Developed GNSS Relative Positioning Software and a Comparison of Relative Positioning and Precise Point Positioning Methods on Kinematic Data

Agata Szeremeta

213 400 148

ESS 5410: Final Project Report

Professor Bisnath & Professor Wang

April 17, 2019

Executive Summary

The main objectives of this project were to develop a relative positioning software and use such software to perform a comparison on the post-processed kinematic (PPK), specifically Relative Positioning, and Kinematic Precise Point Positioning (PPP) methods. In addition, an objective of this project was to study the various factors that affect positioning results, such as observation length, short and long baselines, etc. Unfortunately, due to a error in the final processing step of the software, a fully functioning software was not completed. However, a comparison of the two positioning methods and brief study of the various factors that affect their results was still made. Publicly available software, *RTKLib* and *CSRS-PPP*, were used to obtain relative positioning and precise point positioning results, respectively, regarding recently (self-collected) and previously collected GPS kinematic data. As the conditions of the datasets (i.e. their collection processes) are significantly different from each other, the impact of the factors that could affect results were easily observable in the results.

Findings agree with the expected result of relative positioning providing positional results of a greater accuracy than precise point positioning methods. Study of the factors show that the base station (specifically, its coordinates) play a significant role in the relative positioning process and the position solution of the rover receiver. It was observed that longer observation periods and greater number of epoch satellites, tend to improve the positional accuracies.

Table of Contents

List of Tables

List of Figures

1 Introduction

Satellite-based positioning is the process of determining one's position using ranges to satellite vehicles. In today's world, positioning using this method has become very common and even a necessity for life. Unfortunately to many, the complexity of such positioning is often unknown, with only the result in mind. However, several methods for such satellite-based ranging exist, each with their own advantages and disadvantages. Examples of positioning methods include single point, precise point, differential, relative, and real-time kinematic. Each mode requires their own combinations of Global Navigation Satellite System (GNSS) receivers, radio communication links, observed signals, etc.

The accuracies of positioning are driven by 3 main factors: the processing method, the type of measurement, and the corrections of errors sources in the positioning model [1]. Various combinations of these factors can cause the positional accuracy to range between meter level to millimeter level. For some applications, such as vehicle tracking, a position accurate in the meter level is substantial. However, for others (ex. boring, autonomous vehicles) a high positional accuracy is required. Although desirable, having the highest positional accuracy may not be feasible due to equipment costs or the need for additional equipment and processing.

Post-processed relative positioning and precise point positioning (PPP) methods are known to produce higher accuracy results, compared to the other possible processing modes. This is due to the mathematical ranging models they make use of. However, the two methods greatly differ from each other in their configurations (for example, in their required equipment, processing equations and applied error corrections). The objective of this project was to compare the relative positioning method with the kinematic PPP method. More specifically, the project aimed to compare the obtained positioning results (of a moving receiver) using the two methods and to test the factors that affected the results (such as, stop and go motion of a receiver, static initialization period, etc.). To aid in this, relative positioning software was created to post-process kinematic GPS data.

2 Background

The principal behind satellite-based position follows the general geometry below.

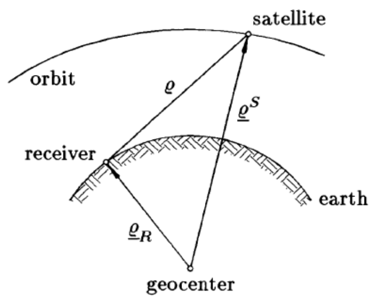


Figure 1: Geometry of Satellite Positioning [2]

As the orbital path of satellites are deterministic, the vector between the geocenter (or center of Earth) to the satellite vehicle is known. Furthermore, the receiver is able to observe satellite transmitted messages and determine the (approximate) vector/range to the satellite. Through subtraction of these two vectors, the vector from the geocenter to the receiver is determined and, thus, the position of the receiver can be found.

In short, range measurements to the satellite are computed from timing measurements (i.e. time shifts) between a receiver's generated replica data and satellite transmitted/received data. Range measurements can be determined from either the transmitted carrier phase or code (pseudorange) observables [1]. The position vector of the satellite vehicle is computed from the satellite's broadcast navigation message. The ranges from four or more satellites are then used in geodetic trilateration (point positioning) computations (either through Least Squares or Kalman Filtering), to determine the position of the phase center of a receiver's antenna. In a point positioning problem, the receiver clock error/offset is also estimated in addition to the receiver position. Although the latter is often not of any interest to the user, it is essential to consider as it affects the time shift measurements and any errors from timing systems will propagate to this difference and affect the final positioning result.

As positioning was performed using carrier phase measurements and not code range observables, positioning with only such measurements is briefly discussed below.

2.1 Satellite-based Positioning Using Carrier Phases

The basic carrier phase (or simply, phase) range model at epoch *t* is defined as

$$\phi_r^s(t) = \frac{1}{\lambda^s}\rho_r^s(t) + N_r^s + \frac{c}{\lambda^s}\left(dt_r(t) - dt^s(t)\right) + \epsilon(t)$$

$$\phi_r^s(t) = \frac{1}{\lambda^s}\rho_r^s(t) + N_r^s + f^s\left(dt_r(t) - dt^s(t)\right) + \epsilon(t)$$

$$\phi_r^s(t) = \frac{1}{\lambda^s}\rho_r^s(t) + N_r^s + f^s\Delta\mathrm{dt}_r^s(t) + \epsilon(t) \qquad (2.\ 1)$$

where $\phi_r^s(t)$ is the measured carrier phase (expressed in cycles) from receiver *r* to satellite *s* at epoch *t*, λ is the wavelength of the satellite carrier (in metres), N is the integer phase ambiguity, c is the speed of light (in metres per second), dt_r is the receiver clock bias (in seconds), dt^s is the satellite clock bias (in seconds), ϵ is the random error of the measurement, and ρ is the geometric range (or Euclidean distance) between receiver and satellite 3D positions (in metres):

$$\rho_r^s(t) = \sqrt{\left(X^s(t) - x_r(t)\right)^2 + \left(Y^s(t) - y_r(t)\right)^2 + \left(Z^s(t) - z_r(t)\right)^2} \qquad (2.\ 2)$$

The frequency of the satellite carrier, f^s, can be substituted in place of $\frac{c}{\lambda^s}$. The *(t)* notation is removed from equations involving simultaneous measurements (i.e. same epoch data) from hereon for notational simplicity. If considering the more complex model with noise and atmospheric errors, (2. 1) can be re-written as

$$\lambda^s\phi_r^s = \rho_r^s + c(dt_r - dt^s) + dtropo_r^s - diono_r^s + \lambda N_r^s + dmulti_r^s + d\delta_r + d\delta^s + \epsilon \qquad (2.\ 3)$$

where $dtropo$ is the correction for the carrier signal due to tropospheric refraction, $diono$ is the correction for ionospheric phase advance of the signal, $dmulti_r^s$ is the correction for the multipath delay, and $d\delta$ is the fractional biases of the receiver or satellite. The atmospheric corrections (troposphere and ionosphere) are estimated from either existing closed-form models, mapping functions, the GPS data itself, or hybrid models [1]. Often, the multipath and hardware delays are difficult to model and are disregarded and incorporated into the measurement error (therefore, are not considered for the positioning estimation process).

Positioning using carrier phase measurements often results in higher accuracy estimates for the unknown parameters (i.e. receiver position and clock bias), compared to that using code range measurements. However, such measurements/models introduce an additional complexity to the processing – an ambiguity through the N variable. The receiver is only capable of measuring the change in phase (from when it 'locks on' to the satellite), not the initial fractional phase (at the both the satellite transmitter and receiver) or integer number of whole cycles at initial receiver tracking loop lock [3, 4]. Thus, to exploit the high accuracy of the phase observables and use them for ranging, the user must also estimate the ambiguity term while estimating the unknown parameters [4]. However, (2. 3) is a mixed domain equation, where the ambiguity term is an integer count and all other variables are float/real-valued. The estimation process would result in float value for the ambiguity term (or simply a "float solution"). Using the float solution would provide inaccurate results, as they do not agree with the constraint of the model/system (i.e. being a whole number count of cycles) [4]. Such "float solutions" must be fixed to integer values. The process of doing so, is commonly referred to as ambiguity resolution.

2.2 Ambiguity Resolution

Ambiguity Resolution (AR) is the process of fixing ambiguity (float) estimates to integer values. It consists of 3 main steps [4]:

1. Generation of potential integer ambiguity candidates
2. Identification of optimal ambiguity candidate
3. Validation/Verification of selected ambiguity

In the first step, candidates for integer ambiguities are estimated from a search space, defined by the uncertainty (or square variance) of the estimated float ambiguity [4]. That is, all possible candidates are the integers that fall within the $\pm\sigma$ range of the float solution. In the second step, the optimal (or best estimate) candidate is identified. Such a candidate is typically the integer value that fits the least-squares principle of minimizing the sum of square residuals of the phase observation equation [4]. That is, each integer candidate is substituted into the observation equation and the residuals of the measurement model are assessed. Finally, in the third step, the ambiguity term that was identified as optimal is confirmed as the best estimate. To do so, typically the statistical difference between the best and second-best candidates (i.e. those that provide the smallest and second smallest measurement residuals) is studied through a ratio test [4]. If the ratio is larger than a predefined threshold, than the best estimate is valid. Otherwise, it can be concluded that no significant statistical difference between the two integer candidates exists and, as such, there is no integer solution to the ambiguity [4]. In additional, residual analyses can be performed – including statistical testing on the residuals themselves [4]. It should be noted that the success of the AR process is strongly affected by the estimation reliability of the error sources in the observation model (ex. tropospheric delay, ionospheric advance, etc.) [4]. Poor error modelling will likely result in less accurate AR, as errors in the search space exist.

