Engineering Journal - Lidar Calibration

Date: 07/29/2017

By: Kenneth Laws

Lidar testing software has been developed by Vishi. Repository is located at:

ssh://snc@source-americas.it.here.com/lidar-testing/git/lidar\_testing

link to the repository:

[https://emea01.safelinks.protection.outlook.com/?url=https%3A%2F%2Fsource-americas.it.here.com%2Fprojects%2F1782-lidar-testing&data=01%7C01%7C%7C9fa4e920823b48b9c1c308d4c55cbddf%7C6d4034cd72254f72b85391feaea64919%7C1&sdata=IQ6jlIxBsvLVBMFwVJocMgEdsSgLUf3yo8%2BV%2B7xSd6k%3D&reserved=0](https://emea01.safelinks.protection.outlook.com/?url=https%3A%2F%2Fsource-americas.it.here.com%2Fprojects%2F1782-lidar-testing&data=01|01||9fa4e920823b48b9c1c308d4c55cbddf|6d4034cd72254f72b85391feaea64919|1&sdata=IQ6jlIxBsvLVBMFwVJocMgEdsSgLUf3yo8%2BV%2B7xSd6k%3D&reserved=0)

make a new folder for repository and use git shell:

git clone https://klaws@source-americas.it.here.com/lidar-testing/git/lidar\_testing

to create a new repository.

Make a new branch so any changes I might make can be merged or controlled later.

Lidar unit that we use is the Velodyne HDL-32E.

Web site has product information available – downloading everything useful to local store

Downloading demo software

Installed demo software “veloView-3.1.1” – runs OK and looks perfect for verifying data flow from a unit.

Obtained an interface box from Cyrus’s desk. It has a cut off cable to a terminal header. The header is labeled. Obtained a lidar unit from the “waiting for test” cage. Obtained a mating connector with 1” pigtail from stock. Now need only wiring diagram to be able to connect pigtail to interface box.

Velodyne web site has the following instructions for using veloView with the lidar:

For “sensor streaming” (live display of sensor data) it is important to change the network settings of the Ethernet adapter connected to the sensor from automatic IP address to manual IP address selection and choose:

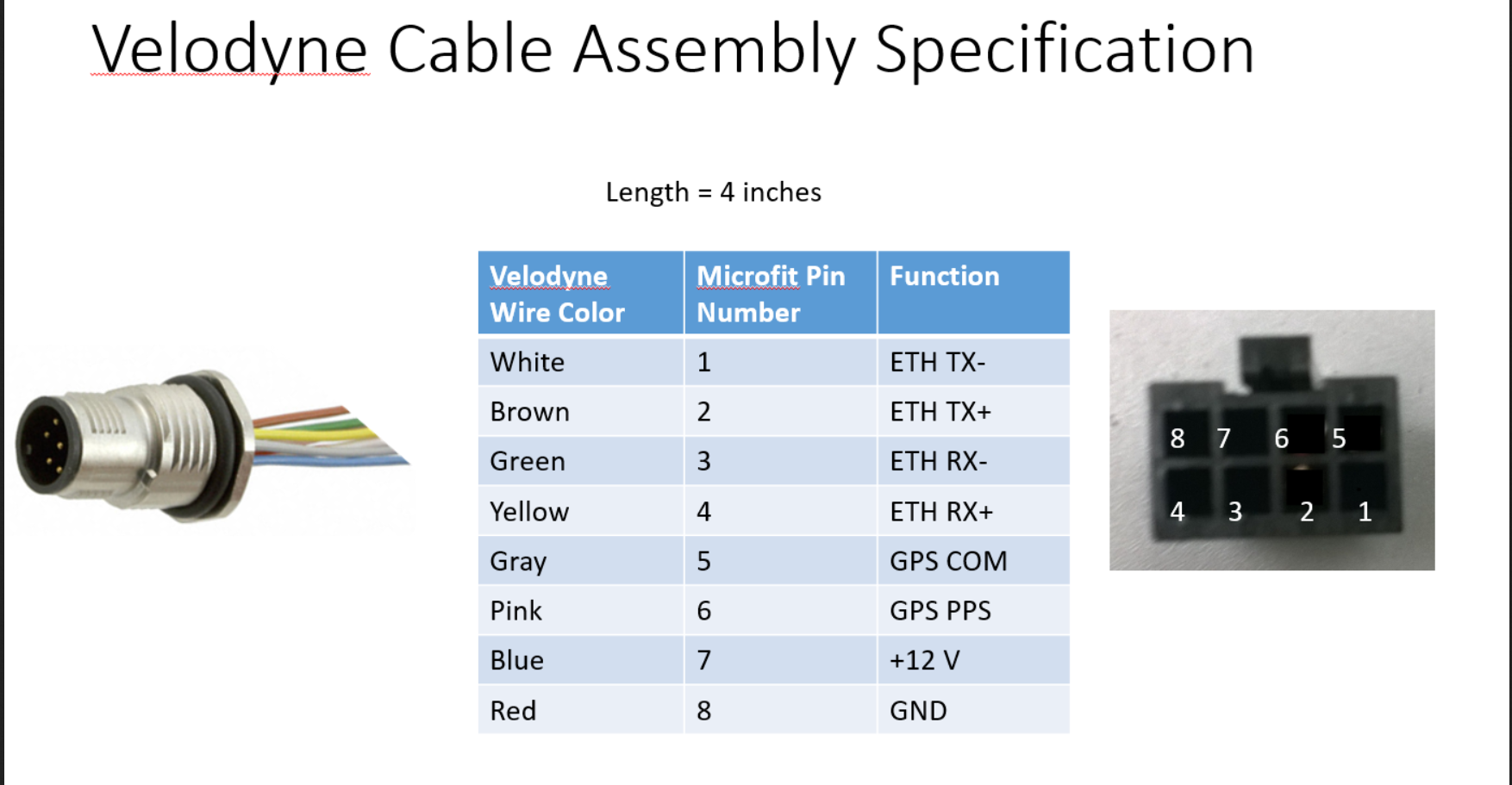
* HDL-32E and VLP-16
  + IP address: 192.168.1.70 (70 as example, any number except 201 works)
  + Gateway: 255.255.255.0
* HDL-64E
  + IP address: 192.168.3.70 (70 as example, any number except 43 works)
  + Gateway: 192.168.3.255

