STATE OF CALIFORNIA • DEPARTMENT OF TRANSPORTATION

TECHNICAL REPORT DOCUMENTATION PAGE

TR0003 (REV 10/98)

ADA Notice

For individuals with sensory disabilities, this document is available in alternate formats. For information call (916) 654-6410 or TDD (916) 654-3880 or write Records and Forms Management, 1120 N Street, MS-89, Sacramento, CA 95814.

1. REPORT NUMBER	2. GOVERNMENT ASSOCIATION NUMBER	3. RECIPIENT'S CATALOG NUMBER
CA21-3746		

4. TITLE AND SUBTITLE	5. REPORT DATE
Network Differential GNSS Corrections for Connected and Autonomous Vehicles	01/20/2021
	6. PERFORMING ORGANIZATION CODE

7. AUTHOR(S)	8. PERFORMING ORGANIZATION REPORT NO.
Dr. Jay Farrell and Wang Hu	

9. PERFORMING ORGANIZATION NAME AND ADDRESS	10. WORK UNIT NUMBER
University of California Riverside 900 University Ave. Riverside, CA 92521	11. CONTRACT OR GRANT NUMBER Contract 65A0767 Task ID 3746

12. SPONSORING AGENCY AND ADDRESS	13. TYPE OF REPORT AND PERIOD COVERED
Division of Research, Innovation and System Information California Department of Transportation 1727 30th Street, 3rd Floor, MS-83 Sacramento, CA 94273-0001	1/1/2020 to 12/31/2020 14. SPONSORING AGENCY CODE

15. SUPPLEMENTARY NOTES

16. ABSTRACT

This report documents the efforts conducted under contract 65A0767, task ID 3746. Under this project the contractor (University of California, Riverside) built, tested and evaluated the Networked Differential Global Navigation Satellite System (N-DGNSS) system. The positioning performance in the receivers that using N-DGNSS surpassed the SAE specification, while the receivers without external corrections failed the vertical positioning portion of the SAE specification. For a moving platform, both the N-DGNSS and Single Point Positioning (SPP) approaches surpass the SAE specifications in horizontal positioning accuracy. The SPP approach did not achieve the SAE 3-meter vertical accuracy specification while the N-DGNSS approach did. Receivers using N-DGNSS correctly determine the vehicle lane for distances up to 1.33 m from lane center (0.47 m from lane edge) and maintained 90% probability of success lane for distances up to 1.4 m from lane center (0.4 m from lane edge).

17. KEY WORDS	18. DISTRIBUTION STATEMENT
GPS, GNSS, High Accuracy GPS, Differential GPS, State Space Representation (SSA)	

19. SECURITY CLASSIFICATION *(of this report)*	20. NUMBER OF PAGES	21. COST OF REPORT CHARGED
Unclassified	20	

DISCLAIMER STATEMENT

This document is disseminated in the interest of information exchange. The contents of this report reflect the views of the authors who are responsible for the facts and accuracy of the data presented herein. The contents do not necessarily reflect the official views or policies of the State of California or the Federal Highway Administration. This publication does not constitute a standard, specification or regulation. This report does not constitute an endorsement by the Department of any product described herein.

For individuals with sensory disabilities, this document is available in alternate formats. For information, call (916) 654-8899, TTY 711, or write to California Department of Transportation, Division of Research, Innovation and System Information, MS-83, P.O. Box 942873, Sacramento, CA 94273-0001.

Final Report:

Network Differential GNSS Corrections for Connected and Autonomous Vehicles

Contract 65A0767

Caltrans FY 2020-2021

Project Period of Performance: 1/1/2020 – 12/30/2020

Authors: Wang Hu and Jay A. Farrell

University of California, Riverside

Executive Summary

This final report summarizes the successful completion of each of the tasks of this project.

The UCR VN-DGNSS server has been built, performed and evaluated. In the evaluation results from the third quarter, stationary and moving testing results for evaluating positioning accuracy were presented. The positioning performance in the receivers that using VN-DGNSS surpassed the SAE specification, while the receivers without external corrections failed the vertical positioning portion of the SAE specification. For a moving platform, both the VN-DGNSS and SPP approaches surpass the SAE specifications in horizontal positioning accuracy. The SPP approach did not achieve the SAE 3-meter vertical accuracy specification while the VN-DGNSS approach did.

The main tasks of the fourth quarter were collaborative performance evaluation of VN-DGNSS within an application and completion of this final report. After consultation with CE-CERT researchers, the lane-determination application was selected, with evaluation results presented herein. Receivers using VN-DGNSS correctly determine the vehicle lane for distances up to 1.33 m from lane center (0.47 m from lane edge) and maintained 90% probability of success lane for distances up to 1.4 m from lane center (0.4 m from lane edge). In comparison, the probability of success for receivers in SPP mode has dropped below 32% at 1.33 m from lane center.

Table of Contents

Project Final Report: Task Summary

Task 1

Task 1 was completed in the second quarter.

Task 2

This task is the implementation of a network DGNSS approach for CAV applications. The task started in March 2020 and ended in November 2020.

The implementation works in a client-server architecture, as depicted in Figure 1. The server accesses DGNSS state-space representation (SSR) correction information in real-time, from which it computes OSR corrections suitable for each client (i.e., user) location. The user connects their receiver to the server through a client program. The client (a) establishes communications with the user receiver through a receiver port; (b) receives an estimate of the receiver location $\mathbf{P}_b$; (c) enables differential operation on that receiver; (d) establishes communication with the server via ethernet TCP/IP, (e) communicates the virtual reference station location $\mathbf{P}_b$ for this client to the server; (f) relays correction information using the RTCM format from the server to the user receiver.

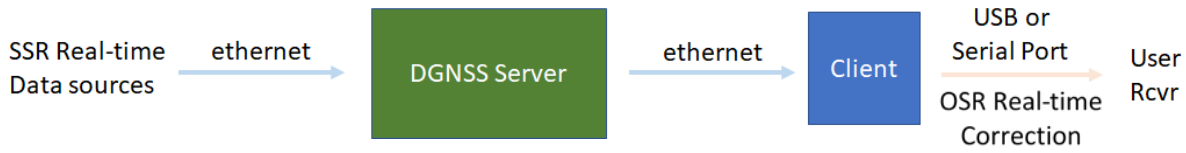


Figure 1. Client Server architecture

A version of the implementation is working, publicly available through GitHub, and being tested. This server and client programs are available publicly for Windows and Linux users at https://github.com/jaffarrell/WADGNSS. The public release includes its source code so that users can adapt it to their particular applications. Test results are discussed later in this report.

This task is competed for the scope of this project. It may be updated based on user feedback. Updates will be release through GitHub.

Task 3

The user manual is complete and has been publicly posted at https://github.com/jaffarrell/WADGNSS. The user manual is attached to this report as an appendix.

This task is competed for the scope of this project. It may be updated based on user feedback. Updates will be release through GitHub.

Task 4

This task evaluated the positioning performance using the UCR Virtual Network DGNSS server. Initial results were included in the third quarter. Those results are augmented herein with additional testing results in a lane recognition application performed collaboratively with research personnel from CE-CERT.

This task is competed for the scope of this project.

Task 5

This task concerns writing the final report, which is this document.

This task is competed for the scope of this project.

Evaluation Results from the Third Quarter

This section presents experimental results for stationary and moving platforms that were performed in the third quarter. They are included to have a comprehensive final report.

