A Multi-GNSS Software Receiver

Chapter 7

A Multi-GNSS Software Receiver

Stefan Söderholm, M. Zahidul H. Bhuiyan, Giorgia Ferrara, Sarang Thombre, Heidi Kuusniemi

7.1 Introduction

In a software defined implementation, the main operations of a GNSS receiver, which have been described in detail in Chapter 1, are delivered by software, leading to a much more flexible design than in hardware architectures. This chapter presents the overall architecture of the multi-GNSS software receiver accompanying this book, the FGI-GSRx, used also in Chapters 2 to 6. In the sequel we will describe how acquisition, tracking and PVT solution computation are implemented, and then show some multi-GNSS processing experimental results obtained with a multi-constellation sample data set, which is also available to the reader as explained in Chapter 12.

The present software receiver is designed for post-processing operations and it does not support real-time processing. In other words, the FGI-GSRx processes stored IF samples that have been previously obtained through the use of a radio front end. The full GNSS receiver chain using the FGI-GSRx is shown in Figure 7.1 [1, 2]. The RF front end converts, in real-time, the received analog RF signals to digital IF data. These samples are stored in memory for later post-processing with the FGI-GSRx, which performs acquisition, code and carrier tracking, navigation bit extraction, navigation data decoding, pseudorange estimation, and position computation.

Within the receiver, the baseband processing block is responsible for processing the stored IF samples in order to extract navigation data and provide pseudorange estimates that are then utilized by the position computation block. The receiver is capable of performing acquisition and tracking of signals from GPS, GLONASS, Galileo, BeiDou and NavIC satellites. After the baseband processing block, the multi-constellation navigation solution block computes the multi-constellation position, velocity and time.

In the following sections, the acquisition, tracking, data decoding, and multi-GNSS position computation, are explained. In particular, we refer to the different configurable parameters in the receiver for each stage. In order to relate the theory of this chapter with the practical implementation in FGI-GSRx, the reader can refer in parallel to Chapter 12, where the MATLAB configuration files and variables are presented, and the way to modify them to obtain different outcomes.

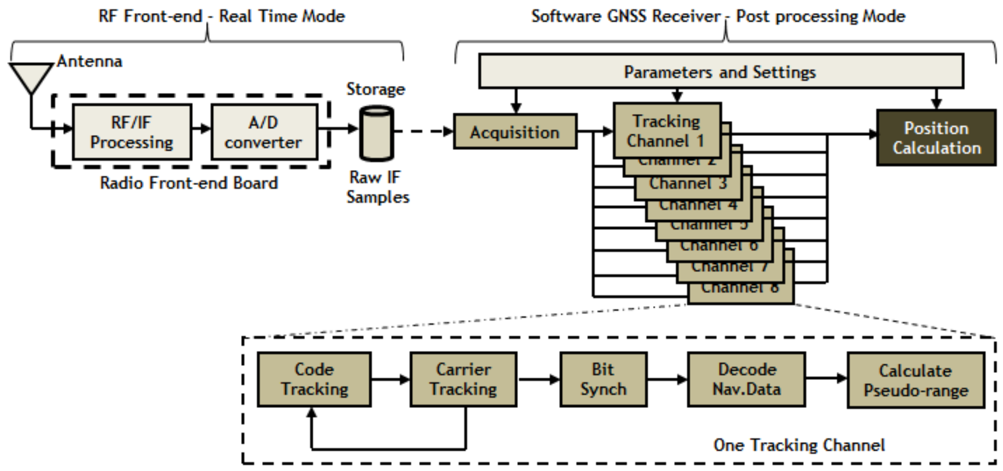

Figure 7.1: GNSS receiver chain using FGI-GSRx as post-processing module.

7.2 Multi-GNSS Signal Acquisition and Tracking

The receiver acquisition engine performs FFT-based parallel code search to identify visible satellites and obtain coarse estimates of their respective Doppler frequency and code delay values to initialize the individual tracking loops. The FFT-based parallel code delay search process was already described step by step at the end of Section 1.6.

For each navigation system, the search window, number of coherent and noncoherent rounds and detection threshold are all configurable parameters. Moreover, the user can also specify the PRN numbers of the satellites to search for. The acquisition is done for one navigation system at the time.

After acquisition has been completed for all systems, the results are handed over to initialize the individual tracking loops for all the acquired satellites. An FLL-assisted PLL for carrier signal tracking and a DLL for PRN code tracking are assigned to each visible satellite identified by the acquisition engine. Tracking is performed with a correlation interval of one code length duration, T_{code}, that can be different for each individual system (i.e., 1 ms for GPS, GLONASS, BeiDou and NavIC, and 4 ms for Galileo). The actual amount of data read from the file is adjusted for each epoch based on the true code frequency so that we always aim to process exactly one code epoch. Essentially, the code phase error is kept as close to zero as possible. In the software receiver, a two-quadrant arctangent discriminator based FLL is implemented, which is insensitive to bit transition, but it has reduced tolerance of frequency uncertainty coming from the acquisition stage. For the phase error, the traditional Costas loop two-quadrant arctangent discriminator is used. Finally, in the code tracking loop, a Normalized Early Minus Late Noncoherent Envelope discriminator is implemented. They are discussed in Section 1.7.