In order for sensor streaming to work properly, it is important to disable firewall restrictions for the Ethernet port. Disable the firewall completely for the ethernet device connected to the sensor or explicitly allow data from that Ethernet port of (including both public and private networks).

When opening pre-recorded data or live sensor streaming data one is prompted to choose a calibration file.

* For HDL-32E or VLP-16 data no calibration file is needed (the calibration values are already incorporated in VeloView) therefore select HDL-32 or VLP-16.
* For HDL-64E data the correct calibration file for that sensor needs to be chosen. The calibration file can be found on the individual product CD that was send with the HDL-64E sensor.

Have connected a sensor using interface box and wiring diagram



And a 12 V power supply (3.3A)

Testing Veloview software:

Can’t get data to stream.

Tried allowing velowview through firewall – didn’t work

Tried to turn firewall off – can’t do that, no privileges

9/6/2017 Tried again with same settings and am able to see the spinning lidar. Will try and record a bit of data for comparison.

The problem was that I had connected rx- to the wrong location on the terminal strip – lesson learned, check for your own mistakes first.

Works fine now and am able to view live data stream and record data. Recorded one data file as a sample.

Next up – look at calibration code, compile and run to collect a data file.

10/17/2017

Reestablish that I can use velodyne software to communicate with lidar.

Plug in radar using interface box and modified cable

Change IP address to static IP

* + IP address: 192.168.1.70 (70 as example, any number except 201 works)
  + Subnet: 255.255.255.0

Lidar is connected and streaming data.

Looking at lidar calibration code:

Ping vishi for some guidance

Look over programs in code base

There is a python script that looks like it only pings the lidar for basic information

My PC does not seem to have python installed, install python

Run python by typing

>py

The py script needs a plugin: urllib2

This is an old library not fully supported by python 3.6, looks like the line in the script should be changed to

import urllib

Next problem:

import rosparam

caused error

10/23/2017

Trying to get test\_lidar to compile in windows

Vishi has used <> form for local header files, “./” not in the system path.

Adding the local path to the system build path.

Worked but new error is that boost tools are required

Downloaded the boost package

Installation instructions are vague, not able to quickly install

Switching to linux to make project

Gsl package not found. Searching for gsl package.