Results obtained from u-blox receivers connected to the UCR Virtual Network DGNSS (VN-DGNSS) server were compared with u-blox receivers operating in single point positioning (SPP) mode without external corrections. The positioning performance was analyzed relative to the Society of Automotive Engineering (SAE) specification which requires 68% of horizontal positioning error to be less than 1.5 m and 68% of vertical positioning error to be less than 3 m. We will additionally quantify the probability of achieving meter-level horizontal positioning accuracy (i.e., position error < 1m).

Position errors are computed in the North, East, Down frame (NED). The horizontal error is computed as the norm of north and east error. The vertical error is computed as the absolute value of down-direction error. The 3D error is also included.

Stationary Results

The hardware set up for the stationary test is shown in Fig 2. Four receivers were connected to one antenna. Therefore, they each have the same ground-truth location and measure the same GNSS signals.

- Two receivers were u-blox M8P Single-frequency receivers. One was connected to the UCR Virtual Network DGNSS (VN-DGNSS) server for receiving real-time corrections. The other was operating in single point positioning (SPP) mode for comparison.
- Two receivers were u-blox ZED-F9P Dual-frequency receivers. One was connected to the UCR Virtual Network DGNSS (VN-DGNSS) server for receiving real-time corrections. The other was operating in single point positioning (SPP) mode for comparison.

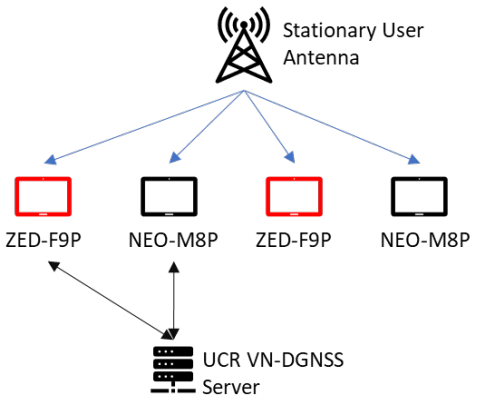


Figure 2: Sketch map for Stationary testing

In this test, the stationary user antenna is the antenna (at a known position) on the roof of the UCR Winston Chung Hall. All receivers reported their computed position every 1 second in UTC. The results are shown in Fig. 3 and 4 and summarized in Tables 1 and 2.

In Fig. 3 and Fig. 4, the u-blox accuracy with UCR VN-DGNSS server information is shown in red and blue. The accuracy in SPP mode (no corrections) is shown in yellow and cyan.

- Table 1 and Figs. 3 (a) and (b) show that the horizontal positioning accuracy increases slightly with the corrections. This is clearer in Table 1 than in the figure.
- Figs. 1 (c) and (e) show that using VN-DGNSS corrections yields significantly better performance in the vertical error. Therefore, Figs. 1 (d) and (f) show that VN-DGNSS improves the three-dimensional positioning performance by several meters.

The statistic for the stationary testing results are summarized in Table 1. The positioning performance in the receivers that using VN-DGNSS surpass the SAE specification, while the receivers without external corrections failed the vertical positioning portion of the SAE specification. With or without VN-DGNSS both approaches achieved meter-level horizontal accuracy in stationary test at a rate of about 50%.

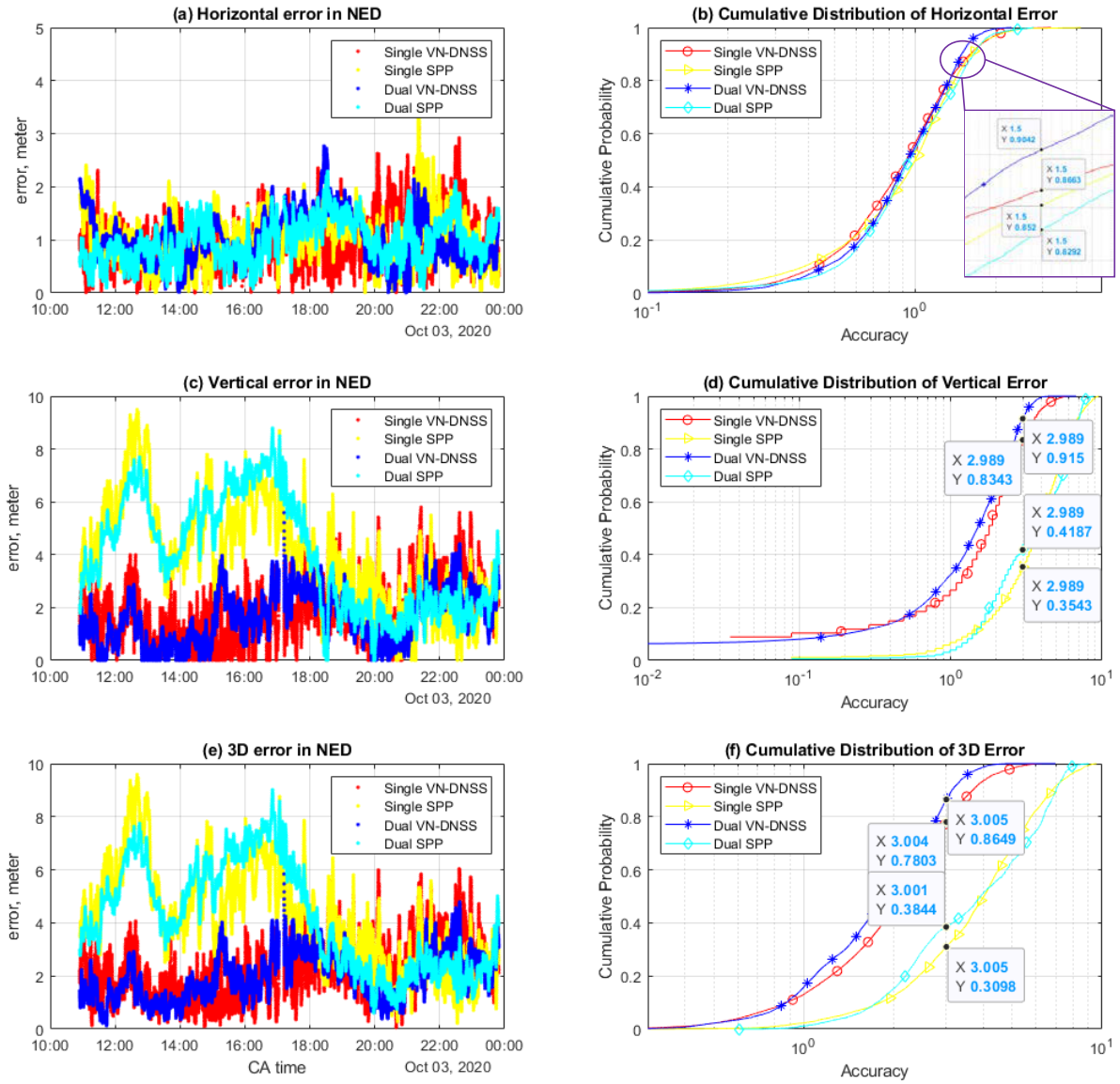


Figure 3: Subplots (a), (c), and (e) show the horizontal, vertical, and 3D position error plotted versus time for the stationary test. The x-axis is the California local time in 24-Hr format. Subplots (b), (d), and (f) show the cumulative distribution of the horizontal, vertical, and 3D position error for the stationary test. The red curve corresponding to the u-blox M8P in DGNSS mode using the RTCM messages sent by the UCR VN-DGNSS server. The yellow curve corresponding to the u-blox M8P in SPP mode. The blue curve corresponding to the u-blox ZED-F9P in DGNSS mode using the RTCM messages sent by the UCR VN-DGNSS server. The cyan curve corresponding to the u-blox ZED-F9P in SPP mode.

