Chapter 9
Implementation of BDS/GPS Dual-Mode Software Receivers

This chapter will discuss the implementation of the BDS and GPS dual-mode receivers in software according to the principles of the GNSS receiver that were explained in previous chapters. This implementation is the software radio receiver solution mentioned in Chap. 2. Compared with the hardware solution, the software solution is considerably more flexible, and is easier to reconfigure and upgrade, so it is ideal for beginners to learn principles of the GNSS receiver. For engineers, it can also be used as a development platform for the GNSS receiver, for related signal processing and algorithms.

The basic principles of the implemented software receivers in this chapter have been covered in the previous chapters. The essential task of software receivers is to implement the most important receiver principles on a computer through a specific programming language. In the implementation process of specific software, the choice of programming language should be carefully considered. In principle, any programming language can be used to implement the principles of GNSS software receivers. However, considering that the readership of this book mainly consists of students and researchers, MathWorks' Matlab scripting language (USA) has obvious advantages, as it is widely used in algorithm research, data visualization, data analysis, and numerical calculation.

This chapter will be divided into three parts. The first part explains the signal source (input signal) of the software receiver, which often takes the form of data files. The data stored in the signal source comes from the AD sampled quantized data of the RF front-end, and is stored as a data file in a certain format, so the principles of the RF front-end introduced in Chap. 8 will be referred to in this part.

The second part of this chapter deals with the code implementation of the software receiver. It explains the Matlab code module with a focus on the main functions of each source file and the signal processing principles involved, which are related to the principles of BDS and GPS signal, signal acquisition and tracking, telegraphy demodulation, satellite position and velocity calculation, and PVT solution. This will

Y. Lu, *BDS/GPS Dual-Mode Software Receiver*, Navigation: Science and Technology 10, https://doi.org/10.1007/978-981-16-1075-2_9

give readers a clearer and more practical understanding of the theoretical knowledge mentioned in the previous chapter.

The third part of this chapter presents the data processing result of the software receiver. The software receiver explained in the second part is applied to process typical BDS and GPS dual-mode IF data, and analyzes the signal processing results. The results of signal acquisition and tracking, and the results of the PVT positioning solution will be given in the form of graphs, offering readers a more visualized understanding of the internal principles of the GNSS receiver, signal processing flow, and various functional modules. At the same time, it can also help readers to understand the principles of GNSS receivers discussed in the first half of the book.

The source code of the software receiver in this chapter is given on the website that accompanies this book (https://www.gnssbook.cn). At the same time, there is a BDS and GPS dual-mode IF data file lasting about 90 s. This data file contains the signals from 15 BDS and GPS satellites. The time period of 90 s ensures that the GPS and BDS satellites can at least contain a complete set of ephemeris data. Readers can develop their own BDS and GPS software receivers and corresponding signal processing algorithms based on their understanding of the receiver's source code.

9.1 Signal Source for Dual-Mode Software Receivers

The source of the dual-mode software receiver comes from the IF data file. Since the IF data file comes from the sampled quantized data of the BD/GPS dual-mode RF front-end, the signal source of the dual-mode software receiver is also from the BD/GPS dual mode RF front-end. In fact, the logical relationship between the three is shown in Fig. 9.1.

The BD/GPS dual-mode RF front-end in Fig. 9.1 contains two RF signal channels, which process the BDS and GPS signals respectively, and complete a series of signal processing tasks such as mixing, filtering, down-conversion, and AD sampling

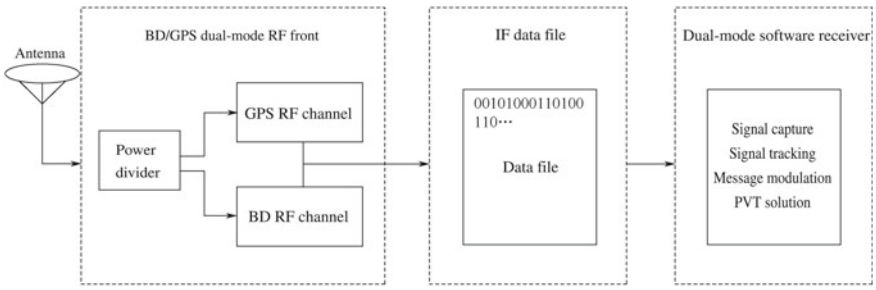


Fig. 9.1 The relationship between the signal input, IF data files, and dual-mode RF front-ends for dual-mode software receivers

quantization. The local clock ensures that the clocks of the BDS and GPS signals are strictly aligned. The two AD-quantized digital signals are stored as intermediate-frequency data files through the high-speed data interface, and then the IF data files are processed by the dual-mode software receiver. Figure 9.1 shows that the dual-mode software receiver still processes the signal of the hardware RF front-end, which is only processing the stored data file. It is not a real-time processing method. Signal processing algorithms have obvious advantages, and are widely used in the early stages of product development for receivers, as well as in scientific research.

In the implementation of the software receiver code, the format of the IF data file needs to be analyzed first to obtain the data sample value. The specific format of the IF data is determined by the AD quantization format of the RF front-end, and the AD output quantization formats of different RF front-ends are different from each other. Hence, only the BDS and GPS dual-mode IF data file formats attached to this book are described here. Readers can analyze data files stored by digital sampling output from other RF front-ends according to their hardware manuals.

The data file attached to this book is in binary format. Each byte contains two sets of sample data. Each set of sample data contains one sample of BDS and GPS. The quantization bit width of each sampled data is 2 bits. The detailed data format is shown in Fig. 9.2.

The dual-mode data format in each byte is shown in Fig. 9.2, where the data is stored in chronological order from high bit (MSB) to low bit (LSB), which readers should note. The reason is that when reading data at the software receiver, the data needs to match the order in which it was stored. Each data sample (GPS and BDS) is two-bit quantization. The high bit is the Sign bit (Sign) and the low bit is the Amplitude bit (Mag). For the specific quantization principle, please refer to Sect. 8.5. If the Sign is 0, the sample value is positive. Otherwise, the sample value is negative. If Mag is 0, the amplitude should be 1, otherwise the amplitude is 3. The specific meaning is shown in Table 9.1.

