Chapter 59
A Code Phase Measurement Method for Snap-Shot GNSS Receiver

Bin Huang, Zheng Yao, Mingquan Lu and Zhenming Feng

Abstract The snap-shot working mode has turned out to be a good choice in location-based services (LBS) applications for it can significantly reduce the calculation and power dissipation in the receiver. However, within a limited snap-shot data, the code loop is difficult to enter into convergence state which leads to a large measurement error. A Kalman filter (KF) based code loop can accelerate convergence but the accuracy is still very poor. In this paper, a novel code phase measurement method is developed and used for snap-shot receivers. A KF based code loop is used to maintaining code tracking, further, a non-causal smoothing estimator is joined to improve measurement accuracy of code phase and obtain more accurate positioning results. Experiment results show that, within a second or less snap-shot data, the method with smoothing can obtain a relatively accurate code phase, and with the same snap-shot length, the proposed method has better estimation accuracy than one without smoothing and the conventional code tracking loop. Finally, the method is applied in a real-time software receiver which working in snap-shot mode.

Keywords Global navigation satellite system · Snap-shot receiver · Kalman filter · Code phase tracking · Non-casual smoother

59.1 Introduction

Usage of global navigation satellite system (GNSS) has become quite common in our society, allowing many applications that are used in our daily life. Conventional GNSS receivers work in a continuous way which at least four satellites are

B. Huang (✉) · Z. Yao · M. Lu · Z. Feng
Tsinghua University, 1109-1110 Wei Qing Building, Beijing 100084, China
e-mail: b-huang09@mails.tsinghua.edu.cn; bing890816@163.com

J. Sun et al. (eds.), *China Satellite Navigation Conference (CSNC) 2013 Proceedings*, Lecture Notes in Electrical Engineering 243,
DOI: 10.1007/978-3-642-37398-5_59,

tracked continuously in order to provide location fixes. In order to provide a fix a conventional GPS C/A code receiver needs at least 30 s for the length of a main frame of navigation message is 30 s. However, for location-based services (LBS) under the demand of these receivers, the computational complexity, resource utilization, power dissipation and the response time of the results are unacceptable.

Under this application background, the receivers with a snap-shot working mode get more and more concerned [1–4]. In snap-shot working mode, the receiver is in a dormant state in most of the time, it turns into the snap-shot operating state only when it receives a user request. Upon each positioning request, the receiver takes a snapshot and performs signal processing to generate a PVT result.

In a snap-shot receiver, the length of the snap-shot data is the key parameter. Considering the application background, the length of the snap-shot data is expected to be as short as possible. However, with a shorter snap-shot length, the conventional Delay-Locked Loop (DLL) is hard to obtain relatively accurate code phase result within less update times. In Driessem [5] and Qian et al. [2], a KF based DLL technique is used to speed up the loop convergence, but the poor accuracy of the code phase is still the most troublesome problem.

In this paper, a non-causal smoother is joined to the KF based DLL to improve the tracking accuracy of the code phase within the limited data. Since the data has been cached, the smoothing result of the code phase can use additional information which contained the future data. The simulation results show that, within a second or less snap-shot data, the method with smoothing can obtain a relatively accurate code phase, and with one hundred milliseconds snap-shot length, the proposed method has better estimation accuracy than one without smoothing and the conventional code tracking loop. In addition, the proposed method has been used on a snap-shot receiver which processes the actual GNSS signals. Actual positioning results show that the snap-shot receiver with smoothing have the smaller RMS error and more focused positioning results. The horizontal RMS error with proposed method is 2.9986 m that has been able to meet most of the LBS applications' demand.

The remainder of this paper is divided into four sections. Section 59.2 gives a brief overview of the snap-shot GNSS receivers. Section 59.3 presents KF base code phase tracking loop models and the proposal loop filter solution. The experiment results and discussions are presented in Sect. 59.4. Finally the conclusions are shown in Sect. 59.5.

59.2 Snap-Shot GNSS Receiver

GNSS positioning has turned out to be an enabler of LBS in recent years. However, the continuous working mode is not suitable for LBS. On the one hand, the positioning information does not actually require a high update rate. On the other hand, in order to ensure the real-time tracking, the receiver must occupy a great many processor resources and have a high priority. In the context of application requirements, a snap-shot working mode is an ideal solution.