Correction Mode	u-blox type	Cumulative probability for stationary results			
		Horizontal error < 1.5m	Horizontal error < 1m	Vertical error < 3m	3D error < 3m
VN-DGNSS	M8P (single freq)	86.63%	56.45%	83.43%	78.03%
	F9P (Dual freq)	90.42%	55.34%	91.50%	86.49%
SPP	M8P (single freq)	85.20%	50.23%	35.43%	30.98%
	F9P (Dual freq)	82.92%	53.83%	41.87%	38.44%

Table 1: Cumulative probabilities of stationary test

Figs. 3 (c) and (e) show a distinct difference in performance between day (10 AM – 7 PM) and night (7 PM to midnight). To further investigate the performance difference between daytime and nighttime. Fig. 4 divides the cumulative distribution results for before and after sunset. As shown in Fig. 4 (a), (c), and (e), in the daytime, VN-DGNSS approach improves both horizontal and vertical accuracy. The positioning performance does not dramatically change at night as shown in Fig. 4 (b), (d), and (f).

Table 2 summarizes the statistics of Fig. 4. The positioning performance in VN-DGNSS approach surpass the SAE specifications in both daytime or at night. The receivers operating in SPP mode have undesirable vertical positioning accuracy during the daytime.

As shown in Tables 1 and 2, the VN-DGNSS approach achieves the SAE spec in all instances, whereas the spec is not always met without the VN-DGNSS information. The improvement is most pronounced in the improvement of the vertical positioning error during the daytime. After sunset, there is no improvement.

These results indicate that the strongest contributor to vehicle position error is delay due to the signals passing through the earth's atmosphere, particularly the ionosphere, which charges during daytime and discharges at night. The satellite ephemeris, tropospheric, and hardware bias calibration errors are less important. It is also worth noting that multipath errors, which can be significant, have different characteristics for stationary and moving platforms. Multipath is caused by nearby reflective surfaces. For stationary antennae and stationary reflectors, multipath can be correlated over long periods of time. For moving platforms, because the receiver moves relative to reflective surfaces, multipath errors change much more quickly, which makes them easier for the receiver to reject in its vehicle state estimation process.

Correction Mode	u-blox type	Partition	Cumulative probability for stationary results in the daytime and at night			
			Horizontal error < 1.5m	Horizontal error < 1m	Vertical error < 3m	3D error < 3m
VN-DGNSS	M8P (single freq)	Before	94.91%	68.14%	94.01%	91.11%
		After sunset	75.07%	40.03%	68.65%	59.57%
	F9P (Dual freq)	Before	94.45%	64.18%	92.39%	88.61%
		After sunset	84.86%	42.67%	90.64%	83.06%
SPP	M8P (single freq)	Before	86.76%	49.88%	6.24%	4.17%
		After sunset	82.99%	50.51%	76.97%	68.83%
	F9P (Dual freq)	Before	83.19%	48.67%	4.89%	2.97%
		After sunset	82.53%	61.23%	94.63%	89.07%

Table 2: Cumulative probabilities of stationary test in different time

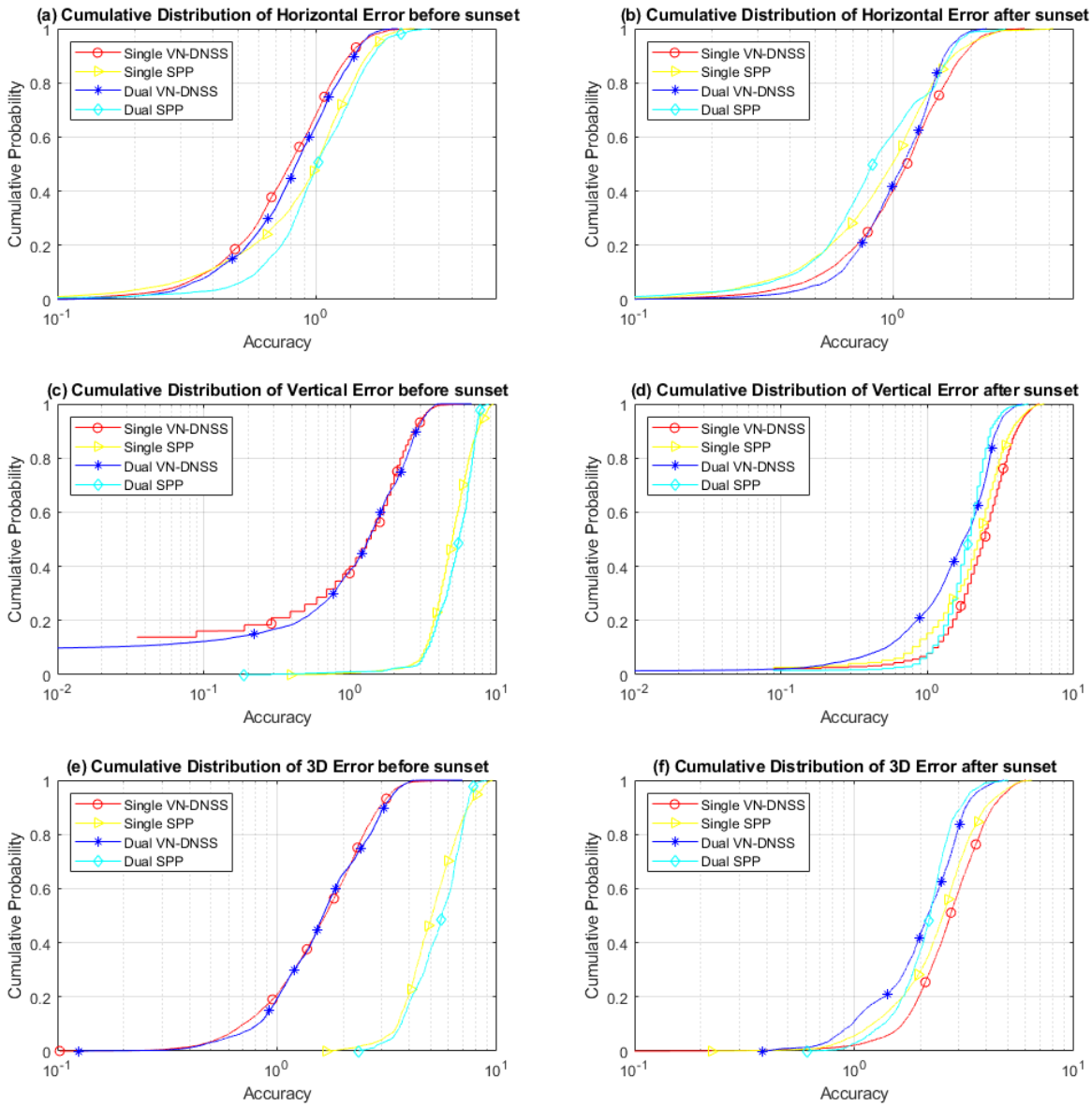


Figure 4: Figure (a), (c), and (e) show the cumulative distribution of the horizontal, vertical, and 3D position error in stationary test before sunset, respectively. Figure (b), (d), and (f) show the cumulative distribution of the horizontal, vertical, and 3D position error in stationary test after sunset, respectively.

