To: Professor Merz
From: Benjamin Nitkin
Subject: IGVC Progress Report
Date: November 13, 2013

This week, the software team created a synchronized pair of cameras and verified operation on the oscilloscope. The team also began to reexamine options for sensors, with a focus on compatibility with ROS and low set-up times. Moving forwards, the team will research options for navigation and begin to work on high-level integration of sensors in the robot.

This past week, I synchronized our two PS3 Eye cameras with some very fine pitch soldering. Each camera has two pins of interest to us: VSYNC and FSIN. VSYNC is usually grounded, but outputs 5v for a few microseconds as the camera begins to capture a frame. FSIN is usually ignored, but when it's attached to an input, it forces the sensor to capture an image when the input goes to 5v. I soldered a wire to VSYNC on one camera and VSYNC and FSIN of the other. Shorting VSYNC to FSIN forces the cameras into a master/slave configuration, as I demonstrate in this video: http://youtu.be/VamHPoZzCMg. A synchronized stereo pair should tolerate movement better than an unsynchronized one.

Our initial set of sensors operated on 3.3v and communicated with the computer through an Arduino. Considering our limited time, the software team decided to switch from serial sensors to slightly more expensive USB variants with native ROS support. Andrew's been looking into sensors that meet our revised requirements.

Now that divergence mapping is working, the software team is beginning to look into high-level integration. Navigation is, in a way, the highest level. The navigation stack draws on a robot transform frame, current position provided by various sensors, obstacle detection, goals, and other information to plot a path.

Transform frames describes the positions of robot parts relative to each other. Although our robot only has two moving parts, the navigation stack needs to know where the cameras are relative to the body, where the wheels are relative to each other, and where our other sensors sit.

Current position (odometry) is provided by most of the on-board sensors. Cameras provide one estimate; the accelerometer provides another; GPS provide a third. ROS will combine the measurements to estimate the robot's current position.

Our current goal is to combine odometry with the point clouds produced by the camera to create a map of our surroundings that can be fed into the navigation stack. Our unusual sensors are the largest problem facing us. Navigation stacks are usually written to use laser scans, rather than pointclouds. We need to find one that'll take a pointcloud instead. rgbd_slam, octomap_server, and viso2_ros all look promising.

Moving forward, we need to build a navigation stack. To do so, we'll need to calibrate the camera's pointclouds, generate an environment map, create a robot pose description, interface all sensors with ROS, and feed the whole lot of data into a navigation stack.