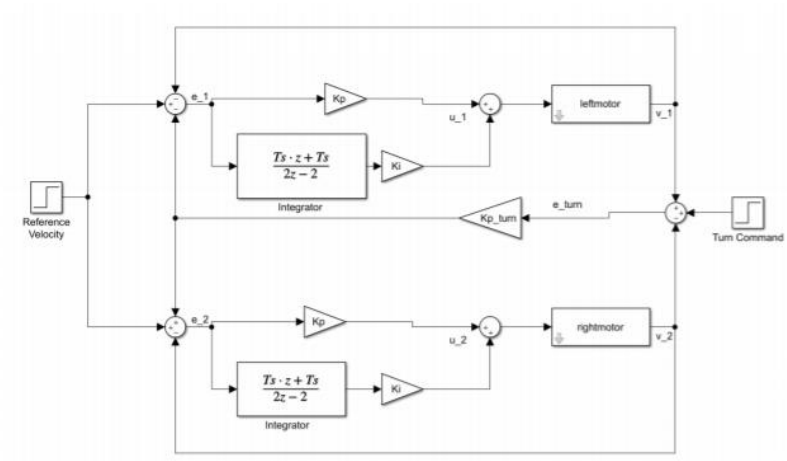
SE 423 Final Project

Jacob Butera

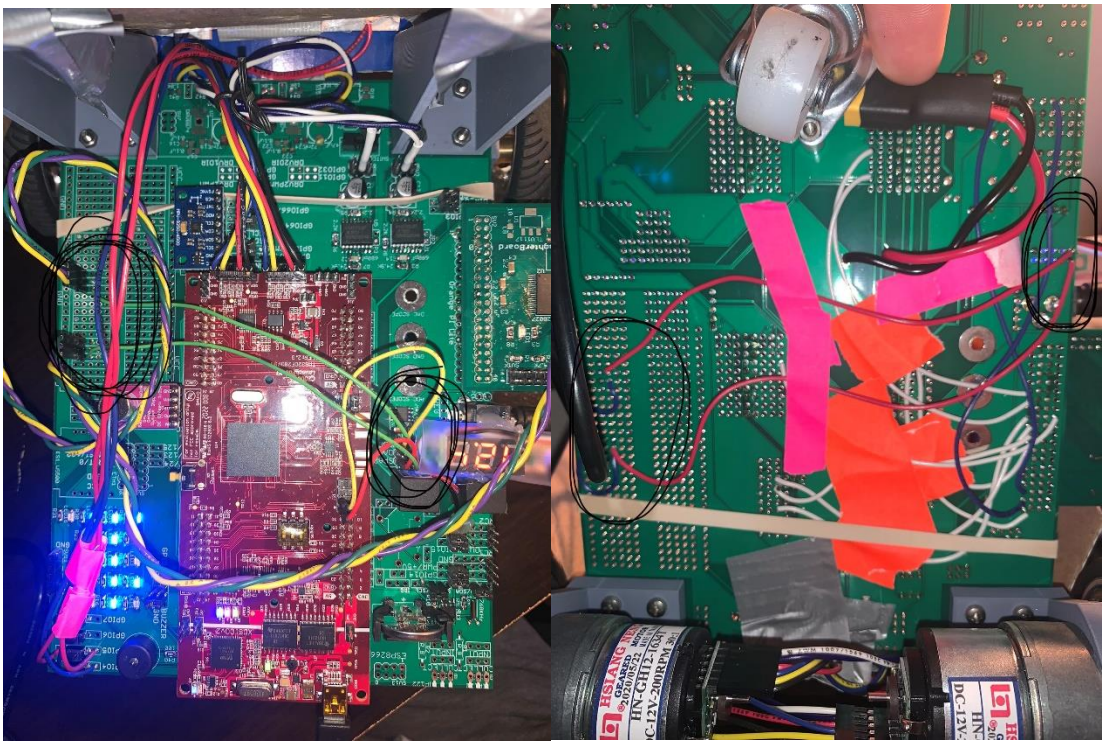
5/11/2021

This project takes advantage of coursework developed throughout the semester. Namely, it uses lab 6 code where I used DC motor speed control and car positioning to steer the 3 wheeled car using keyboard buttons. Lab 6 code took advantage of eQEP peripheral of the TMS320F28379D processor and allowed me to use the DC motor magnetic encoder to sense the angle of the DC motor. This angle was converted into left and right wheel distance, and then into velocity of the car. Finally, a coupled PI speed controller was implemented that allows the steering of the car, while a gyro and accelerometer feedback forces on the car. The PI controller block can be seen below.