Figure 5: Pictures of the experimental set-up and test site

Moving Results

For the moving platform testing, three receivers were connected to a single antenna (as shown in Fig 5).

- One u-blox M8P Single-frequency receiver communicated with the UCR Virtual Network DGNSS (VN-DGNSS) server to receive real-time correction information.
- One u-blox M8P Single-frequency receiver operated in SPP mode, without external corrections.
- One u-blox ZED-F9P Dual-frequency receiver connect to the UCR base station received Multi-GNSS (GPS, Galileo, Beidou, GLONASS) two-frequency pseudorange and carrier phase measurements from a local based station to support real-time kinematic (RTK) fixed solution to achieves centimeter level accuracy. These results serve as the ground truth position for the calculation of position errors for the other two receivers.

All receivers report their computed position every 1 second in UTC.

Figure 6 shows the results of the moving platform experiment. The u-blox accuracy using the UCR VN-DGNSS server information is shown in blue. The accuracy in SPP mode (no corrections) is shown in red. In Figs 6 (a), (c), and (e), the gaps circled in green are caused by loss of the (cell phone based) wireless connection in this period. After reestablishing the wireless communications, the receiver takes several seconds to re-converge to its improved level of accuracy.

- Figs. 6 (a) and (b) show that VN-DGNSS approach can achieve the SAE specified accuracy of 1.5 m on 99% of the epochs and meter-level accuracy on 86% of the epochs. Without VN-DGNSS information, SPP achieves the SAE spec on 88% of the epochs and achieves meter-level accuracy on only 18% of the epochs.
- Figs. 6 (c)-(f) show that the VN-DGNSS approach yields significant improvement in vertical and 3-d accuracy.

Table 3 summarized the positioning performance statistics. In moving platform, both the VN-DGNSS and SPP approaches surpass the SAE specifications in horizontal positioning accuracy. The SPP approach does not achieve the SAE 3-meter vertical accuracy specification while VN-DGNSS approach does.

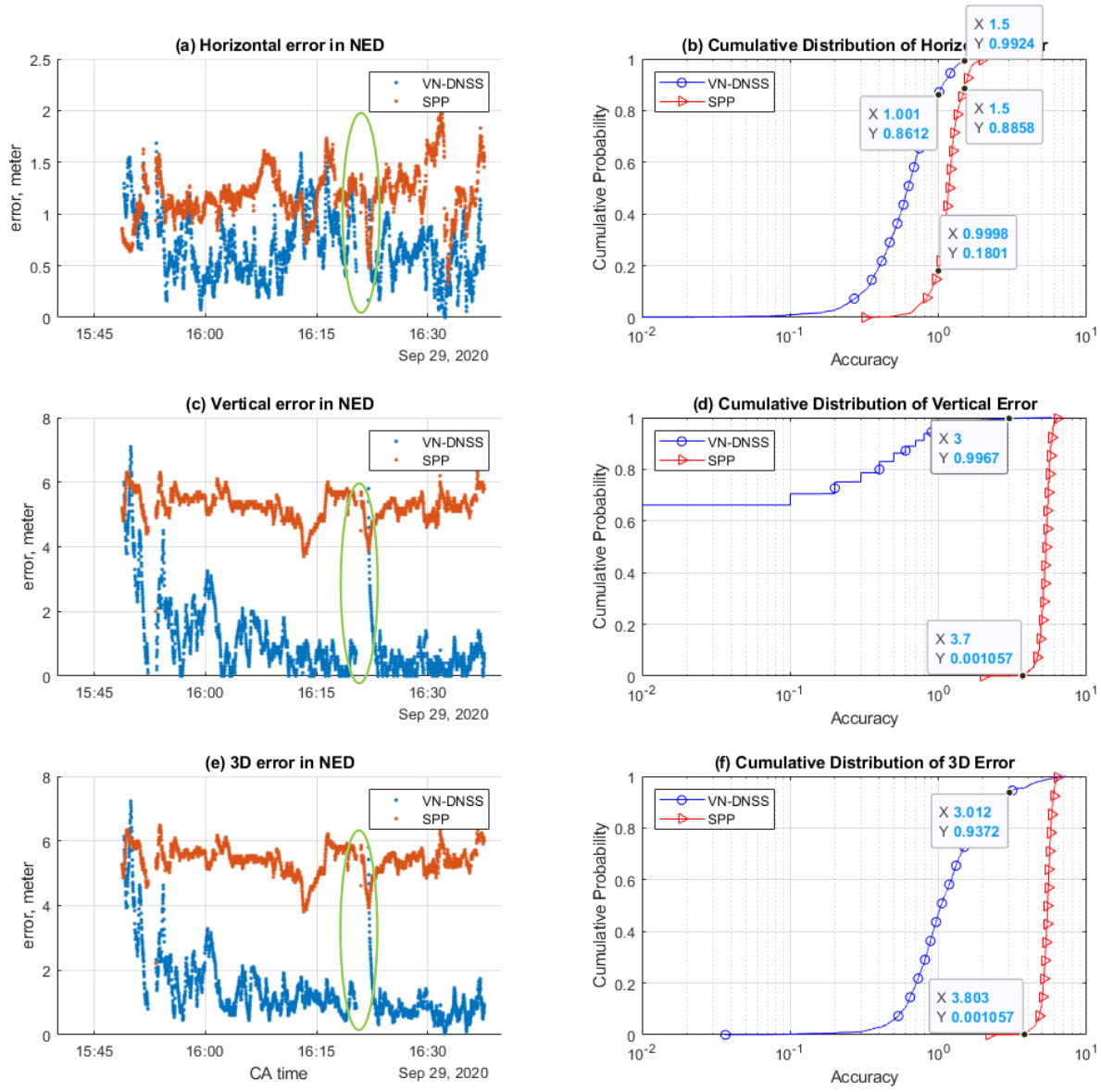


Figure 6: Figure (a), (c), and (e) are the horizontal, vertical, and 3D position error in NED frame plotted versus time respectively. The x-axis shows CA local time in 24-Hr format. Figure (b), (d), and (f) show the cumulative distribution of the horizontal, vertical, and 3D position error respectively. The red curve corresponding to the u-blox in DGNSS mode using the RTCM messages sent by the UCR VN-DGNSS server. The blue curve corresponding to the u-blox in SPP mode.

Correction Mode	ublox type	Cumulative probability for moving results			
		Horizontal error < 1.5m	Horizontal error < 1m	Vertical error < 3m	3D error < 3m
VN-DGNSS	M8P (single freq)	99.24%	86.12%	99.67%	93.72%
SPP	M8P (single freq)	88.58%	18.01%	~0%	~0%

Table 3: Cumulative probabilities for moving test

Evaluation Results from the Fourth Quarter

This evaluation uses the lane matching application. The goal is that the software uses the receiver computed position to determine in which lane the vehicle is situated as it drives along the lane. The system level traffic or intersection controllers can use this lane inventory information to more efficiently control roadway management. Correct lane determination is a function of position estimation accuracy. This evaluation compares application performance for two receivers one with VN-DGNSS corrections and the other without.

Hardware Setup

The receiver configurations are same as the moving test from Fig. 5 using two M8P Single-frequency receivers. One communicated with the UCR VN-DGNSS server, while another one operated in SPP mode, without external corrections. A third u-blox ZED-F9P dual-frequency receiver operated in Multi-GNSS RTK Fix mode to provide a centimeter level accuracy solution that was used as the ground truth vehicle position.