The IF data file in the software receiver has two main functions. The first is to provide IF sampling data, and the second is to provide the working clock of the core processing module. The first function can be easily understood, but the second is

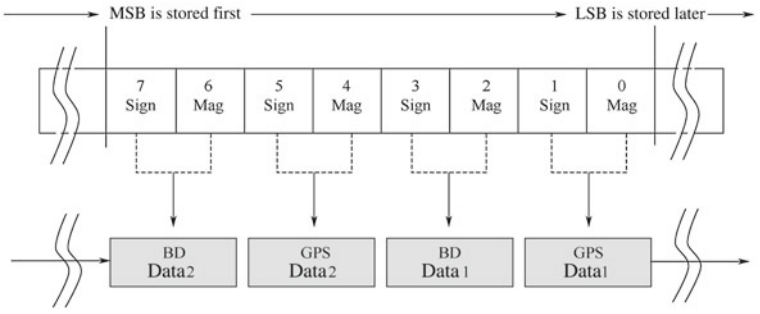


Fig. 9.2 IF data file format

Table 9.1 Mapping table of two-bit quantization and sample values

Sampling value	Sign bit	Mag bit
+1	0	0
+3	0	1
−1	1	0
−3	1	1

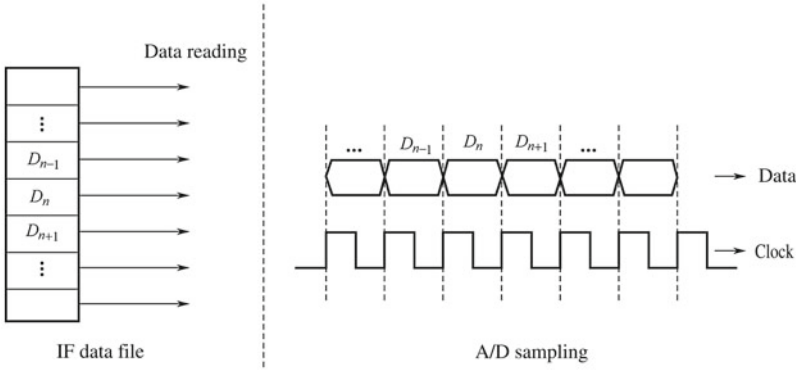


Fig. 9.3 Correspondence between the reading of data files and AD sampling

less simple. To understand it, we only need to grasp the specific process of ADC sampling. Each sampling clock leads to the generation of a sample. In the practical use of BDS and GPS receivers, this sampled data is sent to the subsequent hardware processing stage. However, in the software receiver, the sampled data is stored in the data file. When undertaking the signal processing of the software receiver, each time sample data is read from the data file, it also means the arrival of a sampling clock. This clock will cause each state update of the correlator. The integration of the clock is the local time change, which is used to update the local time.

The above analysis is shown in Fig. 9.3. The left side of the dashed line in the figure is the representation of the IF data in the computer file. Each reading of the data corresponds to the sampling process of the AD to the intermediate frequency signal in the hardware, which is shown to the right of the dotted line in the figure. For the dual-mode software receiver in this book, each piece of data in the figure contains BDS data and GPS data.

9.2 Software Modules and Program Interfaces for Dual-Mode Receivers

As mentioned in the previous section, the code for the software receiver in this book is the Matlab scripting language, so readers should ensure that they understand it

independently. There are many reference books and technical materials explaining it, as well as a variety of free tutorials on the Internet for reference.

In general, the dual-mode software receiver code can be divided into five parts, namely the graphical interface, the signal acquisition, the signal tracking, the telegram demodulation, and the PVT solution. The functions belong to five subdirectories in accordance with their respective functions, namely GUI, Acquisition, Tracking, Navmsg, and PVT. The functions performed by the code modules in each subdirectory are shown in Table 9.2.

The following is a description of the source code files in each subdirectory. It should be noted that this is only a brief introduction to the basic functions of each source code file. For the details and principles of how it works, please read the source code.

1. **GUI Subdirectory**

main_fig.m: Completes initialization of the main program interface.

initial_mainfig.m: Updates the main program interface according to the information from the initial configuration. It only operates once during the running of the program.

GuiUpdate.m: Updates different information areas during the running of the program, including the data file information area, processing progress information area, processing message area, baseband signal processing information area, least-squares result area, and Kalman filtering result area.

pushInfoMsg.m: Shows the key information and intermediate variables during the running of the program in the processing message area.

Table 9.2 Module divisions of software receiver functions

Subdirectory	Function
GUI	Generates the graphical interface. Updates according to the results of data processing during the running of the program
Tracking	Allocates the tracking channel according to the results of signal acquisition, to realize signal tracking bit synchronization and observation measurement of GPS and BD
Acquisition	Receiver initialization, GPS and BD pseudo-random code generation, generation of various LUTs, and capture of GPS and BD signals
Navmsg	Sub-frame synchronization of GPS and BD signals, GPS data check and BCH check of BD data, and demodulation of ephemeris data
PVT	PVT solution, least-squares solution, Kalman filtering solution, ECEF to LLH coordinate transformation, GPS and BD satellite positioning, and velocity calculation

2. **Acquisition Subdirectory**

ReadGnssConfig.m: Reads the name, the IF sampling frequency, the theoretical IF frequency, and the signal format of the input configuration file that users need to modify to suit the specific file information.

gnssInit.m: Completes global variable initialization, including the generation of GPS and BD pseudo-random codes, sine/cosine and signal mapping look-up tables, initialization of tracking channels, least-squares, and initialization of Kalman filtering results.

GpsCodeGen.m: Generates the pseudo-random code of 32 GPS satellites.

BdCodeGen.m: Generates the pseudo-random codes of 32 BDS satellites.

DownSampling.m: Down-samples the original input signal to satisfy the software FFT requirements.

createValueMapping.m: Establishes a signal map look-up table.

BDsearchNH.m: Searches the starting position of the NH code in the D1 code of the BD, which is required for the capture of the BDS IGSO/MEO satellite signal.

AcquisitionByFFT.m: Captures the signal by the time domain parallel FFT algorithm, which is the core file of the signal capture in the software receiver.

AcquisitionEngine.m: Down-samples the original input signal, and then signal acquisition is performed by the FFT algorithm. It is the upper function interface for the main program to call signal capture.

3. **Tracking Subdirectory**

AllocateTrackingChannel.m: Allocates and initializes the tracking channel according to the result of the capture module.

ResetChannel.m: Resets the tracking channel.

BDGeoTrkLoop.m: Updates loops for BDS GEO satellites, including the code loop update and carrier ring update.

BDMeoTrkLoop.m: Updates loops for BDS MEO/IGSO satellites, including the code loop update and carrier ring update.

GPSTrkLoop.m: Updates loops for GPS satellites, including the code loop update and carrier ring update.

SignalTracking.m: The main function entry of the signal tracking, which performs the I and Q integration of the E, P, and L branches for all the assigned tracking channels, and uses their respective loop update functions to conduct loop updates.

SignalTrackingByC.c: Correlates the I and Q integrals of the E, P, and L branches. It is the only C language file in all codes. Its main purpose is to improve the running speed of the program.

TicMeasurement.m: Extracts pseudo-range and Doppler observations after the signal tracking channel completes the sub-frame synchronization.

UpdateTrackingLoop.m: Calls loops to update their respective functions based on the difference in signal systems.

4. **Navmsg Subdirectory**

BdBchDecode.m: Performs BCH decoding of BDS telegram data bits.

BdGeoDecodeEph.m: Performs the ephemeris data decoding of the BDS GEO satellite.

BdMeoDecodeEph.m: Performs the ephemeris data decoding of the BDS MEO/IGSO satellite.

BdNavProcess.m: The main function entry of the BDS baseband data processing.

BdSearchPreamble.m: Performs the synchronization word search and sub-frame synchronization of the BDS message, upon the completion of which the SOW and the current sub-frame number can be known.

