**How to Use RTKLIB and NTRIP on Windows or Ubuntu**

Created by: Qiu

Created on: 2021.08.20

**Software Environments**

Windows 10

Ubuntu 20.04.3 LTS

ROS Noetic Ninjemys

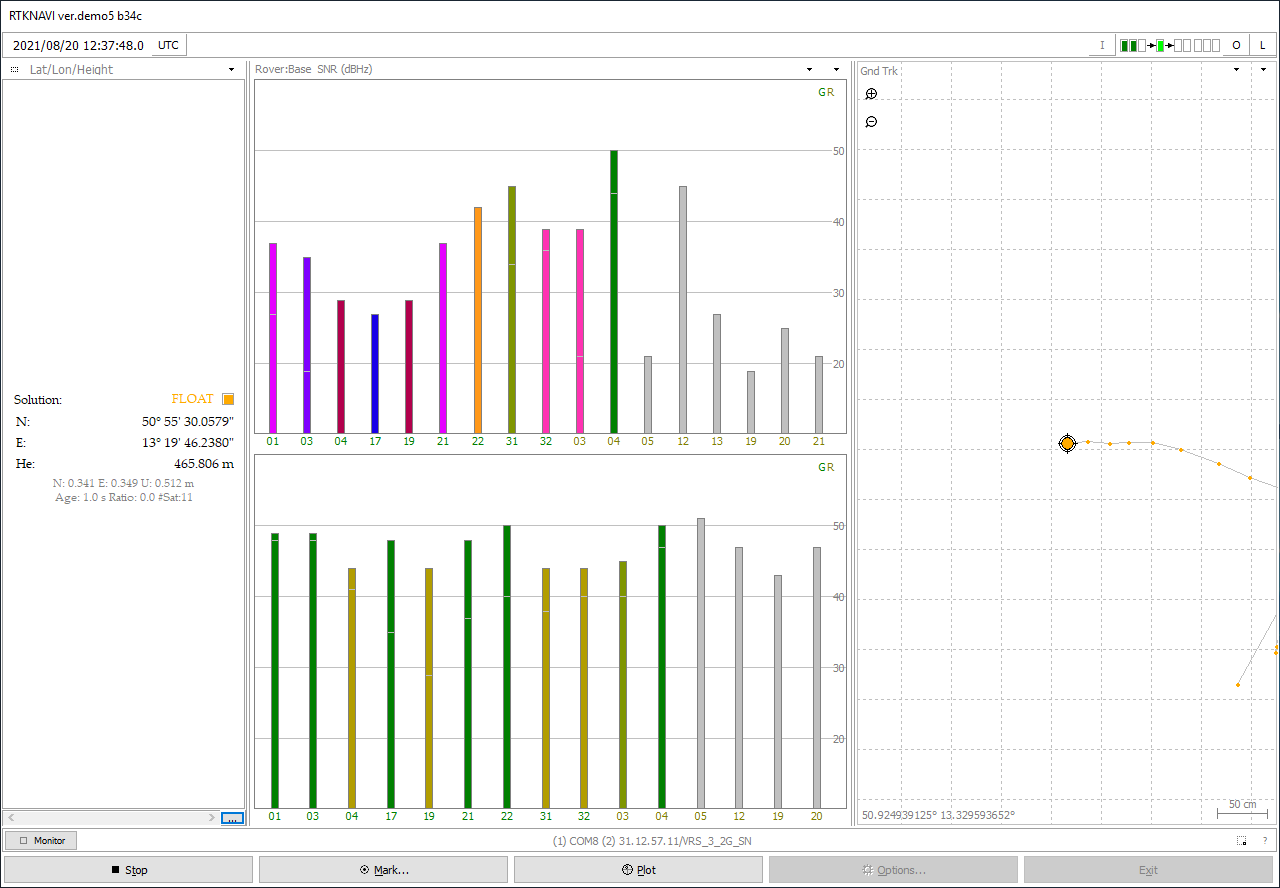
RTKLIB ver. demo5 b34c for Windows [1]

RTKLIB v2.4.2 p13 for ROS on Ubuntu (integrated in rtkrcv\_ros with adjustments [2] [7])

**Hardware Environments**

simpleRTK2B based on u-blox ZED F9P using the hardware configuration file “rover\_config\_for\_rtklib\_and\_ntrip.txt” [[1]](#footnote-1). Do not forget to upload the configuration file to the chip.