Experiment Description

In this experiment, we drove a sedan on a customized road testbed inside the UCR CE-CERT parking lot that contained two lanes each 3.6 meters wide. The length of road testbed is about 270 meters. The vehicle was driven along both lanes three times in each direction yielding a total of 12 lane traverses, as show in Fig. 7. The antenna (red circle) was mounted on the roof of the sedan (blue box) at three different locations for the purpose of simulating driving at different lateral positions relative to the lane centerline. The green arrows indicate 6 straight driving tests from West to East direction and East to West direction, respectively.

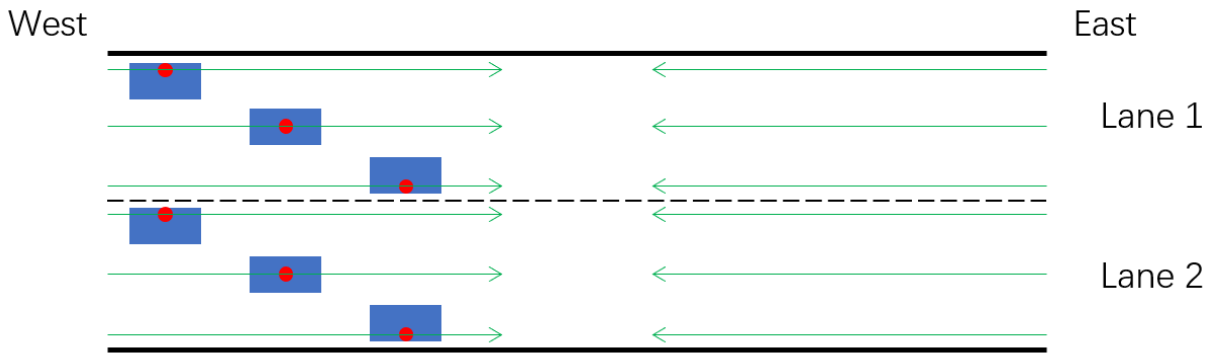


Figure 7: sketch of the lane determination driving test

For analysis, the ground truth position from the u-blox ZED-F9P was used to define the correct lane of the vehicle at each epoch. The lane determinations based on the u-blox M8P receivers with and without VN-DGNSS corrections were compared to this correct lane at each epoch to determine correctness of each decision.

Experimental results

Fig. 8 and Fig. 9 show the trajectories of the 12 lane traverses. The red curves show the ground truth trajectory (ZED-F9P Dual-frequency receiver in RTK mode). The green curves show the trajectory computed by the M8P receiver using the UCR VN-DGNSS server, marked in the legends as “VND”. The blue curves show the trajectory computed by the M8P receiver in standard GNSS SPP mode (no VN-GNSS corrections)).

Fig. 8 shows trajectories driving from East to West. Fig. 9 shows trajectories driving West to East. The red trajectory shows the correct lane, with the first three traverses in the bottom lane (southern) and the last three traverses in the top lane (northern).