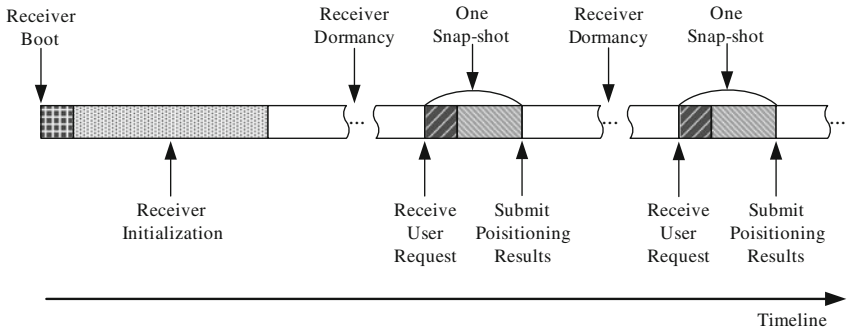

Fig. 59.1 The flow chart of a snap-shot receiver

Figure 59.1 shows the flow chart of the snap-shot receiver architecture [1]. It can be observed that after boot before turning into the snap-shot mode, the receiver need to complete a initialization process, which to get necessary information for the snap-shot positioning such as GNSS time and satellite ephemeris, etc. The initialization process is necessary because the relatively accurate GNSS time is required and it generally difficult to obtain from external auxiliary. For LBS, if receiver can utilize assisted network or internet to obtain the auxiliary information such as current satellite ephemeris and coarse time, both the complexity and the calculation of the initialization process can be greatly reduced.

After the initialization process, the receiver turn into a cycle stage which only in the snap-shot state or dormant state. It turns into the snap-shot working state only when it receives a user request. In the snap-shot working sate, it collects several hundred milliseconds snap-shot data for the latter part of signal analysis. Note that the snap-shot data has been cached, so it is not need real-time signal acquisition and tracking. The receiver can handle these channels serially, which can significantly reduce the occupancy rate of the receiver processor. When code phase estimated values of more than four visible satellites are acquired, a positioning calculating program can be called to obtain positioning results.

Usually, the snap-shot length would be only a few hundred milliseconds for the limited storage of consumer electronics devices. A shorter snap-shot length, however, directly affect the measurement accuracy of code phase thereby affect the positioning results. So, how to solve measurement accuracy of the code phase under the limited snap-shot length is the most critical problem in snap-shot receiver.

59.3 Models of Code Phase Measurements

The conventional GNSS receivers generally use the DLL to track the spreading code and obtain accurate code phase. A wide loop bandwidth make the DLL turn into the locked state faster but it also cause the decline in code phase accuracy.

In Driessem [5] it has been proved that the use of a KF based tracking loop is possible to achieve both rapid full-in and reliable tracking without the tradeoffs associated with conventional fixed loop gain tracking loop. However, the poor accuracy of the code phase is still the most troublesome problem. In snap-shot receiver, the signal data has been cached, which allows a non-causal smoother to be used to further enhance the measurement accuracy of the code phase. In Psiaki [6], a fixed-interval smoother is used which get more accurate than KF, the accuracy increases because the estimate at any given time is based on a larger data set. In Yao et al. [1], it uses a dual update-rate tracking loop with a non-casual smoothing estimator to obtain a relatively accurate estimation. In this section, an equivalent KF based DLL is derived, and the structure of a non-causal fixed-point smoother is presented.

59.3.1 Kalman Filter Model Based Delay-Locked Loop

In order to use the KF theory, the dynamic model of the system must be established. Assuming the signal contains only one order dynamic, the process can be modeled by

$$s_{n+1} = \begin{pmatrix} \phi_{n+1} \\ T \cdot f_{n+1} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \phi_n \\ T \cdot f_n \end{pmatrix} + \begin{pmatrix} 0 \\ u_n \end{pmatrix} = As_n + U_n \tag{59.1}$$

