Credit Exam 2021—Solution

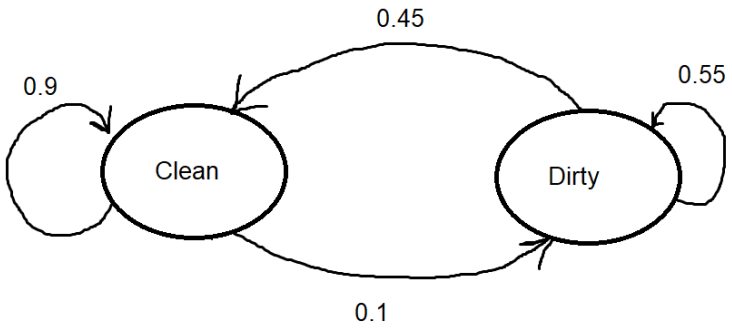
Q1) Compare Between Traditional Robotics & Cognitive Robotics.

Traditional Robotics	Cognitive Robotics
• Operates in controlled environments • Well-understood • Used in mass production	• Operates in dynamic real-life environment • Have cognitive functions normally associated with people or animals • Interpret various kinds of sensor data • Act purposefully and autonomously towards achieving goals • Has a high degree of robustness in coping with unpredictable situations
Example: Robotic hands in factories	Example: Humanoid robots, floor cleaning robots, Google self-driving cars

Q2) What's the probability of the floor being clean $P(x_t = clean|u)$ given that the action performed was clean and knowing that:

$$P(x_t = clean|u = clean, x_{t-1} = dirty) = 0.45$$
$$P(x_t = clean|u = clean, x_{t-1} = clean) = 0.9$$
$$P(x_t = dirty) = 0.8$$

solution:



$$P(x_t = clean) = 1 - 0.8 = 0.2$$

$$P(u = clean|u) = \sum_{x_{t-1}} P(clean|u, x_{t-1})P(x_{t-1})$$
$$= P(clean|u, clean)P(clean) + P(clean|u, dirty)P(dirty)$$
$$= 0.9 * 0.2 + 0.45 * 0.8 = 0.54$$

Q3) Explain prediction—correction cycle in your own words.

This cycle is used by filters in cognitive robotics to determine the probability of a robot state (position).

Given a robot at position x' and taken action u, at prediction step, we are trying to predict x which is the new position of the robot after taking that action. How it's calculated differs from one filter to the other.
In other words, prediction depends on Motion Model: $p(x_t|u_t, x_{t-1})$ to give the belief of state x_t, such that: $\overline{bel}(x_t) = \int p(x_t|u_t, x_{t-1})\ bel(x_{t-1})dx_{t-1}$.

After predicting the new position x, a correction step is done to decrease the error range of that prediction. This is done after taking new observations z given the new predicted position x.
In other words, correction depends on observation model: $P(z_t \mid x_t)$ to give the corrected belief of state x_t, such that: $bel(x_t) = \eta\ p(z_t|x_t)\ \overline{bel}(x_t)$.

Q4) Explain Kalman Filter. (There was a 1D problem missing)

Kalman filter is a discrete linear filter based on Bayes Filter, used for same purpose, but with assumption of Gaussian distribution of data. So, it predicts and corrects $\mu\ and\ \sigma$ for the normal distribution that represents a position x.
For direct-measurements:

Prediction update is:

$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = a_t\mu_{t-1} + b_t u_t \\ \bar{\sigma}_t^2 = a_t^2\sigma_{t-1}^2 + \sigma_{act,t}^2 \end{cases}$$

Correction update is:

$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - \bar{\mu}_t) \\ \sigma_t^2 = (1 - K_t)\ \bar{\sigma}^2 \end{cases} \quad \dots \quad K_t = \frac{\bar{\sigma}^2}{\bar{\sigma}^2 + \bar{\sigma}_{obs,t}^2}$$

For Cindirect measurements:

Prediction update:

$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = A_t\mu_{t-1} + B_t u_t \\ \bar{\Sigma}_t = A_t\Sigma_{t-1}A_t^T + R_t \end{cases}$$

Correction update:

$$K_t = \bar{\Sigma}_t C_t^T (C_t\bar{\Sigma}_t C_t^T + Q_t)^{-1}$$

$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - C_t\,\bar{\mu}_t) \\ \Sigma_t = (1 - K_t C_t)\ \bar{\Sigma}_t \end{cases}$$

Q5) Explain the Counting Model.