**Windows**



Data Acquisition from the u-blox ZED F9P using RTKLIB with SAPOS Sachsen NTRIP Service

**Ubuntu**

Prerequisites

Before starting the program “rtkrcv”, make sure that the read/write access to the USB port of the GNSS sensor has been permitted.

sudo chmod 666 /dev/ttyACM0

The execution permission of two files has to be activated by running commands below:

chmod +x rtkstart.sh

chmod +x rtkshut.sh

There is a very important configuration file that specifies the rtkrcv server. The file is stored under:

./app/consapp/rtkrcv/gcc

*Using a Single Rover without Correctional Data*

Step 1: Specify the configuration file “rtkrcv.conf”

Following changes shall be applied to the file “rtkrcv.conf”:

inpstr2-type =off # switch off the input stream 2

inpstr3-type =off # switch off the input stream 3

inpstr1-path =**ttyACM0**:115200:8:n:1:off # pay attention to the name of the usb port

inpstr1-format =ubx # specify the format of the gnss sensor messages to ubx

pos1-frequency =l1+l2 # u-blox zed f9p is a dual-frequency gnss sensor

pos1-navsys =13 # use GPS, GLONASS, GALILEO

pos1-sateph =brdc # specify satellite ephemeris/clock to broadcast

Additionally, replace *pos1-snrmask =0* with the following codes [5]:

pos1-snrmask\_r =off

pos1-snrmask\_b =off

pos1-snrmask\_L1 =0,0,0,0,0,0,0,0,0

Step 2: Start the rtkrcv server using terminal

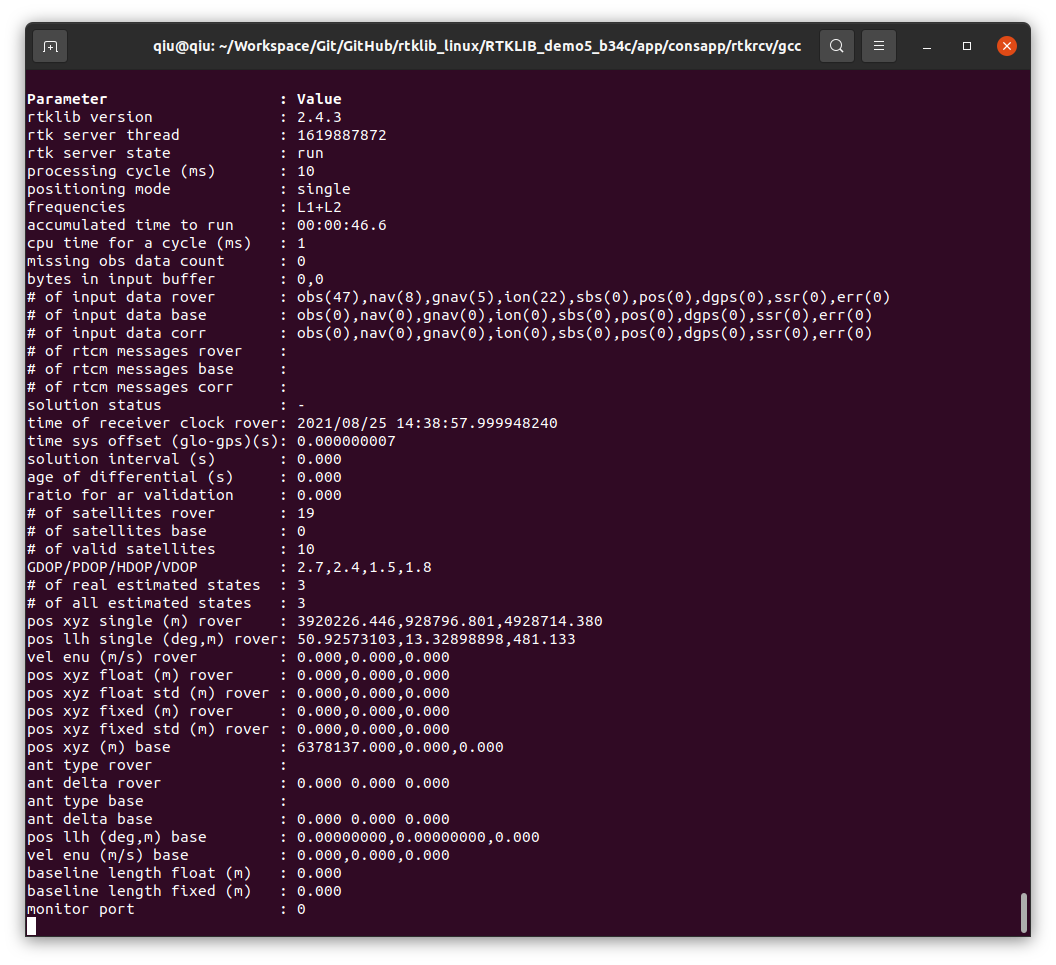
Change directory to ./app/consapp/rtkrcv/gcc, open a terminal in this folder, and input commands below sequentially [5]:

rtkrcv -o rtkrcv.conf # in case that the config file is in the same folder as the rtkrcv server

start

status 1 # refresh the window every second

After that, the terminal will be continuously refreshing the results as shown below. Please check the line “pos llh single (deg,m) rover” for positioning results.



*Using a Single Rover with Correctional Data*

Step 1: Specify the configuration file “rtkrcv.conf”

Apply settings described in Step 1 of Section “Using a Single Rover without Correctional Data”.

Then, do additional modifications as follow (s. [5] and [7] for further details):

inpstr2-type =ntripcli # use ntrip service

inpstr2-path =SBX7000A:iet2021@31.12.57.11:2101/VRS\_3\_4G\_SN # SAPOS Sachsen VRS\_3\_4G\_SN

inpstr2-format =rtcm3 # RTCM3 protocol

inpstr2-nmeareq =single # enable the rtkrcv server to send the rover position to the reference station

ant2-postype =rtcm # enable to receive the antenna position of the reference station with corr. data

pos2-armode =continuous # enable to estimate and solve integer ambiguities continuously [8]

The complete configuration file (*rtkrcv\_float\_fix.conf*) can be found in Appendix A.

Step 2: Start the rtkrcv server using terminal

Change directory to the root of the ROS workspace “ws\_gnss\_f9p”, open the terminal in this folder, input the following commands subsequently:

catkin\_make # compile source files

source ./devel/setup.bash # source the file. This step has to be repeated on new terminal tab.