Some known AR methods include the Least Squares Ambiguity Search Technique (LSAST), Fast Ambiguity Resolution Approach (FARA), Modified Cholesky Decomposition Method, Least-Squares

Ambiguity Decorrelation Adjustment Null Space Method (LAMBDA), and Fast Ambiguity Search Filter Optimal Method for Estimating GPS Ambiguities (FASF OMEGA) [5].

As mentioned previously, AR introduces additional complexities in the processing steps for satellite-based positioning. However, it must be performed to use phase observables for ranging. As such, it is often not applied when performing single point positioning, since the accuracy of the solution itself is already limited (due to the inability to model all other error sources in the observation equation well). Instead, it is often applied when performing relative positioning, as the models in such processing techniques better mitigate/eliminate such errors, thereby allowing for better or more accurate position estimation and justify for the increased processing complexity.

2.3 Relative Positioning

Relative positioning involves determining a receiver's position relative to another receiver (often, with known position). The technique aims to determine the baseline vector between the two receivers (or points), from which the unknown position of the receiver can be indirectly found [5, 3]. That is, if receiver A is the known point and receiver B is the unknown, then the position of B can be determined through the relation between the position vectors $\bar{X}_A$ and $\bar{X}_B$, and the baseline b_{AB}

$$\bar{X}_B = \bar{X}_A + b_{AB} \tag{2. 4}$$

where the baseline is defined as

$$b_{AB} = \bar{X}_B - \bar{X}_A = \begin{bmatrix} X_B - X_A \\ Y_B - Y_A \\ Z_B - Z_A \end{bmatrix} = \begin{bmatrix} \Delta X_{AB} \\ \Delta \mathrm{Y}_{AB} \\ \Delta \mathrm{Z}_{AB} \end{bmatrix}. \tag{2. 5}$$

Rather than correcting individual measurements for errors such as clock errors or atmospheric (i.e. ionosphere, troposphere) effects, simultaneous measurements from two receivers can be combined through relative positioning methods to mathematically eliminate and/or reduce the magnitudes of the errors [3]. This technique is beneficial as it can be difficult to correct individual measurements (i.e. knowledge of the true correction value can be limited and not entirely known). Thus, through the combination of measurements, a more precise position can be determined as compared to that of single point positioning.

There are many methods of performing relative positioning, including Single-Differencing, Double-Differencing, and Triple-Differencing, each of which develops on the previous. Single-Differencing, hereon abbreviated as SD, involves performing the (relative) differencing between receivers (therefore, using 2 receivers and data from 1 satellite). Double-Differencing, or DD, involves performing differencing between receivers and satellites (therefore using 2 receivers and data from 2 satellites). Triple-Differencing, or TD, involves performing differencing between receivers, satellites, and time (therefore, requiring 2 receivers and data from 2 satellites at 2 different epochs. As TD was not applied in this project, it is not discussed below. As with single point positioning, relative positioning techniques can be applied on both code range or carrier phase range data. As the purpose of this research was to determine position using the carrier phase measurements, an explanation is only provided using the latter. However, similar methods hold for processing using code range.

Relative positioning techniques often provide position results with accuracies in the millimeter to centimeter level. Of course, this is dependant on factors such as the equipment used, survey configuration and the order of differencing (1st, 2nd, or 3rd).

2.3.1 Single Differencing

Consider the geometry below, where *A* and *B* as receiver sites and *j* is the satellite transmitting data.

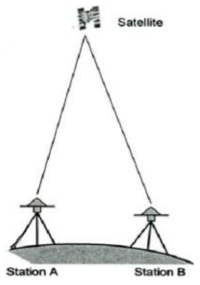


Figure 2: Single-Differencing Relative Positioning Satellite/Receiver Geometry [6]

The phase ranges for each of the receivers, disregarding multipath delays as in (2. 3), are modelled as:

$$\lambda^j \phi_A^j = \rho_A^j + c\left(dt_A - dt^j\right) + dtropo_A^j - diono_A^j + \lambda N_A^j + d\delta_A + d\delta^j + \epsilon_A^j$$
$$\lambda^j \phi_B^j = \rho_B^j + c\left(dt_B - dt^j\right) + dtropo_B^j - diono_B^j + \lambda N_B^j + d\delta_B + d\delta^j + \epsilon_B^j$$

Taking the mathematical difference between the two observation equations:

$$\phi_{AB}^j = \phi_B^j - \phi_A^j$$
$$\lambda^j \phi_{AB}^j = \rho_{AB}^j + cdt_{AB} + dtropo_{AB}^j - diono_{AB}^j + \lambda^j N_{AB}^j + d\delta_{AB} + \epsilon_{AB}^j \tag{2. 6}$$

With this difference, the satellite clock offset, and satellite fractional bias terms are eliminated – since the same satellite is being simultaneously observed and the same errors are seen in each equation. However, with the differencing, the noise term amplifies due to the laws of error propagation. As suggested by Hofmann-Wellenhof in [5], the atmospheric corrections can be disregarded from the model for short baselines, since the refractions (and corrections) are approximately equal at both receiver sites and are likely to cancel out in the differencing process. However, it should be noted that the error of their residuals may propagate into the unknown parameters [5]. Thus, (2. 6) can become:

$$\lambda^j \phi_{AB}^j = \rho_{AB}^j + cdt_{AB} + \lambda^j N_{AB}^j + d\delta_{AB} + \epsilon_{AB}^j \tag{2. 7}$$

To further mitigate/eliminate error sources, a second difference can be applied on this single difference equation.

2.3.2 Double Differencing

Double differencing involves subtracting two simultaneous single differences. Consider the geometry below, where *A* and *B* as receiver sites, and *j* and *k* are the satellites transmitting data.

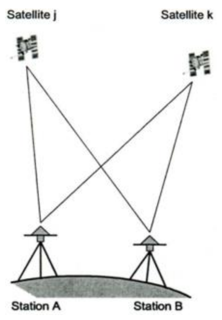


Figure 3: Double-Differencing Relative Positioning Satellite/Receiver Geometry [6]

Assuming data from the same (transmission) frequency is acquired by each receiver from two different satellites ($\lambda^j = \lambda^k = \lambda$) and still considering atmospheric effects, the differencing result can be mathematically represented through:

$$\phi_{AB}^{jk} = \phi_{AB}^{k} - \phi_{AB}^{j}$$
$$\lambda\phi_{AB}^{jk} = \rho_{AB}^{jk} + dtropo_{AB}^{jk} - diono_{AB}^{jk} + \lambda N_{AB}^{jk} + \epsilon_{AB}^{jk} \tag{2. 8}$$

With this double difference, the receiver clock offset, and receiver fractional bias terms are eliminated – since the same receiver errors exist when observing the two satellites. As before, the measurement error amplifies due to the laws of error propagation and, for short baselines, the tropospheric and ionospheric terms can be omitted.

Because of the elimination of the receiver clock biases, the double differencing technique is preferable [5]. However, like with the single point positioning techniques, the model requires an integer ambiguity (difference) term to be solved. This can be completed in a similar manner as explained in the Ambiguity Resolution section above.

2.4 Precise Point Positioning

In the standard point positioning method, the satellite transmitted broadcast message is used to determine position and compute clock information. Precise point positioning is similar to standard, single point positioning except it uses precise orbit and clock information, additional error modelling (ex. for solid earth tides, ocean loading), and code and phase filtering [7]. That is, the method substitutes the common to all, lower quality/accuracy observations for precise data that is more difficult to access. Using higher precision data eliminates/mitigates the error sources in the observation model therefore resulting in lower error in positioning results. This can be better understood through the visual below.

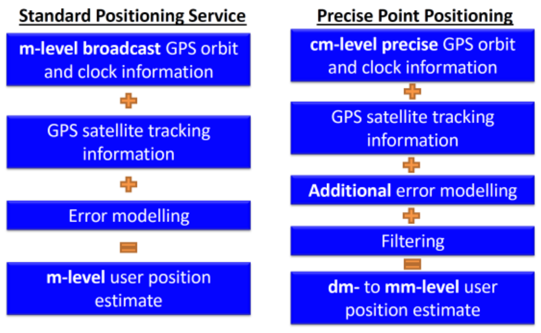

Standard Positioning Service
m-level broadcast GPS orbit and clock information
GPS satellite tracking information
Error modelling
m-level user position estimate
Precise Point Positioning
cm-level precise GPS orbit and clock information
GPS satellite tracking information
Additional error modelling
Filtering
dm- to mm-level user position estimate

Figure 4: Comparison of Standard and Precise Point Positioning Workflows [7]

Unlike relative positioning, PPP only requires a single user receiver. However, a global array of reference stations are still required to obtain precise ephemeris and clock data [7]. The latter is beyond the 'control' of the user – which makes the method appealing, as it has the appearance of producing higher accuracy results with only 1 receiver.

Despite the use of greater precision data in computations, the PPP method is limited in the convergence period and its accuracy [7]. Precise point positioning techniques often provide position results with accuracies in the milliimeter to decimeter level. Like with relative positioning, this is dependant on the quality of equipment used, proximity to the reference station, etc.