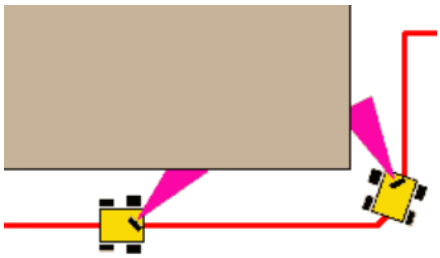
The final project converts the controlled car into a wall-following and obstacle avoiding car. The main sensors involved are 2 IR sensors and 2 switches, along with the 2 motors. The IR sensor readings are referenced using ADCD regs, and the 2 sensor values are referenced using GPIO 4 and 5. The motors are commanded using the ePWM2A and ePWM2B.

The IR sensors had to be wired into the board. I checked the specs on Pololu for the Sharp GP2Y0A21YK0F IR sensors, and found a simple 3-pin setup. From the sensor itself, the three pins are ground, Vo, and VCC. After soldering a pair of 3 pins, I wired ground, Vo, and VCC to the TI Launchpad and Green Board. Pictures of the wire soldering can be seen below circled in black:



The switch sensors were simpler, as the wiring only involved a quick check of which 2 pins created a short when the lever was pressed. For the practicality of my project, a pressed switch is a short. These pins were soldered and then attached to the GPIO 4 and 5 pins around the pushbuttons.

The picture below explains the configuration for testing the car. The two IR sensors are placed at an angle, and the analog voltage received is referenced as a distance. The higher the voltage, the closer the distance.

