

CSCI 4302/5302 Advanced Robotics

Assignment 5.1: Discrete Bayes Filter

Due: April 4th 2025

1 Introduction

In this assignment, you will implement a discrete Bayes filter for robot localization in a grid world environment. The Bayes filter is a fundamental algorithm in probabilistic robotics that maintains a belief state over possible robot locations and updates this belief based on actions and measurements.

2 Environment Description

The robot operates in a discrete grid world environment with the following properties:

2.1 Grid World

- The environment is represented as a 2D grid of cells
- Each cell is either free (0) or occupied by an obstacle (1)
- The robot's state consists of its position (i,j) in the grid
- The robot's heading is discretized into four directions:
 - 0: facing right (positive x)
 - 1: facing up (positive y)
 - 2: facing left (negative x)
 - 3: facing down (negative y)

3 Motion Model

The robot can execute three types of actions:

3.1 Action Space

- Forward (action = 0): Move one cell in the current heading direction
- Turn Right (action = 1): Rotate 90 degrees clockwise
- Turn Left (action = 2): Rotate 90 degrees counterclockwise

3.2 Motion Uncertainty

The motion model includes uncertainty in the following ways:

- Motion noise: There is a probability `motion_noise` (default: 0.2) that an action fails:
 - For forward motion: The robot stays in place
 - For turns: The robot still changes heading
- Heading uncertainty: The robot's true heading is unknown, modeled as a uniform distribution over all four directions
- Collision handling: If a forward action would result in collision with an obstacle or wall, the robot stays in place

3.3 Motion Model Equations

For a forward action, the transition probability is:

$$p(x_t|x_{t-1}, u_t = \text{forward}) = \begin{cases} 1 - p_{\text{noise}} & \text{if } x_t \text{ is valid forward cell} \\ p_{\text{noise}} & \text{if } x_t = x_{t-1} \\ 0 & \text{otherwise} \end{cases}$$

For turn actions, the transition probability is:

$$p(x_t|x_{t-1}, u_t = \text{turn}) = \begin{cases} 1 & \text{if } x_t = x_{t-1} \\ 0 & \text{otherwise} \end{cases}$$

4 Measurement Model

The robot is equipped with a simple beam-based sensor system:

4.1 Sensor Configuration

- Four beam sensors, one in each cardinal direction relative to the robot's heading
- Each beam measures the distance to the nearest obstacle or wall
- Measurements are corrupted by Gaussian noise with standard deviation `sensor_noise_std`

4.2 Measurement Process

For each beam:

- Cast a ray from the robot's position in the beam's direction
- Count cells until hitting an obstacle or wall
- Add Gaussian noise to the count
- Return the noisy distance measurement

4.3 Measurement Model Equations

The measurement likelihood for a single beam is:

$$p(z_i|x) = \begin{cases} 1 - p_{\text{noise}} & \text{if } |z_i - \hat{z}_i(x)| < \text{tolerance} \\ p_{\text{noise}} \cdot \frac{1}{1 + |z_i - \hat{z}_i(x)|} & \text{otherwise} \end{cases}$$

where:

- z_i is the actual measurement for beam i
- $\hat{z}_i(x)$ is the expected measurement from state x
- p_{noise} is the measurement noise parameter
- tolerance is the acceptable difference threshold

The total measurement likelihood is:

$$p(z|x) = \prod_{i=1}^4 p(z_i|x)$$

5 Implementation Tasks

5.1 Prediction Step

Implement the `predict(action)` method in `bayes_filter.py`:

- Apply motion model to current belief state
- Handle uncertainty in robot motion
- Account for different action types
- Ensure proper probability normalization

5.2 Update Step

Implement the `update(readings)` method in `bayes_filter.py`:

- Incorporate sensor measurements into belief state
- Handle measurement noise
- Process beam readings correctly
- Normalize posterior distribution

5.3 Analysis and Visualization

- Use the scripts in the test directory to evaluate if your filter is working
- You can visualize using the visualizer

6 Getting Started

6.1 Installation

Install the package with student dependencies:

```
pip install -e "[assignment1]"
```

6.2 Code Structure

The template file `bayes_filter.py` contains:

- Class definition with required methods
- Docstrings explaining expected behavior
- TODO comments marking implementation points
- Helper functions for common operations
- *Only* change code in the functions with TODO docstrings in the Bayes Filter class.

6.3 Testing

Run the provided test suite:

```
pytest tests/test_bayes_filter.py -v
```

7 Submission Instructions

Create a branch on github with the codebase and your implementations.
Provide a link to the branch. <https://github.com/gyanigk/Advanced-Robotics.git>

```

106     for _ in range(n_steps):
107         # Generate measurements from true state
108         measurements = env.get_measurements(true_state)
109
110         # Update filter
111         bayes_filter.update(measurements)
112
113         # Move true state (example: random walk)
114         action = np.random.randint(0, 3) # Random action (0: forward, 1: turn right, 2: turn left)

```

Installing collected packages: filtering_exercises
 Attempting uninstall: filtering_exercises
 Found existing installation: filtering_exercises 0.1.0
 Uninstalling filtering_exercises-0.1.0:
 Successfully uninstalled filtering_exercises-0.1.0
 DEPRECATION: Legacy editable install of filtering_exercises[assignment]=0.1.0 from file://home/gyanig/advanced_robotics/ROS/bayes_filter_assignment/setup.py (setup.py develop) is deprecated. pip 23.0 will enforce this behavior change. A possible replacement is to add a project tool or build-backend to the setup.cfg file and use setuptools --build-backend setuptools.
 Discussion can be found at https://github.com/pypa/pip/issues/11877
 Running setup.py develop for filtering_exercises
 Successfully installed filtering_exercises-0.1.0

```

filtering_exercises filtering_exercises.egg-info README.md setup.py
<- filtering_exercises
assignment1 bayes_environments init_py __pycache__ utils
+ py.test assignment1 bayes/tests/ -v
platform linux -- Python 3.12.2, pytest-7.4.4, pluggy-1.6.0 -- /home/gyanig/anaconda3/bin/python
cachedir: .pytest_cache
rootdir: /home/gyanig/advanced_robotics/ROS/bayes_filter_assignment
plugins: anyio-4.2.0
collected 6 items

assignment1 bayes/tests/test_bayes_filter.py::test_initial_belief PASSED [ 16%]
assignment1 bayes/tests/test_bayes_filter.py::test_predict_forward PASSED [ 33%]
assignment1 bayes/tests/test_bayes_filter.py::test_predict_turn PASSED [ 50%]
assignment1 bayes/tests/test_bayes_filter.py::test_update PASSED [ 66%]
assignment1 bayes/tests/test_bayes_filter.py::test_full_cycle PASSED [ 83%]
assignment1 bayes/tests/test_bayes_filter.py::test_bayes_filter_visualization PASSED [100%]

```

6 passed in 24.37s

Figure 1: All tests passed