FGI-GSRx supports three tracking states: pull-in, coarse tracking and fine tracking, and the tracking architecture has been designed to be highly configurable with support for the different tracking states. Within one state, the tracking loops use parameters that can be different from other states. Different channels of the receiver can be in different tracking states at any instant. Also, each channel may transit freely from one tracking state to another depending upon the prevalent signal and environmental conditions.

The transitions between states are regulated by the frequency and phase lock indicators (Lock_{FLL} and Lock_{PLL}), which are implemented, respectively, based on [3] and [4]. For each indicator, two thresholds are used ($T_{\text{FLL-W}}$ and $T_{\text{FLL-N}}$ for the frequency lock indicator, and $T_{\text{PLL-W}}$ and $T_{\text{PLL-N}}$ for the phase lock indicator). The value of such thresholds can be configured by the user in the parameter file, with $T_{\text{FLL-W}}$

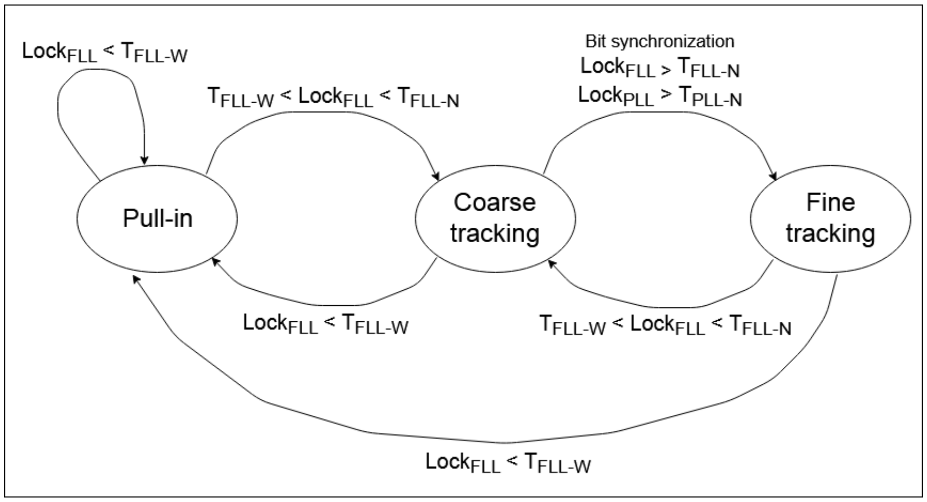

Figure 7.2: Tracking state transitions implemented in the FGI-GSRx multi-GNSS software receiver.

$< T_{\text{FLL-N}}$ and $T_{\text{PLL-W}} < T_{\text{PLL-N}}$. It is good to note that the subscripts 'W' and 'N' stand for 'Wide' and 'Narrow', respectively.

If the frequency lock indicator is smaller than $T_{\text{FLL-W}}$, the tracking is in 'pull-in' state, where a wide loop filter bandwidth is used for the carrier tracking, in order to enable faster convergence to the actual Doppler frequency. When the frequency lock indicator value is between $T_{\text{FLL-W}}$ and $T_{\text{FLL-N}}$, the tracking transitions into the more stable 'coarse tracking' mode, where a smaller loop filter bandwidth is used. If the frequency errors increase beyond the pre-defined threshold, the loop re-enters the 'pull-in' state. Furthermore, upon successful bit synchronization, which is possible only when the receiver is tracking the carrier with sufficient accuracy, and if both frequency and phase lock indicators are greater than their respective thresholds $T_{\text{FLL-N}}$ and $T_{\text{PLL-N}}$, the tracking state transitions to 'fine tracking' where a very narrow loop filter bandwidth is used for the carrier tracking. If at any instant the frequency and phase errors reach unsustainable levels, the probability of data decoding starts to degrade, which in turn causes a loss in bit synchronization. The tracking re-enters the more robust 'pull-in' mode, and the process repeats. The possible tracking state transitions are shown in Figure 7.2.

In order to accommodate the diversity of tracking states and also to ensure that new states can be introduced without significant modification to existing software code, a table-based tracking architecture is adopted which improves its modularity and flexibility. An example of 'pull-in' state tracking table for GPS L1 C/A signal is shown in Table 7.1. For a given tracking state, a set of functions is defined. Each function (column 1) defines the operation of a particular processing unit within the tracking engine, for example C/N_0 estimator, correlator, integrator, discriminator, loop filter etc. The functions at different tracking states may differ in the loop filter bandwidth used for the carrier tracking. The second column of the tracking table denotes the update interval of the function (in ms). Utilizing this kind of an approach, the tracking engine can easily switch between, for example, different kinds of loop filters and accommodate for pull-in, high sensitivity or high dynamic states of the tracking without the risk of unmanageable code.

In case of multi-GNSS receiver tracking, the parameter values may be different from one constellation to the other depending on how the user would like to configure each particular signal tracking. This offers extra flexibility to the user in case he or she would like to configure receiver tracking on per-constellation basis. Some key GPS L1 C/A, Galileo E1-B and BeiDou B1I signal tracking parameters are presented in Tables 7.2, 7.3 and 7.4 respectively.