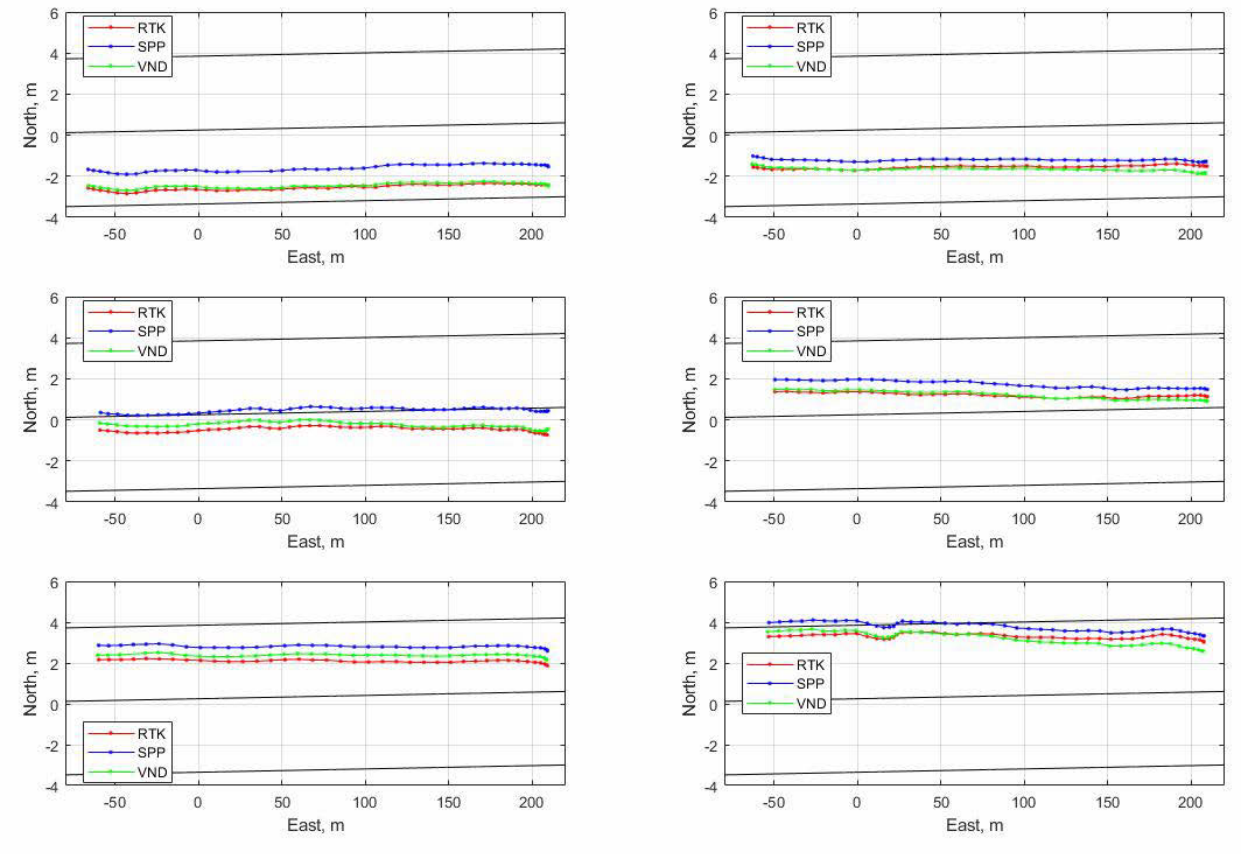
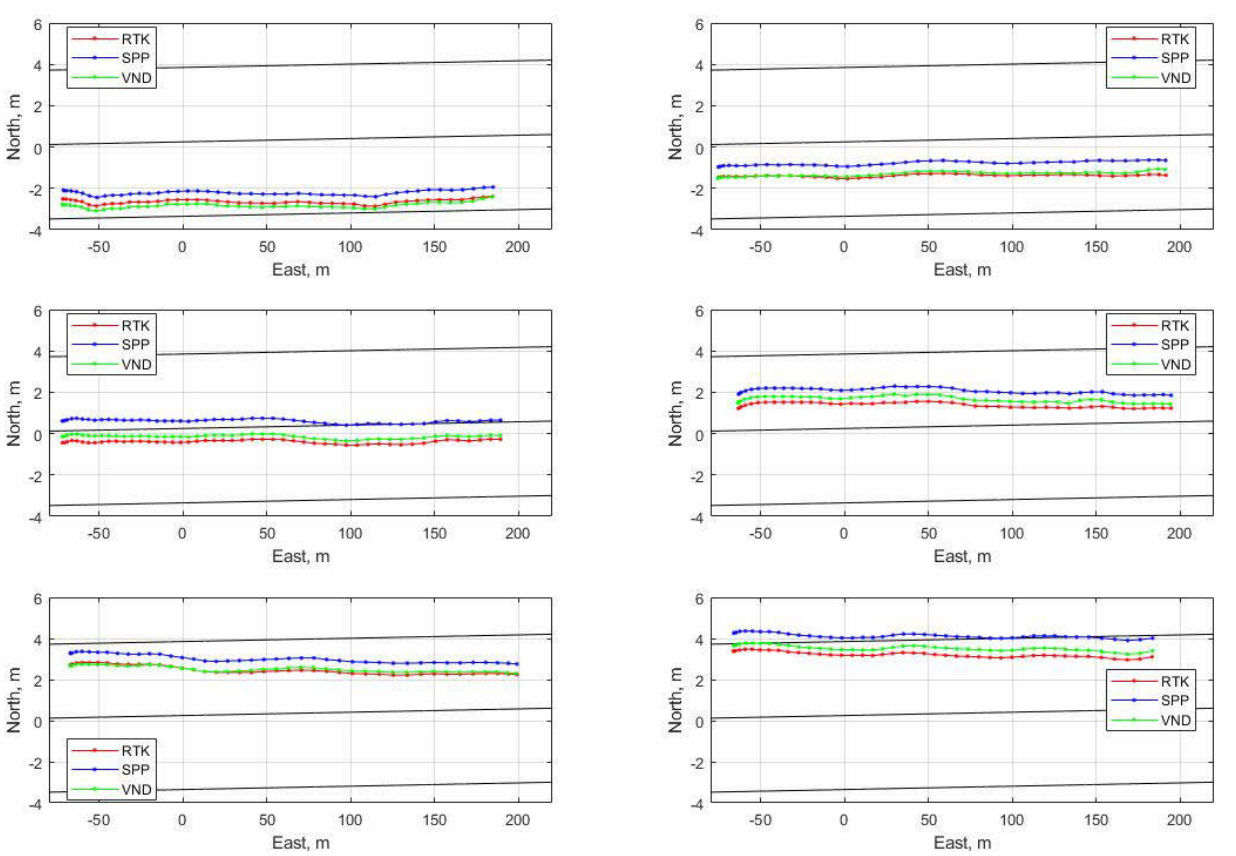
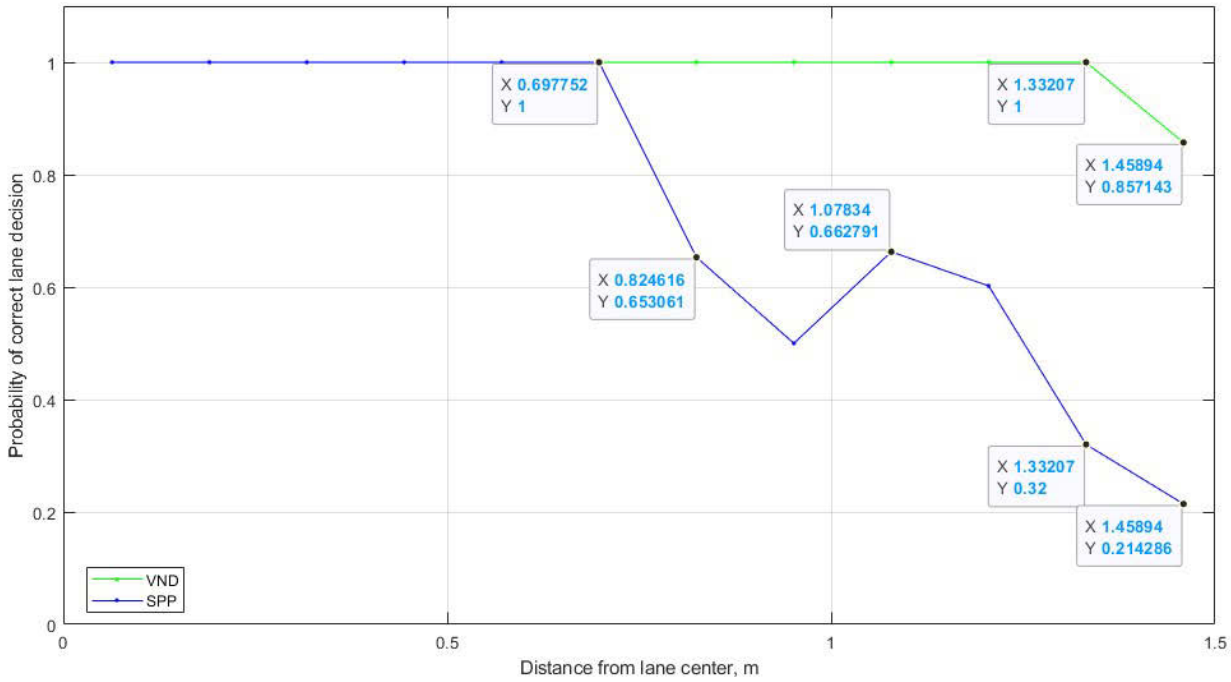


Figure 8 Trajectory of East to West direction driving test

Figs. 8 and 9 allow the reader to visually see whether the SPP and VND computed positions are in the correct lane. It is clear that the VND trajectories are closer to the ground truth RTK solution than are the SPP trajectories. The VND trajectories are also more frequently in the correct lane. Only in the last graph in Fig. 9 does the green curves surpass the edge of upper lane.

Fig. 10 shows the probability of making the correct lane decision using the SPP and VND positions as a function of the distance of the ground-truth trajectory from the lane center. When the vehicle is near the lane center (i.e., less than 0.6 m), both SPP and VND correctly determine the lane. As the distance from the lane centerline increases, the distance to the lane boundary decreases, so that position error begins to affect the lane determination decision. The decrease in probability of making the correct decision drops off earlier and faster for the SPP solution, dropping below 90% at about 0.7 m, which is 1.1 m from the lane edge. The solution using the UCR VN-DGNSS server maintains 90% correct decision through 1.4 m from the lance center, which is 0.4 m from the lane edge.


RTK
SPP
VND
North, m
East, m


X 0.697752
Y 1
X 1.33207
Y 1
X 1.45894
Y 0.857143
X 1.07834
Y 0.662791
X 0.824616
Y 0.653061
X 1.33207
Y 0.32
X 1.45894
Y 0.214286
VND
SPP
Probability of correct lane decision
Distance from lane center, m

Appendix: User Manual

The current draft of the user manual is included as an attached pdf file starting on the next page.

WADGNSS Client User Guide

Wang Hu, Xiaojun Dong, Daniel Vyenielo, Farzana Rahman, and Jay A. Farrell

August 2020

Abstract

This document describes software that enables Wide Area Differential Global Navigation System (WADGNSS) implementation for users on a global basis, without the user needing access to a local GNSS reference station. The software functions in a client-server architecture. Typical users will only use the client portion of the software locally on their computer. That client will connect to a remote server. The remote server accesses State Space Representation (SSR) model parameters for ionosphere, satellite position, satellite clock, and satellite hardware biases via the internet that it uses to construct artificial GNSS satellite measurements for a virtual reference station at a user-specified location $\mathbf{P}_b$. The client software runs on the users computer to perform the following tasks:

1. Establishes a connection to the user GNSS receiver. Any type of physical connection (e.g., serial, USP, ethernet) is feasible. The distributed version of the software supports a serial connection implemented via USB. Any type of GNSS receiver with NMEA protocol and RTCM version 3 protocol can be supported. The distributed version of the software supports u-Blox ZED-F9P and u-Blox M8P.
2. Configures the user GNSS receiver for RTCM differential operations.
3. Obtains an initial user position.
4. Establishes an Ethernet connection to the WADGNSS server.
5. Communicates the desired virtual reference station location $\mathbf{P}_b$ to the server. The location of $\mathbf{P}_b$ should be within approximately 20km of the rover location and can be updated if the rover location changes significantly.
6. Redirects the virtual base station measurements from its Ethernet connection with the server to its physical connection with the rover receiver.