$$x_n = \begin{pmatrix} 1 & 0 \end{pmatrix} s_n + w_n = hs_n + w_n \tag{59.2}$$

$$u_n \sim N(0, \sigma_u^2), \quad w_n \sim N(0, \sigma_w^2), \quad U_n \sim N(0, Q), \quad \mathrm{Q} = \begin{pmatrix} 0 & 0 \\ 0 & \sigma_u^2 \end{pmatrix} \tag{59.3}$$

where ϕ_n and f_n are the code phase and code Doppler rate at the start of the n-th update interval, T is PIT in seconds, s_n is state vector and x_n is the observation of the n-th code phase, u_n is the additive white Gaussian motivate noise, and w_n is the observation noise of code phase. The observation noise is come from the phase discriminator here, so the noise variance is related to the type of the discriminator. When using the classical early-minus-late power (EMLP) discriminator [7], the variance of the observation noise is

$$\sigma_w^2 = \frac{d}{4(C/N_0)T}\left(1 + \frac{2}{(C/N_0)T(2-d)}\right) \tag{59.4}$$

where d is the early-late chip spacing in chips, and C/N_0 is CNR.

Define $K_n = \begin{pmatrix} K_{n0} & K_{n1} \end{pmatrix}^T$ as the Kalman gain vector, $\hat{\mathbf{s}}_n = \begin{pmatrix} \hat{\phi}_n & T \cdot \hat{f}_n \end{pmatrix}^T$ as the estimated value of the n-th update interval, $s_{n|n-1} = \begin{pmatrix} \hat{\phi}_{n|n-1} & T \cdot f_{n|n-1} \end{pmatrix}^T$ as a prediction value for n-th from sample $n-1$, as $M_{n|n-1}$ the minimum mean square error

(MMSE) matrix of the prediction, and $M_{n|n}$ as the MMSE matrix of the filter. The KF form equations are as follows with the initial conditions $\hat{s}_{-1|-1}$ and $M_{-1|-1}$ [8]:

$$\begin{cases} \hat{s}_{n|n-1} = A\hat{s}_{n-1|n-1} \\ M_{n|n-1} = AM_{n-1|n-1}A^T + Q \\ K_n = \frac{M_{n|n-1}h^T}{hM_{n|n-1}h^T + \sigma_w^2} \\ M_{n|n} = (I - K_n h)M_{n|n-1} \\ \hat{s}_{n|n} = \hat{s}_{n|n-1} + K_n(x_n - h\hat{s}_{n|n-1}) = \hat{s}_{n|n-1} + K_n\varepsilon_n \end{cases} \tag{59.5}$$

where ε_n is equivalent to the output of the phase discriminator.

Figure 59.2 shows estimated value of the code phase which using a conventional DLL and a KF based DLL, it can be clearly seen that, for conventional DLL, the 2 Hz loop bandwidth make the DLL turn into the locked state faster but it also cause the decline in code phase accuracy, and the KF based DLL can achieve fast acquisition time and low tracking jitter when the loop turns into steady-state. However, within a snap-shot length less than one second, the accuracy is still very poor.

59.3.2 Fixed-Point Smoother Model

Since the signal data has been cached, which allows a non-causal filter to be used to further enhance the measurement accuracy of the code phase. The extended RTS for fixed-point smoother is based on the KF, it has the following form Andrew and James [9]:

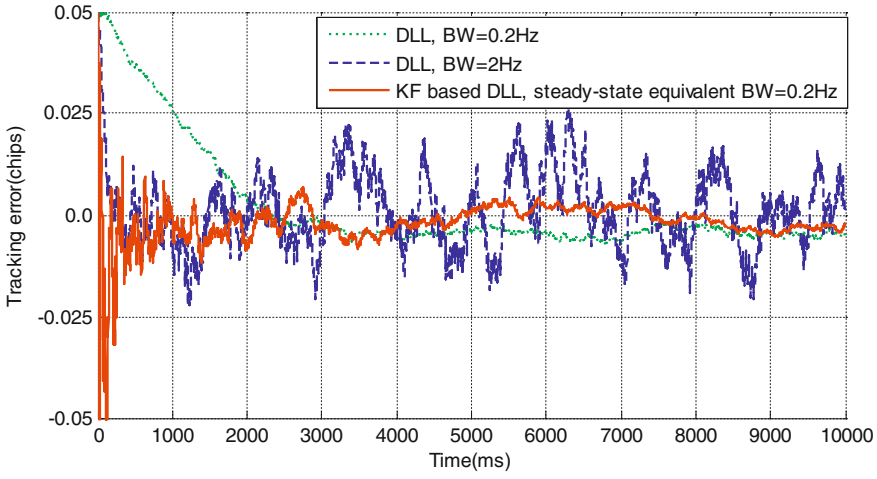

Fig. 59.2 The contrast between a conventional DLL and a KF based DLL

$$\widetilde{s}_{p|n} = \widetilde{s}_{p|n-1} + \Theta_{p,n} K_n \varepsilon_n \,,\ \mathrm{n} > \mathrm{p} \tag{59.6}$$

where t_p is the selected time and p is the fixed-point, and the initial state $\tilde{s}_{p|p}$ is equal $\hat{s}_{p|p}$ which has been estimated at the above KF approach.

$$\begin{cases} \Theta_{p,p} = I \\ \Theta_{p,n+1} = \Theta_{p,n}(I - K_n h) M_{n|n-1} A^T M_{n+1|n}^{-1} \end{cases} \tag{59.7}$$

The error covariance of the smoothing estimator $M_{p,n}$ can be computed by

$$\begin{cases} M_{p,p} = M_{p|p-1} \\ M_{p,n+1} = M_{p,p} - \Theta_{p,n} K_n h M_{n|n-1} \Theta_{p,n}^T \end{cases} \tag{59.8}$$

Assume that N is the total update count of the loop, the smoothing result is $\widetilde{s}_{p|N}$. For fixed-point smoother, there is a problem to be solved is how to select the fixed-point equivalent to how to select p here.

Figure 59.3 shows the typical error variances of KF based DLL. It can be clearly seen that the error covariance after smoothing is smaller than not smoothing. This result is very nature because the smoothing results for fixed-point uses additional information which contained the future data. Besides, it also can be seen that the minimum of the smoothed error covariance can be obtained near the middle time.

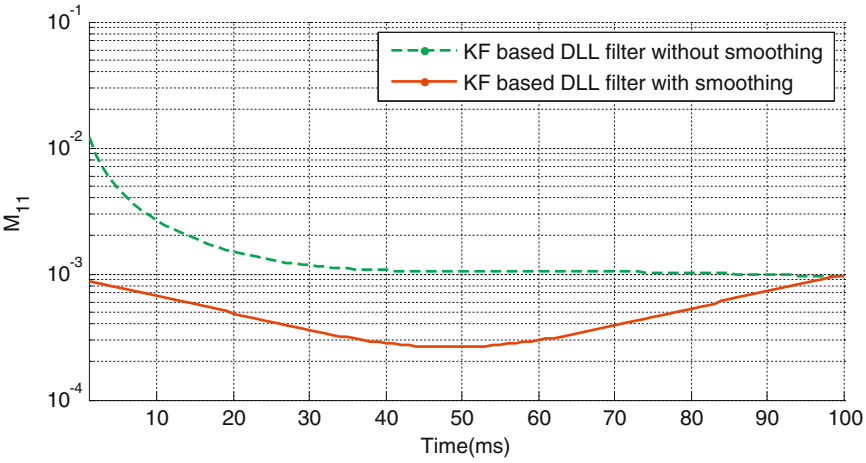

Fig. 59.3 The typical error variances of KF based DLL

59.4 Experiment Results

59.4.1 Code Phase Estimated Accuracy

To illustrate the estimated accuracy of the proposed technique, several simulations will be implemented and the results under different circumstances will be presented. Assuming that the simulation input signal is a GPS C/A signal which is generated by Matlab. The conventional DLL uses a second order loop, and carrier loop is a first order FLL with 10 Hz loop bandwidth.

Figures 59.4 and 59.5 shows the tracking error standard deviations of these loops against CNR. The CNR is from 30 to 50 dB-Hz with 2 dB-Hz increment. The initial code phase deviation is 0.25chips and the initial carrier Doppler deviation is 500 Hz, the accuracy of these values are similar to the capture results of a normal receiver. Each value has been evaluated through a Mote Carlo simulation over 500 tests.