Table 7.1: Example of 'pull-in' state tracking table for GPS L1 C/A signal.

Functions	**Update interval (ms)**
CN0fromSNR	1
freqDiscrimAtan2	1
freqLoopFilterWide	1
phaseDiscrim	1
phaseLoopFilterWide	1
bitSync	20
phaseFreqFilter	1
CN0fromNarrowWide	1
codeDiscrim	1
codeLoopFilter	1
gpsl1BitHandling	1
lockDetect	1
gpsl1UpdateChannelState	1

Table 7.2: GPS L1 signal tracking parameter settings in multi-GNSS receiver.

Parameter	**Value**	**Unit**
Correlator spacing	0.1	chips
Number of tracking states	3	N/A
Tracking states	[PULL_IN; COARSE_TRACKING; FINE_TRACKING]	N/A
DLL loop filter bandwidth	[1; 1; 1]	Hz
FLL loop filter bandwidth	[200; 100; 5]	Hz
PLL loop filter bandwidth	[15; 15; 10]	Hz

7.2.1 C/N_0 Computation

C/N_0 is used as a measure of strength for the received GNSS signal as already discussed in Section 1.7.2. The traditional C/N_0 estimation technique based on the Narrowband and the Wideband Power Ratio (NWPR), which is described in Section 1.7.3, is implemented in the multi-GNSS software receiver. Such technique works just perfectly for GPS L1 C/A signal. However, GNSS signals from other systems have different characteristics because of which they might no longer enjoy similar C/N_0 estimation performance that the NWPR-based technique offers for GPS L1 C/A. In particular, the performance degradation is caused by the frequent bit transitions due to a higher bit rate. For this reason, an additional C/N_0 estimation technique based on the Signal-to-Noise Power Ratio (SNPR) is implemented in FGI-GSRx.

An estimate of the noise power μ_N is obtained by correlating the incoming signal with a $+2$ chips early locally generated replica. The properties of the PRN codes suggest that the autocorrelation values with the same PRN codes outside ± 1 chip delay from the prompt correlator should either be zero or very close to zero due to the limiting length of the PRN codes themselves. A fair choice of $+2$ chips early correlation index is preferred, as this correlation index, $+2$ chips with respect to the prompt correlator, will have no impact from the multipath which usually comes as a delay in the late side of the correlation.

Table 7.3: Galileo E1-B signal tracking parameter settings in multi-GNSS receiver.

Parameter	**Value**	**Unit**
Correlator spacing	0.1	chips
Number of tracking states	3	N/A
Tracking states	[PULL_IN; COARSE_TRACKING; FINE_TRACKING]	N/A
DLL loop filter bandwidth	[1; 1; 1]	Hz
FLL loop filter bandwidth	[75; 35; 5]	Hz
PLL loop filter bandwidth	[15; 10; 10]	Hz

Table 7.4: BeiDou B1I signal tracking parameter settings in multi-GNSS receiver.

Parameter	**Value**	**Unit**
Correlator spacing	0.25	chips
Number of tracking states	3	N/A
Tracking states	[PULL_IN; COARSE_TRACKING; FINE_TRACKING]	N/A
DLL loop filter bandwidth	[1; 1; 1]	Hz
FLL loop filter bandwidth	[300; 100; 5]	Hz
PLL loop filter bandwidth	[15; 15; 10]	Hz

The signal plus noise power $\mu_{S,N}$ is estimated from the prompt correlation output. After each coherent integration period T, the estimate for the SNPR is computed as:

$$\widehat{\text{SNPR}}_i = \frac{\hat{\mu}_{S_i}}{\hat{\mu}_{N_i}} = \frac{\hat{\mu}_{S,N_i} - \hat{\mu}_{N_i}}{\hat{\mu}_{N_i}} \tag{7.1}$$

and C/N_0 is estimated as:

$$\left(\widehat{\frac{C}{N_0}}\right)_i \Big|_{\text{dBHz}} = \left|10\log_{10}\left(\frac{\widehat{\text{SNPR}}_i}{T}\right)\right|. \tag{7.2}$$

As presented in [5], the SNPR-based technique outperforms the NWPR-based estimator in case of higher data bit rate, which is the case of the BeiDou B1I GEO signal and the Galileo E1 signal.

7.2.2 Multi-Correlator Tracking

In the default configuration, only three correlator *fingers* are used for tracking and one finger for monitoring the noise level. The correlator spacing between the adjacent correlators is a configurable parameter. Additionally, the receiver has a feature called multi-correlator tracking, where the user can specify the number of correlators, correlator spacing between the fingers and correlator output rate (e.g., at 1 or 50 or a maximum of 1000 Hz rate). This feature is intended only for analyzing the cross-correlation function of the received signal in more detail and it does not interfere with the actual tracking. A typical multi-correlator function for a code delay window of ± 2 chips is shown in Figure 7.3. In this figure, there are altogether 17 correlators with a correlator spacing of $\delta = 0.25$ chips. The correlation function resembles very well with a BOC-modulated Galileo signal.