3 Software Implementation

Software was written to perform relative positioning using the double differencing technique. A general workflow or functional blocks of the relative positioning generated software structure can be observed in Figure 5: General Workflow of Generated Software. The Python software, with incorporation of the LAMDBA Ambiguity Resolution Matlab software, consists of 5 main functional blocks:

1. File Readers
2. Satellite Position Computer
3. Data Combiner
4. Least Squares Receiver Position Estimator (Using Code Range Data)
5. Least Squares Relative Receiver Position Estimator (Using Phase Range Data)

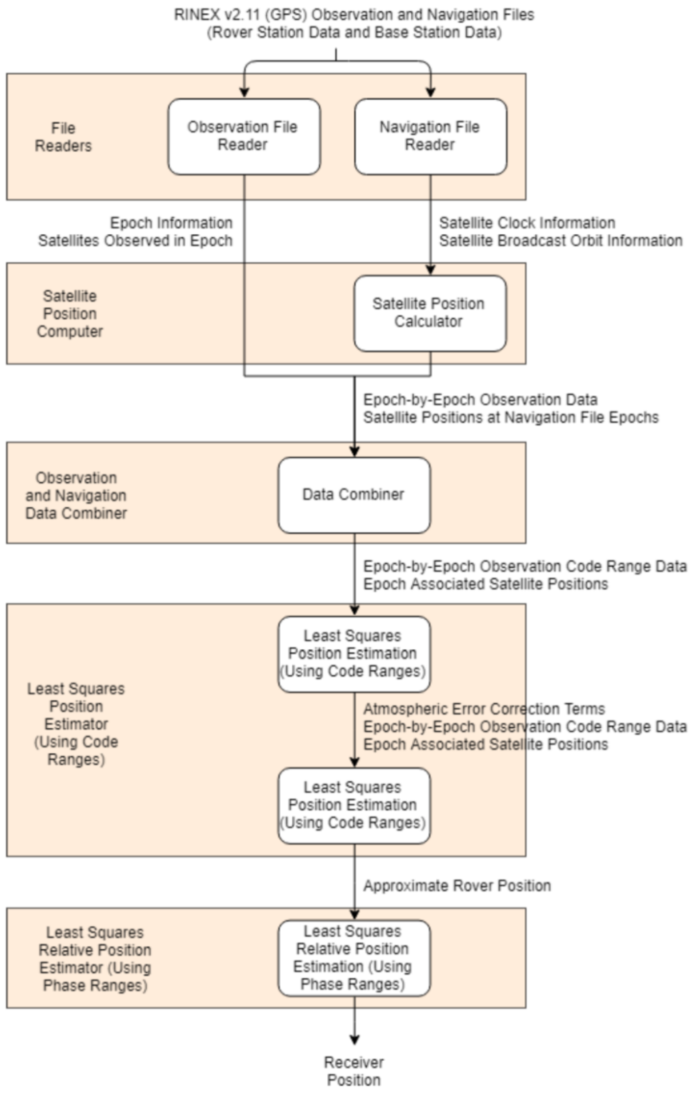

RINEX v2.11 (GPS) Observation and Navigation Files
(Rover Station Data and Base Station Data)
File Readers
Observation File Reader
Navigation File Reader
Epoch Information
Satellites Observed in Epoch
Satellite Clock Information
Satellite Broadcast Orbit Information
Satellite Position Computer
Satellite Position Calculator
Epoch-by-Epoch Observation Data
Satellite Positions at Navigation File Epochs
Observation and Navigation Data Combiner
Data Combiner
Epoch-by-Epoch Observation Code Range Data
Epoch Associated Satellite Positions
Least Squares Position Estimator (Using Code Ranges)
Least Squares Position Estimation (Using Code Ranges)
Atmospheric Error Correction Terms
Epoch-by-Epoch Observation Code Range Data
Epoch Associated Satellite Positions
Least Squares Position Estimation (Using Code Ranges)
Approximate Rover Position
Least Squares Relative Position Estimator (Using Phase Ranges)
Least Squares Relative Position Estimation (Using Phase Ranges)
Receiver Position

Figure 5: General Workflow of Generated Software

The generated software was developed to process Rinex v2.11 data collected by a receiver only capable of ‘understanding’ (i.e. acquiring, tracking, etc.) single frequency signals from the GPS satellite constellation. It should be noted that due to errors in determining the float ambiguities, the software fails to function correctly in the Least Squares Relative Position Estimator (Using Phase Ranges) block. This error is discussed more in the appropriate section. Nevertheless, given appropriate corrections, results of this self-generated software could be validated against *RTKLib*, an open-source program package for GNSS processing. It should be noted that the software was designed assuming a static base (receiver) station and kinematic/mobile rover station, and that the rover position is estimated at each epoch. However, with a minor adjustment to the software, similar methodology can be applied to determine the relative position of a rover to a kinematic base station. Furthermore, the software was designed considering if the coordinates of the base station were unknown (and thus, also required estimation). However, it is known that the errors of the base station estimation process would propagate into the errors of the rover position. At the current time, no additional consideration for this is performed by the generated software. Often in practice, the base station is set over a known control point and, thus, this estimation is not required. Lastly, it should be noted that the user must have the Matlab Engine API for Python installed in their environment prior to running the software. This is necessary by the software to call on the LAMBDA Ambiguity Resolution open source software. Such an API requires the use of a version of Python older than 3.6, as the Matlab Engine API does not function for newer versions.

In addition to this relative position processing software, brief code was written to compare results determined using *RTKLIB* relative positioning and precise point positioning processing methods. This code is not incorporated into the software, and instead needs to be manually run/configured by the user with the appropriate filenames.

3.1 File Readers

First, the software was designed to read and store the essential information transmitted (and decoded) from the GPS satellite to each of the receivers. Again, the software is only capable of reading RINEX v2.11 data formats – particularly only the observation and navigation (broadcast message) files. The two file readers are designed to 'decode' the messages based on the known header and data record formats (i.e. known positions of variables in the 80-character lines). In addition, the variables in each position were decoded following the guides at the following websites:

Navigation File:	http://www.gage.es/sites/default/files/gLAB/HTML/GPS_Navigation_Rinex_v2.11.html
Observation File:	http://www.gage.es/sites/default/files/gLAB/HTML/Observation_Rinex_v2.11.html

A brief summary of the extracted essential content of each data file header and body message is summarized in the table below. The data collected for each receiver (rover and base) is stored independently.

Table 1: Summary of Extracted Data from Rinex v2.11 Observation and Navigation Files

Data File	File Structure	Sample Extracted Important Data Content
Observation	Header	Approximate XYZ Position [of Receiver] Antenna: Delta H/E/N # and Types of Observation [and Their Order]
	Data Body	Epoch/Satellite or Event Flag Epoch Observations [Including Observation, Loss of Lock Indicator, and Signal Strength]
Navigation	Header	Ion Alpha [if Provided] Ion Beta [if Provided] Delta-UTC: A0, A1, T, W [if Provided] Leap Seconds [if Provided]
	Data Body	PRN/Epoch/Satellite Vehicle Clock Errors [Bias, Drift, Drift Rate] 7 Broadcast Orbits

3.2 Satellite Position Computer

Having extracted information from both receiver's navigation files, the software then proceeds with computing the positions of the satellites which transmit broadcast information (through the navigation file). More specifically, the software only computes the satellite position, as seen by the base station. As the base is static, its data is expected to be of higher accuracy (less influenced by relativistic effects, movement and measurement noise, etc.)

Their position is determined by applying the formulas provide in Table 20-IV of the IS-GPS-200H specifications, available at [8]. The equations and variables from the broadcast orbits are summarized in the images below.

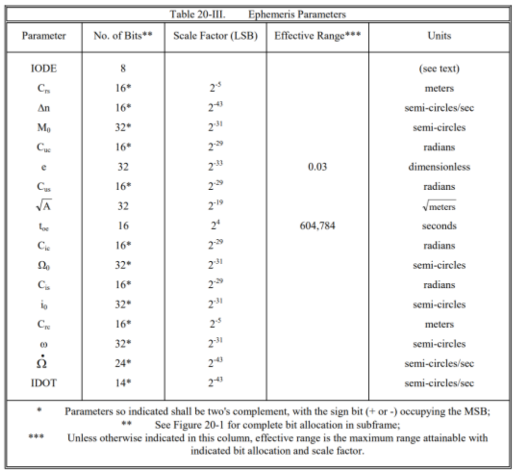


Table 20-III. Ephemeris Parameters

Parameter	No. of Bits**	Scale Factor (LSB)	Effective Range***	Units
IODE	8			(see text)
C_{rs}	16*	2^{-5}		meters
Δn	16*	2^{-43}		semi-circles/sec
M_0	32*	2^{-31}		semi-circles
C_{uc}	16*	2^{-29}		radians
e	32	2^{-33}	0.03	dimensionless
C_{us}	16*	2^{-29}		radians
$\sqrt{A}$	32	2^{-19}		$\sqrt{\text{meters}}$
t_{oe}	16	2^4	604,784	seconds
C_{ic}	16*	2^{-29}		radians
Ω_0	32*	2^{-31}		semi-circles
C_{is}	16*	2^{-29}		radians
i_0	32*	2^{-31}		semi-circles
C_{rc}	16*	2^{-5}		meters
ω	32*	2^{-31}		semi-circles
$\dot{\Omega}$	24*	2^{-43}		semi-circles/sec
IDOT	14*	2^{-43}		semi-circles/sec

* Parameters so indicated shall be two's complement, with the sign bit (+ or -) occupying the MSB;
** See Figure 20-1 for complete bit allocation in subframe;
*** Unless otherwise indicated in this column, effective range is the maximum range attainable with indicated bit allocation and scale factor.