GpsDecodeEph.m: Performs the decoding of the ephemeris data of the GPS satellite.

GpsParityCheck.m: Performs the Hamming code verification of GPS data bits.

GpsSearchPreamble.m: Performs the synchronization word search and sub-frame synchronization of the GPS data. The Z-Count and the sub-frame number can be known after the step is completed.

GpsNavProcess.m: The main function entry for GPS baseband data processing.

5. **PVT Subdirectory**

Ecef2Llh.m: Converts ECEF coordinates to LLH coordinates (latitude and longitude coordinate system).

Llh2Ecef.m: Converts LLH coordinates to ECEF coordinates.

gnssPvt.m: The main function entry of the PVT solution.

initKF.m: Initializes the Kalman filter state with the result of the least-squares solution.

kalmanFix.m: Implements Kalman filtering, including time update and observation update, and the generation of rotation matrix of ECEF to LLH coordinate system and calculation of DOP value. What the Kalman filter model selects is the PV model explained in Sect. 7.2.5.

lsfix_double.m: Performs a least-squares solution (5 state) for dual mode observation.

lsfix_single.m: Performs a least-squares solution (4 states) for single mode observation.

sv_pos_eph.m: Calculates the position, velocity, and clock correction of GPS satellites and BDS MEO/IGSO satellites.

sv_pos_eph_geo.m: Calculates the position, speed, and clock correction of the BDS GEO satellite.

In the source code root directory, the main program script file is startRun.m. In the Matlab environment, the current working directory is set to the source code root directory. Then, the main function (startRun) is run in the command line window, and the main program interface appears. This is shown in Fig. 9.4.

Click the start button in the upper left corner of the interface to start the program. You can stop it by clicking the stop button while the program is running. In the process of running the program, in order to visually display the results of data processing and the status of the program, the program running interface is divided into six parts: the data file information area, processing progress information area, processing message area, baseband signal processing information area, least-squares result zone, and Kalman filter result zone.

The data file information area mainly shows the current data file name, IF sampling frequency value, theoretical IF frequency value, and GPS/BD data file mode (single-mode data or dual-mode data), as shown in Fig. 9.5.

The processing progress information area shows the total duration of the data file currently being processed and the sampling time being processed, as well as the percentage progress of the current data processing, which is shown in Fig. 9.6. The total duration and the time being processed are in milliseconds.

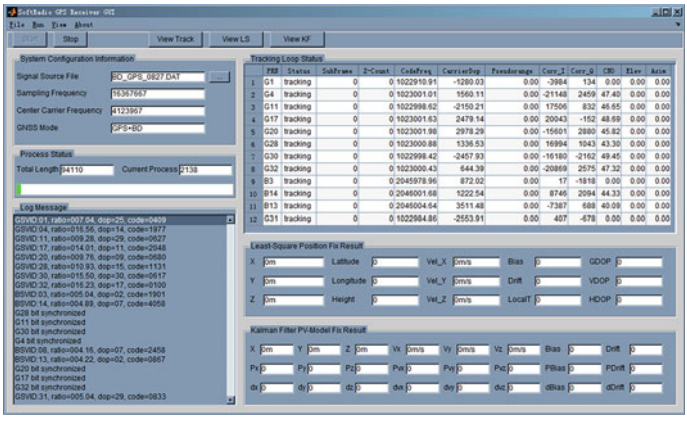


Fig. 9.4 The main program interface of the software receiver

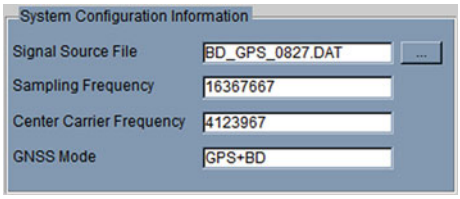


Fig. 9.5 Data file information area

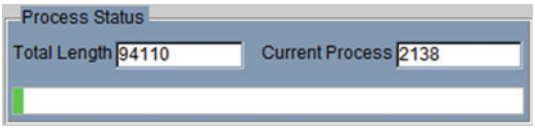
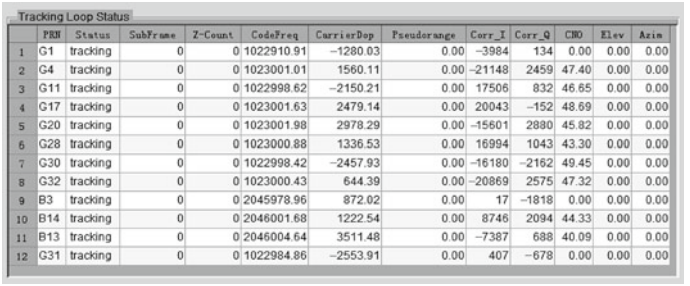


Fig. 9.6 Data processing progress information area

During the running of the program, key information in BDS and GPS signal processing is generated, such as signal acquisition, signal tracking loop status, bit synchronization, sub-frame synchronization results, local time initialization, ephemeris data demodulation results, and PVT solution results. This information is crucial for judging whether the software receiver is working normally, so the signal processing message area is specifically allocated in the main interface to display the information, as shown in Fig. 9.7.

The state quantity during signal tracking is displayed in the baseband signal processing information area, including satellite number, satellite type (G indicates GPS satellite, B indicates BDS satellite), sub-frame synchronization result, Z-Count (or TOW) value, code tracking frequency, carrier tracking frequency, pseudo-range observation, Doppler observation, IQ integral value, CN0, and satellite elevation and amplitude, as shown in Fig. 9.8.

Fig. 9.7 Signal processing message areas


Tracking Loop Status

	PRN	Status	SubFrame	Z-Count	CodeFreq	CarrierDop	Pseudorange	Corr_I	Corr_Q	CNO	Elev	Azim
1	G1	tracking	0	0	1022910.91	−1280.03	0.00	−3984	134	0.00	0.00	0.00
2	G4	tracking	0	0	1023001.01	1560.11	0.00	−21148	2459	47.40	0.00	0.00
3	G11	tracking	0	0	1022998.62	−2150.21	0.00	17506	832	46.65	0.00	0.00
4	G17	tracking	0	0	1023001.63	2479.14	0.00	20043	−152	48.69	0.00	0.00
5	G20	tracking	0	0	1023001.98	2978.29	0.00	−15601	2880	45.82	0.00	0.00
6	G28	tracking	0	0	1023000.88	1336.53	0.00	16994	1043	43.30	0.00	0.00
7	G30	tracking	0	0	1022998.42	−2457.93	0.00	−16180	−2162	49.45	0.00	0.00
8	G32	tracking	0	0	1023000.43	644.39	0.00	−20869	2575	47.32	0.00	0.00
9	B3	tracking	0	0	2045978.96	872.02	0.00	17	−1818	0.00	0.00	0.00
10	B14	tracking	0	0	2046001.68	1222.54	0.00	8746	2094	44.33	0.00	0.00
11	B13	tracking	0	0	2046004.64	3511.48	0.00	−7387	688	40.09	0.00	0.00
12	G31	tracking	0	0	1022984.86	−2553.91	0.00	407	−678	0.00	0.00	0.00

