

Application Note
ECE 480
Design Team 8
Chen Hao
11/15/2015

Summary

This document will provide a guide of how control loop used on the robotic project development. Specifically how to use C++ on the Raspberry Pi.

Keywords

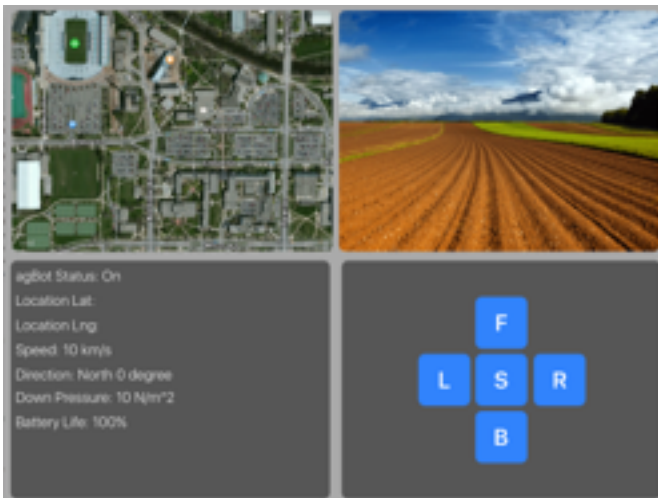
Raspberry Pi, C++

Introduction

Raspberry Pi Model 2B. It runs as a separate computer with 900MHz quad-core ARM Cortex-A7 CPU. It has USB ports, 40 GPIO pins, built in UART, I2C pins and Camera interface (CSI). Raspberry Pi is fully programmable. It has a Linux operating system with the interface.



The user interface is a webpage for monitoring and controlling the robot. It stores on the Raspberry Pi. Since the Raspberry Pi has the wireless connection, it can be a local network server to provide local internet access by using IP from web browser or SSH from terminal.

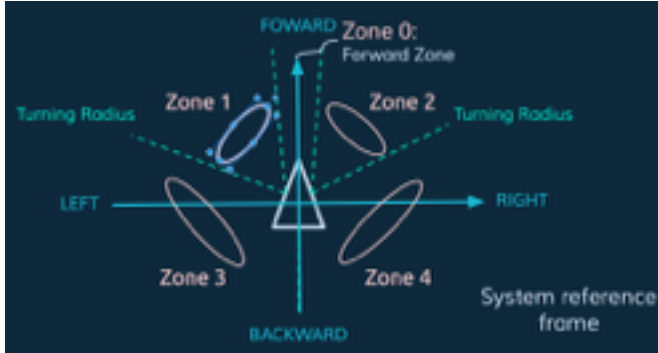


Software Requirement

Xcode, Coda, Terminal

Steps

Control loop logic: In order to determine the comparing location of two objects in programming language, I created a frame for the robot. This frame is keeping updates by the data from GPS and Magnetometer. By calculate GPS coordinate difference, a heading angle can be found on fixed world frame. The facing angle of the robot can be found by the gyroscope. The difference of the heading angle and the facing angle is the parameter that the programming language uses to select next instruction.



```
float desAngleRate = (des_longitude/ des_latitude)/(current_longitude/current_latitude);
float desAngle = atan(desAngleRate);
float gyroAngle = gyroData.selfAngle;
float angleDiff = desAngle - gyroAngle;
if angleDiff > -10 || angleDiff < 10{
    motorForward();
}
else if angleDiff > 10 && angleDiff < 70 {
    motorFL();
}
else if angleDiff < -10 && angleDiff > -70 {
    motorFR();
}
else if angleDiff > 70 && angleDiff < 180 {
    motorBR();
}
else if angleDiff < -70 && angleDiff > -180 {
    motorBL();
}
```

Connect GPIO and C++: On robotic programming, the programming code needs to be able to read and write digital signals on the GPIO pins. For converting data from GPS and Gyroscope sensors, the products provide C++ and python code for easy start. On this project, how to connect the GPIO of motor driver is the key point to control the navigation of it.

After include a third party library,

```
#include <GPIOSerial.h>
```

The functions for initializing GPIO, assigning Pin number, read / write value to Pin are available to use.

```
if gpsNewLng > gpsNewLat * k + c + errorAllowed {  
    if k > 0 {  
        GPIO.setValue(1, forPin: 14)  
        GPIO.setValue(0, forPin: 16)  
    }  
    else {  
        GPIO.setValue(1, forPin: 16)  
        GPIO.setValue(0, forPin: 14)  
    }  
}
```

The function above is part of controlling loop for detecting movement and calibrating steps.



By implement all data reading, frame calculation, routing computation and GPIO instruction to the main.cpp, the base set up of the control loop programming is finished.

Conclusion

This application note provides a step-by-step guide to creating the control loop for robotic project. This is only part of the project. How to use wireless network for monitoring and controlling the robot is the next topic.