Figure 6: Variables Extracted from Satellites Broadcast Orbits in Navigation Message [8]

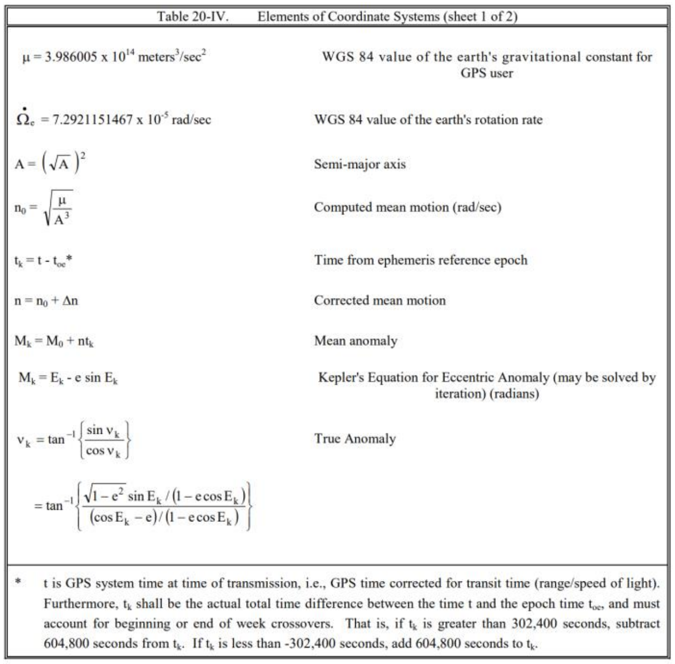


Table 20-IV. Elements of Coordinate Systems (sheet 1 of 2)

$\mu = 3.986005 \times 10^{14}$ meters^3/sec^2	WGS 84 value of the earth's gravitational constant for GPS user
$\dot{\Omega}_e = 7.2921151467 \times 10^{-5}$ rad/sec	WGS 84 value of the earth's rotation rate
$A = \left(\sqrt{A}\right)^2$	Semi-major axis
$n_0 = \sqrt{\frac{\mu}{A^3}}$	Computed mean motion (rad/sec)
$t_k = t - t_{oe}$*	Time from ephemeris reference epoch
$n = n_0 + \Delta n$	Corrected mean motion
$M_k = M_0 + nt_k$	Mean anomaly
$M_k = E_k - e \sin E_k$	Kepler's Equation for Eccentric Anomaly (may be solved by iteration) (radians)
$\nu_k = \tan^{-1}\left\{\frac{\sin \nu_k}{\cos \nu_k}\right\}$	True Anomaly
$= \tan^{-1}\left\{\frac{\sqrt{1-e^2}\sin E_k / (1 - e\cos E_k)}{(\cos E_k - e)/(1 - e\cos E_k)}\right\}$	

* t is GPS system time at time of transmission, i.e., GPS time corrected for transit time (range/speed of light). Furthermore, t_k shall be the actual total time difference between the time t and the epoch time t_{oe}, and must account for beginning or end of week crossovers. That is, if t_k is greater than 302,400 seconds, subtract 604,800 seconds from t_k. If t_k is less than -302,400 seconds, add 604,800 seconds to t_k.

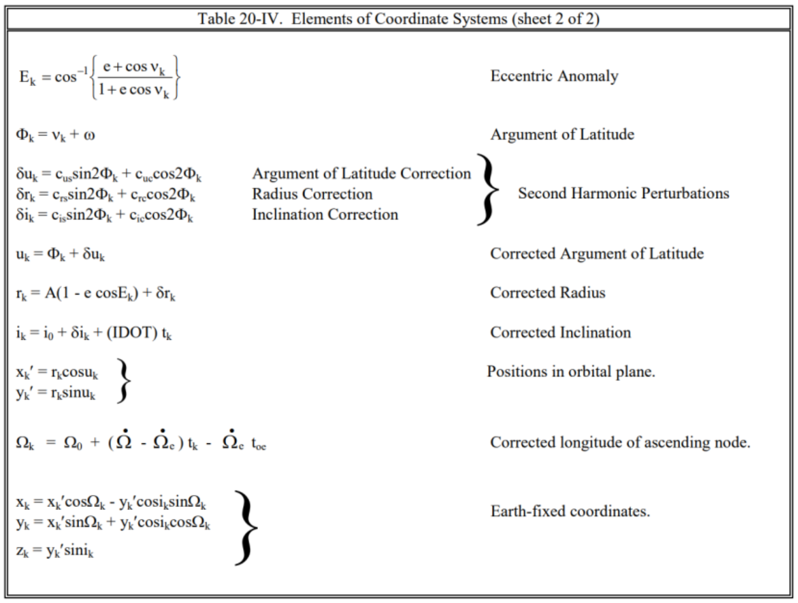


Table 20-IV. Elements of Coordinate Systems (sheet 2 of 2)

$E_k = \cos^{-1}\left\{\frac{e + \cos \nu_k}{1 + e\cos \nu_k}\right\}$		Eccentric Anomaly
$\Phi_k = \nu_k + \omega$		Argument of Latitude
$\delta u_k = c_{us}\sin 2\Phi_k + c_{uc}\cos 2\Phi_k$	Argument of Latitude Correction	Second Harmonic Perturbations
$\delta r_k = c_{rs}\sin 2\Phi_k + c_{rc}\cos 2\Phi_k$	Radius Correction	
$\delta i_k = c_{is}\sin 2\Phi_k + c_{ic}\cos 2\Phi_k$	Inclination Correction	
$u_k = \Phi_k + \delta u_k$		Corrected Argument of Latitude
$r_k = A(1 - e\cos E_k) + \delta r_k$		Corrected Radius
$i_k = i_0 + \delta i_k + (\text{IDOT})\, t_k$		Corrected Inclination
$x_k' = r_k \cos u_k$; $y_k' = r_k \sin u_k$		Positions in orbital plane.
$\Omega_k = \Omega_0 + (\dot{\Omega} - \dot{\Omega}_e)\, t_k - \dot{\Omega}_e\, t_{oe}$		Corrected longitude of ascending node.
$x_k = x_k'\cos\Omega_k - y_k'\cos i_k \sin\Omega_k$; $y_k = x_k'\sin\Omega_k + y_k'\cos i_k\cos\Omega_k$; $z_k = y_k'\sin i_k$		Earth-fixed coordinates.

Figure 7: Formulas to Determine Satellite Positions Given Broadcast Orbits [8]

When generating the software, to validate the correctness of the satellite position computations, the Euclidean distance of the satellite from the XYZ coordinates was computed. This distance was expected to be approximately 26600km from the geocenter, as GPS satellites orbit at an altitude of 20200km above Earth's surface and the earth's radius is approximately 6400km [9]. Additionally, an assessment of XYZ components was performed to ensure calculated positions were logical (ex. Z could not be negative, X and Y should not have been several orders of magnitude different from each other, etc.).

3.3 Observation and Navigation Data Combiner

Next, the software combines the observation and navigation data. More specifically, it determines the corresponding satellite position for each of the received observation messages/measurements.

The workflow of combining observation and navigation data in the software is summarized into the following steps:

1. Determine all epoch dates in Julian days (for both observation and navigation files)
2. Consider epoch of observation data and each of epoch's locked satellites (from which messages were received)
3. Iterating through each satellite PRN in the observation epoch, find indices in base navigation message data, were PRN is matching. Identify (previously) determined satellite position at these indices.
4. Calculate the differences between Julian date of the observation epoch and the Julian dates at each of the indices (where satellite is matching) in the navigation data
5. Matching satellite is the one where this difference is at a minimum. The satellite position at the index of this minimum difference is associated and stored with the observation epoch message for further ranging calculations

If a record of received measurement for a satellite exists for which there is no associated navigation data (at any epoch), the measurement is omitted from processing. For example, if there is a pseudo range measurement associated with satellite G21 in the observation file (at any epoch), but no broadcast message (at any epoch) in the navigation file, then the message is omitted since there is no determinable position for that satellite. It is not possible to perform ranging between the receiver and satellite for that observation. Considering this, the combiner also acts as a filter for extracting only the useful data from the two messages.

Ideally, the satellite position should be known at exactly at the point of transmission of the received message. This can be difficult to do, not only because simultaneous epochs can be rare but because there also exist relativistic effects (ex. motion of the satellite during transmission, etc.) [1]. However, using the above described method of differencing, provides suitable approximate coordinates for further processing.

3.4 Receiver Position Estimator (Using Code Ranges)

Next, the software determines the approximate receiver positions (for both base and rover stations) using the measured code ranges. First, epoch-by-epoch least squares adjustment methods are applied on the basic pseudo range (code) model. The code range model can be mathematically represented through

$$F_r^s = \sqrt{(X^s - x_r)^2 + (Y^s - y_r)^2 + (Z^s - z_r)^2} + c(dt_r - dt^s) \tag{3. 1}$$
$$F_r^s = \rho_r^s + c(dt_r - dt^s)$$

where, like above, XYZ^s are the satellite coordinates, xyz_r are the receiver coordinates (to be estimated), P_r^s is the measured psuedorange, dt^s is the known satellite clock error, dt_r is the receiver clock error (to be estimated), and *c* is the speed of light. The approximate/estimated receiver coordinates are then used when estimating the atmospheric delay correction terms, as they depend on the zenith view angle of the receiver to the satellite. At the time, no correction is applied for the ionospheric delay of the signal through the atmosphere. It is known that, to minimize the observation errors in the model and better estimate the receiver positions, the ionospheric delay should be considered and corrected for. Finally, considering the corrections, the receiver coordinates are once again estimated using the more complex observation model:

$$F_r^s = \rho_r^s + c(dt_r - dt^s) + dtropo_r^s \tag{3. 2}$$

Considering the additional error sources allows for a more accurate position to be found, which in turns becomes beneficial when substituting the approximate coordinates in the phase measurement models for relative positioning (discussed in the following sections).

3.4.1 Tropospheric Delay Correction Computer

The software makes use of the Saastamonein model to determine the refractivity of the transmitted signal in the troposphere. An extensive theory of tropospheric effects is omitted from this report. However, in short, it should be known that the troposphere causes signals to slow down as they pass through that region of the atmosphere and, as such, a corrective term must be added to the observation model. The Saastamoinen model determines the delay (in meters) through [5]:

$$\Delta Trop = \frac{0.002277}{\cos z}\left(p + \left(\frac{1255}{T} + 0.05\right)e - \tan^2 z\right) \tag{3. 3}$$

where *z* is the zenith angle of the satellite, *p* is the atmospheric pressure in millibar, *T* is the temperature in Kelvin, and *e* is the partial pressure of water vapor in millibars. The software is designed to require user input on these environment parameters.