Fig. 9.8 The baseband signal processing information area

With the completion of the ephemeris demodulation, the determination of the local time, and the extraction of the pseudo-range and carrier frequency observations, the receiver software starts the PVT solution. The initial solution algorithm is completed by the least-squares method, and the program will adopt either single-mode or dual-mode algorithms based on the combination of observation measurements. The results of least-squares calculation include information in the ECEF coordinate system and the latitude and longitude coordinate system, the speed of the ECEF coordinate system, clock difference and clock drift, and the geometric accuracy factor. A further PVT solution is completed by Kalman filtering. The Kalman filter in the software receiver is in system state, which is [*x, y, z, b, vx, vy, vz, dr, Tgb*], where [*x, y, z*] are the positions in the ECEF coordinate system, [*vx, vy, vz*] are the amounts of velocity in the ECEF coordinate system, and [*b, dr, Tgb*] are the clock difference, the clock drift, and the system time deviation of GPST-BDT respectively. The results of least-squares and Kalman filtering are shown in the least-squares result area and the Kalman filter result area, as in Fig. 9.9.

During the running of the program, you can click the three View buttons at the top of the interface to implement the function similar to "Oscilloscope". The View Track button can show the I/Q integral and E, P, and L branch integration results of each tracking channel. Thus, we can make a real-time observation of the entire process of transient response and loop update during signal tracking, which is shown in Fig. 9.10.

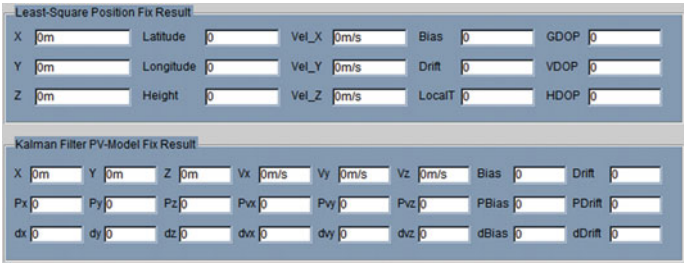


Fig. 9.9 The least-squares result area and Kalman filter results area

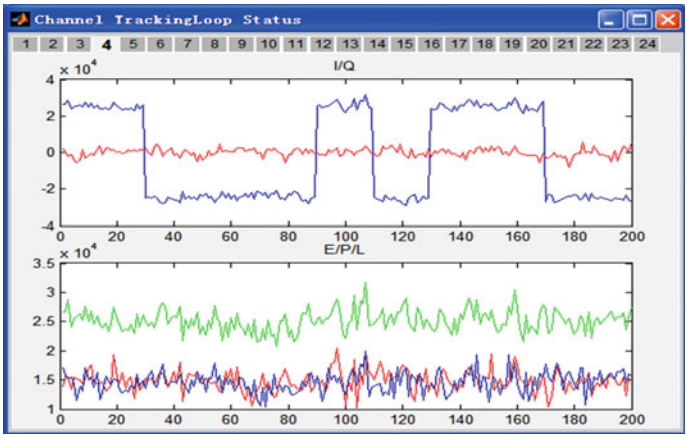


Fig. 9.10 I\Q integral and E, P, and L integration results of the tracking channel

The View LS button shows the curve of the least-squares positioning result on the time axis. The View KF button shows the value of the system state of the Kalman filter, the correction amount, and the curve of the covariance matrix element on the time axis. They are shown in the left and right halves of Fig. 9.11 respectively. In the least-squares result, there is information such as receiver position, speed, clock drift, GPST-BDT system time difference, and DOP value. They are the values of nine system state quantities in the Kalman filter result display, and display their respective correction amount and variance value.

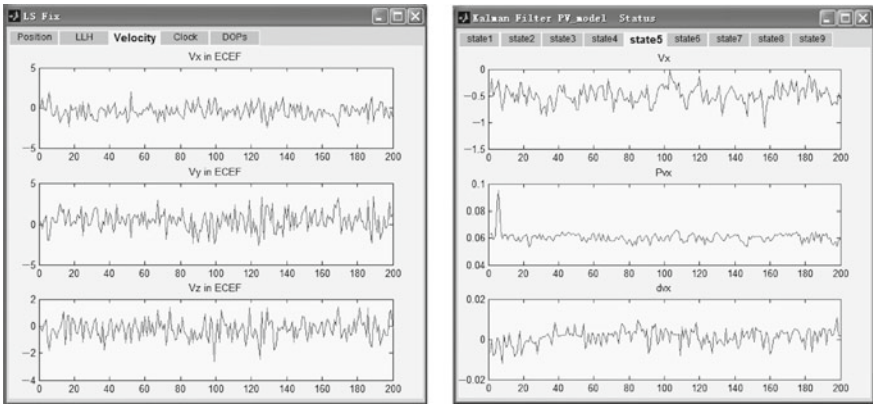


Fig. 9.11 Least-squares results (left) and Kalman filter results (right)

9.3 Data Processing in Dual-Mode Receivers

As well as the Matlab code for BDS and GPS dual-mode software receivers, a section on the BDS and GPS dual-mode IF data file is also provided in this book. This file comes from the UTREK210 satellite belonging to Beijing Jetstar Quanta Technology Co., Ltd. The data acquisition system obtains the actual satellite signal acquisition, with a sampling frequency of 16.367 6 MHz, a theoretical intermediate frequency of 4.130 4 MHz, and two-bit quantization. The data storage format is consistent with the description in Fig. 9.2. Combined with the source code of the dual-mode software receiver and the dual-mode IF data file, readers can directly debug and modify the implementation of the software receiver. At the same time, the software algorithm can be programmed to process the actual satellite signal, which can deepen the theoretical understanding of the GNSS receiver.

In this section, the software receiver is used to process the above IF data files, including key processing such as signal acquisition, tracking, message demodulation, and navigation solution. The results are presented in the form of graphical curves. Readers can run the software receiver code independently, and analyze and compare the intermediate results and variables that are of interest.

Figure 9.12 shows the result of signal acquisition for dual-mode IF data files. Due to the characteristics of the software receiver, there is no a priori information at the beginning of the software receiver operation. Therefore, the signal acquisition method adopts the "cold start blind search" strategy. Figures 9.12 and 9.13 show the results of capturing all GPS and BDS satellites. The specific signal acquisition algorithm uses the time domain parallel FFT acquisition algorithm from Sect. 4.1.4. The figure only shows the capture results of the satellites with stronger signals, from which the obvious signal peaks can be seen.

According to the results of Figs. 9.12 and 9.13, the pseudo-code phase and Doppler frequency corresponding to the satellite signal peak after successful acquisition can be calculated. Then, the information is submitted to the signal tracking channel through a certain logic conversion, and the carrier channel of the tracking channel is initialized. The pseudo-code NCO is set to enter the tracking processing of the signal.