Installed the gsl package and added the path to the makefile

Now cannot find the boost libraies

Installing boost using sudo apt-get install libboost-all-dev

Trouble using the gsl stuff, ‘pkg-config’ cannot find the file needed to provide path information “gsl.pc”.

Found this file manually using “find /usr/local -name gsl.pc”

Added the path for this file to pkg-config search path using

“export PKG\_CONFIG\_PATH=/usr/local/include/gsl/lib/pkgconfig/”

Then put the above line in the make file since this fix is not retained after reboot.

That error went away but now there is a new error

“/usr/bin/ld: VelodyneMonitor.o: undefined reference to symbol 'pthread\_create@@GLIBC\_2.2.5'

//lib/x86\_64-linux-gnu/libpthread.so.0: error adding symbols: DSO missing from command line

collect2: error: ld returned 1 exit status

Makefile:41: recipe for target 'test\_lidar' failed”

Fix for this problem was to add “-lpthread” to the LFLAGS (linker flags) in the makefile.

The code now compiles.

The code still doesn’t run

updating the repo to add Vishi’s latest additions

created a new branch ‘warehouse’ for warehouse development

adding gsl2.4 to the repository

compiling gsl from that folder – appears to compile OK

Not sure if all this was necessary. I still had an error about not being able to find a shared object file from the gsl library.

Problem was fixed by setting an environment variable:

export LD\_LIBRARY\_PATH=/home/earthmine/KEL/Projects/Lidar/lidar\_testing/gsl-2.4/.libs

the above path was determined by manually finding the file that test\_lidar was looking for.

10/25/2017 Testing with lidar.

* Set IP in test\_lidar.cpp, 192.168.1.201.
* Set IP on computer to manual, 192.168.1.70,
* set Subnet: 255.255.255.0

tested connection to lidar with ping – OK

errors:

get\_lidar\_info.py no module named rosparam

In get\_lidar\_info.py, comment out import rosparam – this seems to have worked (this module was not actually used in th py script

Have collected reference data set (apparently), now trying to test the same lidar against the reference set.

Program is taking some time to run. There is no feedback to tell if it is working properly.

It was not running, I had just hit return key and the program was still waiting for any key?