7.2.3 Data Decoding

The sign of the prompt finger is copied into the data decoding buffer and when the buffer is full the receiver correlates with the data frame preamble for the respective system. After successful correlation, the start of

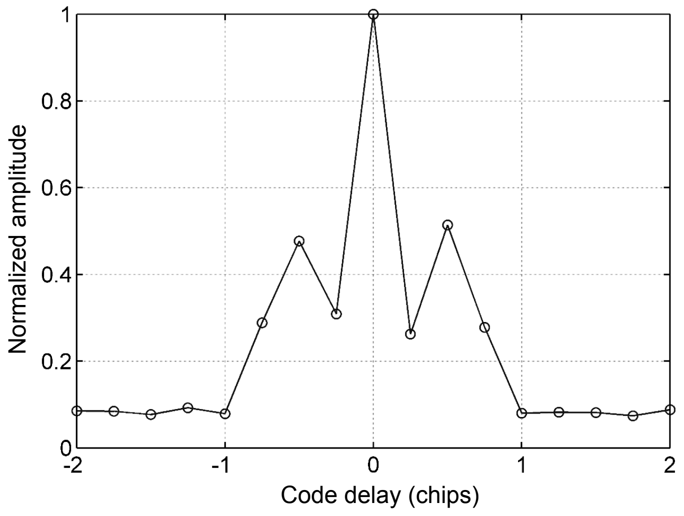

Figure 7.3: Example of Galileo E1-B correlation function on the BOC(1,1) contribution.

a data frame is found and the raw data bits can be extracted from the signal. The final decoded data contains the transmission time for the beginning of the data frame for each channel. Since we know the sample at which the data frame started, we can precisely link the transmission time for each signal to a specific sample count.

7.3 Computation of Multi-Constellation Position and Time

When tracking is completed, the position, velocity and time can be computed. The input is the decoded data and the observations for each channel from the tracking engine. Since the observations from each channel are only aligned with the bit edge of that channel at this stage, but these observations are not taken at the same time instant. The decoded data frame in each channel n will provide the transmission time T_n for the sample S_n that the receiver acquired at the beginning of the frame. In order to obtain synchronization, it is necessary to extrapolate the transmission time for all channels to one common sample S_0. This is shown in Figure 7.4.

The obtained transmission time for each channel T'_n refers then to the same sample S_0 in the incoming data. It is worth noting that since the transmission times are extrapolated from the time stamp in the data frames for each channel, they are in different time domains, namely GPS Time (GPST), GLONASS Time (GLONASST), Galileo System Time (GST), BeiDou Navigation Satellite System Time (BDT), and NavIC system time, depending on the signal which occupies that channel.

To obtain the initial receiver time estimate T_{rx} at sample S_0, it is assumed that the signal with the shortest travelling distance for each system has travelled for 80 ms. The accuracy of this receiver time estimate is not critical for the position solution and our time solution will give the final accurate receiver time. The estimated receiver time is a vector with as many components as the satellite systems. For example, when processing signals from GPS, Galileo and BeiDou, such vector is:

$$T_{\text{rx}} \doteq \begin{bmatrix} T_{\text{rx}}^{(\text{G})} \\ T_{\text{rx}}^{(\text{E})} \\ T_{\text{rx}}^{(\text{B})} \end{bmatrix} \tag{7.3}$$

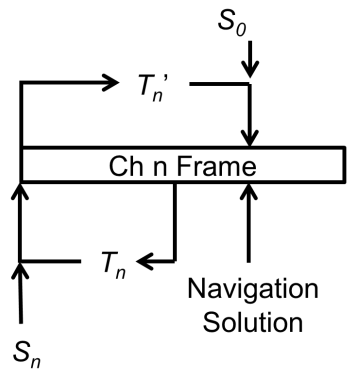

Figure 7.4: Transmission time T_n for channel n extrapolated to the sample at which we will compute our position. Note that T'_n is channel specific and system-dependent.

which are the estimated receiver times in GPST, GST, and BDT for the same sample S_0. The superindices in (7.3) indicate the satellite systems (G for GPS, E for Galileo, and B for BeiDou).

The pseudoranges $\rho_n^{(k)}$ can then be computed as

$$\rho_n^{(k)} = (T_{\mathrm{rx}}^{(k)} - T'_n)c \tag{7.4}$$

where n is the channel index, k the system index (GPS, Galileo, or BeiDou) and c is the speed of light. Note that, as we are discriminating between satellite systems, we use a slightly different notation than in Chapter 1.