3.4.2 Least Squares Position Estimator

The receiver positions and their corresponding clock errors for each epoch are estimated through a non-linear parametric least-squares adjustment. Considering the basic observation model in equation (3. 1), best estimates for the unknowns (denoted $\hat{x}$) are found by iteratively applying the below equations until the corrective term δ less than the defined threshold.

$$\hat{x} = x_0 + \delta \tag{3. 4}$$

$$\delta = (A^T PA)^{-1} A^T PAw \tag{3. 5}$$

x_0 denotes the approximate estimates for the unknowns, *w* is the misclosure vector (i.e. vector of errors between the measured code ranges in the epoch and those estimated with the approximate estimates for the unknowns), *P* is the weight matrix, and *A* is the first design matrix composed of the partial derivatives of each epoch observation equation with respect to the unknowns.

$$A = \begin{bmatrix} \frac{\partial F_1}{\partial x_r} & \frac{\partial F_1}{\partial y_r} & \frac{\partial F_1}{\partial z_r} & c \\ \frac{\partial F_2}{\partial x_r} & \frac{\partial F_2}{\partial y_r} & \frac{\partial F_2}{\partial z_r} & c \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial F_n}{\partial x_r} & \frac{\partial F_n}{\partial y_r} & \frac{\partial F_n}{\partial z_r} & c \end{bmatrix} \qquad w = \begin{bmatrix} F_1 - \rho_{0_1} \\ F_2 - \rho_{0_2} \\ \vdots \\ F_n - \rho_{0_n} \end{bmatrix} \qquad \hat{x} = \begin{bmatrix} \widehat{x_r} \\ \widehat{y_r} \\ \hat{z}_r \\ \widehat{dt_r} \end{bmatrix}$$

The software assumes equal observation weight and that no stochastic information is provided. Therefore $P = I$, otherwise known as the identity matrix [10]. With the consideration of the additional corrective terms in the complex model (2nd iteration of the least squares estimation process), the design matrix derivatives are adjusted according to (3. 2). Again, such estimation is applied independently on both the rover data and base station data.

Because the base station is static, its coordinates do change from epoch-to-epoch. Of course, different epoch estimates can be computed due to the errors in observations, corrective terms, etc. Thus, upon applying this epoch-by-epoch adjustment, the software considers the average of all computed coordinates and clock errors as the single parameter estimates for the base station receiver. These coordinates are considered the known base station coordinates, for further computations.

3.5 Receiver Position Estimator (Using Phase Measurements)

As mentioned previously, due to errors in estimating the float ambiguities for the ambiguity resolution and final least squares processing, the software is unable to correctly perform the latter portion of this functional block of the positioning method. It is therefore, not fully functional and cannot correctly compute results. Nevertheless, the methodology behind the processing is discussed as, given code modifications, the software has the potential to correctly implement relative positioning through this procedure.

Having approximate coordinates for the two receiver stations (both static and epoch-by-epoch), the software would apply relative positioning techniques using the epoch phase measurements to (better) estimate the epoch-by-epoch rover receiver positions. The positioning is performed through the following general steps:

1. Simultaneous epoch phase measurements between the base and rover stations are matched
2. Least Squares techniques are applied to estimate the double differenced float ambiguities and their variance-covariance, in addition to the baseline vector components
3. Ambiguity resolution is performed using open-source LAMBDA software in Matlab
4. Least squares techniques are once again applied to determine the best estimates for the baseline vector components
5. The best estimated rover position is found relative to the base station by adding the determined baseline vector to the latter coordinates

3.5.1 Rover and Base Station Phase Measurement Data Matching

To perform relative positioning, simultaneous phase measurements between the rover and base stations need to be considered. Similar as in the data combiner, the software identifies the base station phase measurement that corresponds to the phase measurement provided by the epoch satellite. The general (iterative) workflow is as follows:

1. Identify the epoch base station that is simultaneously observed with the rover epoch of interest.
2. Identify the satellites that are observed by both receivers at such epochs.
3. Extract the phase measurements of only the matching satellites of both receivers.
4. Extract the corresponding satellite positions for each matched satellite

First, the software identifies the index of the epoch of the base station observation data that is as close to the epoch (of interest) of the rover station observation data. That is, the difference between Julian dates of the rover and base observation epochs are computed and the position of the minimum difference is identified. This epoch is considered the 'simultaneous' measurement and the base station data is extracted. Next, the satellite PRNs of the two corresponding epochs are assessed. The PRNs observed in both datasets are identified and their transmitted phase message is extracted. If no phase measurements are recorded by either the rover or base, the satellite providing the data is removed from consideration. Lastly, the satellite positions of the remaining matched epoch satellites (at that particular epoch – as identified through the data combiner) are extracted.

3.5.2 Double Differencing Equation Former

Given rover and base phase observation matches, the software proceeds with forming the double differenced equations required for relative positioning at each epoch. Again, double differencing essentially differences between 2 single differences. It requires simultaneous measurements between two receivers to two different satellites.

The software forms the double difference equations by first assessing the single differences between the rover and base for each matched satellite PRN in the epoch. That is, re-considering equation (2. 7),

$$\lambda^j \phi_{AB}^j = \rho_{AB}^j + cdt_{AB} + \lambda^j N_{AB}^j + d\delta_{AB} + \epsilon_{AB}^j$$

ϕ_{AB}^j, N_{AB}^j, and ρ_{AB}^j terms are found. The software does not consider the single difference fractional receiver bias, as it would be eliminated through the double differencing process. Then, iterating through each possible j and k PRN combination of that epoch, the double difference equation for that combination is determined.

3.5.3 Non-Functional/Implemented Software Functions

Having the difference equations and matching components, the next step of the software would proceed by estimating float ambiguities, fixing these ambiguities, and applying the determined values to calculate the baseline vector (between rover and base) and, indirectly, the rover position.

The float ambiguities should be determined as explained above. Unfortunately, applying the least squares process, float ambiguities are estimated as negative valued. This is an immediate indicator that there is an error in the code, as (integer) ambiguities are a total count of the whole number of cycles transmitted from the satellite to the receiver. Being a count, it is not possible to be a negative value (a satellite cannot transmit a negative number).

Due to the failure of the float ambiguity estimation, the software is not developed further and, as such, fails to provide the final determined receiver position. The expected way it would have done so would have been to implemented open-source LAMBDA ambiguity resolution software to 'fix' the float integer ambiguity estimates and then use the best identified and validated value (through least square and ratio testing) to determine the baseline vector. Knowing the baseline, the rover receiver position would be indirectly determined by applying the baseline vector 'change' to the known base station coordinates.

As mentioned previously, calling the function from *Matlab* required the use of the Matlab Engine API in Python. This in turn, requires the use of Python 3.6 or older.

4 Additional Commentary on Software

4.1 Software Processing Inputs and Considerations

The (external) inputs required for the software to function is summarized in the table below:

Table 2: Summary of External Inputs Required for Software

Software Component	External Input	Other Comments
File Readers	Rinex v2.11 Observation Data File Rinex v2.11 Navigation Data File	From each receiver (base and rover)
Receiver Position Estimator (Using Code Ranges)	Data regarding troposphere at observation time (for atmospheric corrections step)	Includes variables *p*, *e*, and *T*
	Base Coordinates	(If known), otherwise computed from input base station files

As mentioned above, the software is designed around single-frequency receivers capable of only receiving signals transmitted from satellites of GPS constellation. More specifically, it only functions for signals transmitted on the L1 carrier (at a frequency of 1575.42MHz). Furthermore, it requires the receiver data (both observation and navigation) files to be logged in Rinex v2.11 format.

Because assumptions are made concerning a short baseline throughout the processing steps, the two receiver stations should not be separated by a great distance. As mentioned previously, in [5], Hofmann-Wellenhof suggests that atmospheric corrections at the two stations are approximately equal if the stations are separated by < 20km. As the accuracy of positions determined through relative positioning techniques degrades with further baselines (due to the introduction of errors), the receivers should be separated by even less than this.

Lastly, despite not applying atmospheric corrections during the relative positioning, the software also requires user to define troposphere related variables for the atmosphere correction applied when approximating the user position using code ranges. The software will prompt the user to input these variables when on that functional block. These inputs include:

p - atmospheric pressure in millibar

T - temperature in Kelvin

e - partial pressure of water vapor in millibars

4.2 Software Processing Outputs

Again, since not fully functional, the software does not yet provide any meaningful results. However, the main output of the software would be the XYZ – ECEF coordinates of the rover receiver at each epoch. In addition, the software would the provide associated variance-covariance matrices and residual vectors of the final least-squares adjustment processes. To assess the precisions, the software would provide data regarding the precision dilution factors (i.e. DOPs), including the Geometric Dilution of Precision (GDOP), Positional Dilution of Precision (PDOP), Vertical Dilution of Precision (VDOP), Horizontal Dilution of Precision (HDOP) and Time Dilution of Precision (TDOP). For understanding, the software would also provide the user with a plot of the rover and base station positions (for their visual analysis and confirmation).

5 Sample Data for Processing

5.1 Field Setup

Data was collected using 2 Emlid Reach RTK Modules. Both RTK modules consisted of a U-Blox NEO-M8T GNSS timing module and an Intel Edison processor. The Emlid Reach module was beneficial to use as it provided a user-friendly, online interface (titled *ReachView)* for observing the results of the data collection process, for each receiver. Through this interface, the parameters for the survey could be established and the raw data could be collected in the desired format. Each module also integrates *RTKLib*, so that a Real-Time Kinematic (RTK) positioning solution could be obtained directly in the field. Results of this RTK processing can be saved and compared to results of the generated software (not performed for this project, as it was desired to manually control the parameters and running of *RTKLib*).