It's a sensor-model used to determine probability or belief of the occupancy of the cell, or rather the reflection probability of it.

We count two number for each cell:

(1) Hits: number of beams ended at this cell

(2) Misses: number of beams passed through this cell

Then we have $Bel(m^{[xy]}) = \frac{hits(x,y)}{hits(x,y)+misses(x,y)}$, which is the probability of reflection of the cell (value of interest).

Q6) Explain the Odometry Model.

Odometry-based systems are those equipped with wheels or joints encoders.

Odometry model is a model that represents motion of Odometry-based systems. In 3D, it has 6 degrees of freedom, $(x, y, z, roll, pitch, yaw)$. For simplicity, it's usually considered in 2D environments, where there is only 3 degrees of freedom, (x, y, θ).

Odometry model represents motion from $(\bar{x}, \bar{y}, \bar{\theta})$ to $(\bar{x}', \bar{y}', \bar{\theta}')$.

Given information is $u = (\delta_{rot1}, \delta_{trans}, \delta_{rot2})$, where rot1 represents angle of the robot before translation, and rot2 represents its angle after the translation.

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

$$\begin{aligned}\hat{\delta}_{rot1} &= \delta_{rot1} + sample(\alpha_1|\delta_{rot1}| + \alpha_2|\delta_{trans}|) \\ \hat{\delta}_{trans} &= \delta_{trans} + sample(\alpha_3|\delta_{trans}| + \alpha_4(|\delta_{rot1}| + |\delta_{rot2}|)) \\ \hat{\delta}_{rot2} &= \delta_{rot2} + sample(\alpha_1|\delta_{rot2}| + \alpha_2|\delta_{trans}|)\end{aligned}$$

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

(x', y', θ') is the new state.

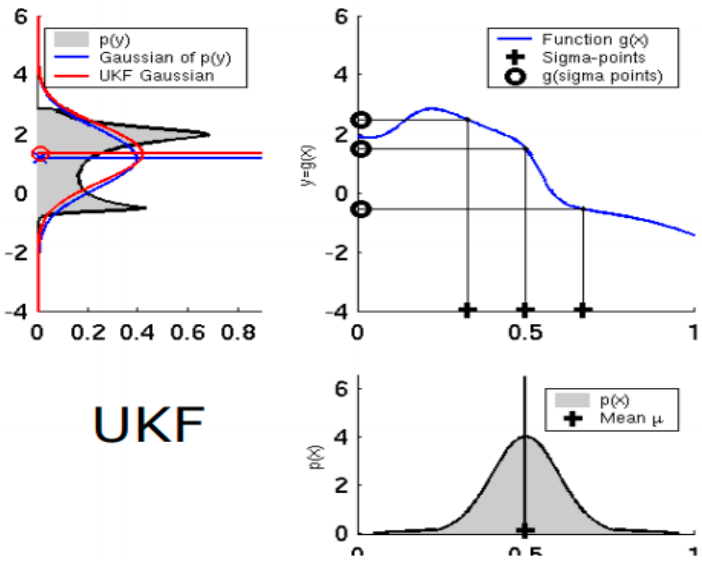
Q7) Compare between Counting Model & Occupancy Grid Model.

Counting Model	Occupancy Grid Model
• Each cell is represented by its reflection probability • Simpler • Reflection with transparent obstacle causes an issue & isn't accurate	• Occupancy grid map discretize the space into independent cells • Each cell is a binary random variable estimating whether the cell is occupied (1) or not (0) • Static state binary bayes filter is applied for each cell

	• Mapping with known poses is easy • More complex • Odd log notation is used to make computations faster • No need for predefined features

Q8) Compare between UKF and KF, sketch the idea of UKF.

