eInfochips – An Arrow Company

Internship Documentation

By

Akshay Mahesh Laddha, Engineering Intern (Sensor Fusion)

Manager:
Trevor Bingham

Team:
Gurpreet Singh
Naitik Nakrani
Akash Parikh
Amit Gupta

Internship Completion Date:
August 11th, 2022

Table of Content

1. **Introduction** ---------- *3*
2. **Preface** ---------- *4*
3. **State-Space Models?** ---------- *5*
4. **Sensor Fusion and Algorithms** ---------- *6 - 9*
   - 4.1**.** Kalman Filter
     - 4.1.1**.** Extended Kalman Filter
     - 4.1.2. Unscented Kalman Filter
   - 4.2. Particle Filter
   - 4.3. Bayesian Networks
5. **Why Extended Kalman Filter?** ---------- *10*
6. **Implementing Kalman Filter** ---------- *12*
   - 6.1. Python Implementation
   - 6.2. MATLAB Implementation
7. **ROS Packages for EKF** ---------- *13 - 14*
   - 7.1. robot_localization
   - 7.2. robot_pose_ekf
8. **Simulating the turtlebot3** ---------- *15 - 17*
   - 8.1. Launching
   - 8.2. Moving the turtlebot
   - 8.3. Checking for topics
   - 8.4. Incorporating EKF
9. **Introducing Noise** ---------- *18 - 19*
   - 9.1. Odometry Noise
   - 9.2. IMU Noise
10. **Tuning EKF parameters** ---------- *20 - 22*
    - 10.1. Covariance matrix
    - 10.2. Sensor Configuration matrix
    - 10.3. Sensor diffrential parameter
    - 10.4 Rejection Threshold
11. **Surface friction of Gazebo** ---------- *23*
12. **Conclusion** ---------- *24*
13. **Future Scope** ---------- *25*
14. **References** ---------- *26*

Introduction

Observing an autonomous vehicle that stops at the sign of red lights shows the rapid response of the vehicle and such a response is generated by the fusion the output from multiple sensors. AV and mobile robots are the latest players in this ecosystem of sensor fusion aiming at basically combining sensors that assists in tracking both stationary and moving objects in order to simulate human intelligence. In the context of sensor fusion, predict equation is the one that predicts the state of the car, and update equation is the one that continuously updates that prediction. The predict equation uses the previous prediction of the state (the range of possible state values calculated from the last round of predict-update equations) along with the motion model to predict the current state. This prediction is then updated (via the update equation) by combining the sensory input with the measurement model. Based on the above explaination, It can be deduced that performing sensor fusion provides us with much more accurate results as opposed to using an individual sensor for simulation. Therefore, the topic of focus will be performing Sensor fusion using ROS based packages on turtlebot3. For the given task, we will be using IMU and odometry sensor and aim to introduce certain type of elements that would present us with a more realistic simulation environment including noise addition to the sensors in any form including weather and environmental noise to surface friction and more.

Preface

Observing an autonomous vehicle that stops at the sign of red lights shows the rapid response of the vehicle and such a response is generated by the fusion the output from multiple sensors. AV and mobile robots are the latest players in this ecosystem of sensor fusion aiming at basically combining sensors that assists in tracking both stationary and moving objects in order to simulate human intelligence. In the context of sensor fusion, predict equation is the one that predicts the state of the car, and update equation is the one that continuously updates that prediction. The predict equation uses the previous prediction of the state (the range of possible state values calculated from the last round of predict-update equations) along with the motion model to predict the current state. This prediction is then updated (via the update equation) by combining the sensory input with the measurement model. Based on the above explaination, It can be deduced that performing sensor fusion provides us with much more accurate results as opposed to using an individual sensor for simulation. Therefore, the topic of focus will be performing Sensor fusion using ROS based packages on turtlebot3. For the given task, we will be using IMU and odometry sensor and aim to introduce certain type of elements that would present us with a more realistic simulation environment including noise addition to the sensors in any form including weather and environmental noise to surface friction and more

State-Space Models

A state-space model is basically a mathematical equation Observing an autonomous vehicle that stops at the sign of red lights shows the rapid response of the vehicle and such a response is generated by the fusion the output from multiple sensors. AV and mobile robots are the latest players in this ecosystem of sensor fusion aiming at basically combining sensors that assists in tracking both stationary and moving objects in order to simulate human intelligence. In the context of sensor fusion, predict equation is the one that predicts the state of the car, and update equation is the one that continuously updates that prediction. The predict equation uses the previous prediction of the state (the range of possible state values calculated from the last round of predict-update equations) along with the motion model to predict the current state. This prediction is then updated (via the update equation) by combining the sensory input with the measurement model. Based on the above explaination, It can be deduced that performing sensor fusion provides us with much more accurate results as opposed to using an individual sensor for simulation. Therefore, the topic of focus will be performing Sensor fusion using ROS based packages on turtlebot3. For the given task, we will be using IMU and odometry sensor and aim to introduce certain type of elements that would present us with a more realistic simulation environment including noise addition to the sensors in any form including weather and environmental noise to surface friction and more

a. GRCNN (Generative Residual CNN)

A lightweight, fully convolutional network which predicts the quality and pose of antipodal grasp at every pixel in an input depth image allowing for fast execution and single-loop control with its single-pass generative nature. The proposed goal is to train the GGCNN expert on the Cornell grasp dataset and further combine it with other experts to achieve diverse results. Here, I will step down points required to perform the training.

- Downloading and extracting the Cornell grasp dataset (https://www.kaggle.com/oneoneliu/cornell-grasp)
- Creating a virtual environment and installing the required python libraries mentioned in the .txt file
- utils.dataset_processing.generate_cornell_depth to generate depth images of the point cloud files
- Extracting the GGCNN weights pre-trained on the Cornell grasping dataset using the depth images
- train_ggcnn.py being the primary training for generative grasping
- Architecture of GGCNN -

b. Resnet50 (Residual Neural Network)

A lightweight, fully convolutional network which predicts the quality and pose of antipodal grasp at every pixel in an input depth image allowing for fast execution and single-loop control with its single-pass generative nature. The proposed goal is to train the GGCNN expert on the Cornell grasp dataset and further combine it with other experts to achieve diverse results. Here, I will step down points required to perform the training.

- Downloading and extracting the Cornell grasp dataset (https://www.kaggle.com/oneoneliu/cornell-grasp)
- Creating a virtual environment and installing the required python libraries mentioned in the .txt file
- utils.dataset_processing.generate_cornell_depth to generate depth images of the point cloud files
- Extracting the GGCNN weights pre-trained on the Cornell grasping dataset using the depth images
- train_ggcnn.py being the primary training for generative grasping
- Architecture of GGCNN -

IoU Results for Individual experts