Figure 9.14 shows the coherent integration result of the in-phase path (I way) and the orthogonal path (Q way) of the signal tracking loop output. The integral length is 1 ms, so the horizontal axis in the figure is the time axis in ms. The axis is the integral amplitude value. The results in the figure are from GPS satellite signals and have similar patterns to the BDS satellite signal, except that the data hopping period of the BDS GEO satellite is 2 ms. The upper part of the figure is the I way integral result, and the lower part is the Q way integral result. From the curves of the I and Q way integral results with time, the following two points can be seen:

① At the beginning of the loop (within 500 ms), the signal energy gradually increases, which means that $\sqrt{I^2 + Q^2}$ gradually increases. This is because the loop adjustment makes the carrier frequency difference smaller, so the energy loss caused by the frequency difference becomes smaller.

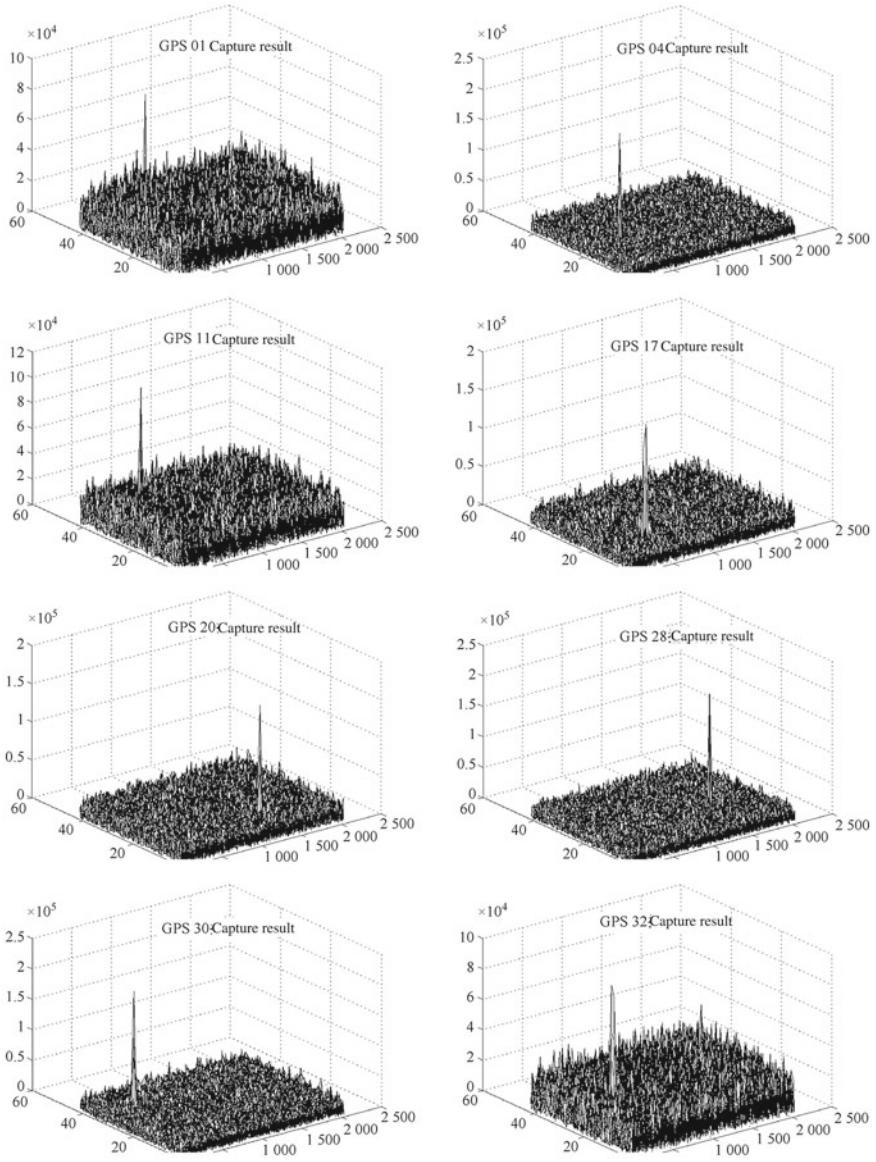


Fig. 9.12 Results of signal acquisition for all GPS and BDS satellites

② After the loop enters stable tracking (after 500 ms), the integral amplitude of the I path is stable, and the integral amplitude of the Q channel becomes smaller. This is because the phase lock is started when the frequency difference is small to a certain degree. After the phase lock is realized, the energy is concentrated