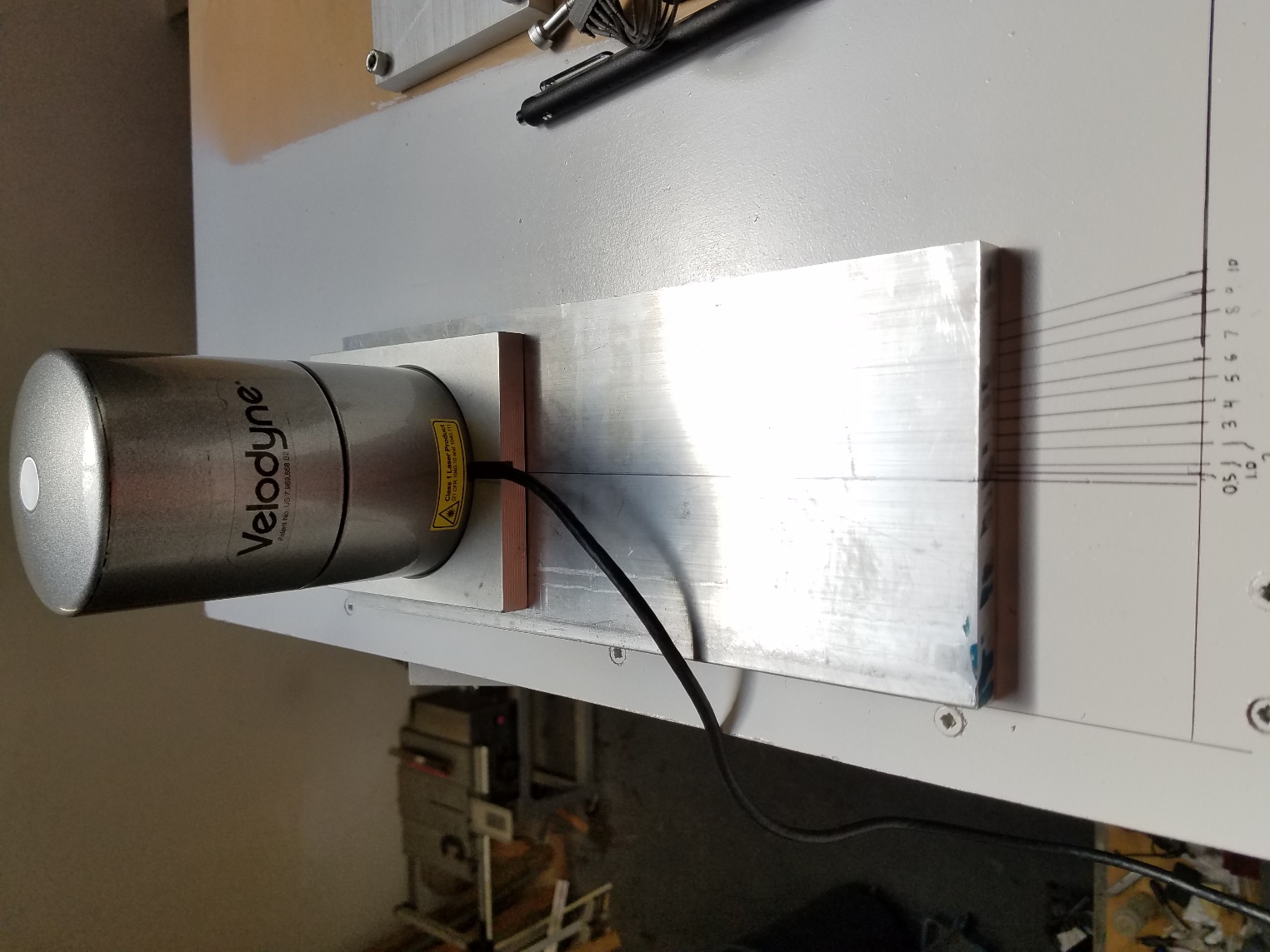
Running again

# Code compiles and runs

Vishi resolved the problems. Major issue was compatibility between different ubuntu versions, a typo in Vishi’s code, path problems and a problem with the installation of the GSL package library.

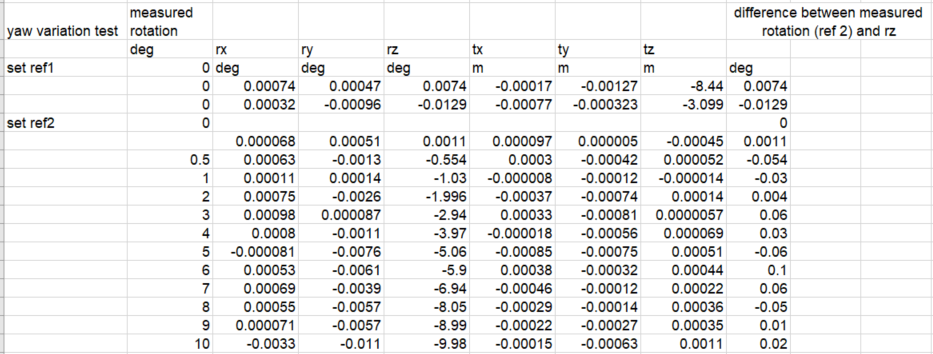
Testing the software

Using the test fixture outside the warehouse second floor office window. The fixture was modified by fabricating a plate for yaw adjustment with a pivot point at the lidar center. The plate is long to improve angle accuracy. A registration mark was placed on the side of the plate and lines were carefully drawn on the fixture at 0.5 degrees and 1.0 – 10.0 degrees in 1 degree increments.



Jig for measuring yaw rotation

A reference image was collected at zero degrees rotation. Subsequent images were collected over the range of angles. These results are collected below.



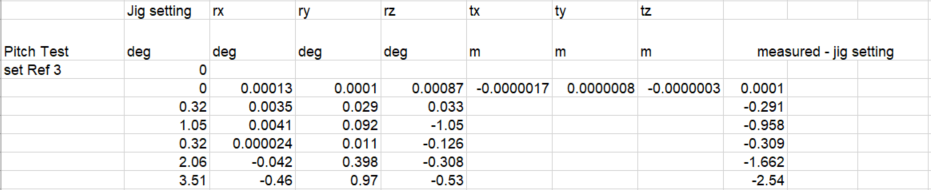
Yaw rotation is apparently defined as clockwise positive from above. Accounting for the sign error in the table, angle measurements agree well, mean err = 0.008 deg, std = .05 deg.



