UDACITY

‹ Return to "Computer Vision Nanodegree" in the classroom

Landmark Detection & Tracking (SLAM)

REVIEW

CODE REVIEW 1

HISTORY

Meets Specifications

Excellent job! You passed this project! 🎉 👍

Good luck!

`robot_class.py`: Implementation of `sense`

Implement the `sense` function to complete the robot class found in the `robot_class.py` file. This implementation should account for a given amount of `measurement_noise` and the `measurement_range` of the robot. This function should return a list of values that reflect the measured distance (dx, dy) between the robot's position and any landmarks it sees. One item in the returned list has the format: `[landmark_index, dx, dy]`.

The sense function has been correctly implemented and returns a list of values that reflect the measured distance between the robot's position and the landmarks it sees. There is a suggestion though, which you can read in the code review comments.

Notebook 3: Implementation of `initialize_constraints`

Initialize the array `omega` and vector `xi` such that any unknown values are `0` the size of these should vary with the given `world_size`, `num_landmarks` and time step `N` parameters.

should vary with the given `world_size`, `num_landmarks`, and time step, `N`, parameters.

The initialize_constrains function has also been correctly implemented to create and initialize a constraint matrix of 2 * (N + num_landmarks) dimension.

Notebook 3: Implementation of `slam`

The values in the constraint matrices should be affected by sensor measurements *and* these updates should account for uncertainty in sensing.

Constraint matrix values have been affected by sensor measurements and the updates do account for uncertainty in sensing.

The values in the constraint matrices should be affected by motion `(dx, dy)` *and* these updates should account for uncertainty in motion.

Constraint matrix values have also been affected by motion (dx, dy) and the updates do account for uncertainty in motion.

The values in `mu` will be the x, y positions of the robot over time and the estimated locations of landmarks in the world. `mu` is calculated with the constraint matrices `omega^(-1)*xi`.

The mu has been correctly calculated with the formula omega ^ (-1) * xi.

Compare the `slam`-estimated and *true* final pose of the robot; answer why these values might be different.

Great job comparing the final pose estimated by the slam with the true final pose! There is indeed a very small difference. Your intuition justifying the difference is right, but keep in mind that matrix inversion also adds to the uncertainty due to floating point errors, which is not in your control.

There are two provided test_data cases, test your implementation of slam on them and see if the result matches.

The test cases do indeed match the results from the slam output! 🎉

DOWNLOAD PROJECT

1 CODE REVIEW COMMENTS ❯

RETURN TO PATH

Rate this review