An open source GPS/GNSS vector tracking loop – Implementation, filter tuning, and results

Conference Paper · January 2011

2 authors:

Sihao Zhao
Tsinghua University
8 PUBLICATIONS **21** CITATIONS
SEE PROFILE

Dennis Akos
University of Colorado Boulder
98 PUBLICATIONS **1,483** CITATIONS
SEE PROFILE

An Open Source GPS/GNSS Vector Tracking Loop – Implementation, Filter Tuning, and Results

Sihao Zhao, Dennis Akos, *University of Colorado*

BIOGRAPHY

Sihao Zhao obtained his B.E degree in Electronic Engineering at Tsinghua University, China, 2005. Since 2007, he has been working as a Ph.D. candidate in the Department of Electronic Engineering, Tsinghua University. Currently, he is working as a visiting researcher in the Department of Aerospace Engineering Sciences, University of Colorado.

Dennis Akos completed the Ph.D. degree in Electrical Engineering at Ohio University within the Avionics Engineering Center. He has since served as a faculty member with Luleå Technical University, Sweden, and then as a researcher with the GPS Laboratory at Stanford University. Currently he is a faculty member with the Aerospace Engineering Sciences Department at the University of Colorado, Boulder.

ABSTRACT

This paper first briefly introduces the advantages of vector tracking loops (or vector lock loops - VLL) over traditional scalar tracking loops (or scalar lock loops - SLL) in a GNSS receiver. Then, some principles and the mechanism of how VLL works are illustrated. Based on an open source GPS software-defined receiver code, the VLL is implemented and tested with three different data sets including two from a GPS simulator and one from a drive test. To compare the performance, a traditional SLL is also used to process these three datasets. Their results are included. Comparisons between scalar loop and vector loop observables including carrier frequency results and position/velocity results are presented. Based on the details illustrated in this paper, one can easily understand the mechanism of the vector tracking algorithm and incorporate this in their receiver code. The tests carried out in this paper are not exhaustive but provide solid insights into the implementation and tuning associated with the vector loops. Test results show that the vector tracking loops have definite advantages over traditional scalar loops including better position accuracy and higher robustness under low signal-to-noise ratios. However, to obtain these benefits, the vector loop parameters should be carefully tuned depending on different scenarios the receiver is going through. Meanwhile, some criteria of tuning these parameters are given empirically in this work.

1. INTRODUCTION

Starting with GPS, GNSS systems have become widely used in a variety of navigation applications. However, in many cases such as urban canyons, dense foliage, high dynamics and others, the reliability of a GNSS receiver is severely challenged. Among the components of a GPS receiver, tracking loops are one of the most critical elements. These loops process received signals with an ultimate goal of a position, velocity and time (PVT) solution. However, the code and carrier tracking loops are vulnerable to low signal to noise ratio and high dynamics. Based on the principles of calculating navigation solutions, at least 4 satellites are needed to provide 3-dimensional positions and clock bias[1]. Therefore, when the amount of available satellites drops below 4 due to interference or blockage, the receiver will fail to navigate.

In a traditional GPS receiver, tracking loops work independently for all channels. The inherent connections based on the same user position and velocities between channels are neglected. Thus, there is no information exchange between channels which makes aiding from channels with strong signals to those with weak signals impossible.

The concept of vector tracking lock loops was first proposed by Spilker [2]. It is an appealing and advanced structure which can provide a performance improvement over traditional scalar lock loops. As opposed to scalar loops, vector loops use all signals from different satellites both to track carrier frequency/code phase and to calculate navigation solutions in the position domain [3]. Vector tracking loops simultaneously process signals from all channels and meanwhile give out the navigation solutions. In vector tracking loops, the feature of jointly processing all channels can exploit more information buried in signals, make it possible for channels to aid each other and thus enhance the performance in low carrier to noise ratio environments [4].

The basic idea of vector tracking loops is that every channel's code phase (or pseudorange) and carrier frequency (or pseudorange rate) are connected to the user position and velocity. In a traditional receiver, we calculate those navigation solutions using pseudoranges and pseudorange rates. Inversely, code phases and carrier frequencies can be calculated by projecting the relative position and velocity between the receiver and corresponding satellite on the line-of-sight (LOS) direction. Therefore, all channels' signals are connected with one another through the same user position and velocity.

In the research of GNSS, software-defined receivers (SDR) are widely recognized and utilized for their configuration flexibility and easiness. The open source Matlab SDR code composed by Borre and Akos [5] is widely spread and used in education and research. This code post-processes collected GNSS intermediate frequency data using traditional SLL and handles different channels sequentially. In its tracking functions, DLL and PLL, whose parameters such as bandwidth and damping ratio can be configured, are utilized to offer the code phase/frequency and carrier phase/frequency. Iterative least squares method uses pseudoranges and range rates as inputs to generate navigation solutions. Many intermediate variables can be extracted for the purpose of understanding a GNSS receiver or realizing new algorithms by modifying this open source SDR code. However, when dealing with large datasets with tens of minutes data or high sampling frequency, the SDR code seems to have slightly low efficiency. But its flexibility offers enough convenience for implementation of our VLL here.

2. MECHANISM OF VLL

In a digital GPS receiver, to obtain pseudoranges and pseudorange rates, a section of received data samples are used to correlate with local replicas. At a specific time epoch (denoted as *k*) in the received data samples, we assume that the code phase, code frequency, carrier frequency, signal transmit time, clock bias/drift and user position/velocity at the beginning point are perfectly known. Denote them as $\varphi_{j,k}$, $f_{code,j,k}$, $f_{j,k}$, $t_{tr,j,k}$, $t_{b,k}$, $t_{d,k}$, X_k and V_k, the subscript '*j*' represents the *j*th satellite and '*k*' represents the time epoch of the data sample.

