Homework 8 – IMU Localization

Submitted in partial fulfillment of the requirements for the course of

ENPM809T – Autonomous Robotics

By

Adheesh Chatterjee

and

Zach Zimits

Course Faculty
Prof. Steven Mitchell

A. JAMES CLARK
SCHOOL OF ENGINEERING

INTRODUCTION

The aim of this project is to integrate the use of the Arduino and IMU to correct the positioning and turning of the robot and remove errors in measuring ticks to control it. A script is written to control the robot based on IMU, finally, the orientation from the gyroscope and position from the accelerometer of the IMU is used to get the actual position and orientation of the robot

APPROACH

We first assemble the IMU and the arduino as instructed on the Pirate4WD. Then a python class is created to control the encoder and the motors based on the IMU readings. The robot is made to drive in a square – with and without IMU readings. Both the outputs are compared and contrasted

CODE AND OUTPUT VIDEO

The entire repository containing the problem statement and the data can be viewed on https://github.com/adheeshc/raspi-IMU

The output video can also be viewed on YouTube - https://youtu.be/OcxM2eUv5B0

OUTPUT

We observe that in both cases, the motion of the robot isnt perfectly smooth. This is mainly due to the wheel slip. Although there is a drastic improvement in performance when using the IMU. We get perfect orientation angles and the robot knows when it has reached the orientation. Since this doesn't depend on the ticks and only on the IMU, the effect of wheel slip is considerably reduced.

However, on plotting the actual positions, we observe that in both cases the robot thinks its working perfectly. In fact, in the first case without IMU, when we dump the ticks into a pickle file and view them later, we observe that the robot think it moved exactly 1m everytime and rotated a perfect 90deg. A similar case occurs when using the IMU as well. We are considering either using the IMU for position tracking as well to reduce errors in driving forward and back, or we may consider using Visual odometry with the camera to generate optical flow and track the position of the robot