The rover station made use of this module with a Tallysman TW4721 antenna, while the base station made use of a NovaTel antenna. Errors discovered with past use of the module suggested that the Tallysman was not suitable as an antenna for the base station. That is, it introduced errors into the observations which degraded the position of the base and, in turn, the relative position (as errors would propagate into the rover receiver position computations). To minimize such errors, a different antenna was used.

Being single frequency and collecting data for only a short interval, it was expected that the position of the base station would not be highly accurate. Such errors would propagate into the relative position of the rover station and, as such, were considered in the final results of the relative positioning method.

5.2 Main Dataset

The main dataset, used for both software development and comparison of relative positioning and kinematic PPP (in the following section), consisted of a simple linear rover path and static base station, as shown in the figure below.

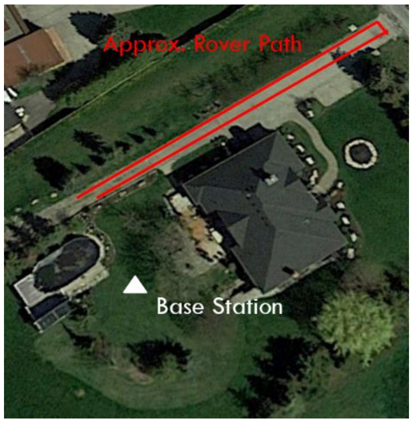


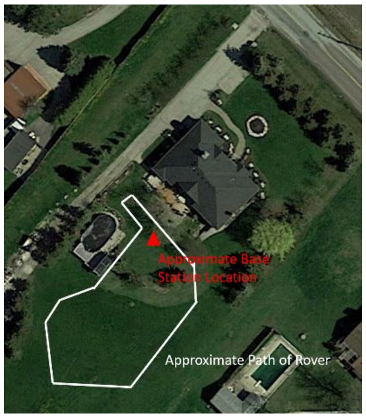
Figure 8: Approximate Rover Path of Main Dataset Used

The rover was held by hand above the user and was moved in an arbitrary path (like that above). The single-frequency (L1) data on each receiver was collected at a frequency of 1Hz. The elevation cut-off and signal to noise ratio masks for the two receivers were left at their default values of 10° (out of 30°) and 35 (out of 40). Data was collected by both the rover and base for slightly over 2 minutes. The base station remained static for the duration of the experiment. The distance between the rover and base station throughout the duration of all experiment was less than 70m. This ensured that similar atmospheric effects were impacting the two receivers and therefore similar corrective terms could be applied in the observation models. In the above scenario, the distance from the rover starting/ending positions to the base station was approximately 15m, while the length of one segment of the rover path was approximately 60m.

5.3 Additional Datasets for Positioning Technique Comparison

In addition to that above, a total of 3 other datasets were used to confirm findings on the positioning techniques. Two of the 3 were collected specifically for this project, while the 3rd consisted of data previously collected.

The former two datasets were configured in a similar manner as the previous, with the exception of different rover motion. In the first, the rover was moved in a path with clear open sky view. In the second, the rover was moved in a path more likely affected by multipath, as it was nearby a building and trees.



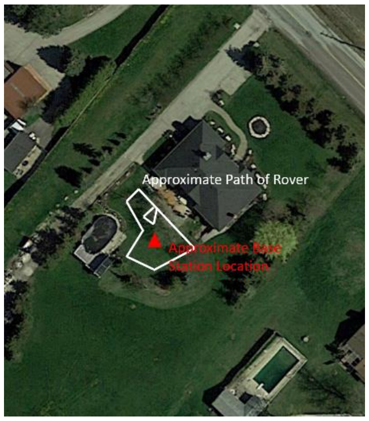


Figure 9: Additional Datasets Used for Comparison and Validation (Additional Data 1 – Left, Additional Data 2 – Right)

In addition to these recently collected datasets, a previously collected dataset (which used the same modules and antennas) was used for the comparison. Data was collected on York University campus. From previously determined results, it was known that the rover was moved the approximate following path.

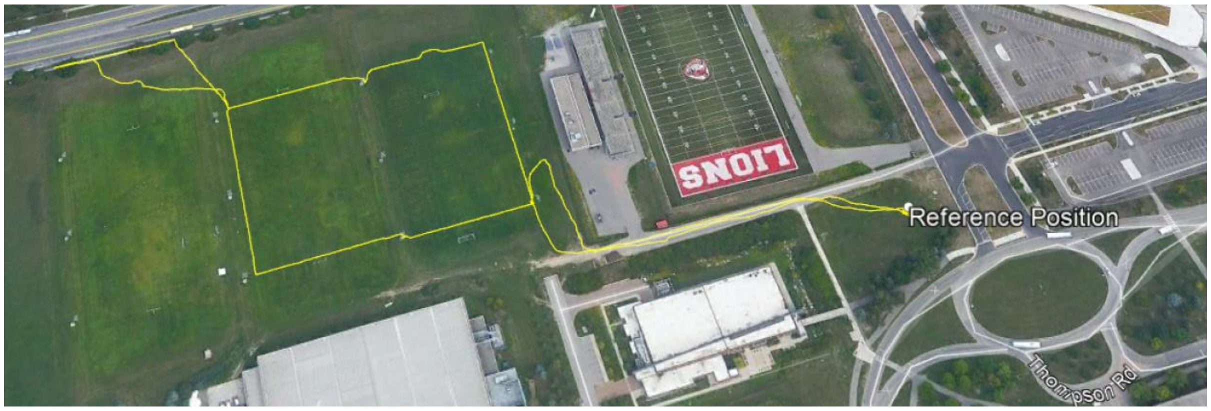


Figure 10: York University Campus Data Rover Path

Unfortunately, because the true path of the rover is unknown, only a comparison is made on the resulting accuracies of the two methods. The dataset was of interest for this project as it was collected for approximately 50 minutes and the base station was positioned at a known control point. This collection time was significantly greater than the previous 2 minutes and, therefore, could allow for an analysis on how station occupation time may impact results.

6 Comparison of Results using Post Processed Kinematic and Precise Point Positioning Techniques

As errors existed within the self-generated software, which prevented it from providing a positioning result, *RTKLib* was used to provide relative positioning results on the main dataset described above. To compute results using the kinematic PPP processing method, the *CSRS-PPP* tool provided by Natural Resources Canada’s Canadian Geodetic Survey (CGS) was used. The online tool can be accessed at the following link.

https://www.nrcan.gc.ca/earth-sciences/geomatics/geodetic-reference-systems/tools-applications/10925

This tool allows for the computation of higher accuracy positions of raw GNSS data by using '*precise orbit ephemerides to produce corrected coordinates of a constant "absolute" accuracy*.' [11]. It was necessary to use this external software, as *RTKLib* presented issues with processing the datasets using the Kinematic PPP method. In addition to positional results, both software provided useful information for each computed rover position. This included, any detected cycle slips, errors in processing, and the accuracies of positional results.

6.1 Software Configurations

For a representative comparison of results between the various processing methods, it was necessary to ensure that their settings were equivalent (or as similar as possible).

RTKLib was configured as suggested by the link:

https://docs.emlid.com/reach/common/tutorials/gps-post-processing/

Few modifications were made, as per the configuration of the receiver during the field survey and the self-generated code. The positioning mode was set as Kinematic (for all moving rover cases, i.e. static for static case), the elevation mask as 10°, and the atmospheric correction models were set to be Broadcast (Klobuchar) and Saastamoinen for ionosphere and troposphere, respectively. Because the software required the coordinates of the base station to be known (unlike the self-generated code), the approximate coordinates of the station provided in its observation file header data were input as the coordinates. It was expected that this may introduce additional errors into the results and were further considered when comparing results. All other parameters were kept at their default value.

As the PPP tool provided by the Government of Canada, required data to be 'sent-away' for processing, the number of controllable processing settings was limited. However, information regarding the configuration was stored in the results output file and considered during the comparison.

6.2 Main Dataset Results and Discussion

The *RTKLib* software provided results in either the XYZ-ECEF coordinate system or geodetic (latitude, longitude, ellipsoidal height) system. The *CSRS-PPP* tool only provided results in the geodetic coordinate system. As the angular (geodetic system) is not intuitive and easy to understand, results were converted into the ECEF Cartesian system. A brief code was written to convert the geodetic coordinates to Cartesian and is included in the provided software package.

The output summary file of the CSRS-PPP tool suggested that the tool made use of both the rover's phase and code observations and considered an elevation cut-off angle of 7.5°. This lower cut-off angle suggested a greater volume for observation (i.e. greater conical area observed around the receiver), compared to the *RTKLib* and receiver 10° angle. However, since the receiver was configured to have such a cut-off angle, the additional 3.5° in the kinematic PPP was not expected to affect results, as no additional measurements could be possible.

The *CSRS-PPP* tool identified the satellite sky distributions as in the figure below. This plot showed the track of each satellite in the sky relative to the receiver antenna (or determined position).

