|  |  |
|---|---|
| To: | Professor Merz |
| From: | Benjamin Nitkin |
| Subject: | IGVC Progress Report |
| Date: | November 6, 2013 |

---

This week, the software team finalized divergence mapping and began to look forward. As of Tuesday, the cameras generate a fairly dense divergence map, and should do better outdoors. Their main weakness is movement. Moving forwards, we're going to start looking at other aspects of the robot's electrical systems. Most of our focus will remain on sensors, but the power electronics need to be roughly designed so that the chassis team can move forward with their models.

Since last week, the software team made a few tweaks to the cameras that vastly improved divergence mapping.[1] First, the camera's baseline was halved from ~40cm lens-to-lens to ~20cm. Since the matching algorithm can only explore so much of one image to find each matching pixel, reducing the difference between the images reduces the amount of processing power to find a match. A shorter baseline helps short-range mapping at the cost of long-range accuracy.

Second, the cameras are now statically mapped. Previously, the left camera was referred to as `/dev/video0`, and the right was `/dev/video1`. Under Linux conventions, `video0` is the first camera to be plugged in; `video1` the second, so divergence mapping depended on the cameras being attached in order. The cameras are now mapped to USB ports the computer has a rules file that maps the device in one USB port to `/dev/videoleft` and another port to `/dev/videoright`. ROS uses `videoleft` and `videoright` in divergence mapping. As long as the cameras stay in the same ports, their orientation will be preserved in software.

Third, we attempted to synchronize the cameras, without success. An online tutorial suggested shorting the VSYNC (Vertical Synchronization) of one camera to the FSIN (Synchronization input) of the other. Connecting the suggested locations didn't work.

Moving forward, the software team is going to begin looking at sensors and white-line tracking. Once all data streams are functional, we'll start on navigation algorithms. For sensors, we're planning to upgrade from serial sensors to USB ones that will interface directly with ROS. They may be slightly more expensive, but native compatibility will shave weeks off of development time. White-line tracking will be developed using either stock ROS or by developing a custom ROS node. White-line tracking and a depth map will be combined to create a local map; a target heading and distance from GPS will allow navigation. The other sensors help the robot drive straight and match the software-requested velocity.

As the chassis team finalizes their model for the robot, they need our input on any large electronic parts. Most of the sensors are small enough that they don't need dedicated space, but a few power components will need to be included in the Inventor model. These include batteries, a power distribution board, motor controllers, motors, and the laptop.

1 I write about both "divergence mapping" and "depth mapping". The former finds distance between matching pixels in a stereo pair; the latter maps divergence to real depth. Mapping divergence to real depth should be a simple calibration once the final camera mast is complete.