Jig for setting rotation about the y axis of the radar. Pitch angle was introduced using shims of different thicknesses and calculated given the known length of the jig mounting plate.

Length of the mounting plate was 35 cm. Blocks of different heights were placed under the end nearest the front of the photo.

Results for measuring y rotation are collected below.

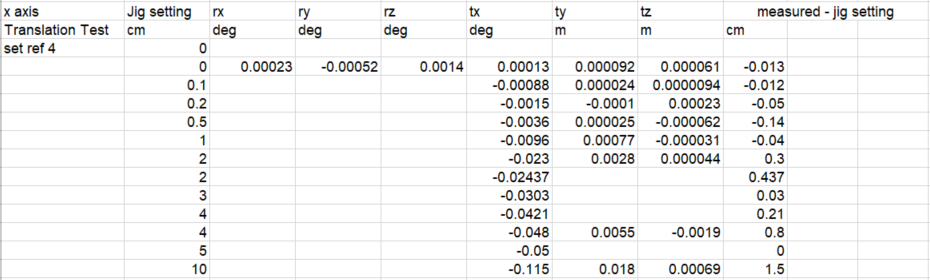


There is little apparent correlation between the pitch setting and the pitch measurement (y axis)

Roll (rotations about x axis) were not measured.

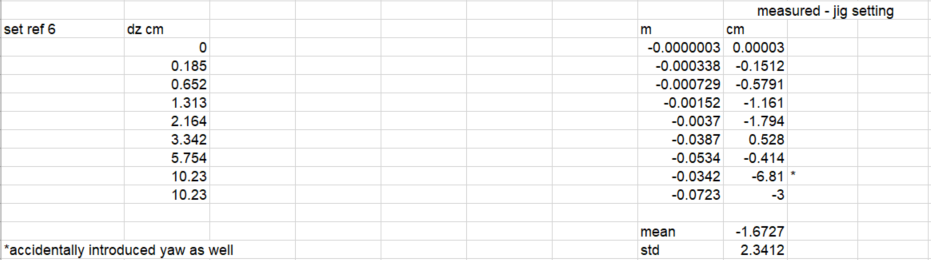


Translations in x were measured by placing a registration mark on the lidar mounting plate and clamping a piece of bar stock with the same thickness and also with registration mark to the fixture. Shifts were introduced by sliding the mounting plate along the bar stock and measuring the distance between registration marks.



Measurements of translations in x had a mean error of 0.28 cm and a std of 0.51 cm.

Translations in z axis were introduced by inserting blocks of known height under the lidar mounting plate. Results below.



Measurement of z axis translations have higher error than x axis results. More data would be good here as results are not consistent enough to draw conclusions.

Conclusions

Yaw (z-axis) rotations are accurately measured, mean err = 0.008 deg, std = .05 deg. Pitch axis measurements did not show conclusive correlation with pitch settings over the range investigated. X,y translations are accurately measured and had a mean error of 0.28 cm and a std of 0.51 cm. Translations in z axis were inconclusive with apparent correlation but higher uncertainty and error. More measurements of z axis translations would be needed to better characterize the results.

11/3/2017

Looking into adding intensity evaluation.

Created new branch for intensity development, tested compile and run – OK

12/5/2017

Lidar intensity testing:

setting up ethernet interface

address: 192.168.1.70

netmask: 255.255.255.0

gateway: blank

comms are good, collecting reference scans for:

# description

1 my test lidar

2 unknown

3 unknown

4 unknown

5 unknown

6 unknown

R1 red lidar 1

R2 red lidar 2

S1-3 standard good lidar (3 repetitions)

Getting the c-program to compile on a new laptop

Compilation fails – no GSL library found

Installing GSL from the included package directory as per instructions in INSTALL

This library is gsl-1.16

chmod u+x configure

sudo ./configure

makefile

This compiled OK but the lidar program still does not compile

Add environment varialbe

PKG\_CONFIG\_PATH='./gsl-1.16/'

seemed to help but still doesn’t compile

cd gsl-1.16/

make

make clean

make

sudo make install

sudo apt-get install libboost-dev

sudo apt-get install libboost-program-options-dev

sudo apt-get install libboost-filesystem-dev

sudo apt-get install libboost-thread-dev

chmod u+x test\_lidar

chmod u+x get\_lidar\_info.py

3/6/2018 Setting up a new repo that I will take ownership of, for the lidar code and scripts