7.3.1 Position Solution

Using an a priori estimate for the user's position $\mathbf{p}_{t-1} \doteq [x_{t-1}, y_{t-1}, z_{t-1}]^T$ such that $\mathbf{p}_t \doteq [x_{t-1} + \Delta x_t, y_{t-1} + \Delta y_t, z_{t-1} + \Delta z_t]^T$ for some coordinates increments $[\Delta x_t, \Delta y_t, \Delta z_t]^T$, and using the decoded ephemeris, it is possible to iteratively solve for the user position as already described in Section 1.10. In the present case, however, we need to extend the formulation to the multi-GNSS position solution. To this end the satellite positions and the estimated pseudorange between the user and each satellite, $\hat{\rho}_i^{(k)}$, are needed in order to form the observation vector (observed minus computed values),

$$\boldsymbol{\Delta}\rho \doteq \begin{bmatrix} \rho_1^{(\mathrm{G})} - \hat{\rho}_1^{(\mathrm{G})} \\ \vdots \\ \rho_n^{(\mathrm{G})} - \hat{\rho}_n^{(\mathrm{G})} \\ \rho_1^{(\mathrm{E})} - \hat{\rho}_1^{(\mathrm{E})} \\ \vdots \\ \rho_m^{(\mathrm{E})} - \hat{\rho}_m^{(\mathrm{E})} \\ \rho_1^{(\mathrm{B})} - \hat{\rho}_1^{(\mathrm{B})} \\ \vdots \\ \rho_k^{(\mathrm{B})} - \hat{\rho}_k^{(\mathrm{B})} \end{bmatrix} \tag{7.5}$$

In $\hat{\rho}_i^{(k)}$, the subscript i indicates the satellite number and superscript k indicates the satellite system: G for GPS, E for Galileo, and B for BeiDou. The observation vector is identical regardless of the number of systems we have. In the geometry matrix $\mathbf{A}$ containing the directional cosines we need to add one clock term for each enabled system. A typical row in $\mathbf{A}$ can therefore be written as:

$$[\mathbf{A}]_{i,:}^{j} = \begin{bmatrix} \frac{\delta x_i^j}{\hat{\rho}_i^j} & \frac{\delta y_i^j}{\hat{\rho}_i^j} & \frac{\delta z_i^j}{\hat{\rho}_i^j} & g & e & b \end{bmatrix} \tag{7.6}$$

with δX, δY and δZ are the differences between the satellite coordinates and the a priori user coordinates. The parameter $g = 1$ if $j =$ (G) else zero, $e = 1$ if $j =$ (E) else zero, and $b = 1$ if $j =$ (B) else zero. To obtain the updates to the a priori user position, it is necessary to solve the observation equations similarly to the procedure already described in Section 1.10. First of all, the observation vector is modeled to be linearly related to the vector $\mathbf{x}$ containing the updates from the previous to the current time instant,

$$\boldsymbol{\Delta}\rho = \mathbf{A}\boldsymbol{\Delta}\mathbf{x} \tag{7.7}$$

with

$$\boldsymbol{\Delta}\mathbf{x} \doteq \begin{bmatrix} \Delta x & \Delta y & \Delta z & c\Delta b^G & c\Delta b^E & c\Delta b^B \end{bmatrix}^T \tag{7.8}$$

containing the user's position increments from the a priori estimate to the current one, Δt^j are the different clock offsets for the three GNSS systems and c is the speed of light. Finally, the error vector ϵ is assumed to be zero-mean and the solution to (7.7) can be found through the Least Squares (LS) principle to be given by,

$$\hat{\boldsymbol{\Delta}\mathbf{x}} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\boldsymbol{\Delta}\rho \tag{7.9}$$

The a-priori user position estimate $\mathbf{p}_{t-1}$ is now updated by the estimated coordinate entries in (7.9):

$$\hat{\mathbf{p}}_t = [x_{t-1} + \Delta x_t, y_{t-1} + \Delta y_t, z_{t-1} + \Delta z_t]^T \tag{7.10}$$

The iteration (7.3)–(7.10) is repeated until the norm of the change in the estimated $\hat{\boldsymbol{\Delta}\mathbf{x}}$ is sufficiently small. Note that we have one additional unknown per additional GNSS due to the time offset, and therefore we need a measurement from an additional satellite per additional GNSS. In order to avoid that, a receiver can also use the inter-GNSS time offset broadcast in the navigation message, as for example the GGTO (Galileo-GPS Time Offset) in the case of Galileo. Notice also that, as shown in Chapter 1, Table 1.1, different GNSS give their orbits with respect to different reference frames, e.g. WGS84 for GPS and NAVIC, and GTRF for Galileo. The differences in reference frames may introduce an error in a multi-constellation position if the frame offset is not taken into account. However, this error is very small and generally it is neglected for code-based positioning.

7.3.2 Time Solution

The observations from all systems are aligned to the same sample count, but the receiver time is a vector such as the one in (7.3) with the time in each system separately. The navigation solution will provide us with the clock offset, Δt for each system as shown in (7.11) and we can accurately determine the true time for that particular sample count in each system's time domain by correcting the initial estimate with Δt.

$$T_{\text{rx}}^{\text{true}} = \begin{bmatrix} T_{\text{rx}}^{(\text{G})} - \Delta t^{(\text{G})} \\ T_{\text{rx}}^{(\text{E})} - \Delta t^{(\text{E})} \\ T_{\text{rx}}^{(\text{B})} - \Delta t^{(\text{B})} \end{bmatrix}. \tag{7.11}$$

