CriSp Indoor Lab

Jens-Patrick Langstrand, Maben Rabi,
Björn Andersson, Sundarrajan Gopalakrishnan

June 25, 2021

1 Current actions

- Soldering of the rangefinder sensors.
- Test the laser rangefinders and see how they reflect on the material we are thinking of using for walls. Will it still detect reflections if the beam is at an angle to the wall? Try it with the pool noodle directly, if it does not perform well try with some textile that could cover the noodle.
- There is still a lot of equipment in the classroom taking up space. See what can be done about that. Pushing and stacking tables on one side was suggested. Extra chairs that cannot be placed on top or under the tables could perhaps be moved into a neighbouring classroom with a note saying where they belong and that they should not be used.
- The build process of both the vehicles and the track should be documented. It could be a short 1-2 page manual. What would be required to perform changes to the track? For instance, increasing elevation at some location on the track. It would be good if the manual answers questions like this.
- It would be good if a GitHub page was maintained for this work.

2 Software architecture

What is required for the software architecture? What is needed? Which programming language should be used? Perhaps we can start with Python or C++ as there should be some available packages. Which OS should be used on the vehicle? From discussions it seems like Linux is a good choice for the vehicle OS.

At a later stage we should be able to add-on the 3D camera as part of the architecture. The system should be able to account for 4wheeled vehicles and 4wheel vehicles with trailers. The vehicles should be able to follow a path. High speed will be required. The vehicle should be able to maintain its speed with small variations allowed. The extent of the variations allowed can be

determined from experiments. To follow the track the vehicle could attempt to stick to a wall at a certain distance, or use visual camera. A scheme for setting up a scenario and running an experiment is needed. It should be able to start from the workstation machine. The workstation should also be able to stop the vehicle at any time.

The programming language to use. Perhaps he should start with Python, C++ would also be ok.

3 Vehicle

From discussions it was decided that any language could be used for the programming of the vehicle. However using ROS would be beneficial as many existing packages already exist which can perform required parts of the vehicle control and sensing. It would save time as we do not need to program all the blocks required to control the vehicle and interface with the sensors from scratch. It contains for instance nodes that communicate with the Jetson framework, nodes that implements wheel odometers and sensor fusion. The sensors can be fused directly in ROS to give a whole dataset directly. Since ROS runs natively in the system it should be fine to use for real-time use together with Ubuntu.

The vehicle is driven by one common motor via gearing and power transfer. We cannot read individual wheels directly without adding additional sensors. But that might be problematic with regards to positioning on the vehicle and available inputs to the control board. Maben thinks reasonable accuracy can still be achieved by using the gear ratios etc. It should be possible to estimate the angle of all the wheels and the wheels move in a synchronised manner.

3.1 Vehicle assembly

During assembly some of the parts were found to be wrong. The correct parts were ordered. The vehicle itself did not seem that difficult to assemble so far. The Nvidia Jetson TX2 requires Ubuntu and ROS to flash the board.

3.2 Questions concerning the vehicle

Is there a possibility of changing the vehicle from front-wheel drive to rear-wheel drive was explored? It was determined that this was not easily achievable. Is it possible to change tires for more realistic? The indoor model will have smaller weights, stiffness of tires used, can we change tires? It turns out that there are many possible tires that can be purchased ranging from slick to big tread patterns. Can we alter the tyre friction by applying for instance a thin strip of cello tape on the tyre surface? Should we add weight on the vehicle to scale down more properly?

4 Track

Pool noodles will not be used for the walls of the track after all. Some kind of cardboard or paper will be used instead. If we use the yoga mat pieces on the floor without any attachment mechanism, will they move around due to the vehicle? Maben thinks that if they are arranged in a grid that movement would not happen easily.

4.1 Track challenges

As the room where we want to setup the track cannot be permanently reserved we must be able to move the track. Therefore this must be part of the track design. The design of the track stays the same, but the material used to build it is changed to some form of EVA-foam. For instance yoga mat type material that are sold in pieces that can be assembled together like a puzzle. These pieces have been ordered and has arrived. The pieces can be cut to fit the track using a vinyl cutter.