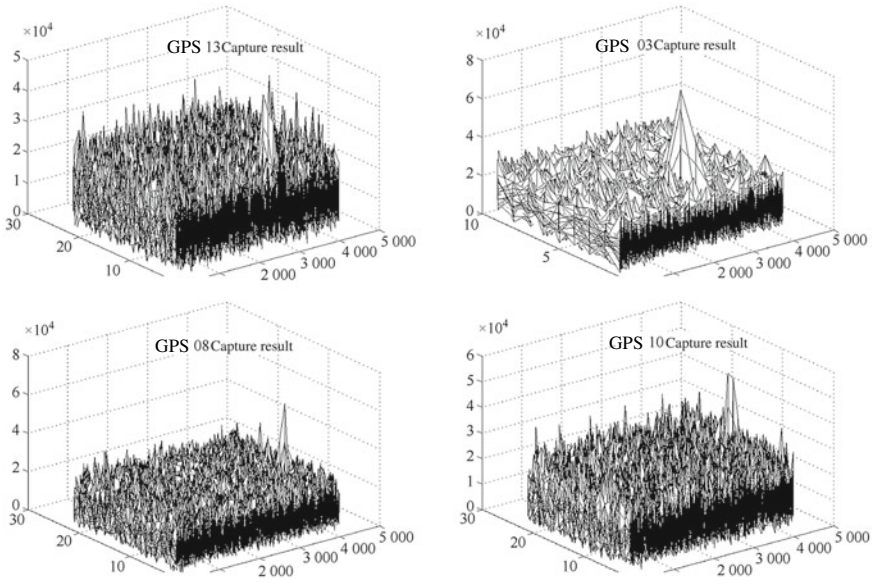


Fig. 9.13 Results of signal acquisition for all BDS satellites

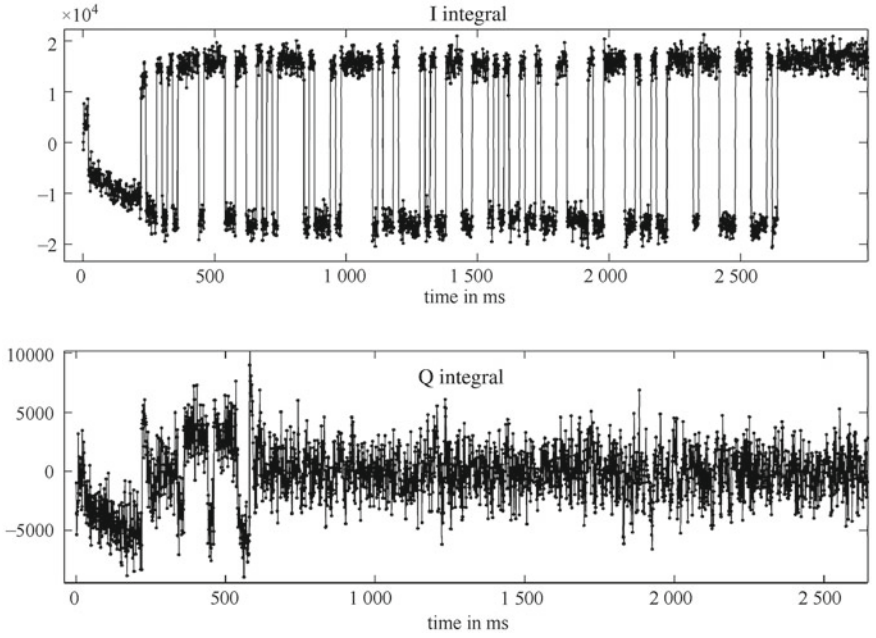


Fig. 9.14 The I and Q integral results of the signal tracking loop

in the I path, and the Q path only contains the noise component. At this time, the navigation signal can be demodulated by the integral value of the I path.

Figure 9.15 shows the NCO frequency value of the carrier tracking loop and the pseudo-code tracking loop in the signal tracking process. The upper part is the carrier tracking frequency value, and the lower part is the pseudo-code tracking frequency value. From Figs. 9.15 and 9.14, corresponding conclusions can be made. At the beginning of the loop, the variance of the carrier frequency and the pseudo-code frequency is large, indicating that a relatively drastic adjustment is being made at this time. The variation of the carrier frequency and the pseudo-code frequency after entering the stable tracking is much smaller than in the initial stage. In particular, the pseudo-code frequency is greater due to the role of carrier assistance, resulting in a very smooth pseudo-code frequency.

For loop update strategies at different stages of loop tracking, please refer to the source code to understand the details. Adjust the parameters and strategies to improve the loop performance.

After the tracking loop enters a steady state, the sub-frame synchronization and the demodulation of the navigation message can be started. Synchronization of the BDS and GPS sub-frames can be realized by referring to the methods described in

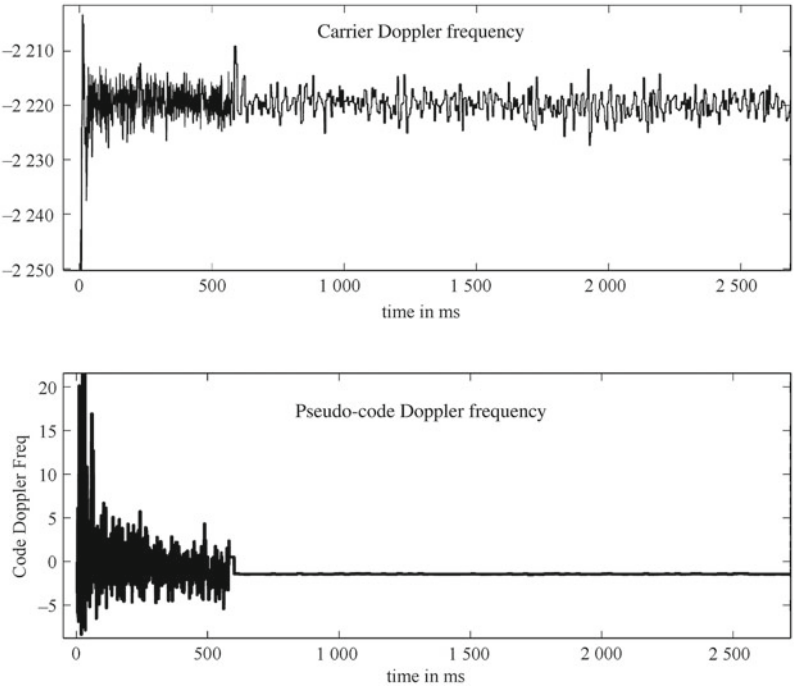


Fig. 9.15 Carrier frequency of the carrier tracking loop, and pseudo-code frequency of the pseudo-code tracking loop

Sects. 3.3.2 and 4.2.7 of this book. Comprehensive navigation in telemetry demodulation includes TOW demodulation, ephemeris data demodulation, almanac data demodulation, UTC parameter demodulation, and ionospheric parameter demodulation. Due to limitations of the data file length, the software receiver code only implements TOW demodulation with ephemeris data, but this is sufficient for subsequent PVT solvers.

After the TOW time of the satellite signal is acquired, the observation measurement can be performed. Here, the pseudo-range observation and the Doppler frequency observation are extracted. The PVT solution can be achieved after obtaining more than four satellite observations and pieces of ephemeris data.

The PVT solution strategy is to perform the least-squares positioning solution first, then initialize the Kalman filter state quantity with the result of the least-squares method, and then start the Kalman filter positioning solution. The least-squares method in the specific implementation will either adopt the 4-state model or the 5-state model according to the situation of single-mode or dual-mode observation, wherein the 4-state model is for the single-mode (single GPS or single BDS) and 5-state model is for the dual-mode observation. The 5-state model has one GPST-BDT system time deviation more than the 4-state model. For the specific principle, refer to Chap. 7 of this book.

Figure 9.16 shows the receiver position coordinates calculated by the least-squares method, expressed in the ECEF coordinate system. In order to display the range of variation of the coordinates, the difference between the calculated coordinate result and its mean value is shown in the figure. Since the software receiver code extracts the observation every 100 ms, the horizontal axis of Fig. 9.16 is the time axis in units

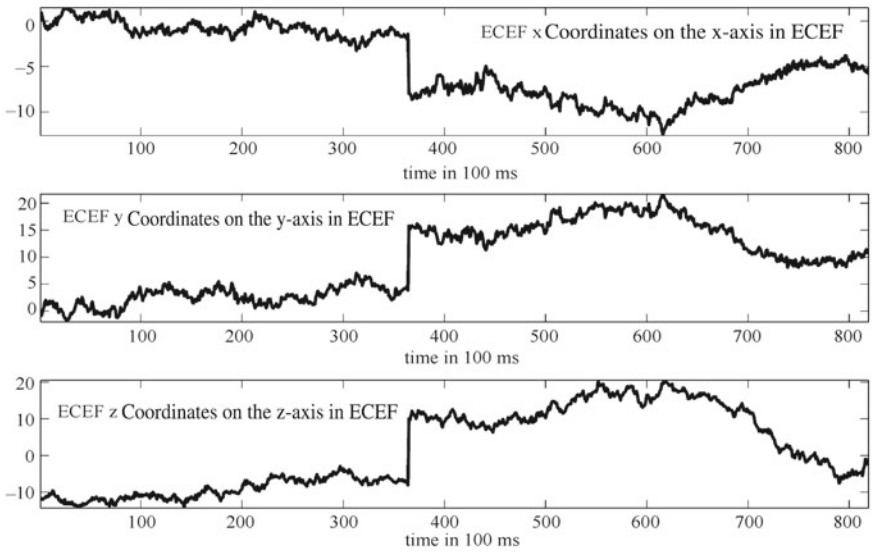


Fig. 9.16 Location calculated by the least-squares method

of 100 ms, and the vertical axis is in meters. It can be seen that the positioning result of the least-squares method has large jumping phenomenon.

