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

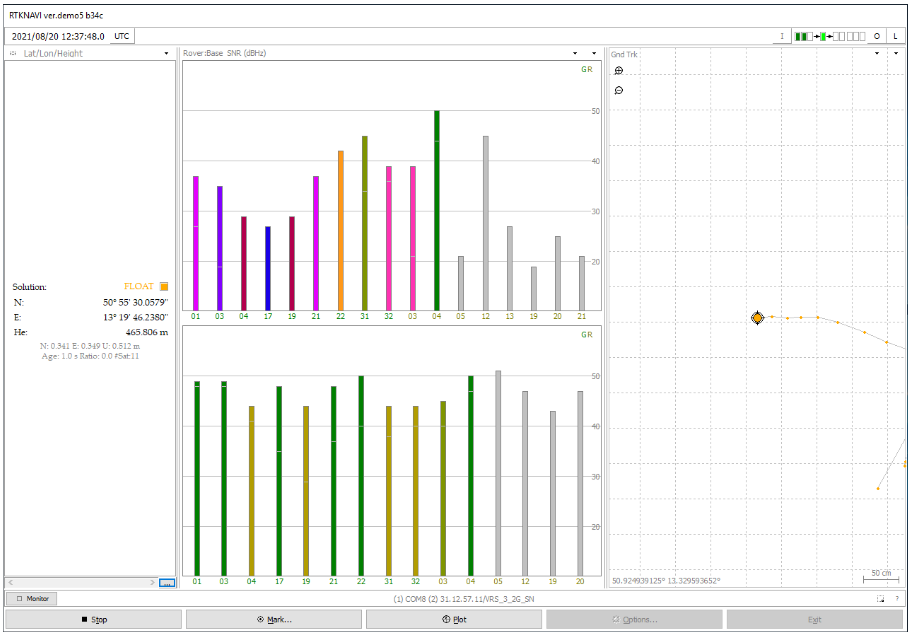
RTKLIB ver. demo5 b34c for Windows [1]

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]. Do not forget to upload the configuration file to the chip.

Windows



RTKNAVI ver.demo5 b34c

2021/08/20 12:37:48.0 UTC

Solution: FLOAT

N: 50° 55' 30.0579"

E: 13° 19' 46.2380"

He: 465.806 m

N: 0.341 E: 0.349 U: 0.512 m
Age: 1.0 s Ratio: 0.0 #Sat:11

50.924939125° 13.329593652°

(1) COM8 (2) 31.12.57.11/VRS_3_2G_SN

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

[1] The hardware configuration file can be found in Appendix A. Refer to [10] for how to upload a configuration file to the chip.

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

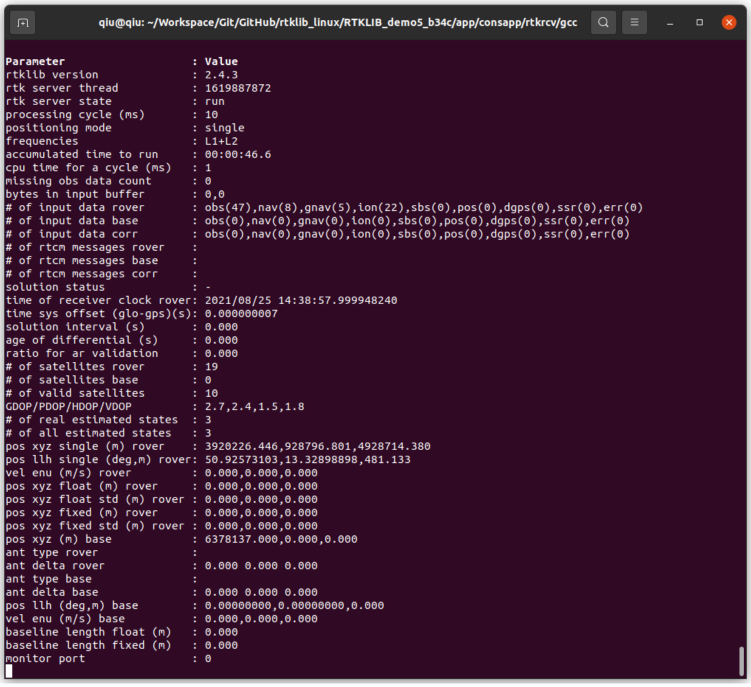
Change directory to ./app/consapp/rtkrcv/gcc, open a terminal in this folder, and input commands below sequentially [5]:

```
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.


qiu@qiu: ~/Workspace/Git/GitHub/rtklib_linux/RTKLIB_demo5_b34c/app/consapp/rtkrcv/gcc