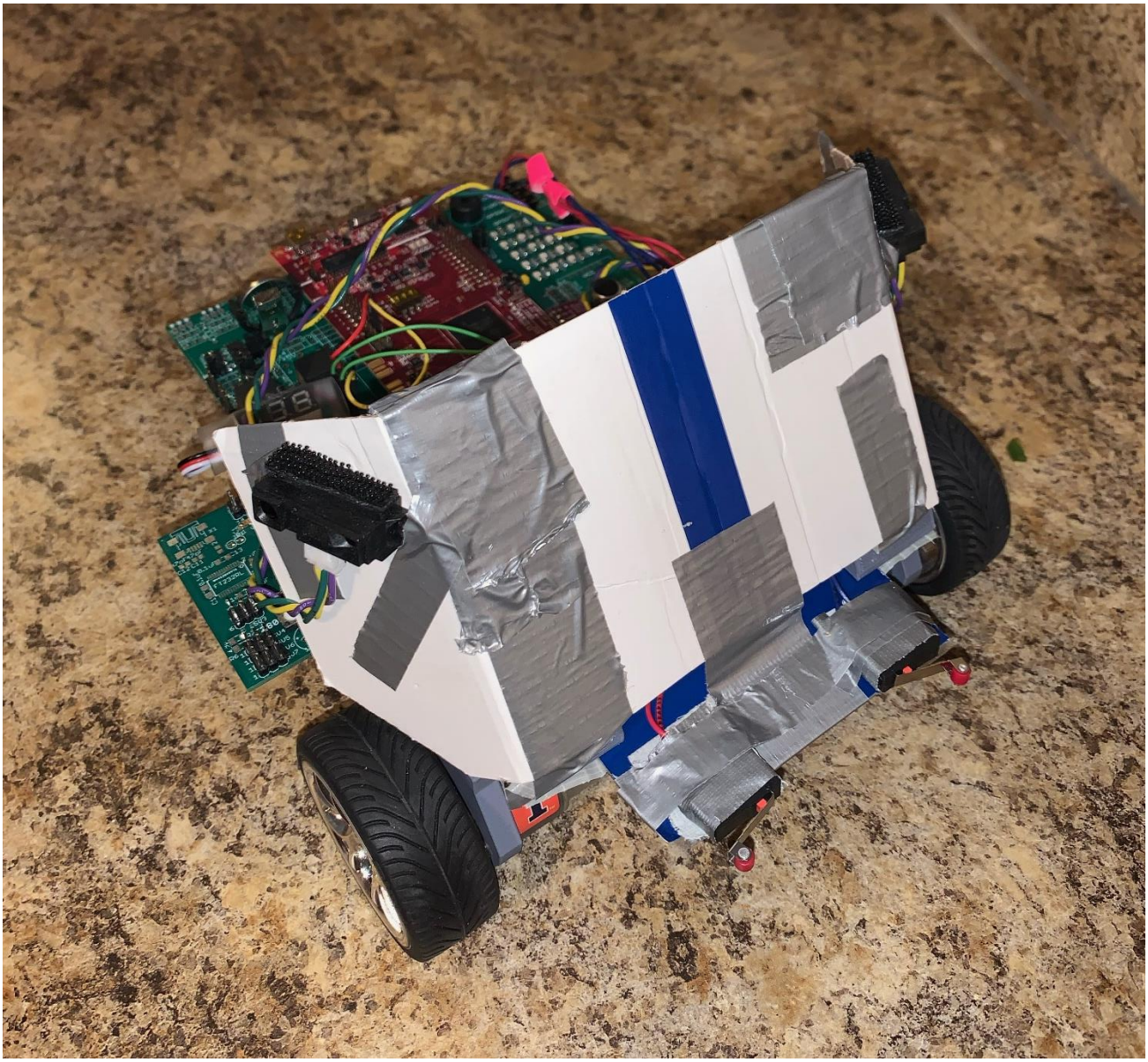


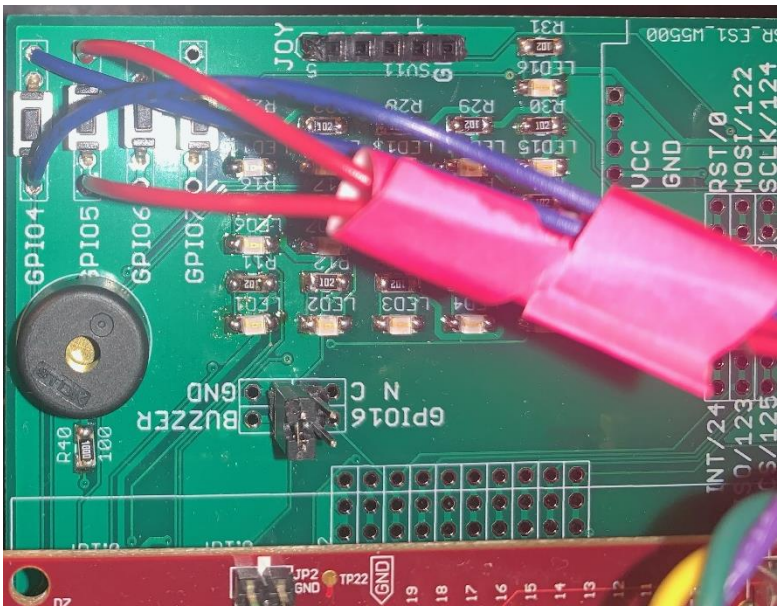
The output voltages from the IR sensors are coupled with the PI controller to turn the car right when too close to an object on the left , and vice-versa. This is accomplished using the “turn” setpoint from lab 6 code. Turn essentially controls the amount one motor ‘s speed exceeds the other. This setpoint can be integrated easily into wall avoidance, simply, when the left IR distance is within a threshold distance of 1 I set the turn value to to -.75 to turn the left wheel and slow the right wheel speed. With this, the car turns right and avoids the left wall. The same process is implemented for the right wall using a turn value of .75. This turn value was tested, as a value of 1 caused the car to turn too much and not follow a wall. This turn value an be changed to make the car an obstacle avoider using a higher turn magnitude, and a wall-follower with a less severe turn.

The two switches in front are essentially a backup incase the car runs head-on into an obstacle and avoids the angled IR sensors. When the switches are pressed, the car reverses for a set time and resumes driving forward. The switches are controlled using the readSwitches function from lab 4, where a value of hex 0x1 and 0x2 control the first and second switch. When these hex values are activated with a press, the VRef value from the PI controller is changed to a negative for a certain time.

Through testing I was able to flash the code onto the Launchpad, and have the car avoid obstacles and ride a wall using my sensors. The setup works rather well but can be refined better into a sensor state machine. Additionally, the reference distance can be further refined to allow the car to follow walls smoother. Lowering the reference distance will make the car follow closer to the wall. The IR and switch sensors were held in place using cardboard and adhesive. The heights at which they are placed were finalized at the optimal position for the IR sensors to read properly.

The next pictures include the switch sensor setup and and the entire setup of the car:




GPIO4
GPIO5
GPIO6
BUZZER
GND
GPIO16
VCC
GND
MOSI/122
JP2
GND
TP22
GND