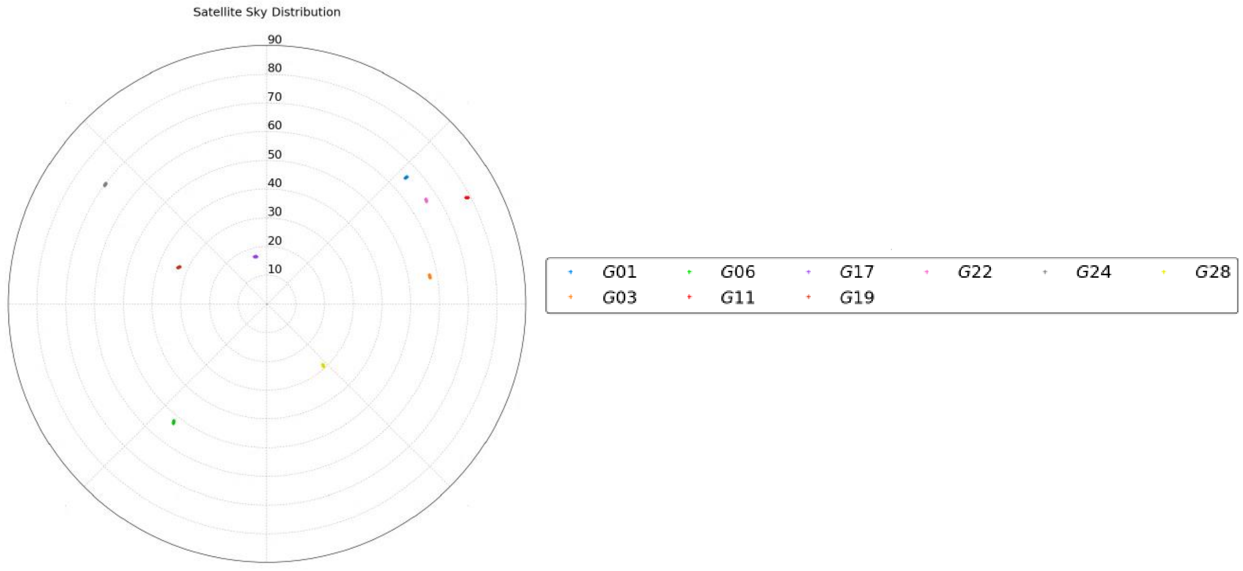


Figure 11: Satellite Skyplot Identified Using CSRS-PPP Tool

In general, a reasonable satellite geometry is observed. Satellites appear to be positioned entirely around the receiver. The greater number of satellites in the upper right quadrant compared to other quadrants may bias the results slightly. Little motion is observed, due to the short observation period (of approximately 2 minutes). Although these results were specific to the PPP solution, similar skyplot results are expected for the relative positioning.

The ground path/track of the dataset's rover, identified through processing with *RTKLib* and the *CSRS-PPP* tool (after the coordinate conversions), is shown below. For understanding, coordinates are provided in the common UTM Zone 17N coordinate from, for reader orientation. Unfortunately, a conversion to a local coordinate system (that is, with ENU directional components could not be made).

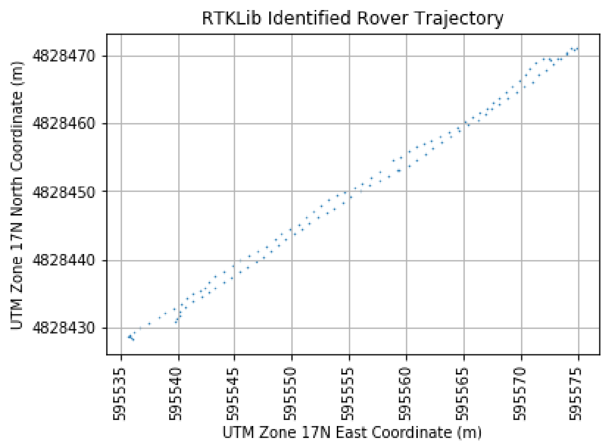


Figure 12: Ground Track of Main Dataset Identified by RTKLib (UTM Zone 17N Coordinates)

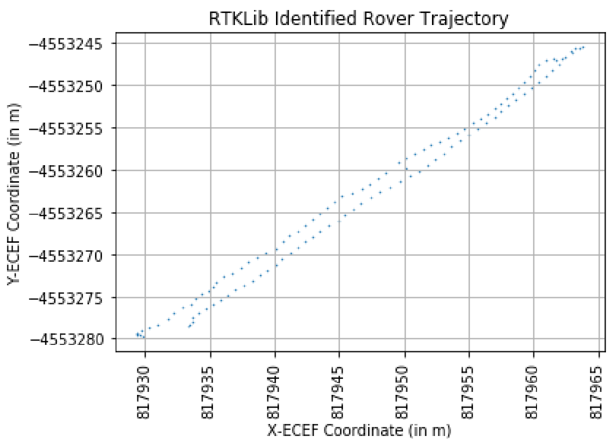


Figure 13: Ground Track of Main Dataset Identified by RTKLib (ECEF Coordinates)

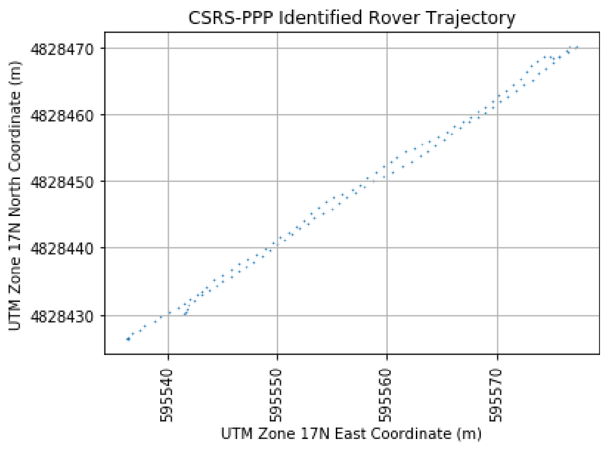


Figure 14: Ground Track Identified by CSRS-PPP (UTM Zone 17N Coordinates)

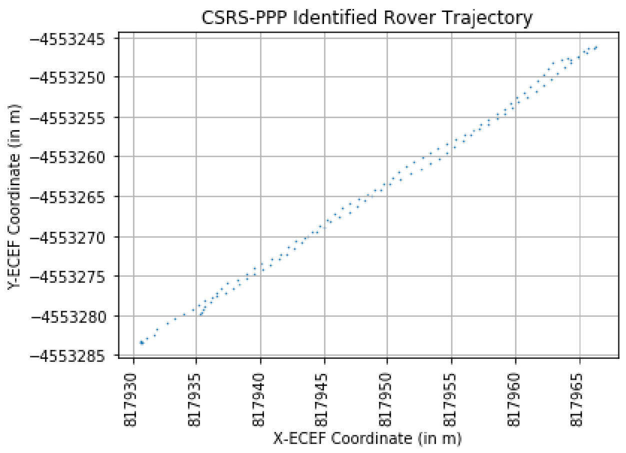


Figure 15: Ground Track of Main Dataset Identified by CSRS-PPP (ECEF Coordinates)

The two plots share a close resemblance. That is, they both followed the expected linear path, as shown in Figure 8: Approximate Rover Path of Main Dataset Used. Of immediate notice is the slight difference in sale of the two paths.

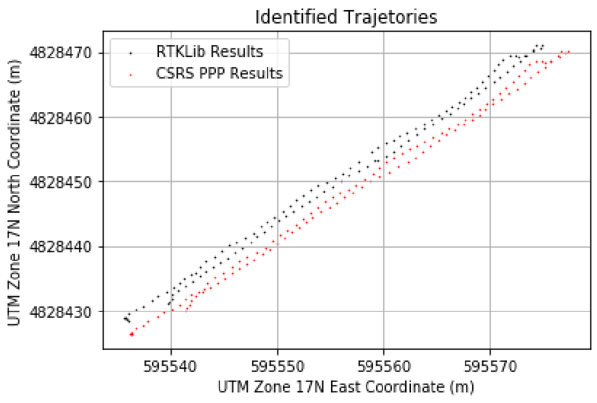


Figure 16: Comparison of Resulting Main Dataset Trajectories

A shift (of approximately 5m) if observed between the two results. Comparing the two plots in Google Earth:

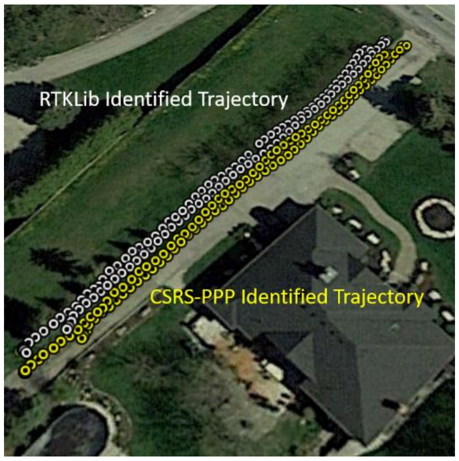


Figure 17: Identified Main Dataset Trajectories in Google Earth

The identified trajectory by the CSRS-PPP tool seems to be a better approximate to the true path taken. Given that results of the relative positioning technique tend to be more accurate than those of the PPP technique, such results are odd to observe. It was expected that the positions determine by the *RTKLib* tool would be much better than those of the *CSRS-PPP* tool. However, considering the field setup, this contradiction can be supported. As mentioned above, a single frequency base station receiver was used, and its position was determined from only a short observation interval. The determined position of the base station was likely incorrect, even more so given that it was taken as the approximate position given in the header of the RINEX file when imported into *RTKLib*. As the determined relative position of the receiver can only be as good as the determined position of the base station, the rover positions are also poor. This is because the errors propagated through the differencing model. Because the true coordinates of the base station were not known (it was not set up over a known control point), an analysis could not be performed on the difference to confirm that this is the reasoning. Since, PPP techniques do not make use of a ‘user’ base station and instead use well-defined reference stations, such base position errors do not propagate into the receiver position.

By considering the difference in each positional component between the two sets of results as:

$$Difference_{X,Y,Z} = CSRS\_PPP_{X,Y,Z} - RTKLib_{X,Y,Z} \tag{6. 1}$$

a statistically analysis was performed. Results are summarized in the table below.

