HCR project 2

Robot wall following
Simulation environment

- Gazebo
- ROS
- Triton
How to make the robot follow the wall?

Q table: matrix size of states * actions

Continuous states/actions ⇒ Infinite Q table

Discretized states and actions ⇒ Finite Q table

Task1: Manually designed Q-table

Region: left, front, right. Distance: close, medium, far. How many states?

Action: turn left, go forward, turn right

Task2: Let the robot learn the Q-table
Code Structure

Initialize simulation environment and Q table

For each episode:
  Reset robot position and orientation

For each step:
  Choose action based on the current state
  Take action and observe the new state
  Calculate reward
  Update Q table
  Check termination
Initialization

- Launch ROS core
- Launch Gazebo
- Init ROS node
- Declare ROS publisher and service handles
  See Gazebo-ROS communication:
  http://gazebosim.org/tutorials/?tut=ros_comm#Tutorial:ROSCommunication
- Initialize Q table
Reset

- `/gazebo/reset_simulation`
- Optional: Reset the robot to a random position and orientation
  - `/gazebo/set_model_state`
  - `tf.transformations.quaternion_from_euler`
Observation

- Read laser data: /scan
- Suggested observation space discretization
  - Right: [-90:-30]
    - Too close: \( d_{min} < 0.5 \)
    - Close: \( 0.5 \leq d_{min} < 0.6 \)
    - Medium: \( 0.6 \leq d_{min} < 0.8 \)
    - Far: \( 0.8 < d_{min} \leq 1.2 \)
    - Too far: \( d_{min} > 1.2 \)
  - Right-front: [-60:-30]
    - Close: \( d_{min} \leq 1.2 \)
    - Far: \( d_{min} > 1.2 \)
  - Front: [-30:30]
    - Too close: \( d_{min} < 0.5 \)
    - Close: \( 0.5 \leq d_{min} < 0.6 \)
    - Medium: \( 0.6 \leq d_{min} < 1.2 \)
    - Far: \( d_{min} > 1.2 \)
  - Left: [30:90]
    - Close: \( d_{min} \leq 0.5 \)
    - Far: \( d_{min} > 0.5 \)
- Orientation
  - Approaching the wall
  - Parallel to the wall
  - Moving away from wall
  - Undefined

Action

- Suggested action space discretization:
  - Forward: $v_x = 0.3$, $w = 0$
  - Left: $v_x = 0.3$, $w = \pi/4$
  - Right: $v_x = 0.3$, $w = -\pi/4$

- Useful services and messages:
  - /gazebo/unpause_physics
  - /gazebo/pause_physics
  - geometry_msgs.msg.Pose2D
  - sensor_msgs.msg.LaserScan
Reward

\[ r = \begin{cases} 
-1 & \text{if } R = \text{too close} \lor R = \text{too far} \lor F = \text{too close} \lor L = \text{close} \\
0 & \text{on any other situation}
\end{cases} \]

*Proc. Towards Autonomous Robotics Systems. Univ. of Essex, UK.*
Q learning

- Update Q table:

\[ Q^{new}(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \cdot \left( r_t + \gamma \cdot \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right) \]

- Suggested parameters:
  - \( \alpha = 0.2 \)
  - \( \gamma = 0.8 \)

https://en.wikipedia.org/wiki/Q-learning
Epsilon greedy

- Choose action:
  \[ a_t = \begin{cases} 
  \arg \max_a Q(s_t, a) & P(1 - \epsilon) \\
  \text{random action} & P(\epsilon) 
  \end{cases} \]

- Decayed epsilon
  \[ \epsilon = \epsilon_0 d^n \]

- Suggested parameters
  - \( \epsilon_0 = 0.9 \)
  - \( d = 0.985 \)
  - \( n \) is episode number
Termination

- Trapped
  The robot stays in the same position in 3 consecutive steps

- A good policy is learned
  The robot follows the wall in 1000 consecutive steps
  Or the robot has executed 10000 steps in an episode
Demo(3x speed)

Drunk robot :-p
Demo (3x speed)
Demo requirement

Demo should show the robot's ability to:

1. Follow straight wall
2. Turn left 90 degree at L shape corner
3. Turn 180 degree at I shape corner
4. Turn right 90 degree at L shape corner
5. Turn 180 degree at U shape corner
Misc.

- Make sure previous Gazebo instances are closed before launching a new one:
  - killall -9 gzclient
  - killall -9 gzserver
  - killall -9 rosmaster
  - killall -9 roscore
- The robot should be able to follow the wall after 1~2 hour training (around 150 episodes).
- Useful tools:
  - ffmpeg
  - kazam