The GPST is semi-synchronized to UTC time is such a way that the time difference can be defined as:

$$\text{UTC} - \text{GPST} = -\text{leap}_G + C_0 \tag{7.12}$$

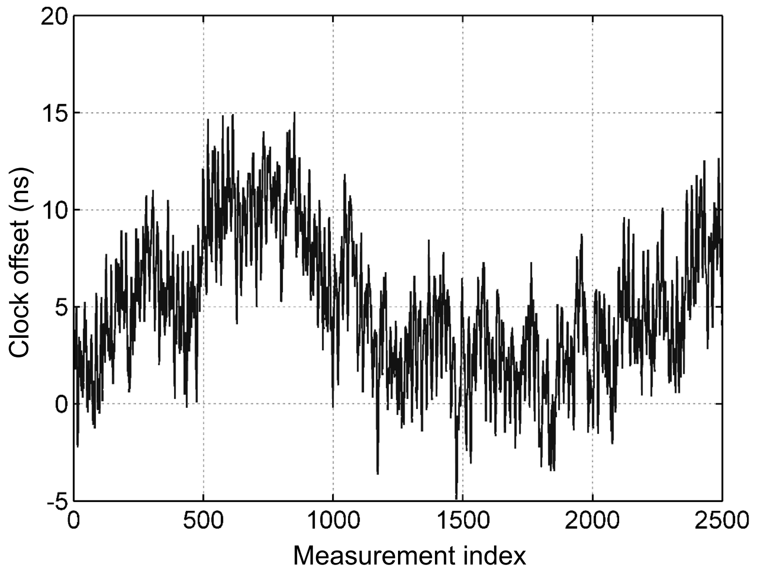

Figure 7.5: Difference between GPS and Galileo clock offsets in ns.

where $\text{leap}_{(\text{G})}$ is the number of leap seconds specified for a particular time and date. At the time of writing, the number of leap seconds for GPS was 18. The value of the constant C_0 is continuously monitored by the GPS ground segment and parameters for a UTC model are broadcast as part of the GPS almanacs. The value of C_0 is targeted to be less than 1 μs, but it is typically smaller than 100 ns. Galileo System Time (GST) is defined in a similar fashion:

$$\text{UTC} - \text{GST} = -\text{leap}_{(\text{E})} + C_1. \tag{7.13}$$

The number of leap seconds is the same for Galileo as for GPS and the difference between C_0 and C_1 is typically less than 50 ns. Similarly, BeiDou time is defined as:

$$\text{UTC} - \text{BDT} = -\text{leap}_{(\text{B})} + C_2. \tag{7.14}$$

The value of the constant C_2 is kept less than 100 ns and for BeiDou the number of leap seconds is 4. An example of time domain differences are shown in Figures 7.5 and 7.6. As shown in Figure 7.6 the difference between the two constants C_0 and C_2 for this test was about 190 ns.

7.3.3 Velocity Solution

The velocity solution is computed in an analogue fashion as the position solution. The observation matrix in this case is the difference between the observed Doppler frequency obtained directly from the PLL and the theoretical Doppler computed from the a priori user velocity and the satellite ephemeris. The geometry matrix $\mathbf{A}$ in (7.6) is the same as for the position computation, and the solution is obtained using (7.5)–(7.7). Instead of obtaining a position solution we obtain a velocity solution:

$$\mathbf{\Delta v} \doteq \begin{bmatrix} \Delta v_x & \Delta v_y & \Delta v_z & \frac{c}{L_G}\Delta f^{(\text{G})} & \frac{c}{L_E}\Delta f^{(\text{E})} & \frac{c}{L_B}\Delta f^{(\text{B})} \end{bmatrix}^T \tag{7.15}$$

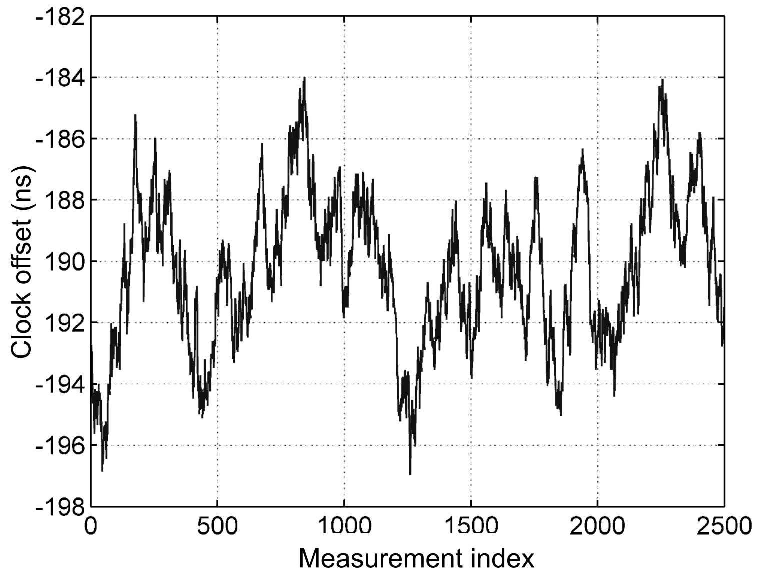

Figure 7.6: Difference between GPS and BeiDou clock offsets in ns. The leap second difference between GPS and BeiDou has been removed.

where c is the velocity of light, L is the center frequency for each system and Δf is the frequency offset from the nominal intermediate frequency. After every iteration we obtain the true user velocity and the frequency offsets for each system.