In contrast, Fig. 9.17 shows the position coordinates calculated by the Kalman filter, in which the mean value is also deducted for the same reason. It can be seen from the comparison of the position results of the least-squares method that the position result of the Kalman filter is smoother, and the position jump range is much smaller in the entire positioning time. The reader can adjust the R and Q matrix parameters to obtain different positioning results based on the understanding of the Kalman filter source code. The reader can also consider the effect that the result of the Kalman filter obtains from the R matrix and the Q matrix.

The velocity component results of the Kalman filter are given in Fig. 9.18. The upper, middle, and lower curves are the velocity components in the X-axis, Y-axis, and Z-axis directions respectively. Since it is a stationary antenna, the velocity component has a mean value of 0, and the velocity deviation in the Z-axis direction is slightly larger. The three curves in Fig. 9.19 are the local clock error, local clock drift, and GPST-BDT system time deviation, where the local clock and system time deviation are in meters, and the local clock drift is in m/s. The local clock difference and the system time deviation can be converted to the speed of light in seconds, and the speed of light at the local clock can be converted to ppm.

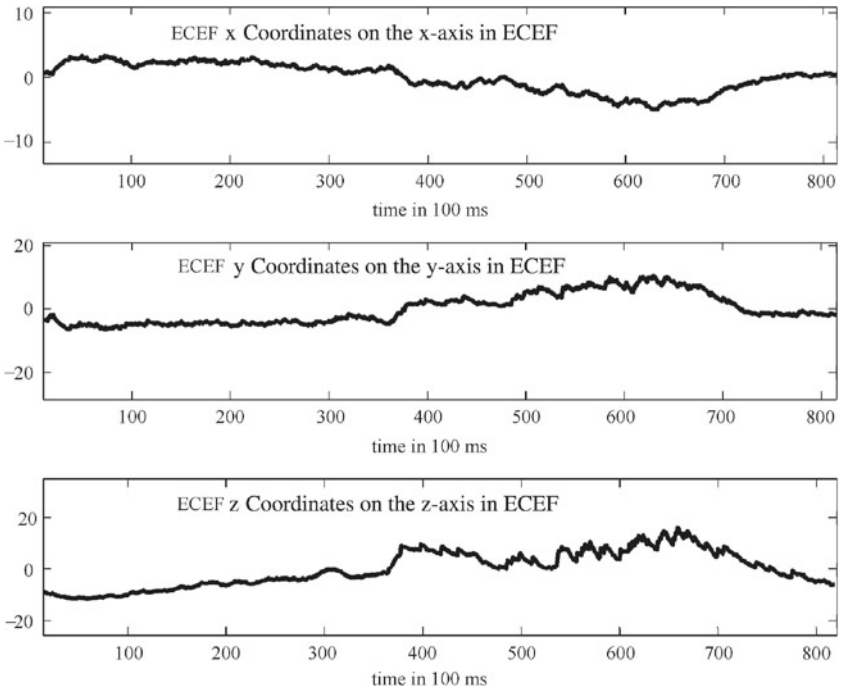


Fig. 9.17 Location calculated by Kalman filtering

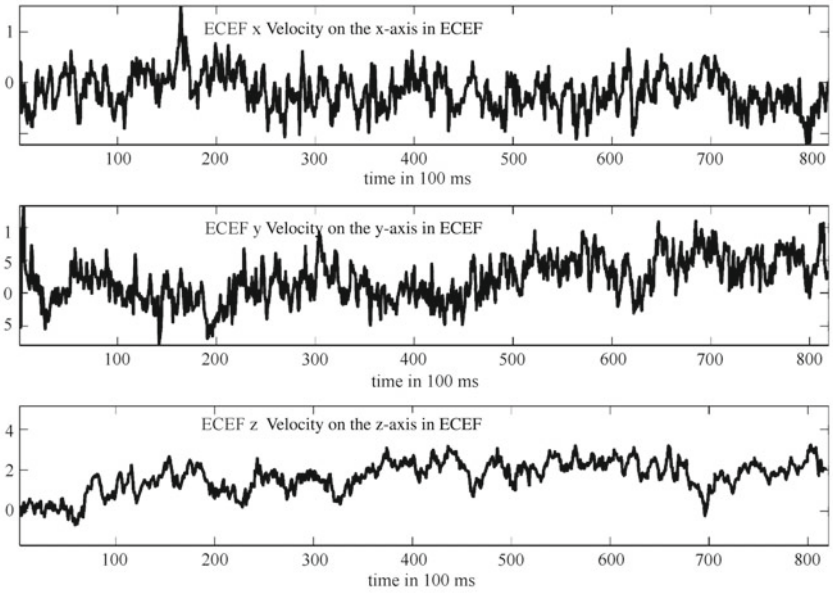


Fig. 9.18 Speed of Kalman filtering

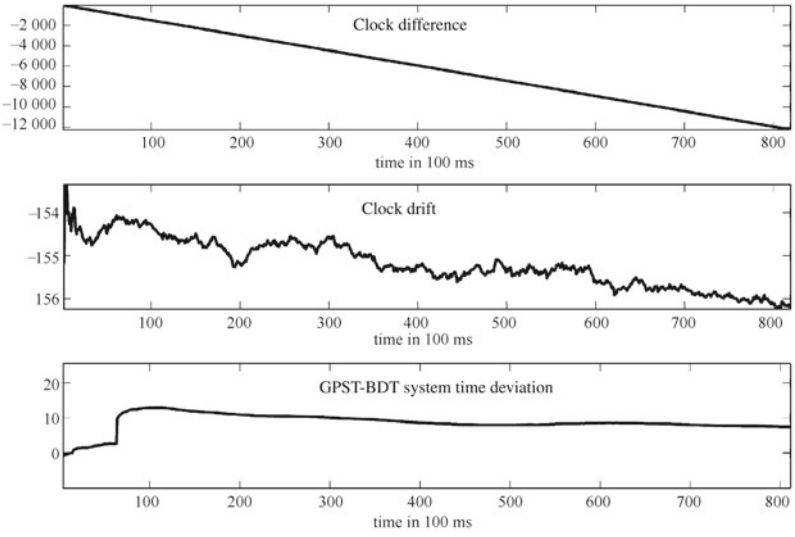


Fig. 9.19 Clock time difference and clock drift of Kalman filtering, and GPST-BDT system time deviation

The local clock difference is the integral of the local clock drift. This conclusion is determined by the physical nature of the clock difference and the clock drift. This can also be easily verified from the values in the curve in Fig. 9.19. It can be seen from Fig. 9.19 that the value of the GPST-BDT system time deviation is around 0 (if expressed in seconds), which means that BDT and GPST are basically synchronized. If the observation error is subtracted, the value of the system time deviation will be smaller. The figure shows that the variation of the time deviation of the GPST-BDT system is small and remains basically unchanged during the observation.