This client/server implementation delivers Observation Space Representation (OSR) corrections to the rover in the form of synthetic measurements coputed for the virtual reference station location $\mathbf{P}_b$. Those measurements are distributed in RTCM format through message types 1004 and 1005 using the NTRIP protocol.

At present, the method works only for the GPS constellation. The theoretical approach behind the software extends to other GNSS systems. At present, the SSR data required to compute the OSR corrections in real-time is only available for GPS.

At present,the client software is available in executable formats at `https://github.com/jaffarrell/WADGNSS`. Later, the source code will also be release under the MIT license.

Contents

Acknowledgements

This software was developed under funding from the California Department of Transportation under the project "Network Differential GNSS Corrections for Connected and Autonomous Vehicles." (Contract 65A0767). The server software utilizes:

- BKG Ntrip (https://igs.bkg.bund.de/ntrip)
- RTKLIB (https://github.com/tomojitakasu/RTKLIB).

The server software is not currently availble for public release. Neither the BKG Ntrip or RTKLIB software is included in the client.

1 Introduction

The client server architecture is illustrated in Figure 1. The server computes synthetic measurements applicable to a virtual reference station at location $\mathbf{P}_b$ for use by the user receiver. The client software is hosted on a user computer with a direct connection to the user receiver. The purpose of the client software is to facilitate communications between the server and user receiver. At start up, the client establishes a connection with the user receiver, configures the receiver for differential operation, establishes communications with the server, and communicates the users IP address and virtual base location $\mathbf{P}_b$ to the server. After that point, the client receives the virtual base station measurements from the server via Ethernet and sends them to the user receiver over the local connection (e.g. serial port or USB).

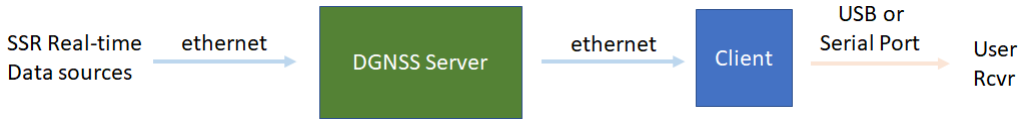


Figure 1: Client Server DGNSS Architecture.

2 Client

The following sections discuss compilation, installation, GPS receiver setup, and execution of the `WADGNSS_Client` application. We assume most users will read the guide sequentially once and then use it only as a reference document.

We recommend that most users begin from Section 2.2 on Installation. It describes the process for installing a pre-compiled executable application. We will also distribute the source code. Users who wish to start from the source code should begin with Section 2.1 which describes the source code installation and compilation processes.

2.1 Client Software Compilation

This section describes compilation on the Ubuntu 18.04 operating system. The procedure may be compatible with other Linux distributions, but is untested. We are actively developing on Ubuntu and Windows. For compilation on windows we recommend using a cmake compatible integrated development environment like Visual Studio.

1. Download the software necessary to compile C++ source code. You will need the `cmake` build tool and a `C++` compiler such as `g++`. If not already installed on your local system, you can run the following commands to download the most recent releases.

   ```
   $ sudo apt update && sudo apt upgrade
   $ sudo apt clean && sudo apt auto-remove
   $ sudo apt install make cmake g++
   ```

2. Download the WADGNSS Client software onto your machine. The download URL is in the abstract of this user manual.[1]

3. Compile the client software on your local machine.

   (a) Navigate to the directory that contains the `WADGNSS_Client` source code (e.g., .../WADGNSS/client/Code). This directory contains the `CMakeLists.txt` file.

   (b) Execute the following to make a build directory and navigate into the newly created build directory.

   ```
   $ mkdir build && cd build
   ```

   (c) Compile the `WADGNSS_Client` by executing the following commands. The first command will configure the `cmake` build system and the second will invoke `make` to start the compilation process.

   ```
   $ cmake .. && make
   ```

   Include the '..'

   (d) If compilation is successful an executable named `WADGNSS_Client` will be visible in the build directory. The client software is now successfully compiled.

You can now proceed to Section 2.3.

[1] At the time of the writing of this document, the source code has not yet been posted. When it is posted, this document will be revised accordingly.

2.2 Client Software Installation

This section describes the recommended installation procedure for the `WADGNSS_Client` executable. At present, we support installation on Ubuntu 18.04 (Ubuntu 18.04.5 LTS) and Windows. This procedure has also been successfully tested on Debian 10.

1. In any directory on your machine create a new directory and navigate into the new directory with the following command.

   ```
   $ mkdir WADGNSS && cd WADGNSS
   ```

2. Copy the `WADGNSS_Client` executable into the `WADGNSS` directory. The `WADGNSS_Client` executable is now successfully installed.

You can now proceed to Section 2.3.

2.3 GPS Receiver Setup

Before running the `WADGNSS_Client` executable, the user must initialize and configure their receiver using its specific software or other interface. At present, `WADGNSS_Client` software supports automated configuration for u-Blox ZED-F9P and u-Blox M8P. All other receivers need to be configured manually. Please note that `WADGNSS_Client` currently only supports GPS receivers compatible with serial input and output communications. Serial messages are used to serve corrections.

The following example on manual configuration is for a u-blox receiver using U-center. Manual configuration of other receivers can be accomplished by configuring the same parameters using the device specific configuration software:

- Enable GPS and disable all non-GPS navigation systems.
- Enable GxGNS messages using the NMEA protocol.
- Enable message output from the serial port of the GNSS receiver.
- Enable RTCM version 3 message input through the serial port of the GNSS receiver.

An example for u-blox receivers using U-center software is shown below.

Step 1: Enable GPS and set GPS only.

Go to *View* → *Messages View* → *UBX* → *CFG* → *GNSS*. Select enable GPS and disable others. Then click "*Send*". See Fig. 2.

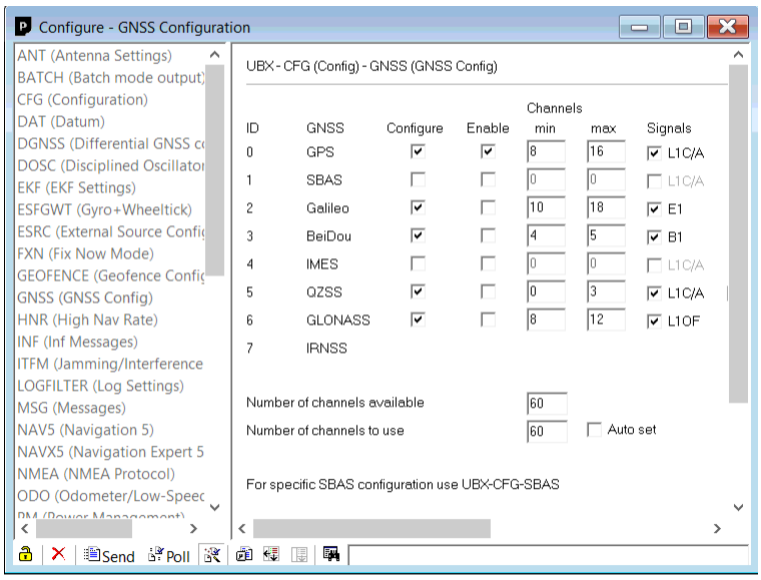


Figure 2: Step 1

Step 2: Enable PUBX 00 via NMEA protocol.