It can be observed that by use of smoothing, the code phase estimate has a higher accuracy. This is primarily because that the smoother uses additional information which contained the future data than the predication. So if the smoothed estimate value of code phase is used to calculate the position, it is possible improve the accuracy of the results. In Fig. 59.4, the KF based DLL has the better tracking performance than the conventional DLL. It is because the KF based DLL has a faster convergence rate. In Fig. 59.5, with a 10,000 ms snap-shot length, it can be seen obviously from the results that the KF based DLL is equivalent with the conventional loop when they are both into the steady-state, and even if they all enter the steady-state, the proposed result still has the highest

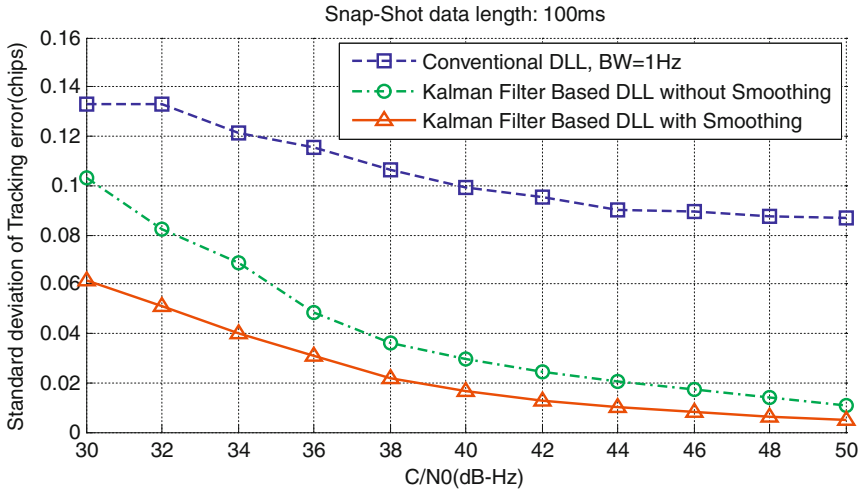

Fig. 59.4 The standard deviation of the tracking error, 100 ms snap-shot

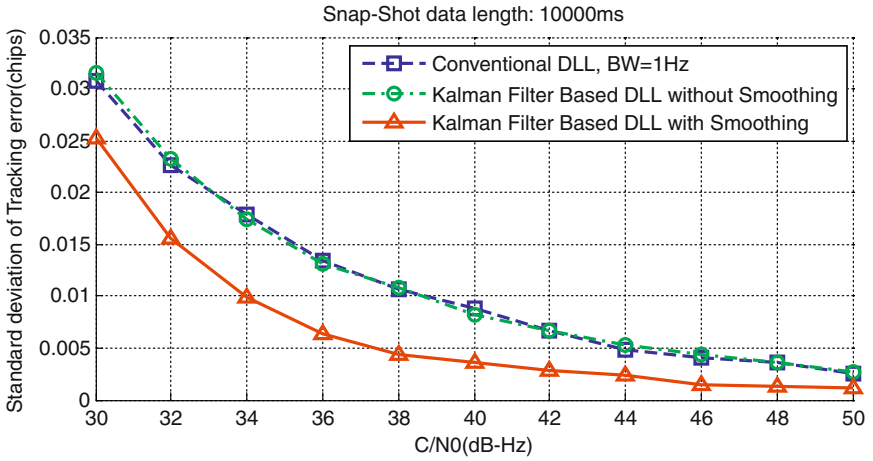

Fig. 59.5 The standard deviation of the tracking error, 10,000 ms snap-shot

accuracy. But 10,000 ms snap-shot is too long, generally for LBS, the length less than 1,000 ms can be accepted.

Figure 59.6 shows the impact of the snap-shot length T_t on the tracking accuracy when C/N_0 is 40 dB-Hz. It can be seen that for all three methods, increasing T_t can improve the tracking accuracy especially when T_t is smaller than 2 s. Ihe loop with smoothing can achieve the better tracking accuracy in the same snap-shot length. The code phase accuracy of proposed method has 3.3 dB larger than the one without smoothing and the conventional loop. It can be clearly seen from Fig. 59.6 that a 500 ms snap-shot for the loop with smoothing can obtain the same accuracy with over 5 s snap-shot for the other two methods. It means in order to achieve the same accuracy, the proposed method required less input data and less memory resource.