KF	EKF	UKF
• Linear Model • Uses Gaussian Distribution • Computational cost is low • Optimal for Linear Gaussian	• Non-linearity is approximated to Locally Linear Model • Uses Gaussian Distribution • Computational cost is low if Jacobians are computed analytically, and medium if they are computed numerically • Accurate in first terms only of Taylor Expansion • Not optimal • Can diverge if non-linearities are large	• Non-linear Model using sigma point linearization • Uses Gaussian Distribution • Computation cost is medium • Better linearization than EKF • Slower than EKF in some applications • Derivation-free & has no Jaccobians • Accurate in first two terms of Taylor Expansion • Not optimal



Q9) Describe FastSLAM

FastSLAM is an improvement on SLAM (simultaneous localization and mapping) that adds some features to it and makes it more robust. It uses Particle Filter to model the robot's path, then for each particle it computes an individual map of landmarks.

Key Steps of FastSLAM 1.0:

- Sample a new pose for each particle
- Compute the particle weights
- Update belief of observed landmarks (EKF update rule)
- Resample

It also considers the data association between particles, which gives a great advantage over EKF.

Feature-Based FastSLAM 1.0:

- Factors the SLAM posterior into low-dimensional estimation problems
- Models the robot's path by samples
- Computes the landmarks given particle pose
- Per-particle data association
- No robot pose uncertainty in the data association

FastSLAM complexity is O(NM), where N is number of particles, and M is number of map features.

FastSLAM 2.0 improves upon FastSLAM by using Improved proposal (prediction) that gives:

- More precise sampling
- Less particles maps
- More accurate maps

Its Key Idea is:

- Perform scan-matching for each particle using its own map
- Fit Gaussian by sampling points around the max of the scan-matcher
- Calculate importance weights using measurement likelihood relative to samples points
- Selective Resampling

Q10) What is the problem of Particle Filters in Grid-Based FastSLAM?

- Too many samples are needed to sufficiently model the motion noise
- Increasing the number of samples is difficult as each map is quite large

Solution is done in FastSLAM 2.0:

- Improve the pose estimate BEFORE applying the particle filter using scan-matching

Scan-matching is a pre-correct short odometry sequences using scan matching and use them as input to FastSLAM.

Q11) Explain Rao-Blackwellization.

Rao-Blackwellization is a method used in SLAM to solve the dimensionality problem, where it notices the dependencies between the given data & the dimensions of the state space.
It uses factorization to exploit dependencies between variables: $p(a,b) = p(b|a)\,p(a)$.
If $p(b|a)$ can be computed efficiently, represent only $p(a)$ with samples and compute $p(b|a)$ for every sample.

This result in factorization of the SLAM posterior as follows:

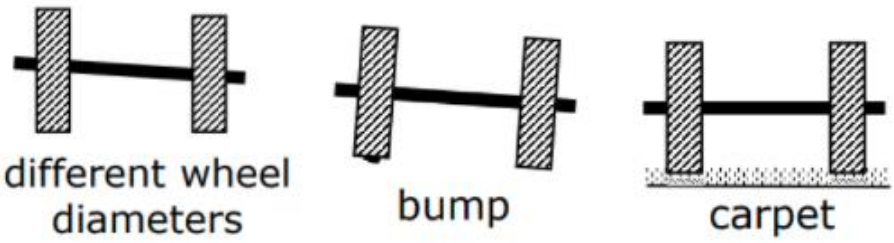
$$p(x_{0:t}, m_{1:M}|z_{1:t}, u_{1:t}) = p(x_{0:t}|z_{1:t}, u_{1:t}) \cdot \prod_{i=1}^{M} p(m_i|x_{0:t}, z_{1:t})$$

Such that the multiplication term is a 2D EKFs, the first term is computed with PF similar to MCL.

Q12) Explain errors in wheeled robots.