Predictions of user position and velocity at time epoch *k+1* are generated based models using these initial values. Denote the predictions by topping '^' on variables such as $\hat{X}_{k+1}$. In this work, we predict position and velocity as the following equations:

$$\hat{X}_{k+1} = X_k + t_{k,k+1} V_k \quad (1)$$

$$\hat{V}_{k+1} = V_k \quad (2)$$

where $t_{k,k+1}$ is the time interval between *k*th and *k*+1th epoch.

Based on the predicted user position/velocity and satellite positions/velocities from the ephemeris, code phases and carrier frequencies of all satellites at time epoch *k+1* can be predicted. And the local code and carrier replicas from time epoch *k* to *k+1* could be generated. After correlating them with the corresponding section of the received signal, code phase and carrier frequency errors plus noises are output from discriminators. The reason why we only use carrier frequency discriminator instead of carrier phase discriminators is that carrier phase tracking needs more critical conditions than frequency discriminators and does not fit for low C/N0 environments. Furthermore, in vector lock loops, we no longer care about the difference between carrier phases or navigation data bits which can be obtained from scalar lock loops, and thus a frequency discriminator is sufficient. The output of the code phase discriminator contains the average code phase error plus noise of the entire sample section. Usually, if the section has a short length (e.g. 1ms), this average code phase error approximately represents the error at the middle point of the section and can be converted to the code phase error at the last sample point in the data section. Similarly, the output of frequency discriminator contains the average frequency error plus noise of the same section and can also be converted to the error at the last sample point.

The code phase error and frequency error from discriminators contain the corresponding LOS projection of the errors between the true position/velocity and their predicted counterparts, i.e. $\left(X_{k+1} - \hat{X}_{k+1}\right)^T a_{j,k}$ and $\left(V_{k+1} - \hat{V}_{k+1}\right)^T a_{j,k}$, where '$a_{j,k}$' is the unit LOS vector between the receiver and the *j*th satellite at *k*th epoch. Once the errors $X_{k+1} - \hat{X}_{k+1}$ and $V_{k+1} - \hat{V}_{k+1}$ are obtained, we

can use them to fix $\hat{X}_{k+1}$ and $\hat{V}_{k+1}$ to get the final 'true' position and velocity. The relationship between the code phase error and the user position error can be written as:

$$\begin{aligned} E_{code,k} &= \hat{\varphi}_{j,k} - \varphi_{j,k} + \eta_{j,k} \\ &= t_{b,k} + \left(X_k - \hat{X}_k \right)^T a_{j,k} + \eta_{j,k} \end{aligned} \quad (3)$$

where, $E_{code,k}$, φ, $t_{b,k}$ are in meter, η is the noise term.

The relationship between carrier frequency error and user velocity error is:

$$\begin{aligned} E_{carrier,k} &= f_{j,k} - \hat{f}_{j,k} + w_{j,k} \\ &= \Delta t_{d,k} + \left(V_k - \hat{V}_k \right)^T a_{j,k} + w_{j,k} \end{aligned} \quad (4)$$

where, $E_{carrier,k}$, f, $\Delta t_{d,k}$, are in the unit of m/s, w is the noise term, and $\Delta t_{d,k}$ is the difference between $t_{d,k}$ and $t_{d,k-1}$.

As the analysis and the equations shown above, if the position/velocity errors can be estimated, the navigation solution could be obtained. Therefore we choose the position/velocity errors as states and use Kalman filtering to estimate them. Denote the position/velocity errors as δX and δV. For each time duration between the kth and k+1th epoch, we assume that the position at time epoch k is known, so the position error at time epoch k+1 is totally determined by the integration of velocity error during kth~k+1th time epoch. If time duration is short enough, we can assume the velocity error remains constant. Thus, the position error is modeled as:

$$\delta X_{k+1} = t_{k,k+1} \delta V_k + n_{k+1}$$

where n_k represents the un-modeled error terms which can be considered as a WGN sequence.

Similarly, at time epoch k, the velocity is known, so the velocity error can be modeled as a WGN sequence.

$$\delta V_{k+1} = v_{k+1}$$

Given the position and velocity, code phase/frequency and carrier frequency prediction equations for time epoch k+1 are:

$$\hat{\varphi}_{j,k+1} = \varphi_{j,k} + \left(\Delta X_{j,k,k+1} - t_{k,k+1} v_k \right)^T a_{j,k+1} + t_{k,k+1} c \quad (5)$$

$$\hat{f}_{code,j,k+1} = \left[1 + t_{d,k} + \left(V_{j,k} - v_k \right)^T a_{j,k+1} \right] f_{code,j,k} / c \quad (6)$$

$$\hat{f}_{j,k+1} = \left[1 + t_{d,k} + \left(V_{j,k} - v_k \right)^T a_{j,k+1} \right] f_n / c \quad (7)$$

where φ is in meter, $t_{k,k+1}$ is in second, $\Delta X_{j,k,k+1}$ is the displacement vector of the jth satellite during kth and k+1th epoch, $t_{d,k}$ is in m/s, $V_{j,k}$ is the velocity of the jth satellite during k and k+1 epoch, f_{code} and f_n are the nominal code and carrier frequency (1575.42MHz for L1 and 1.023MHz for C/A code), c is the speed of light.

