$

For other land mark, use the same step, we can get their distribution as well.

P VII

1.

1. get a uniform sample (x_S, y_S) where X_S from $x_s \in [x_{m2} - r_2, x_{m2} + r_2]$, $y_s \in [y_{m2}, y_{m2} + r_2]$.
2. sample c from $[0, f_{max}]$
3. if $f(x_s, y_s) > c$, keep the sample, otherwise reject this sample

2.

1. for all samples x_i:
2. calculate the weight w according to p(r1|x) and p(r3|x) by multiplying them
3. after sampling, normalize sample's weight

3.

1. genetate cdf, for all n sample, let $c_i = c_{i-1} + w_i$,
2. initial a threadhold $u_i \in U[0, n^{-1}]$, i=1, S' = none
3. for all samples
4. while($u_j > c_i$) begin
5. i = i+1
6. $S' = S' + < x_i, n^{-1} >$
7. $u_{j+1} = u_j + n^{-1}$
8. return S'

P VIII

To finish tasks, robots have to be able to accommodate the enormous uncertainty that exists in the physical world. There is a number of factors that contribute to a robot's uncertainty.

- First and foremost, robot environments are inherently unpredictable,

- Sensors are limited in what they can perceive.
- Some uncertainty is caused by the robot's software.
- Uncertainty is futher created through algorithmic approximations. Robots are real-time systems. This limits the amout of computaion that can be carried out.

Since the complex of real physical world, along with robot moving into the open world, use probabilisitic models to represent uncertainty is the most important thing.