7.3.4 Experimental Results

This subsection shows multi-GNSS processing experimental results obtained with a multi-constellation sample data set which is available with the book as described in Chapter 12. The navigation solution is computed after every 100 ms interval, i.e. with a 10-Hz navigation update rate, as shown in Table 7.5. The navigation solution is further smoothed with 10 navigation solution samples per epoch. Statistics in Table 7.6 are generated with different combinations of GNSS constellations for a standalone static GNSS user with a 1-Hz navigation update rate. At the end of the processing, coordinate transformations, time corrections, and satellite elevation and azimuth angles are computed. The satellite elevation is used after the initial position estimate in order to omit satellites below a user defined cut-off angle. The update rate for the position computation is defined by the user. The default rate is 10 Hz, i.e. one position every 100 ms.

Analyses were performed with various combinations of satellite systems and the performance metrics are presented in Table 7.6. The number of satellites in view was 8 for GPS, 4 for Galileo, and 7 for BeiDou. This is reflected in the HDOP and VDOP values, which are higher for the cases with less satellites, leading also to higher errors. For the ionospheric corrections, the receiver used the default values, based on GPS's Klobuchar model [6].

From Table 7.6 we see the vertical component is offset relatively much for the Galileo-only solution and for the BeiDou-only solution which is most likely due to the geometry. This affects the vertical offset

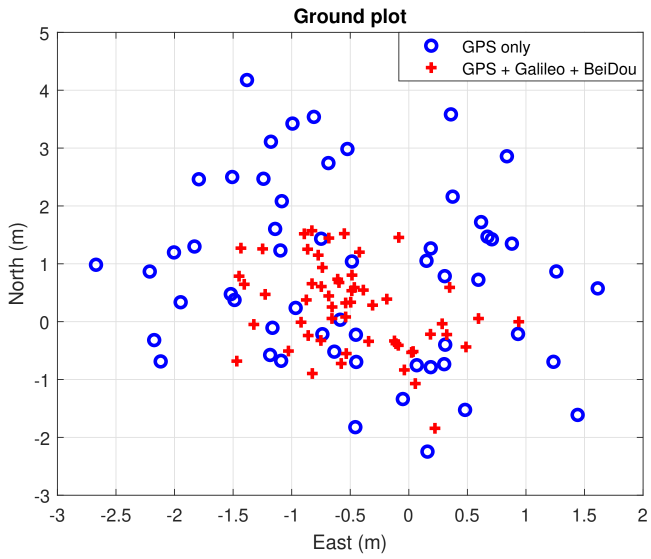

Figure 7.7: Horizontal plot of GPS solution based on 6 satellites versus a combined solution based on GPS & Galileo & BeiDou on 14 satellites

Table 7.5: Multi-GNSS navigation parameter settings.

Parameter	Value	Unit
Navigation solution period	100	ms
Number of smoothing samples	10	N/A
Elevation mask	5	degree
C/N_0 mask	30	dB-Hz

of the overall multi-GNSS solution. However, when using the three-GNSS solution, an improvement in the horizontal component is achieved, as it can also be seen from Figure 7.7. The addition of more GNSS generally increases accuracy, but this may not be the case if the added GNSS has higher measurement errors, which seems to be the case in this data capture particularly for the BeiDou case. Another interesting effect is that, with good visibility conditions, moving from two constellations with enough visible satellites (GPS and BeiDou in our case) to three (GPS, BeiDou and Galileo), does not greatly improve geometry, as seen in VDOP/HDOP values.

Notice that the purpose of the Multi-GNSS experiment is to show the reader what can be performed with the receiver, but not to optimize the position solution, which the readers are encouraged to do. These results also illustrate that, while an all-GNSS position can improve the position performance, it may not always be the case.

7.4 Front End and Other Practical Information

7.4.1 Front End

As explained in the previous Chapters 2 to 6, the dual-frequency Stereo front end from NSL was used to capture the real GNSS data. Among the two front ends in the Stereo, the MAXIM 2112 front end was configured to receive GPS L1 and Galileo E1 at 1575.42 MHz and the MAXIM 2769B was configured to

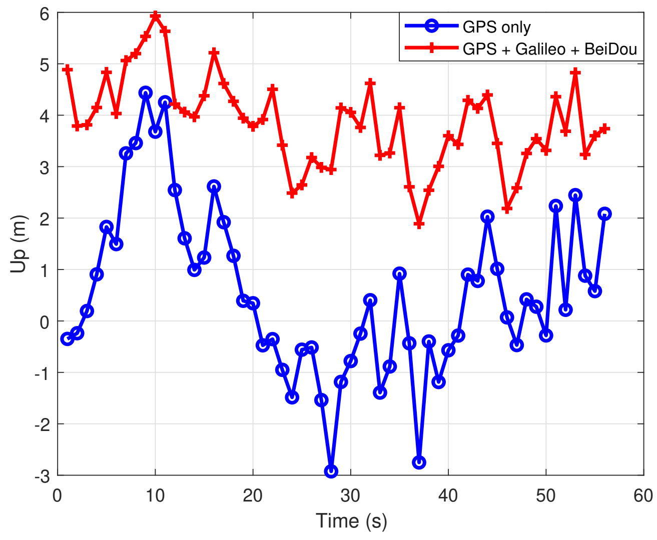

Figure 7.8: Altitude error of GPS solution based on 6 satellites versus a combined GPS & Galileo & BeiDou on 14 satellites.