As for the receiver clock error terms, the clock bias during every interval between kth and k+1th epoch is caused by the clock drift, and can be modeled as follows,

$$t_{b,k+1} = t_{b,k} + t_{k,k+1} \left(t_{d,k+1} - t_{d,k} \right) = t_{b,k} + t_{k,k+1} \Delta t_{d,k}$$

We model clock drift as

$$t_{d,k+1} = t_{d,k} + \Delta t_{d,k} \quad (8)$$

where $\Delta t_{d,k-1}$ is the difference between $t_{d,k+1}$ and $t_{d,k}$. Oscillators used in GPS receivers always have relatively stable drift during short time period (e.g. 1ms), so we can model $\Delta t_{d,k}$ as a WGN sequence, i.e. $\Delta t_{d,k} = r_k$.

Therefore, the states of the system include δX, δV, t_b and Δt_d. The discrete process equation can be written as:

$$\begin{bmatrix} \delta x_{k+1} \\ \delta y_{k+1} \\ \delta z_{k+1} \\ \delta v_{x,k+1} \\ \delta v_{y,k+1} \\ \delta v_{z,k+1} \\ t_{b,k+1} \\ \Delta t_{d,k+1} \end{bmatrix} = F_{k,k+1} \begin{bmatrix} \delta x_k \\ \delta y_k \\ \delta z_k \\ \delta v_{x,k} \\ \delta v_{y,k} \\ \delta v_{z,k} \\ t_{b,k} \\ \Delta t_{d,k} \end{bmatrix} + W_k \quad (9)$$

where,

$$F_{k,k+1} = \begin{bmatrix} 0 & 0 & 0 & t_{k,k+1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & t_{k,k+1} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & t_{k,k+1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & t_{k,k+1} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The outputs of code phase and carrier frequency

discriminators are employed as measurements. Based on equations (3) and (4), the measurement equation is:

$$Z_k = HX_k + V_k = \left[z_{code,1,k} \quad z_{carrier,1,k} \quad \cdots \quad z_{code,n,k} \quad z_{carrier,n,k} \right]^T_{1\times 2n} \tag{10}$$

where the terms of H matrix are determined by the following equations:

$$z_{code,j,k} = a_{jx,k}\delta x_k + a_{jy,k}\delta y_k + a_{jz,k}\delta z_k + t_{b,k} + \eta_{j,k}$$

$$z_{carrier,j,k} = a_{jx,k}\delta v_{x,k} + a_{jy,k}\delta v_{y,k} + a_{jz,k}\delta v_{z,k} + \Delta t_{d,k} + w_{j,k}$$

3. IMPLEMENTATION OF VLL

Figure 1 shows the flowchart of how VLL works. Different from SLL, feedback from the navigation filer to the local replica generator enables all channels to share information based on the same navigation solutions so they no longer work individually. In accordance with this principle, the original SDR code should be modified. First, the tracking and navigation functions are combined, and then the Costas discriminator is replaced by a frequency error discriminator. Next, to obtain velocity solutions, satellite and user velocity calculation functions are added into the orignal SDR code.

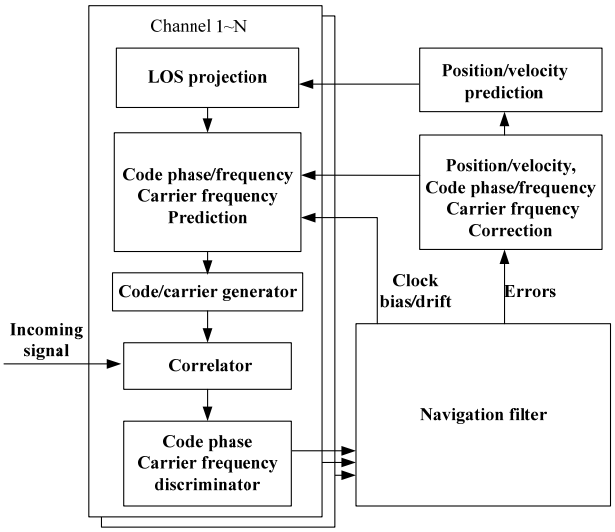


Figure 1. Work flowchart of VLL

To initialize the VLL, some values are needed including code phase and frequency, carrier frequency from each channel, user position and velocity and signal transmit time corresponding to the VLL starting sample point in the received signal. Therefore, the receiver should obtain these values using traditional scalar loops to decode the navigation bits before VLL starts. Using the original SDR code, we could obtain user and satellite position, code phase/frequency, carrier phase/frequency, ephemeris data and signal transmit time. We still need user velocity and satellite velocity. Therefore, we employ the derivative of the satellite position broadcast in ephemeris data to determine satellite velocity and least square equations using frequency results from PLL to determine user velocity.

Once initialized, VLL begins to 1) predict navigation parameters, 2) generate local replicas and correlate them with the incoming signal, 3) estimate the errors and correct the predictions and start over again to form a closed loop.

1) Prediction. When the navigation solutions of kth epoch are known, we can predict the navigation parameters for k+1th epoch based on any models – from simply using the navigation parameters of the kth epoch to leveraging a complicated IMU dynamic model if external inertial sensor is available. In this work, we add the integration result of k~k+1th epoch interval's velocity to the kth epoch's position to get the k+1th epoch's position as equations (1) and (2). Also, code phase and carrier frequency for each channel should be predicted as equations (5), (6) and (7). With ephemeris data and transmit time for the k+1th epoch, satellite positions and velocities can be obtained. Using predictions from step 1), the LOS vectors could be calculated. We should notice that, the LOS vectors may have some errors caused by the inaccurate prediction of the user position. However, due to the long distances between the user and satellites, these errors in LOS vectors can be ignored.