59.4.2 Actual Positioning Results

The snap-shot positioning mode is implemented on a GPU based real-time GNSS software receiver which can process all the civil GNSS signals. In snap-shot mode, the receiver complete a initialization process, which to get necessary information for the snap-shot positioning such as GNSS time and satellite ephemeris, etc. The GPS C/A signal is selected as actual measured signal here. The snap-shot data length is 500 ms with the 5 Mchips/s sampling rate, and the steady state bandwidth of KF based DLL is 1 Hz.

To illustrate the accuracy of the positioning results in snap-shot mode, we use the software receiver to receive actual satellites signals and positioning in real-time. A high precision receiver is used to calibrate the position of the antenna in advance. The software results are contrasted with the measurement position to

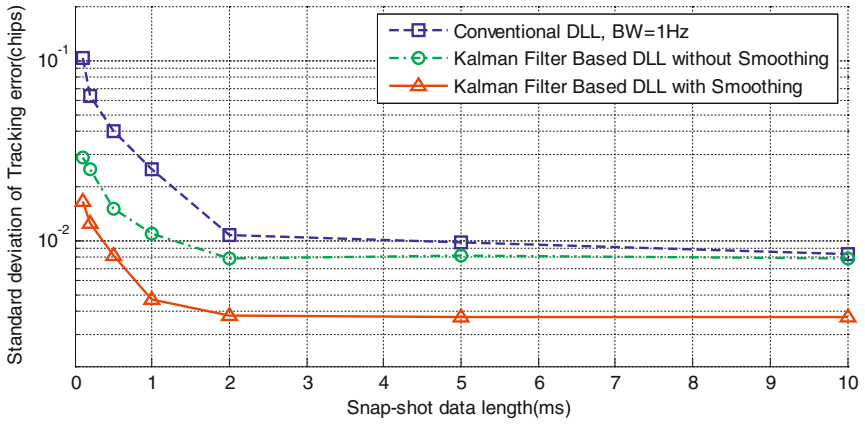

Fig. 59.6 The standard deviation of the tracking error versus the snap-shot length

obtain the positioning error. Meanwhile, in order to illustrate the performance of the receiver, the continuous tracking results are also used to make the contrast.

Figure 59.7 shows the north and east positioning error of the continuous tracking mode and the snap-shot working mode. In snap-shot working mode, due to the limited snap-shot data, so the positioning accuracy would be worse than continuous tracking mode. In the snap-shot receiver, the RMS of the horizontal positioning error is 4.1415 m without smoothing and 2.9986 m with smoothing. If the length of the snap-shot data is increased, then, the equivalent loop width may be further reduced, and thereby it is possible to obtain higher accuracy of the positioning results. However, it also increases the computational complexity and resource utilization. It can be clearly seen from Fig. 59.7 that the snap-shot receiver with smoothing have the smaller RMS error and more focused positioning results. The horizontal RMS error with proposed method, 2.9986 m, has been able to meet most of the LBS applications' demand.

Figure 59.8 shows the cumulative probability distribution of the horizontal positioning error of the continuous tracking mode and the snap-shot working

