Obstacle Avoiding Mobile Vehicle with Rewind

Jonathan A. Garcia-Mallen

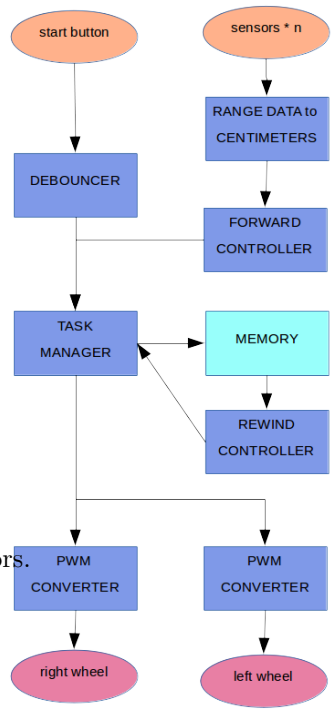
31 October 2017

Overview

The goal is to have a robot go forward and avoid any obstacles in the way. After a certain period of time has passed, the robot will stop. Then a button can be pressed to initiate rewind. When the robot rewinds, it returns to its start pose via the same path it used to reach its end pose. Any turns it makes on the forward trip are done in reverse for the return trip. While rewinding, it performs no obstacle avoidance. To avoid obstacles, three to five ultrasonic sensors, or three to five IR emitters paired with IR receivers can used as rangefinders. The robot itself is supplied by the student.

System Architecture

The task manager coordinates the system. It receives the start signal from the start button and passes wheel commands from the controllers to the PWM modules. The forward controller receives sensor data and calculates wheel commands to send to the task manager. The rewind controller pulls the commands from memory, reverses them, and sends wheel commands such that the robot traverses the path it just followed, but in reverse. The memory itself will likely be BRAM, and will store a subset of the wheel commands received by the task manager from the forward controller. The sample rate will also be stored. PWM converter is a module; we will have a left instance and a right instance. They take wheel commands from the task manager and convert them to waves to control the servo motors.



System Inputs and Outputs

Start button starts. We have n sensors, where n can go from three to five. The sensors are IR range sensors.

Forward Controller

This takes input from the raw_to_dist module, which will do cleanup on the sensors. It receives raw data as input and outputs clean range data in centimeters. Clean data is data that does not jump around. If multipath issues with the IR sensors, raw_to_dist cleans them away. raw_to_dist will be instantiated n times in the forward controller, once for each IR sensor.

The forward controller receives this clean sensor data as input. It is a simple proportional controller. The sensors on the right side of the robot push the robot away from obstacles on the right; sensors on the left push the vehicle away from obstacles on the left. This is done with the following calculation:

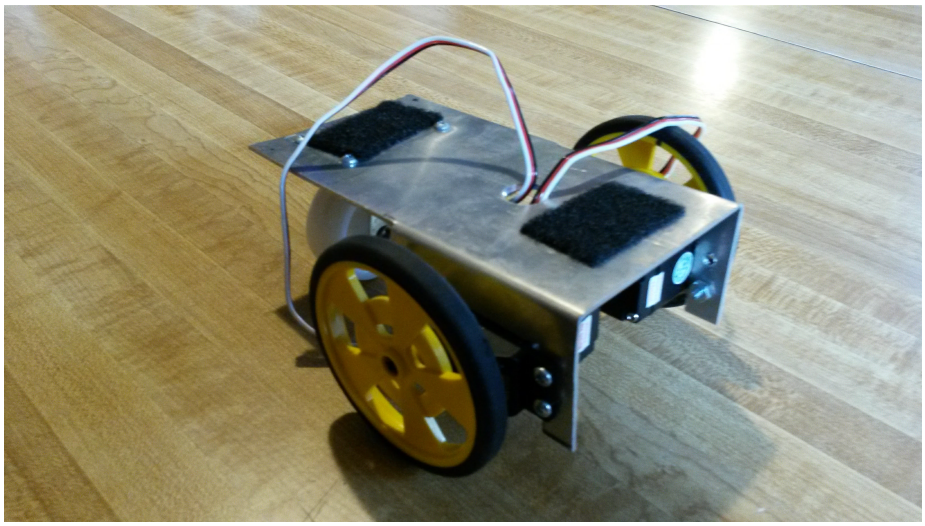
$$wheel\_l = speed * (\sum_{i=0}^{\lfloor n/2 \rfloor} p_i r_i) \qquad (1)$$

$$wheel\_r = speed * (\sum_{i=\lceil n/2 \rceil}^{n} p_i r_i) \qquad (2)$$

The value r_i is the clean sensor data. The values speed and p_i are parameters passed into the forward controller module. Both wheel values range from zero to 127, and are 7-bit wires. Note that the vehicle only ever moves forward. We will make no fast turns.

Rewind Controller and Memory

PWM Converter and System Outputs



Task Manager

The task manager runs a small three-state finite state machine. The robot is in either IDLE, FORWARD, or REWIND states. The most important parameter it receives is the time of the run. This may be determined by the switches on the Nexys 4, but will initially be set at ten seconds.

The system transitions from IDLE to FORWARD when the task manager receives the debounced start signal from the button. This module then sends a forward_enable signal to the forward controller to begin receiving wheel commands. These commands are downsampled and saved into the memory.

Complexity Analysis

Bill of Materials

- wheels and chassis - battery pack - SpringRC SM-S4303R Continuous Rotation Servos - IR

Timeline