2) Local replicas. After projecting the predicted position and velocity onto LOS directions, we get the predicted code phases/frequencies and carrier frequencies at k+1th epoch which are put into code and frequency generators to produce local replicas.

We choose a noncoherent early minus late envelope normalized by E+L function as the code phase discriminator to remove amplitude sensitivity [1].

$$\frac{1}{2}\frac{E-L}{E+L}$$

where $E = \sqrt{I_E^2 + Q_E^2}$, $L = \sqrt{I_L^2 + Q_L^2}$, I_E and I_L represent the inphase early and late correlator outputs during time epoch k and k+1, similarly, Q_E and Q_L represent the quadrature early and late correlation results over the same duration.

We use a normalized decision directed frequency discriminator to eliminate the data bit transition sensitivity.

$$\frac{cross \times sign(dot)}{2\pi(t_2 - t_1)(I_{P2}^2 + Q_{P2}^2)}$$

where $cross = I_{P1}Q_{P2} - I_{P2}Q_{P1}$, $dot = I_{P1}I_{P2} - Q_{P1}Q_{P2}$, I_{P1} and I_{P2} represent the inphase prompt correlation results over epochs *k*-1~*k* and *k*~*k*+1, Q_{P1} and Q_{P2} represent the quadrature prompt correlation results over the same duration.

We should notice that the final results of navigation solutions should correspond to a certain sample point in the received signal. Thus, code phases and carrier frequencies of different satellites should be aligned to this sample point. However, signals transmitted at the same time from different satellites reach the receiver antenna at different instants. This makes it possible for us to use one particular section in the received signal to correlate with different local replicas for different satellites; but it is possible that one certain satellite's signal has data bit transition in this particular signal section. In the worst case, the bit transition occurs in the middle of the signal section and ruins the entire correlation results. To avoid bit transition in the middle of the section, we can find all the sample positions in the received signal for all satellites where their C/A code phases are close to zero. For different satellite signals, select different sections starting at these corresponding sample positions and use them to correlate with local replicas. Then, the discriminator outputs are not aligned to the same sample point but can be adjusted using some fitting method. Here, we assume that between adjacent instants when discriminators output, the code phase and carrier frequency error change linearly with time, so we can calculate the discriminator outputs at any point during this time interval.

3) Error estimation and compensation. Discriminator outputs of all satellites are input into Kalman filter as measurements. Process and measurement equations are shown as (9) and (10). After those states are estimated, we use them to correct the predictions generated in step 1), as

Position: $X_{k+1} = \hat{X}_{k+1} + \delta X_{k+1}$

Velocity: $V_{k+1} = \hat{V}_{k+1} + \delta V_{k+1}$

Code phase: $\varphi_{j,k+1} = \hat{\varphi}_{j,k+1} + \delta X_{k+1}^T a + t_{k,k+1} c + t_{b,k}$

Code frequency:

$$f_{code,j,k+1} = \hat{f}_{code,j,k+1} + \left(t_{d,k+1} + \delta v_{k+1} a\right) f_{code} / c$$

Carrier frequency: $f_{j,k+1} = \hat{f}_{j,k+1} + \left(t_{d,k+1} + \delta v_{k+1} a\right) f_n / c$

Using these corrected values as initial values, the loop can be started over again.

4. PARAMETERS TUNING

In a Kalman filter, the process and measurement noise covariance matrices are the most important parameters determining the performance of the filter. In the VLL proposed in this paper, the *Q* matrix represents the uncertainty in the dynamics of the user and the *R* matrix represents the noise terms in the discriminator outputs. This section will discuss the tuning of the two matrices mainly on an empirical basis.

Denote Q_k as the covariance matrix of process noise W_k.

$$Q_k = \begin{bmatrix} Q_{dyn} & 0 \\ 0 & Q_{clk} \end{bmatrix}$$

where Q_{dyn} corresponds to the three-axis user dynamics and Q_{clk} corresponds to the equivalent dynamics caused by clock errors.

$$Q_{dyn} = diag\left(\sigma_x^2, \sigma_y^2, \sigma_z^2, \sigma_{v_x}^2, \sigma_{v_y}^2, \sigma_{v_z}^2\right)$$

$$Q_{clk} = \begin{bmatrix} Q_{11} & 0 \\ 0 & Q_{22} \end{bmatrix}$$

where Q_{11} is the clock bias noise term and Q_{22} is the clock drift error term. As equation (8) shows, the clock drift error is modeled as a WGN sequence, so Q_{22} can be set based on the variance of clock drift given by SLL.

To determine the terms for user dynamics, we should have some a priori knowledge of the user velocity and acceleration which highly depend on the specific scenario the receiver is taking. For example, with some moderate movement, setting σ_x^2 and $\sigma_{v_x}^2$ to the order of 10 and 1 respectively is reasonable. However, these values are very coarse and empirical estimations, and need to be considered with care in different scenarios. From the view of a traditional tracking

loop, increasing Q_{dyn} (relying less on process models so as to improve the ability for tracking user dynamics) equals to widening the loop bandwidth, vice-versa.

