Due by 5 pm Tues. Feb. 9 on BCourses.

The purpose of the project proposal is to give you early feedback on your design ideas, to help to steer you away from some of the common design flaws. Now is the time to plan what functions you want. The project proposal is a non-trivial amount of work; we would like to see ≈ 8 hours of effort per team member. Please read the NatCar rules from the UC Davis site. Here is an outline for you to follow:

1. Overall Strategy (1 page) (30 %)

1.1 (10%) Think about "winning" strategies. A "winning" strategy is one which results in a functional, thoroughly tested, and reliable vehicle with a minimum of effort. Separately list essential and "wouldn't it be great to have" features. A good rule of thumb is that you would like to be 80% complete with 20% of budget.

1.2 (15%) Now speed is of course an important consideration, but speed without stability and robustness won't get you far. Will your vehicle rely on raw speed, expert navigation, expert braking, etc? How does your strategy affect the types of sensors, signal processing, control, and path planning strategies you will use? Briefly discuss algorithms to consider.

1.3 (5%) Which parts of the project are least understood?

2. Hardware Design (35%)

2.1 Input/Output Connections (10%)

Make a table showing ESP32-WROOM pin number, Huzzah32 pin number, and pin functions for all input/output needed for the car. This includes steering servo, motor ESC, motor velocity sensor (using SparkFun RedBot Sensor), and TSL1401 line camera.

2.2 Input/Output Connections (5%)

Which Huzzah32 IO pins if any are not available for general purpose use, as they are dedicated for other functions? For example if a JTAG debugger were added, what pins would it use?

2.3 Velocity Sensor Mounting (10%)

Include labelled photos which indicates a good location for mounting the SparkFun RedBot Sensor and an encoder wheel. (With appropriate scale, powerpoint can be used with a superimposed image of the sensor on an image of the car.)

2.4 Camera Height and Angle Model (10%)

Assume the camera has a field of view of 0.5 m at a distance of 0.5 m (or 0.1m field of view at 0.1 m distance, etc.) The camera has a 128 pixel line sensor. The camera mounting height and angle determines lateral resolution, and preview time. A high camera will pick up cones and possibly parallel tracks (algorithms can filter these out).

Choose an initial camera height and angle. Show in a drawing of geometry what the expected field of view is. If the tape is 2.5 cm wide, how many pixels wide will it be? If the car is going 3 m/s, how much time is there between seeing the track and the front wheels reaching that location?

3. Software Structure (1 page) (35%)

3.1 (10 pts) Make a first pass, high-level block diagram of the software system. (A block diagram should be like Fig. 5 in Thrun et al. [1] showing functional modules and their interconnection, **not a flow chart**.) Be sure to include UI/debugging block.

3.2 (15 pts) Make a table listing each module, key input/outputs, how often the module should execute, expected data rate for module input/output, and relative priority. For each module, specify whether the module runs as part of the `control_task()` loop, a separate scheduled task, or an interrupt (see for example `timer-group0_isr()` in SkeletonHuzzah). The tutorial [2] discusses design trade-offs.

3.3 (5 pts) For each module in part 3.2, Explain how it will communicate with other modules, e.g. queues, global variables, dual buffers, etc.

3.4 (5 pts) List those things you need to understand more completely before you will be able to design your software.

References

[1] S. Thrun et al., "Stanley: The robot that won the DARPA Grand Challenge." *Journal of Field Robotics,* 23(9), 661-692, 2006.

[2] "Real Time Application Design Tutorial" https://freertos.org/tutorial/index.html