```
Parameter                    : Value
rtklib version               : 2.4.3
rtk server thread            : 1619887872
rtk server state             : run
processing cycle (ms)        : 10
positioning mode             : single
frequencies                  : L1+L2
accumulated time to run      : 00:00:46.6
cpu time for a cycle (ms)    : 1
missing obs data count       : 0
bytes in input buffer        : 0,0
# of input data rover        : obs(47),nav(8),gnav(5),ion(22),sbs(0),pos(0),dgps(0),ssr(0),err(0)
# of input data base         : obs(0),nav(0),gnav(0),ion(0),sbs(0),pos(0),dgps(0),ssr(0),err(0)
# of input data corr         : obs(0),nav(0),gnav(0),ion(0),sbs(0),pos(0),dgps(0),ssr(0),err(0)
# of rtcm messages rover     :
# of rtcm messages base      :
# of rtcm messages corr      :
solution status              : -
time of receiver clock rover: 2021/08/25 14:38:57.999948240
time sys offset (glo-gps)(s): 0.000000007
solution interval (s)        : 0.000
age of differential (s)      : 0.000
ratio for ar validation      : 0.000
# of satellites rover        : 19
# of satellites base         : 0
# of valid satellites        : 10
GDOP/PDOP/HDOP/VDOP          : 2.7,2.4,1.5,1.8
# of real estimated states   : 3
# of all estimated states    : 3
pos xyz single (m) rover     : 3920226.446,928796.801,4928714.380
pos llh single (deg,m) rover: 50.92573103,13.32898898,481.133
vel enu (m/s) rover          : 0.000,0.000,0.000
pos xyz float (m) rover      : 0.000,0.000,0.000
pos xyz float std (m) rover : 0.000,0.000,0.000
pos xyz fixed (m) rover      : 0.000,0.000,0.000
pos xyz fixed std (m) rover : 0.000,0.000,0.000
pos xyz (m) base             : 6378137.000,0.000,0.000
ant type rover               :
ant delta rover              : 0.000 0.000 0.000
ant type base                :
ant delta base               : 0.000 0.000 0.000
pos llh (deg,m) base         : 0.00000000,0.00000000,0.000
vel enu (m/s) base           : 0.000,0.000,0.000
baseline length float (m)    : 0.000
baseline length fixed (m)    : 0.000
monitor port                 : 0
```

__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

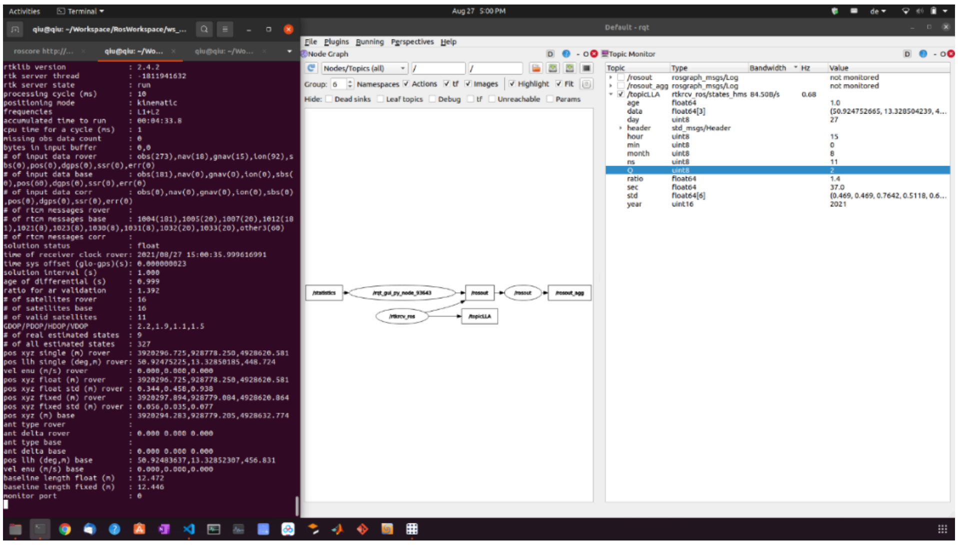
rosrun rtkrcv_ros rtkrcv_ros_node # run a node named “rtkrcv_ros_node” from the package “rtkrcv”

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