Denote R_k as the covariance matrix of the measurement noise V_k, and we assume that outputs from different discriminators are independent so R_k is a diagonal matrix.

$$R_k = \begin{bmatrix} R_{code,1} & & & & \\ & R_{carrier,1} & & & \\ & & \cdots & & \\ & & & R_{code,n} & \\ & & & & R_{carrier,n} \end{bmatrix}_{2n\times 2n}$$

Similarly to process noise, R_k contains terms for code phase and carrier frequency errors. An empirical setting method for R_k is based on the variance of measurements. These values can be obtained by calculating the variances of a certain time length of discriminator outputs. It is worth noting that theoretically the longer time length gives closer estimate to the true variances, but the estimate cannot keep up with the change of values if we choose a very long time length. In our experiments for this work, we use the continuous 500ms measurements to get the variances. From the view of a traditional tracking loop, increasing the terms in R (relying less on measurements so as to gate off more measurement noises) equals to narrowing the loop bandwidth, vice-versa.

5. EXPERIMENT RESULTS

5.1 EXPERIMENT OVERVIEW

In this part, we use three datasets collected from both a simulator and a drive test to demonstrate the performance of VLL. The first two datasets were from a Spirent 12-channel GPS simulator simulating a vehicle moving in figure 8 trajectories. The differences between them are that in the first dataset, all satellite signal powers remain constant at about 50dB-Hz until 50s, and then all begin dropping down at -0.25dB/s; and in the second dataset, only PRN11 and PRN 19's signal power decreases from 50s at -0.17dB/s. The third dataset was collected in Boulder, Colorado, USA, using a Record and Play-back System and a NovAtel Span system as truth reference. In all three experiments, we set the VLL parameters as in table 1. The satellite sky plots of the three datasets are shown in figure 2 and figure 3.

Table 1. Settings of Vector tracking loops for all datasets

Q			
Position error $\sigma^2(m^2)$	Velocity error $\sigma^2(m^2/s^2)$	Clock bias $\sigma^2(m^2)$	Clock drift $\sigma^2(m^2/s^2)$
20	3	1e-7	0.1
R			
Code phase error $\sigma^2(m^2)$		Carrier frequency error $\sigma^2(m^2/s^2)$	
Variance of 500ms discriminator outputs		Variance of 500ms discriminator outputs	

We also use the original SDR code with traditional SLL to process the three datasets and compare the results with VLL. The SLL settings for all datasets are summarized in table 2.

Table 2. Common SLL settings for all datasets

Code/carrier pre-detection time	1ms
Code phase discriminator	$\frac{1}{2}\frac{E-L}{E+L}$
Carrier phase discriminator	$\tan^{-1}\left(Q_p / I_p\right)$

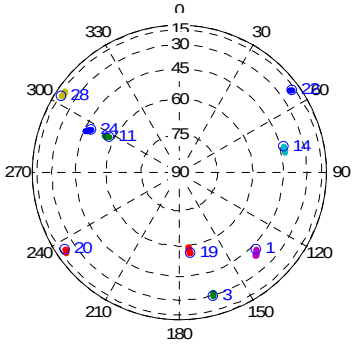


Figure 2 Sky plot of dataset 1 and dataset 2 (Simulated 1 & 2)

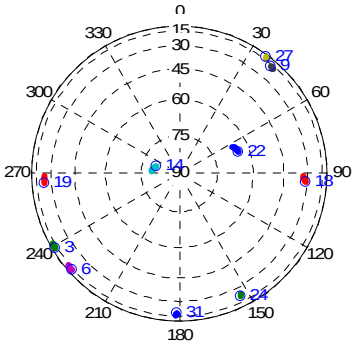


Figure 3 Sky plot of dataset 3 (Live data)

5.2 RESULTS OF SIMULATED DATA 1

As mentioned in section 5.1, the signals in this dataset start to

decrease their power since 50s. To verify the vector tracking loops' performance in low C/N0 environment, we only demonstrate the results of VLL and SLL from 105s to 185s. To make the comparison more convincing, we set the PLL bandwidth down to 8Hz and DLL bandwidth down to 1Hz for SLL which can barely track the dynamics in the signal.