Table 7.6: Horizontal and vertical statistics for various GNSS combinations.

GNSS Signal	Horizontal				Vertical				3D
	RMS (m)	Mean (m)	Max (m)	Mean HDOP	RMS (m)	Mean (m)	Max (m)	Mean VDOP	RMS (m)
GPS	1.84	1.84	4.39	1.38	1.89	1.26	4.43	1.25	2.64
Galileo	6.59	4.69	10.56	2.59	3.78	4.43	11.22	2.55	7.60
BeiDou	4.25	2.00	5.10	1.26	6.14	6.91	11.57	1.61	7.46
GPS & Galileo	1.43	1.39	3.25	1.20	2.02	1.47	5.07	1.12	2.48
GPS & BeiDou	1.78	1.48	2.93	0.91	3.61	3.55	5.44	0.96	4.02
Galileo & BeiDou	4.79	1.82	3.46	1.13	5.59	6.88	11.86	1.34	7.36
GPS & Galileo & BeiDou	1.95	1.05	2.03	0.86	3.50	3.73	5.75	0.90	4.01

receive BeiDou B1I signal at 1561.098 MHz, as mentioned in Table 7.7. The front end configurations used to collect GPS L1, Galileo E1-B and BeiDou B1I raw data samples are also given in this Table. The GPS L1 and Galileo E1-B signals are saved as real data with sample size of 8 bits, whereas BeiDou B1I signal is saved as complex data with sample size of 16 bits (i.e., 8 bits for Q, and 8 bits for I).

7.4.2 Data Collection

The signals were captured using a G5Ant-3AT1 active antenna located on the roof of the Finnish Geospatial Research Institute in Kirkkonummi, Finland.

GPS, Galileo and BeiDou processing was enabled in FGI-GSRx. A suitable time for the test was selected so that a minimum of 4 satellites were visible from all three systems. 60 s of data were logged and the positions were computed at a rate of 10 Hz. The ground plot and the altitude variations are shown in Figure 7.7 and Figure 7.8, respectively. Both figures show the GPS-only and multi-GNSS solution results.

Table 7.7: Configuration of the two front ends included in the NSL stereo v2 for collecting GPS L1, Galileo E1-B and BeiDou B1I raw data samples.

Parameter	**front end 1**	**front end 2**
Chipset	MAXIM 2769B	MAXIM 2112
GNSS Signal	BeiDou B1I	GPS L1, Galileo E1-B
Bandwidth (MHz)	8	10
Sampling Frequency (MHz)	26	26
Intermediate Frequency (Hz)	6500000	353
Number of quantization bits	2	3
Down-converted signal type	Real	Complex
IQ signal orientation	N/A	Q,I
Raw sample size (I+Q)	8	8 bits

7.4.3 MATLAB Configuration and Functions

The parameter file name for multi-GNSS receiver is 'default_param_MultiGNSS_ Chapter7.txt'. The user needs to pass this parameter file with the main function in the MATLAB command prompt as follows:

```
>> gsrx('default_param_MultiGNSS_Chapter7.txt')
```

In case of multi-GNSS signal processing, the enabled GNSS signals are 'gpsl1', 'gale1b' and 'beib1'.

References

[1] S. Söderholm, M. Bhuiyan, S. Thombre, L. Ruotsalainen, and H. Kuusniemi, “A Multi-GNSS Software-defined Receiver: Design, Implementation, and Performance Benefits,” *Annals of Telecommunications*, vol. 71, pp. 399–410, 2016.

[2] S. Thombre, M. Bhuiyan, S. Söderholm, M. Kirkko-Jaakkola, L. Ruotsalainen, and H. Kuusniemi, “A Software Multi-GNSS Receiver Implementation for the Indian Regional Navigation Satellite System,” *IETE Journal of Research*, pp. 246–256, 2015.

[3] C. Ma, G. Lachapelle, and M. E. Cannon, “Implementation of a Software GPS Receiver,” in *Proc. of ION GNSS*, pp. 21–24, Sept. 2004.

[4] B. W. Parkinson and J. J. Spilker, *Global Positioning System: Theory and Applications Volume I*. Washington, USA: American Institute of Astronautics, 1996.

[5] M. Bhuiyan, S. Söderholm, S. Thombre, L. Ruotsalainen, M. Kirkko-Jaakkola, and H. Kuusniemi, “Performance Evaluation of Carrier-to-Noise Density Ratio Estimation Techniques for BeiDou B1 Signal,” in *Proc. of Ubiquitous Positioning, Indoor Navigation and Location-Based Services Conference*, (TX, USA), Nov. 2014.

[6] J. A. Klobuchar, *Global Positioning System: Theory and Applications*, ch. Ionospheric Effects on GPS. American Institute of Aeronautics and Astronautics (AIAA), 1996.