Go to *View* → *Messages View* → *NMEA* → *GxGNS*. Right click "*GxGNS*" then select "*Enable Message*". See Fig. 3.

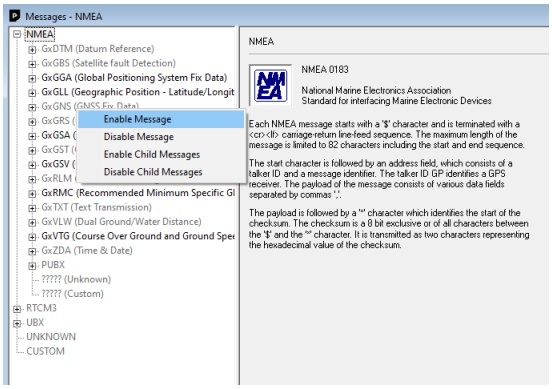


Figure 3: Step 2

Step 3: Enable a receiver USB port for RTCM version 3 input.

Go to *View* → *Messages View* → *UBX* → *CFG* → *PRT*. Select the options based on Fig. 4. Then click "*Send*". See Fig. 4.

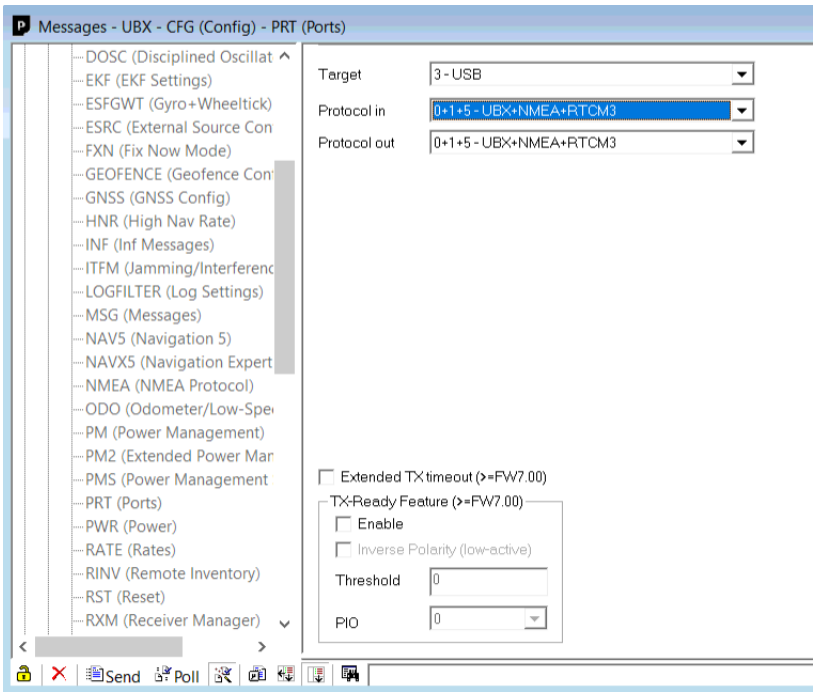


Figure 4: Step 3

We recommend saving the configuration to the GPS receiver. If configurations are saved the receiver should remain configured the next time it starts following power off. In U-center, configurations can be saved to the GPS receiver by navigating to the *CFG* menu item on the *Configuration View*. Then, select *Save current configuration*. Finally, press the *Send* button. The configurations will save to the receiver. The steps for saving configurations in other GPS receiver configuration software will differ.

2.4 `WADGNSS_Client` Execution

The `WADGNSS_Client` executable expects four command line arguments: com port, configuration option, server hostname and server port. An example usage template is shown below.

```
$ ./WADGNSS_Client [com port] [configuration] [server hostname] [server port]
```

The following sections discuss these command line parameters and how to find the correct values for your system.

2.4.1 Parameter: [com port]

The [com port] option designates on which of the computers USB communication ports the WADGNSS_Client software communicates with the GPS receiver. Determining this port number is operating system specific. It may take a couple attempts to discover the correct filepath.

In Ubuntu, serial data from the GPS receiver is available at a filepath matching the pattern \dev\ttyACMX where X is a number in the range [0-9]. For example, if your receiver is connected to your computer at \dev\ttyACM0 you would provide the filepath \dev\ttyACM0 in place of the [com port] argument.

In Windows, the com port can be determined by opening Device Manager and expanding Ports (COM & LPT) option. For example, suppose your GPS receiver is visible as device COM6. Then, the filepath COM6 is provided in place of the [com port] argument.

2.4.2 Parameter: [configuration]

The [configuration] parameter indicates the desired configuration of the supported GPS receiver. If supported, the WADGNSS_Client will initialize the connected GPS receiver to the corresponding configuration. We currently support these automated configuration options:

- 0: Receiver configured by its software,
- 1: u-blox M8P module
- 2: u-blox ZED-F9P module

2.4.3 Parameters: [server hostname] and [server port]

The [server hostname] and [server port] parameters indicate the network location of the WADGNSS_Server to which the WADGNSS_Client will connect. We recommed connecting to the official WADGNSS server hosted at pppdgnss.engr.ucr.edu on port 2101. Substitute these values for [server hostname] and [server port] parameters when executing the WADGNSS_Client, respectively.

2.4.4 Example Execution

Bringing all this together, an example execution command for the WADGNSS_Client where USB data is available at \dev\ttyACM0, the connected GNSS receiver is u-blox M8P, and the client will connect to a WADGNSS_Server at pppdgnss.engr.ucr.edu on port 2101 is as follows:

```
$ WADGNSS_Client \dev\ttyACM0  1  pppdgnss.engr.ucr.edu  2101
```

Congratulations you have successfully launched the WADGNSS_Client. To stop the WADGNSS_Client press "Ctrl + c" keys, simultaneously.

2.5 Client Output Files

The client produces two output files.

ROVERLOG_YYYYMMDD.log: In The file name, YYYY for year, MM for month, and DD for day. This file contains the position information output by the user receiver (i.e., the rover). For the default configuration (as distributed), the rover outputs the $GxGNS messages in NMEA protocol. The user may edit the client source code to change the receiver output messages or to redirect such receiver output information through computer ports to other programs or devices.

running_log.log: This file contains client connection status messages. For example, a typical sequence of start-up messages is as follows:

```
[2020-09-23 03:31:02]: Start connecting receiver.
[2020-09-23 03:31:02]: Succeed.
[2020-09-23 03:31:03]: Initialize u-blox type: 2 (1 for M8P, 2 for ZED-F9P).
[2020-09-23 03:31:03]: Build connection with server succeed.
[2020-09-23 03:31:03]: Start read pos info ($GxGNS) from rover.
[2020-09-23 03:31:04]: Send to server: $POSECEF -2427457.4792 -4709287.4204 3540190.5200
```

The first three messages relate to establishing communications with and initializing the receiver. The last three messages related to establishing communications with the server and sending it the position $\mathbf{P}_b$.