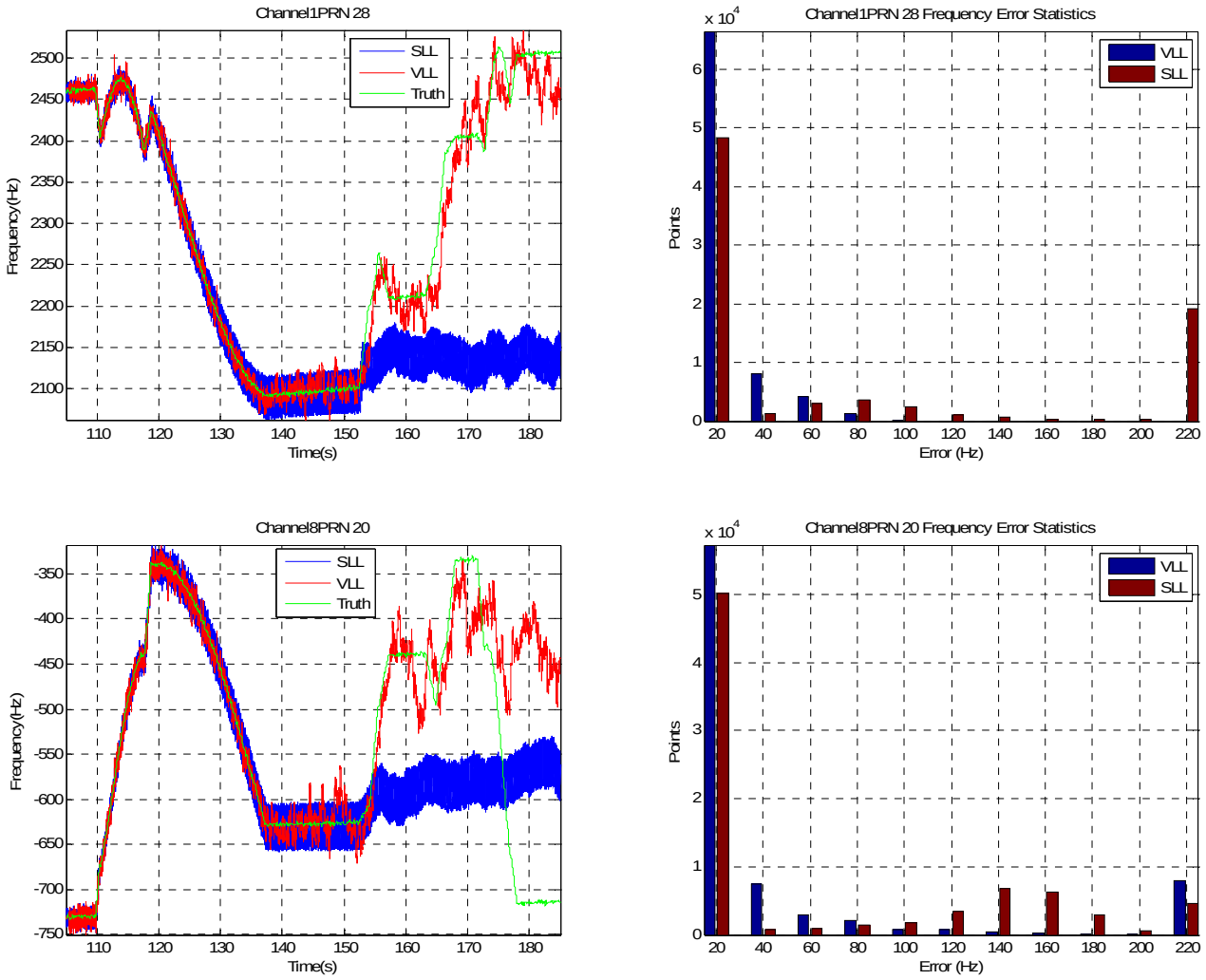
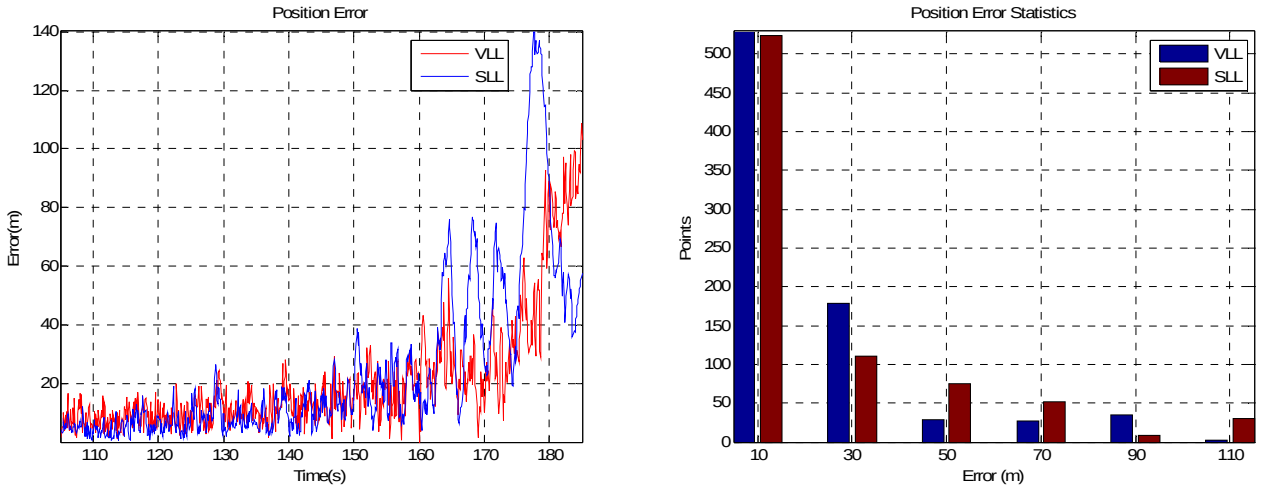


Figure 4. Frequency results of PRN28 and PRN20 (Simulated 1)