Figures 9.20, 9.21 and 9.22 show the respective curves of the variance of the various system states of the Kalman filter over time. The Kalman filter in the software receiver uses the PV model explained in Sect. 7.2.5. The system state vector is $[x, y, z, b, vx, vy, vz, d, T_{GB}]$, which consists of three position quantities, three speed quantities, and three clock quantities. The P matrix is the covariance matrix of the nine system state variables, while Figs. 9.20, 9.21, and 9.22 give the time curve of the nine diagonal elements of the P matrix. The unit of the vertical axis in Fig. 9.20 is m, the unit of the vertical axis in Fig. 9.21 is m/s, and the unit of the vertical axis in Fig. 9.22 is m (top), m/s (middle), and m (bottom). The horizontal axis in these three figures is the time in 100 ms.

Figures 9.20, 9.21, and 9.22 show that the time curves of the diagonal elements of the P matrix of the Kalman filter are consistent. The initial values are large. They gradually become smaller as time passes, and finally converge near a stable value. This is because when the Kalman filter is initialized by using the result of the least-squares method, the P matrix is generally initialized to a larger value, which

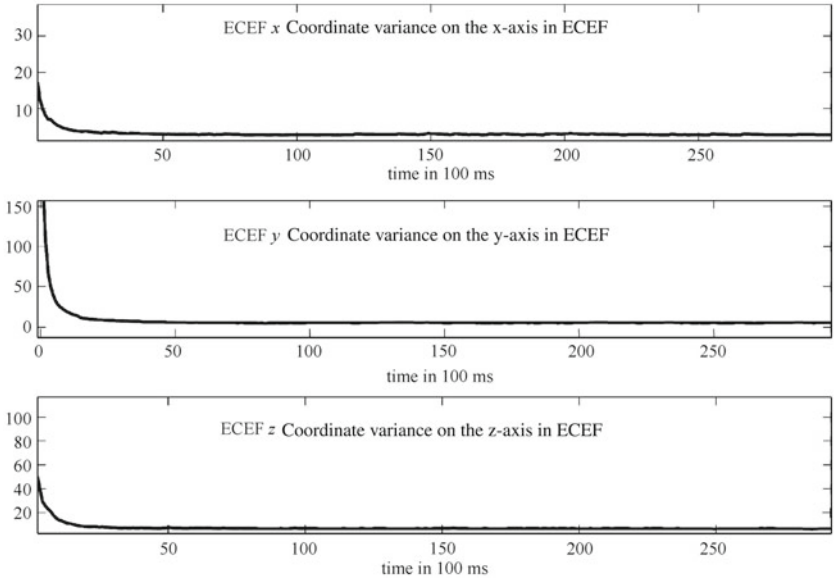


Fig. 9.20 The variance of the positional state of the Kalman filter output

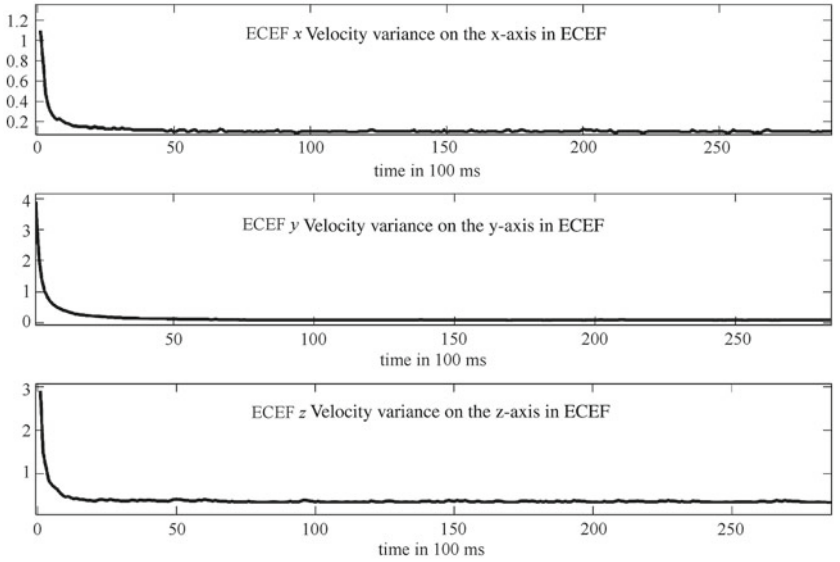


Fig. 9.21 The variance of the velocity state of the Kalman filter output

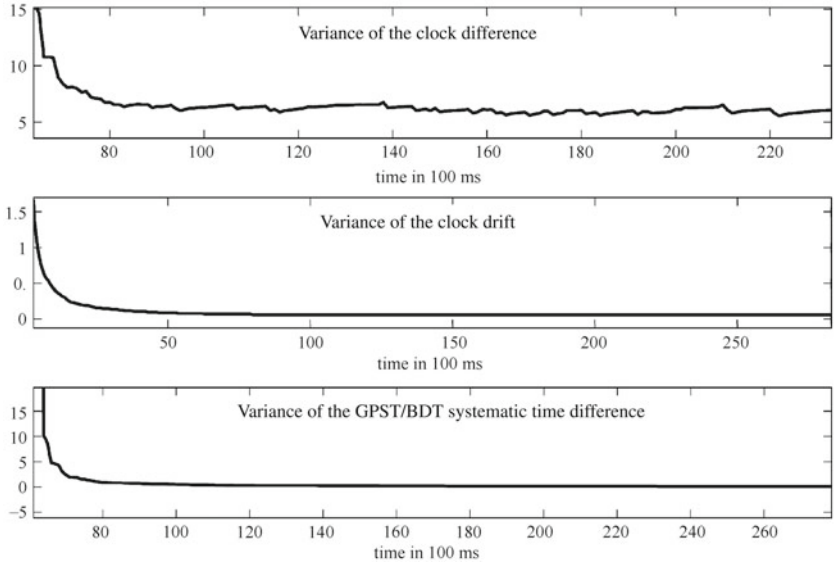


Fig. 9.22 The variance of the time state of the Kalman filter output

allows the observation update to quickly correct the local state. As time passes and the observation quantity update continues, the covariance matrix of the local system state will gradually converge. At this time, the observed noise variance and the P matrix jointly determine the Kalman gain matrix K. The correction for the local system state will be the weighted result of the local state variance and the observed noise variance. The final P matrix will also stabilize near a convergence value. In the subsequent update of the Kalman filter, the observation noise situation and the system processing noise matrix Q will affect the convergence value of the P matrix, such as when there are no observations for a period of time (a typical scenario in which this happens is when the receiver enters an underground garage or tunnel), or if there is a decrease in the quality of the measurement results in an increase in the P matrix.

A simple explanation has been offered on the processing of the actual IF data file by the dual-mode software receiver in this book. This analysis covers only a small part of the intermediate result in the software receiver, and some implementation details and debugging processes have been skipped. We hope that readers will understand the software receiver code implemented in this book, and grasp its theoretical principles. Modifying and optimizing the existing code, and observing the changes in output will improve future work and research.