```
rtklib version              : 2.4.2
rtk server thread           : -1811941632
rtk server state            : run
processing cycle (ms)       : 10
positioning mode            : kinematic
frequencies                 : L1+L2
accumulated time to run     : 00:04:33.8
cpu time for a cycle (ms)   : 1
missing obs data count      : 0
bytes in input buffer       : 0,0
# of input data rover       : obs(273),nav(18),gnav(15),ion(92),s
bs(0),pos(0),dgps(0),ssr(0),err(0)
# of input data base        : obs(181),nav(0),gnav(0),ion(0),sbs(
0),pos(60),dgps(0),ssr(0),err(0)
# of input data corr        : obs(0),nav(0),gnav(0),ion(0),sbs(0)
,pos(0),dgps(0),ssr(0),err(0)
# of rtcm messages rover    :
# of rtcm messages base     : 1004(181),1005(20),1007(20),1012(18
1),1021(8),1023(8),1030(8),1031(8),1032(20),1033(20),other3(60)
# of rtcm messages corr     :
solution status             : float
time of receiver clock rover: 2021/08/27 15:00:35.999616991
time sys offset (glo-gps)(s): 0.000000023
solution interval (s)       : 1.000
age of differential (s)     : 0.999
ratio for ar validation     : 1.392
# of satellites rover       : 16
# of satellites base        : 16
# of valid satellites       : 11
GDOP/PDOP/HDOP/VDOP         : 2.2,1.9,1.1,1.5
# of real estimated states  : 9
# of all estimated states   : 327
pos xyz single (m) rover    : 3920296.725,928778.250,4928620.581
pos llh single (deg,m) rover: 50.92475225,13.32850185,448.724
vel enu (m/s) rover         : 0.000,0.000,0.000
pos xyz float (m) rover     : 3920296.725,928778.250,4928620.581
pos xyz float std (m) rover : 0.344,0.458,0.938
pos xyz fixed (m) rover     : 3920297.894,928779.084,4928620.864
pos xyz fixed std (m) rover : 0.056,0.035,0.077
pos xyz (m) base            : 3920294.283,928779.205,4928632.774
ant type rover              :
ant delta rover             : 0.000 0.000 0.000
ant type base               :
ant delta base              : 0.000 0.000 0.000
pos llh (deg,m) base        : 50.92483637,13.32852307,456.831
vel enu (m/s) base          : 0.000,0.000,0.000
baseline length float (m)   : 12.472
baseline length fixed (m)   : 12.446
monitor port                : 0
```

Topic	Type	Bandwidth	Hz	Value
☐ /rosout	rosgraph_msgs/Log			not monitored
☐ /rosout_agg	rosgraph_msgs/Log			not monitored
☑ /topicLLA	rtkrcv_ros/states_hms	84.50B/s	0.68	
age	float64			1.0
data	float64[3]			(50.924752665, 13.328504239, 4...
day	uint8			27
header	std_msgs/Header			
hour	uint8			15
min	uint8			0
month	uint8			8
ns	uint8			11
Q	uint8			2
ratio	float64			1.4
sec	float64			37.0
std	float64[6]			(0.469, 0.469, 0.7642, 0.5118, 0.6...
year	uint16			2021

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.

Appendix A 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

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.

[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/