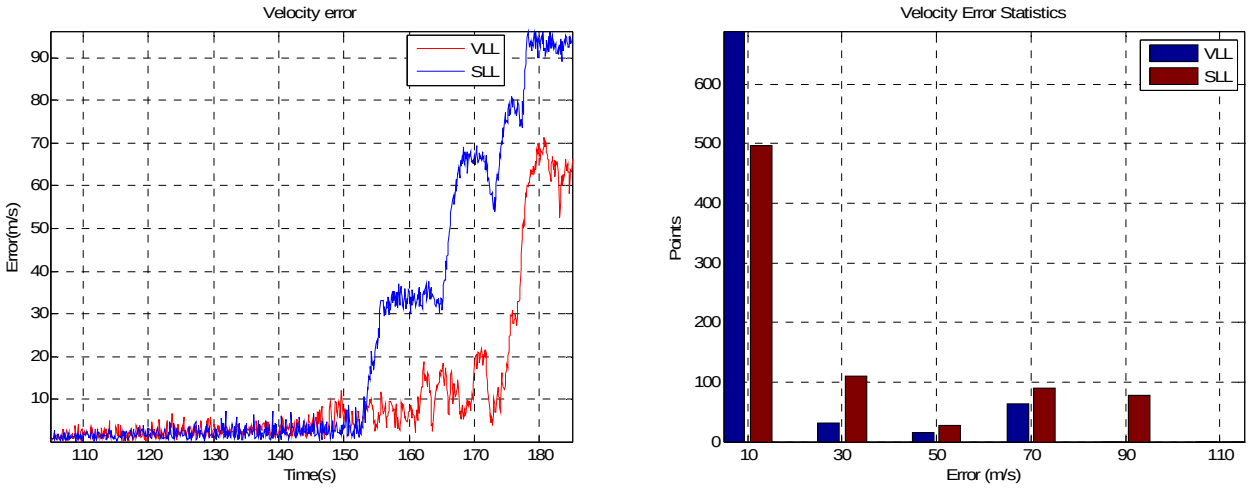
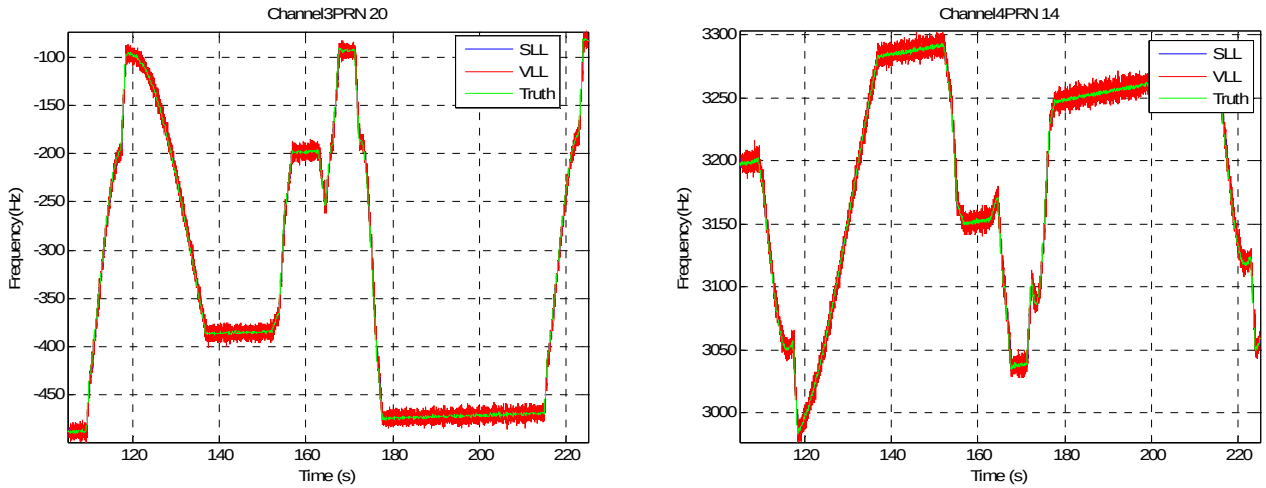


Figure 5. Navigation solutions (Simulated 1)

We only list the frequency results of two channels in fig 4, other channels are similar and not listed here. From fig 4, we can observe that the SLL loses lock at about 153s and VLL at about 170s which illustrates that VLL can maintain frequency lock longer than SLL, and provided the signal attenuating rate 0.25dB/s, the gain of VLL in this experiment is about 4dB over SLL. As to navigation solutions, the accuracy of position solutions is slightly improved while velocity solutions are improved more significantly.

5.3 RESULTS OF SIMULATED DATA 2

In this experiment, only two satellites' (PRN11 and PRN19) signal power decreases since 50s in the dataset. Here, we only use the combination of 4 satellites (PRN11, PRN19, PRN14 and PRN20) to generate navigation solutions for both SLL and VLL. For SLL, we set the PLL bandwidth to 10Hz and DLL bandwidth to 1Hz which are the lower limit that can barely track the signal dynamics. Based on the principle of GPS navigation, at least 4 satellites are needed to obtain the navigation solutions including X, Y and Z axis positions and clock bias. Therefore, when the signal power of 2 satellites becomes too weak and only the other 2 satellites are available, the traditional receiver with SLL cannot provide navigation solutions. However, when we use VLL, this situation can be improved and the results of both SLL and VLL are shown in fig 6 and fig 7.



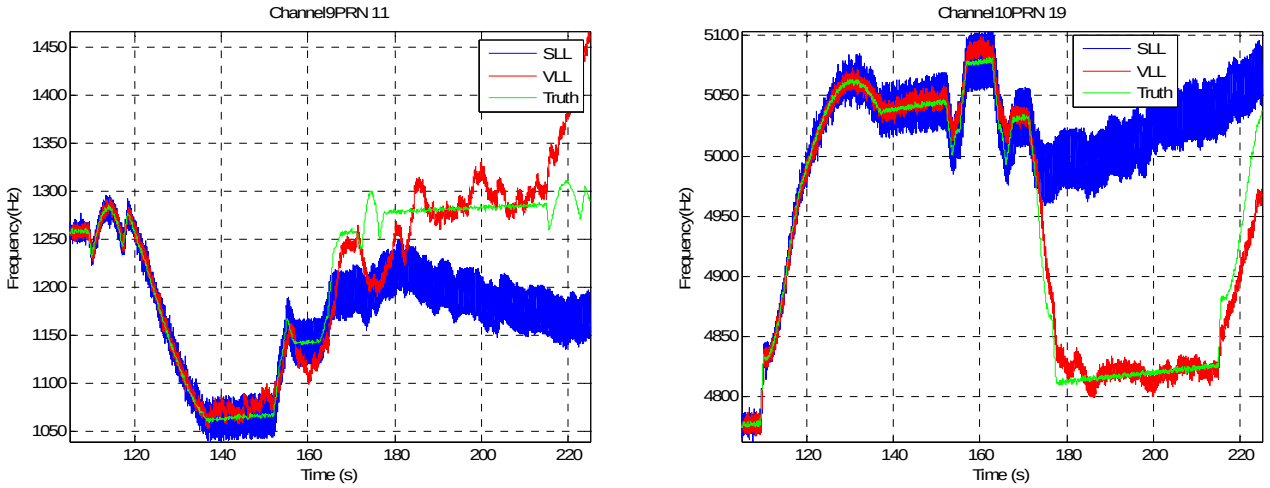


Figure 6. Frequency results (Simulated 2)