Table 3: Statistics on Differences in Components of Results of Main Dataset

Component	**Minimum Difference Value (m)**	**Maximum Difference Value (m)**	**Mean Difference (m)**	**Standard Deviation of Difference (m)**	**Variance of Difference (m)**
X	-4.9593	9.0825	1.9395	5.7960	33.5933
Y	-7.9516	5.9581	-1.7251	4.9378	24.3815
Z	-5.7396	5.5202	-0.0079	5.7960	21.2383

Plotting differences in components for each epoch (result) yielded:

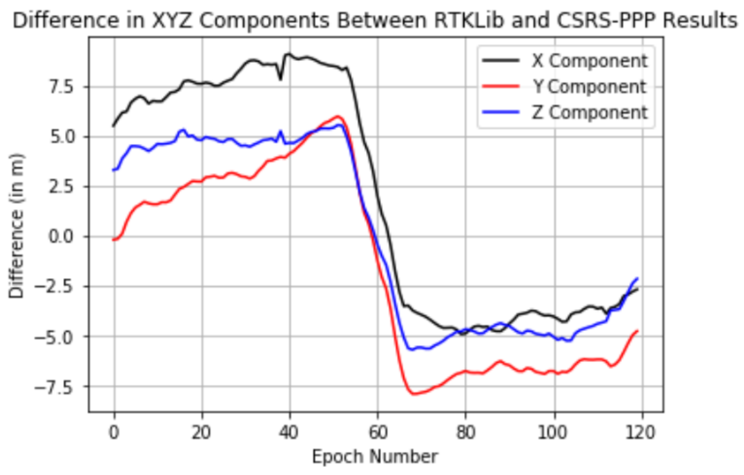


Figure 18: Differences in Components Between RTKLib and CSRS-PPP Results on Main Dataset

In general, a rather large range for (or spread of) difference occurs. Based on the applied difference equation, it is observed that the Y component in the PPP result is, on average, estimated as a lower value than that of the *RTKLib*, whilc the X and Z components are estimated as greater. Furthermore, a common trend is observed between all 3 components. That is, the differences decrease rather greatly around the 60th epoch. Unfortunately, the reasoning behind this change is still unknown. Again, the reasoning of the difference between the *CSRS-PPP* and *RTKLib* results could be explained by the use of poor base station coordinates in the relative positioning computation (and therefore poor position results of the receiver).

When running the software (expecting outputs in the latitude/longitude format for *RTKLib*), standard deviation values in the North (N), East (E), and Up (U) directions were provided for each estimated epoch position. These values were as follows:

Table 4: Main Dataset Standard Deviations of Two Methods

Standard Deviation Component	***RTKLib* Value**	***CSRS-PPP* Value**
σ_N	0.3381m	0.7205m
σ_E	0.2901m	0.3754m
σ_U	0.6237m	1.0350m
σ_{dt_R}	N/A	2.560290833

It should be noted that, since double differencing removed the receiver clock offset term, it is not estimated in the positioning process like it is in all other methods, and therefore does not have an associated estimation error.

It is observed that the average *RTKLib* standard deviations in each directional component are of lower magnitude than those of the *CSRS-PPP* tool. Despite not positioning the receiver close to the true path, we observe a greater 'confidence' in each computed epoch for the relative positioning method, than with the precise point positioning method. Although accuracies are not within the expected ranges of mm-cm (instead fall within the dm range), such results agree with the expectation that relative positioning techniques in a post-processed environment have positional accuracies that are better than precise point positioning techniques. Also, such results further support the hypothesis that the errors in the estimated position of the receiver arise from the ‘relative’ aspect of the technique. That is, from errors propagated from the base station rather than the processing technique.

6.3 Results of Additional Datasets

The standard deviation values obtained from processing the additional data collected specifically for the project were:

Table 5: Average Standard Deviations of Epoch Rover Positions of Additional Dataset 1

Standard Deviation Component	***RTKLib*** **Value**	***CSRS-PPP*** **Value**
σ_N	0.3207m	0.9999m
σ_E	0.2561m	0.7290m
σ_U	0.5768m	1.772m
σ_{dt_R}	N/A	4.6035nm

Table 6: Average Standard Deviations of Epoch Rover Positions of Additional Dataset 2

Standard Deviation Component	***RTKLib*** **Value**	***CSRS-PPP*** **Value**
σ_N	0.4354m	2.0691m
σ_E	0.3351m	1.59441m
σ_U	0.8494m	3.8423m
σ_{dt_R}	N/A	10.1637nm

As in the main dataset, rather large deviations for the processing methods are observed. Interestingly, a significant increase in the deviations is observed in the last additional dataset. Upon further investigation, it was found that this dataset consisted of the least number of (processed) epochs – at only 70 epochs. The main dataset consisted of 120 epochs, while the first additional dataset consisted of 147. Such findings suggest that there is a correlation between the number of epochs (or observation length) and the positional accuracies/results. With similar observation numbers, the main dataset and first additional dataset share similar standard deviation values.

Studying the *RTKLib* output file for the dataset likely affected by multipath and with ‘obstacles’ blocking satellite visibility, it is observed that there is constant fluctuation between the number of satellites in view – ranging from 5 to 8. Fewer measurement per epoch measurements, can explain why the positional accuracy is degraded. On the contrary, the first additional dataset (with most of the path having a clear sky), has a more constant number of satellites in view – ranging from 6 to 9. The main dataset seemed to have a constant 8 satellites, with only a minimal number of deviations.

The standard deviation values obtained from processing the data collected on York University campus were:

Table 7: Average Standard Deviations of Epoch Rover Positions of York University Dataset

Standard Deviation Component	***RTKLib*** **Value**	***CSRS-PPP*** **Value**
σ_N	0.0471m	0.2191m
σ_E	0.0457m	0.2096m

σ_U	0.0948m	0.4697m
σ_{dt_R}	N/A	1.2092ns

The decrease in magnitude compared to the self-collected datasets is immediately observed. Results suggest that the overall quality of measurements of the recently collected datasets is poor, thus resulting in poorer accuracies. Knowing that the base station was occupied for longer, base station coordinates had a greater time to converge to the ‘correct’ position. With this better position estimate, less errors propagate into the receiver position.

Furthermore, it is noticed that, unlike in the self-collected datasets, the YorkU dataset results contains epochs which have had fixed integer ambiguities estimated. That is, the recently collected datasets were not able to have their ambiguities resolved during their processing. Instead, they have their determined receiver position based on float estimates. This could be a crucial explanation for why this dataset has higher accuracies compared to all others. With integer ambiguities, the observation model is constrained (as it should be) and a correct phase measurement model can be used to position. With only float estimates, the model applied in the position determining equations is incorrect and therefore an incorrect position will be determined. From a logical perspective, the errors associated with each determined position would then have to be greater to encompass the correct position.

7 Conclusion

In summary, it was attempted to create a relative positioning software that determined a GPS receiver’s position using only L1 signals. Unfortunately, due to errors in the software’s ambiguity estimator, which solved for the double difference equations of relative positioning, output negative float ambiguity estimates values for the float ambiguity estimate. This issue could not be resolved in time for project submission, and as such the code remains non-functional.

Although a fully functional relative positioning software was not developed, a comparison between relative positioning and precise point positioning methods was still made. Relative positioning methods in a post-processed environment are expected to produce results with accuracies ranging between the centimeter and millimeter level, while precise point positioning methods are expected to produce results in the decimeter to millimeter level [7]. Of course, such accuracies also depend on the type of equipment being used, accuracy of corrective terms in their respective mathematical models, etc. In this project, Emlid Reach RTK modules were used. Such a module is a low cost GNSS receiver that claims to provide positions (by applying relative positioning techniques) with centimeter level accuracies [12]. Unfortunately, rather poor accuracies were observed using both methods on self-collected data. In these cases, contrary to expectations, the kinematic PPP technique positioned the rover receiver closer to the true trajectory. This was determined to be likely due to the propagated errors from the base station coordinates into the receiver in the relative positioning technique. Despite positioning more correctly in the self-generated dataset cases, the accuracies of the kinematic PPP technique were poorer than those of the relative positioning technique. Although not meeting the expectations in terms of magnitude, results agree with the belief that relative positioning results are more accurate. In the case of the provided York University data, accuracies that fell within the expectation (magnitude) levels for both relative and precise point positioning were determined. This difference in positional accuracies of the datasets was expected to be a result of the better configured field survey.

8 References

[1] S. Bisnath, "Lecture 2: GNSS Error Sources," York University, Toronto, 2019.

[2] B. Hofmann-Wellenhof, H. Lichtenegger and J. Collins, Global Positioning System, 5th, Revised Edition ed., Graz, Rockville, Styria, Maryland: Springer-Verlag Wien, 2001.

[3] S. Bisnath, "Lecture 10: Relative Positioning," York University, Toronto, 2018.

[4] S. Bisnath, "Lecture 9: Ambiguity Resolution," York University, Toronto, 2019.

[5] B. Hofmann-Wellenhof, H. Lichtenegger and E. Wasle, GNSS - Global Navigation Satellite Systems, Graz: SpringerWienNewYork, 2008.

[6] J. Wang, "Lecture: Differential GPS - Concepts, Mathematical Models, Accuracies and Applications," York University, Torono, 2019.

[7] S. Bisnath, "Lecture 10: Network RTK and PPP," York University, Toronto, 2019.

[8] "Global Positioning Sytsems Directorate Systems Engineering & Integration Interface Specification IS-GPS-200," 2014.

[9] S. Bisnath, "Lecture 1: Point Positioning," York University, Toronto, 2019.

[10] S. Bisnath, "Lecture 8 - GNSS Point Positioning," York University, Toronto, 2018.

[11] Government of Canada, "Natural Resources Canada," 22 02 2019. [Online]. Available: https://www.nrcan.gc.ca/earth-sciences/geomatics/geodetic-reference-systems/tools-applications/10925. [Accessed 14 04 2019].

[12] ArduPilot Dev Team., "Emlid Reach RTK GPS Receiver," 2019. [Online]. Available: http://ardupilot.org/rover/docs/common-reach-rtk-receiver.html. [Accessed 14 04 2019].