Errors caused by:

(1) Industrial defects: example, wheels are not of same diameter.
(2) Bumpy ground: examples, a small stone on the ground.
(3) Friction of the ground: example, carpet or ceramic ground.



Q13) Explain end-point model for expected laser distance measurements that is used for probabilistic localization. Briefly sketch a situation where this model and the Beam-based sensor model (ray casting) give qualitatively different results.

Main idea of end-point model, is taking the reading of the sensor only at the end-point of the beam, and not along it all. It precomputes a likelihood field (distance grid).
Its probabilistic localization is a mixture of (with assumption of independence between different components):

- A Gaussian distribution evaluating the distance to the closest obstacle
- A uniform distribution for random measurements
- A small uniform distribution for max range measurements

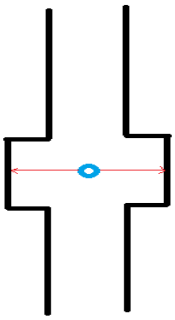
It's considered a highly-efficient model as it uses 2D tables only, but ignore the physical properties of beams & treats them as if they can pass through walls.

The resulting distance grid is smooth w.r.t. to small changes in robot position.

Example:

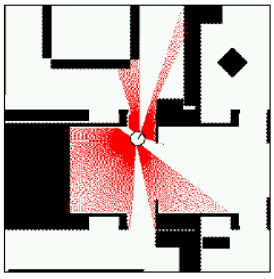
Given the robot pose and the measurements 1m to the left and 1m to the right

The endpoint model will assume the robot is anywhere on the map since it only cares about the end point of the beam, while the Beam-based model gives the result of being in the wide area of the corridor.



Q14) Explain Beam-Based Model.

The beam-based model consists of K measurements $z = \{z_1, \dots, z_k\}$. Each measurement is a result of a "beam", like a laser emitting from the robot in a specific angle, with a maximum range. These measurements are result of the knowing the velocity of the beam (speed of light) and the elapsed time taken from emit to reflection and back (ray casting). These beams consider the physical properties of beams, such that it considers first obstacle along line of sight.



It assumes that the measurements are individual and independent of the robot's pose:

$$p(z|x,m) = \prod_{k=1}^{K} p(z_k|x,m)$$

There are four kinds of measurements & their errors that occur in beam-based model measurements:

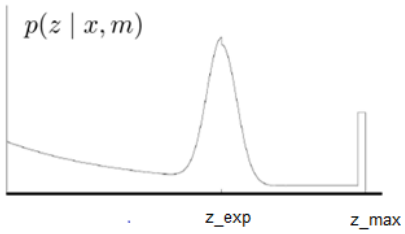
(1) Beams reflected by known obstacles
(2) Beams reflected by moving objects/people
(3) Random measurements
(4) Maximum range measurements

Hence, the Beam-Based Proximity Model obtains mixture of these four components:

(a) Measurement Noise—represented in a Gaussian distribution at the $z = z_{exp}$
(b) Unexpected objects—represented in $\eta\lambda e^{-\lambda z}$ for $z < z_{exp}$
(c) Random measurements—represented in uniform distribution for $z < z_{exp}$
(d) Max Range error—represented by a vertical error at max-range of the beam

Resulting mixture distribution is:

$$p(z|x,m) = \begin{pmatrix} \alpha_{hit} \\ \alpha_{unexp} \\ \alpha_{max} \\ \alpha_{rand} \end{pmatrix}^T \cdot \begin{pmatrix} p_{hit}(z|x,m) \\ p_{unexp}(z|x,m) \\ p_{max}(z|x,m) \\ p_{rand}(z|x,m) \end{pmatrix}$$



Implementation:

- Learn parameters based on real data
- Different models should be learned for different angles at which the sensor beam hits the obstacle (sonar)
- Determine expected distances by ray casting
- Expected distances can be pre-computed

Disadvantage:

- Beam-based model is not smooth at edges
- Not very efficient

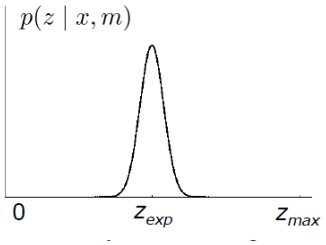
Q15) Compare between Particle Filters and Kalman Filters.