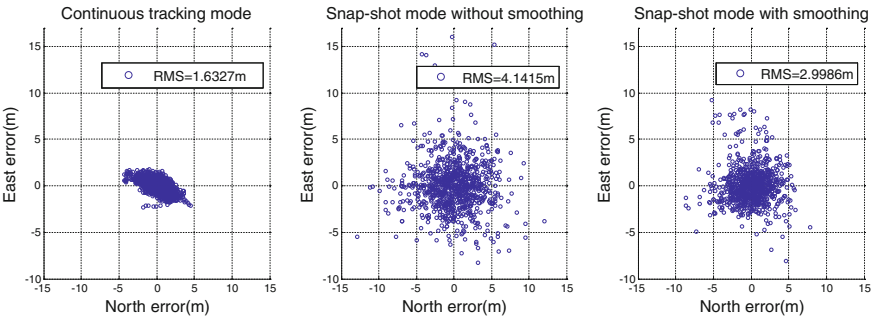

Fig. 59.7 The scatter plot of north and east positioning error

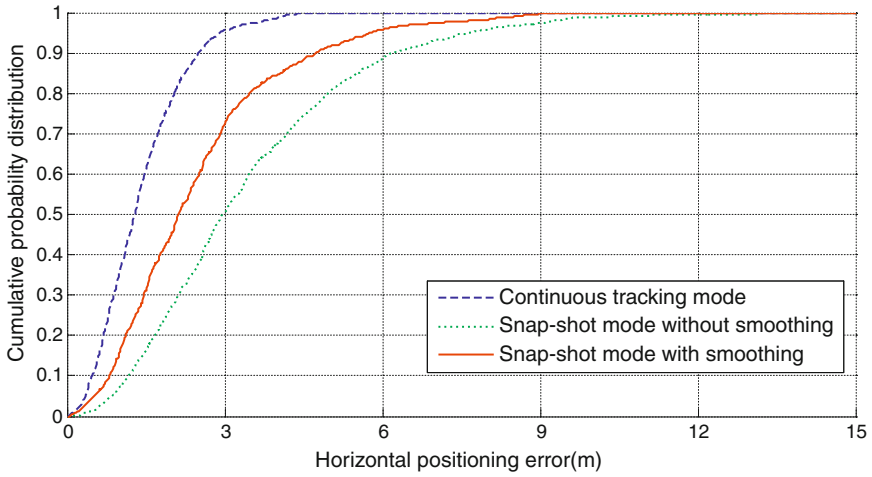

Fig. 59.8 The cumulative probability distribution of horizontal positioning error

mode. It can be observed that by use of smoothing, most of the positioning results are more concentrated near the true value, and more than 90 % of the positioning error is less than 5 m which results cannot be obtained without smoothing. The proposed method achieves the better performance without increasing the snap-shot length. So the required resource and the computational complexity can be reduced.

59.5 Conclusions

In this paper, for the poor accuracy of a snap-shot receiver within a limited snap-shot length, a novel code phase measurement technique has been designed. In this technique, the code phase measurement and extraction is complete in two steps, including a KF based DLL and a non-casual fixed-point smoother. The KF based DLL is used to maintaining code tracking, further, the non-causal smoothing estimator is added to improve measurement accuracy of code phase and obtain more accurate positioning results. This proposed method has been tested using experiment and actual GPS C/A data. The simulation results show that with the same snap-shot length, the proposed method has better estimation accuracy than the loop without smoothing and the conventional loop in common CNR environments. After all loops enter the steady-state, the accuracy of proposed method has 3.3 dB higher than the other two methods. Actual positioning results show that with 500 ms snap-shot length, the snap-shot receiver by use of smoothing has the smaller horizontal RMS error and more focused positioning results. With the proposed method, the horizontal RMS error is 2.9986 m and more than 90 % of the positioning error is less than 5 m. The result is enough for most of the LBS applications.

References

1. Yao Z, Lu M, Feng Z (2010) Spreading code phase measurement technique for snapshot GNSS receiver. IEEE Trans Consum Electron 56(3):1275–1282
2. Qian Y, Cui X, Lu M et al (2008) Snapshot positioning for unaided GPS software receivers. ION GNSS 2008, Savannah, US, pp 2342–2350
3. Knight J, Turetzky G, Norman C et al (1998) SiRF's low power receiver advances. ION GPS 1998, Nashville, US, pp 299–305
4. David JB, Nuria BD, Gustavo LR et al (2006) Innovative techniques for GPS indoor positioning using a snapshot receiver. ION GPS 2006, FortWorth, US, pp 2944–2955
5. Driessem PF (1994) DPLL bit synchronizer with rapid acquisition using adaptive Kalman filtering techniques. IEEE Trans Commun 43(9): 2673–2675
6. Psiaki ML (2001) Smoother-based GPS signal tracking in a software receiver. ION GPS 2001, Institute of Navigation, Alexandria, pp 2900–2913
7. Parkinson BW, Spilker JJ (1996) Global positioning system: theory and applications, vol 1. American Institute of Aeronautics and Astronautice, US
8. Kay SM (1993) Fundamentals of statistical signal processing estimation theory, vol 1: estimation theory. Prentice-Hall, New York
9. Andrew S, James M (1971) Estimation theory with applications to communications and control. McGraw-Hill Book Company, US