A suggestion to use pool noodles and 3D printed holders for flexible and rejiggable side walls was made. The 3D holder can be made to support two pool noodles in height and be fastened to the foam. This would allow flexibility to change the course as needed.

5 Experiments

As part of experiments the start and end of the segment of interest should be specified. The yaw angle of the vehicle might be one option to how we can determine if the vehicle has passed the start segment, moved through the road section and finally having passed the end of the segment. The yaw angle might be one option to check, it can be used as a function of distance instead of time, and mapped to the predefined path. Another option could be to use x,y positions as a function of distance and assume a single road lane.

The first stage should be to run a simple scenario and verify that data can be extracted.

5.1 Experiment tasks

- Task 1: Localisation, trajectory control. Find orientation and speed. Adjust motors to stick to track. Three basic ways were suggested for localisation and orientation estimation. Using laser range finders to stay some distance away from the wall. Using visual camera to follow some tape along the track. Using the Nvidia Jetson to detect the track and walls and control the vehicle to stick to the track. Information from the IMU ir distance unit can also be added. A hybrid combination of approaches can also be considered.

- Task 2: Then make sure to record all the sensor data that we need. IMU, Steering angle, Wheel speeds.
- Task 3: The addition of sensors for ground truth measurements of vehicle motion. IR position tracking systems (Must be purchased) and a 3D camera already available at Hiof has been suggested.
- Task 4: Once data has been collected the difference between the intended vehicle motion and the actual vehicle motion is of interest.

5.2 Experiment safety

Once the track has been setup we need to make sure that if a crash does occur, it should not be a bad crash. An emergency stop button on the workstation is needed. We should also surround the track with protection. Plastic bags filled with sand was suggested.

The vehicle should detect conditions where it is not doing what it should and stop as a result. One condition could be that it can not see any tape. We could use green tape on the edges of the track, so that if it sees green the vehicle should stop. Yellow tape could be used in the bends of the track and white tape to define the desired path. Then when turning if it sees both white and yellow it is ok. Otherwise if it only sees yellow it should stop. Another approach could be to use the vehicle IMU sensors. High values could indicate bad behaviour and be used to stop the vehicle. A difficulty with this approach is that the vehicle will be pushed to its limits, so it might be hard to find good cutoff thresholds.

6 Potential algorithms

- Pure pursuit: Want to go to the next target all the time. Based on position, make calculations to find turning angle etc. Can make use of the IMU and a camera for this.

7 Data

7.1 Data recording

All recorded data should at the least be saved locally on the vehicle. Ideally it should also be able to transmit the data to the workstation. As the vehicle runs through a scenario all signals should be recorded, then we can extract required signals for analysis, and add unused signals if desired. Signals that should be recorded:

- Vehicle IMU
- Steering angle
- Wheel information, angle, velocity etc

- Ground truth for vehicle motion

7.2 Data analysis

The recorded signals should be analysed to decide if some goal has been achieved, this time frame is important and should at the longest only take .5seconds, perhaps including communication with the work station. That time is borderline acceptable, 10ms would be better. Once the data has been recorded we are interested in the difference between the intended direction of travel and the actual direction of travel.

8 Workstation

There should be a workstation next to the track with a machine that communicates with the embedded board on the vehicle. The machine should be able to setup and run experiments.

9 Ideas to explore

Can we make an ice track for the vehicle to go on? Maybe we can ask to lend ishallen or the outdoor skating park to run experiments on ice? We might need to bring some wooden railings, or perhaps we can reuse the pool noodle approach.

10 Resources

Useful resources that has been found are collected here.

- An article on creating a software encoder for speed and steering angle. [Article]
- Found a paper on estimations without wheel encoders. [Paper]
- Found a paper from Chalmers on autonomous RC: Engström, Patrick, et al. “High Performance Autonomous Racing with RC Cars.” High Performance Autonomous Racing with RC Cars, 14 May 2018. [Paper.]