Senior Design
ENG EC 464

Memo

To: Professors Pisano, Hirsch, Alshaykh
From: Team 9 (Giacomo Coraluppi, Alex Zhou, Will Krska, Jonathan Ye, Nick Marchuk)
Date: 4/21/2023
Subject: Final Test Report

1.0 Equipment and Setup

1.1 Equipment

- Demonstration vehicle
  - 3D-printed, drone motor-controlled wheels attached to car base
  - Interactive driver input box: steering wheel and throttle pedal
  - LEDs attached to rear wheels to indicate car slippage
- Electronics
  - 12V power supply
  - Two weather-sealed Polycase containers
    - One houses the auxiliary controller, the other holds the main and watchdog controllers
    - Supporting components in each case, including relay switches, DAC converters, and accelerometers
  - Wiring to connect Polycase containers with driver inputs and car motors
- Software
  - Simulink Scripts
    - PID implementation on controller
  - C code scripts for main, watchdog, and motor controllers
    - ESP-IDF libraries
    - Configurations for UART, I2C, and other serial communication
  - Arduino code scripts for auxiliary controller
    - Adafruit ISM330DHCx Arduino library

1.2 Pre-Testing Setup Procedure

Compile and upload the respective C-code for the appropriate controllers from the user's source computer to the microcontrollers. *aux_controller.c* goes to the auxiliary controller, *main_controller.c* corresponds to the main controller, *watchdog.c* will be for the watchdog controller, and *motor_controller.c* will be for the demonstration vehicle.

The code files should be located under “SrDes...” folders within our Github repository. Compiling and uploading the code will involve an installation of the ESP-IDF interface, followed by the following commands within the project directory folders:

(1) “. ./export.sh”
(2) “idf.py set-target esp32s3”
(3) “idf.py build”
(4) “idf.py -p [PORT] flash monitor”

After the code is successfully uploaded to the hardware with no compilation errors, set the demonstration vehicle upright so that it is on a flat surface and that the wheels are sitting above a flat surface and away from other objects. Turn on the 12V power supply generator that is connected to the vehicle, and after 1-3 seconds plug the LiPO pouch-style battery into the demonstration vehicle. A circular knob can be found on the main controller casing, which turns on/off the electronic components as well. Once LED lights are visible within the hardware parts, user can proceed with driver inputs testing.

2.0 Measurements Taken

2.1 Testing Procedure

Locate the steering wheel and throttle pedal box interface. The steering wheel can be turned to the left or to the right. The throttle pedal can be pushed down for power input. Users can first push the throttle pedal all the way down to simulate the car going at maximum speed. For the steering wheel, the user can turn the car all the way to the left or all the way to the right to maximize the turning angle of the car. The user can also mix and match the inputs, such as adding a moderate amount of throttle power while turning the wheel in any direction.

2.2 Measurable Criteria

- If the throttle pedal is pushed all the way down, the wheels on the car should be turning at the fastest rate possible to simulate the maximum speed.
- If the steering wheel is turned all the way to the left, the right front wheel should have more power applied (rotates faster) than the left front wheel.
- Vise versa, if the steering wheel is turned all the way to the right, the left front wheel should have more power applied (rotates faster) than the right front wheel.
- For any other combination of driver inputs, the demonstration vehicle should be outputting behavior that resembles the patterns above to a lesser extent of power.
- LED lights attached next to the rear wheels light up when car slippage occurs from driver inputs.

- Driver input data is sent successfully from the steering and throttle to the controllers, then into the expected output of the demo car's behavior.

2.3 Results

- We were able to show in our simulation that our system is working correctly and could be applied to the actual BU Terrier Motorsports car once it will finally get built.
- The inputs to the system for the demo are the steering wheel, the throttle pedal, and the simulated speed of the vehicle.
- Starting from rest, the user can press down on the accelerator pedal to increase the speed at which the wheels are turning, and this can be seen because the wheels are turning more quickly and are making more sound. This also acts as a normal car because if the user stops pressing the pedal, the wheels will keep turning but will slowly come to a halt. Also, the maximum RPM at which the wheels on our model can spin is the maximum RPM at which the wheels on the actual car can spin, representing the maximum speed of the vehicle.
- The auxiliary controller (arduino NANO) is working correctly as the inputs coming from the steering and throttle inputs are converted to scaled signals which can be sent to the main controller over UART with wires the appropriate length that will be used in the actual car. The accelerometer data, which is calculated by the auxiliary controller, is also sent to the main controller.
- The main controller (ESP32-S3) is working correctly as it uses the inputs to model what the car would be sending to the left and right rear wheels under current conditions. The data from the main controller is sent to the model car after the DACs convert the output signals from digital to analog.
- The drone motors on the model car show our system working, as the rear wheels are independent and change speeds depending on whether the car is going straight, turning left, or turning right. For example, if the car is turning to the left, the right rear wheel will be spinning faster than the left rear wheel, which in the real car would reduce traction and slip. The opposite goes when the car is turning to the right. The angle at which the driver is turning and the speed at which the car is currently moving also impact these calculations as could be seen in the demo, as there is a larger difference between the speed of the rear wheels when the car is taking a tighter turn or traveling at a greater speed.
- We show that the Matlab Simulink Linear Model of the car is working through a Matlab simulation, which shows how the car would respond under specific conditions by graphing power to left and right wheels respectively over time based on current inputs and speed.
- We show the two boxes are assembled and sealed from the outside environment. No components are loose.

3.0 Conclusions

In our final project testing, we successfully completed our system and demonstrated what the outputs would look like in a rear car, once it actually gets built. We built the entire system to be easily implemented on the rear car eventually, as we used the same sensors that the car itself would be using and used all the appropriate wiring.