## Optical Odometry Engineering Notes

Kenneth Laws

02/12/2018

Preparing for second drive experiment:

Want to add a centimeter resolution acoustic rangefinder and implement better GPS

Need a method to record serial data from the rangefinder, planning to use python.

Spec for rangefinder

height for first drive test ~ 25 cm

My best guess for mounting height by looking at vehicles is 30 to 60 cm, (1 ft to 2 ft)

3/12/2018 Able to parse a Novatel span cpt file, bestxyza, bestvela

From closer look at the way Novatel reports velocity and INS attitude, it seems that it will be better to log longitude, lattitude and altitude and to do all computations in a locally flat coordinate system with z parallel to the perpendicular (gravity vector)

3/19/2018 Adding Aquity AR700RP-8 laser rangefinder

Initial testing:

AR700 clamped to a cart apprx 9.25” above the floor

Connected through serial to usb cable, 9600 baud

On power up, laser spools data, reading is 1.235”. This must be height above 8”. This sensor is a -8 model, 8” span with minimum distance = 8”

Test this by placing spacer blocks under the beam: 8” is minimum reading = 0.

The sensor I ordered is a -16 model with a 16” span and 21” offset

Minimum measurement distance = standoff – span / 2 = 21 – 8 = 13”

Range = 13” to 29”

Note: it is now apparent that the laser beam will likely be within the footprint of the translation camera image area.

3/20/18 Developing python script to set rangefinder parameters and read data.

Got something working, able to read the serial port at 9600 baud

change baud rate to 115200 command B9

save parameter change command w1234

set sample rate to 300 Hz or roughly 2 x the camera sample rate

S = 200,000 / F = 200,000 / 300 = 667

adapt python script to sample data for a finite interval

Adapt python script to control sample rate of sensor

3/22/18

goal is to generate a python script that can start all data collection

make a c program to collect camera data for 5 seconds

improve c program to store data into 10 second folders - done

put the unix time number in the file name – done

establish ability to run c program from python script - done

make a python script to collect span data for 5 seconds

make a master python script to start the other scripts

3/26/2018

Using udev rules

Not able so far to get this rule to work:

in /etc/udev/rules.d

created file

99-usb-serial.rules

SYSTEM=="tty", ATTRS{idVendor}=="0403", ATTRS{idProduct}=="6015", SYMLINK+="ttyAR700"

The above rule has a typo, should be

SUBSYSTEM=="tty", ATTRS{idVendor}=="0403", ATTRS{idProduct}=="6015", SYMLINK+="ttyAR700"

Then get the system to reread the rules,

sudo udevadm control --reload-rules

and unplug/replug the device

Now the symlink is there

/dev/ttyAR700

Working on getting the span working using python: Not able to get it working using same commands as worked when using realterm. I am selecting the soft link “ttyGPS1” and using the port selection USB2 to get it to work.

Working with serial comms in python is really frustrating and full of issues. It appears that characters remain in a buffer

Problem solved, involving 2 changes, single quotes, \r\n instead of \n

self.\_deviceSerial.write('UNLOGALL USB1 TRUE\r\n')

Note that I do want to use full ascii return messages so that time is included in the message header, e.g., "LOG USB1 BESTPOSA ONTIME %0.1f\r\n" % period

Also note that it doesn't seem to make any difference whether single or double quotes are used.

Mostly working now, but to get what I want, I need to thread the data logging processes so they can run concurrently.

Processes are threaded, and run independently, but when running in parallel, serial port operation of second thread times out.