Particle Filter	Kalman Filter
Advantages: • The most-efficient method for non-linear & non-gaussian models • Non-parametric	**Advantages:** • Highly-efficient: $O(k^{2.376} + n^2)$ • Optimal for linear gaussian systems
Disadvantages: • Intensive computations • Effective in low-dimensional spaces	**Disadvantages:** • Restricted to linear systems • Optimal only in case of white noise • Parametric

Q16) Describe the four components of the Beam-Based Sensor Model & sketch the resulting mixture distribution. You don't have to give the formulas.

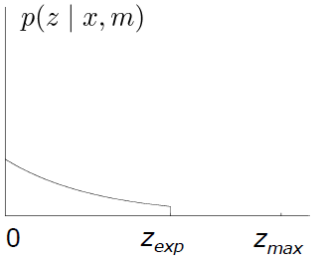
(1) Measurement noise

$$p_{hit}(z|x,m) = \begin{cases} \frac{\eta}{\sqrt{2\pi\sigma}} e^{\frac{(z-z_{exp})^2}{2\sigma^2}} & if\ z \le z_{max} \\ 0 & otherwise \end{cases}$$



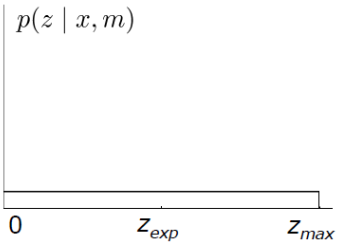
(2) Unexpected objects

$$p_{unexp}(z|x,m) = \begin{cases} \eta\lambda e^{-\lambda z} & if\ z < z_{exp} \\ 0 & otherwise \end{cases}$$



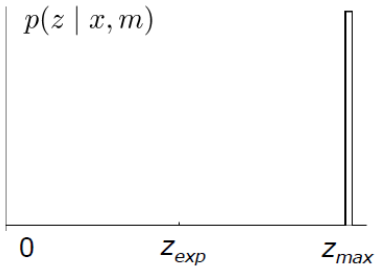
(3) Random Measurement

$$p_{rand}(z|x,m) = \begin{cases} \frac{1}{z_{max}} & if\ z < z_{max} \\ 0 & otherwise \end{cases}$$



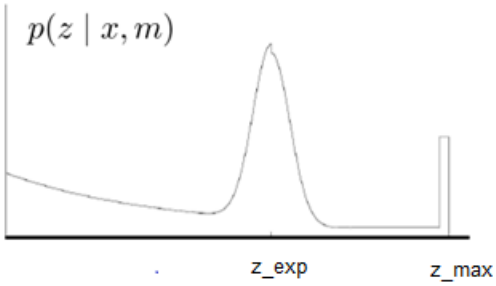
(4) Max Range

$$p_{max}(z|x,m) = \begin{cases} \frac{1}{z_{small}} & if\ z \in [z - z_{small}, z_{max}] \\ 0 & otherwise \end{cases}$$



Resulting Mixture Density:

$$p(z|x,m) = \begin{pmatrix} \alpha_{hit} \\ \alpha_{unexp} \\ \alpha_{max} \\ \alpha_{rand} \end{pmatrix}^T \cdot \begin{pmatrix} p_{hit}(z|x,m) \\ p_{unexp}(z|x,m) \\ p_{max}(z|x,m) \\ p_{rand}(z|x,m) \end{pmatrix}$$



Q17) Explain the problem of the kidnapped robot.

This problem occurs when a robot is in an environment, which is already building and running SLAM, but due to any reason, it suddenly was placed somewhere else, imagine it like taking off the robot, and putting it in a different environment. This will cause the robot to not be able to continue the SLAM or triangulation or whichever method it was using for mapping, and it wouldn't have time to update its belief, till it recovers & starts from initial state again at the new environment.

Q18) – Q20) are missing ☹