roscore # start the ROS

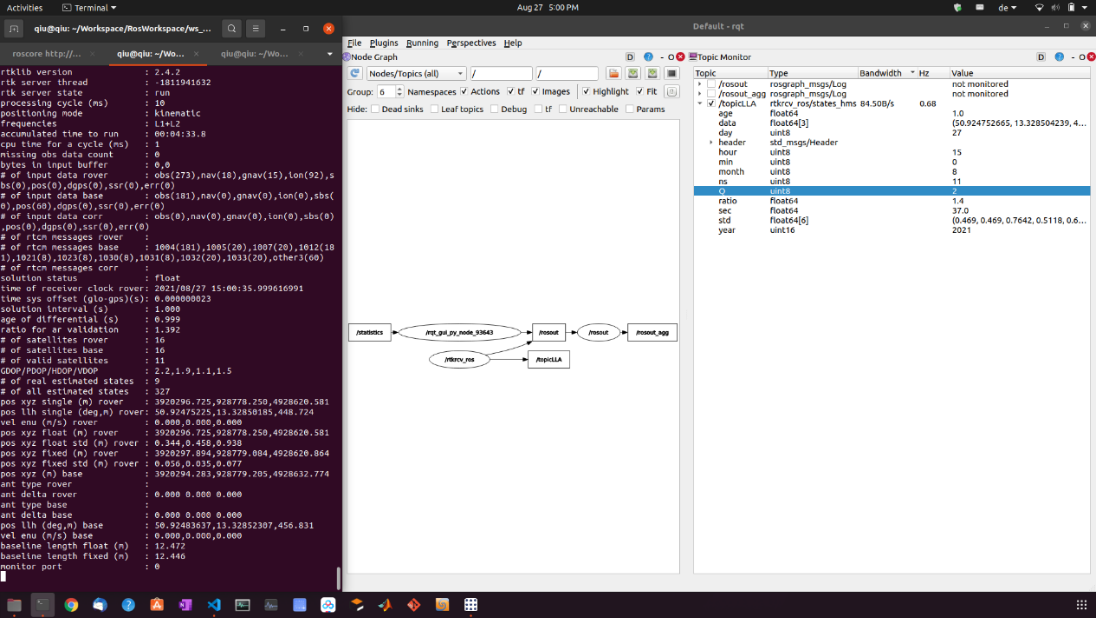
rosrun rtkrcv\_ros rtkrcv\_ros\_node # run a node named “rtkrcv\_ros\_node” from the package “rtkrcv”

load /home/<username>/rtkrcv.conf # load the configuration file [7], please adjust the path

restart # restart the rtkrcv server [7]

status 1 # refresh the window every second

rqt # show ROS node graph



The picture above illustrates positioning results of an experiment performed on Aug. 27th, 2021. The experiment was running in ROS using RTKLIB and NTRIP. The signal quality of “float” has been achieved.

The project of the ROS workspace can be found in Appendix A.

1. **ROS workspace for u-blox ZED F9P sensor with RTKLIB**

On Windows

[\\zfs1.hrz.tu-freiberg.de\etechnik\P\_SteigtUM\200\_Mobile\230\_A3 Autonomes Parken u. Folgen\Repositories\ws\_gnss\_f9p](../../../../../..//zfs1.hrz.tu-freiberg.de/etechnik/P_SteigtUM/200_Mobile/230_A3%20Autonomes%20Parken%20u.%20Folgen/Repositories/ws_gnss_f9p)

On Ubuntu

<smb://zfs1.hrz.tu-freiberg.de/etechnik/P_SteigtUM/200_Mobile/230_A3%20Autonomes%20Parken%20u.%20Folgen/Repositories/ws_gnss_f9p>

**References**

1. RTKLIB Demo5 b34c: <http://rtkexplorer.com/downloads/rtklib-code/>
2. rtkrcv\_ros: <https://github.com/ajbfinesc/rtkrcv_ros>
3. Dual-frequency PPK solutions with RTKLIB and the u-blox F9P: <https://rtklibexplorer.wordpress.com/2019/08/24/dual-frequency-ppk-solutions-with-rtklib-and-the-u-blox-f9p/>
4. DGPS mit RTKLIB: <https://www.dirkkoller.de/dgps-mit-rtklib>
5. RTKLIB auf dem Raspberry Pi: <https://www.dirkkoller.de/rtklib-auf-raspberrypi>
6. ZED-F9P Base and Rover Configuration: <https://www.youtube.com/watch?v=FpkUXmM7mrc&ab_channel=RoboRoby>
7. Ferreira, A. et al.: Real-time GNSS precise positioning: RTKLIB for ROS. International Journal of Advanced Robotic Systems. 2020. [DOI: 10.1177/1729881420904526](https://journals.sagepub.com/doi/10.1177/1729881420904526).
8. rtklib User Manual v2.4.2 p13 (available under the installation folder)
9. rtklib User Manual demo5 (available under the installation folder)
10. ArduSimple Configuration Files: <https://www.ardusimple.com/configuration-files/>

1. The hardware configuration file can be found in Appendix A. Refer to [10] for how to upload a configuration file to the chip. [↑](#footnote-ref-1)