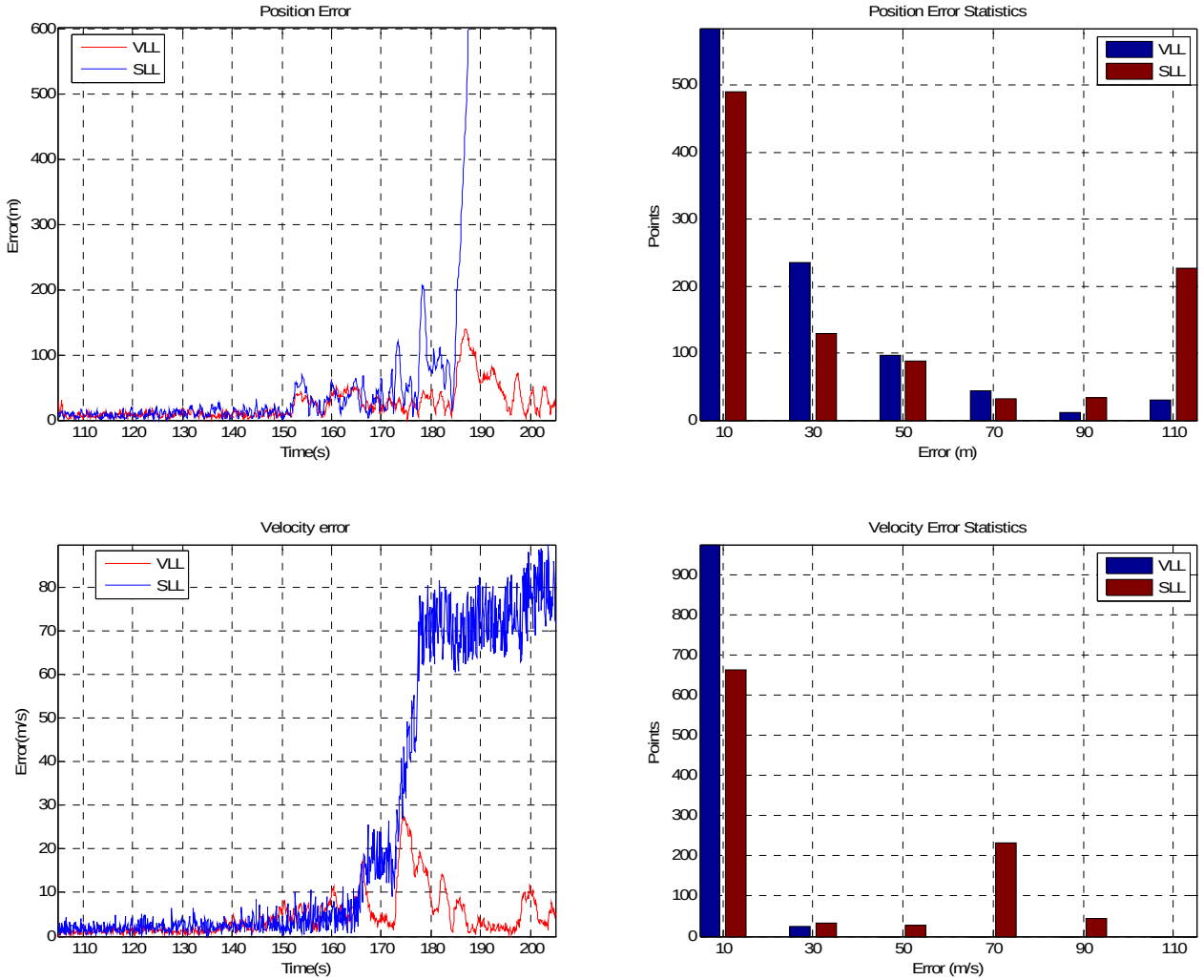


Figure 7. Navigation solutions (Simulated 2)

From the frequency results, we can observe that VLL maintains lock longer than SLL for PRN11 and PRN19. In the lower left of fig 6 showing the frequency results of PRN19, SLL loses lock at about 175s and VLL loses lock at about 220s. As the signal power decreasing rate is 0.17dB/s, we can roughly estimate that for PRN19 in this scenario, VLL is able to work under a 7dB lower C/N0 environment than SLL. Moreover, VLL also provides better position and velocity solutions than SLL. Other satellites combinations have similar results which are not shown here.

5.4 RESULTS OF LIVE GPS SIGNAL

We collected this dataset from a drive test in Boulder, Colorado, USA. In the first 100s, the car was in an open-sky parking lot to ensure that the receiver outputs stable navigation solutions and the IMU is calibrated. Then the car drove into the canyons West of Boulder where in many places satellite signals are attenuated or even blocked. The true trajectory of the 1670s dataset is obtained from a GPS/INS truth reference system and is shown in red line in fig 8.

Figure 8. True trajectory of live data on Google map

Fig 9 shows the amount of visible satellites that can be acquired during the entire dataset. The red line and green line represent 4 satellites and 3 satellites respectively.

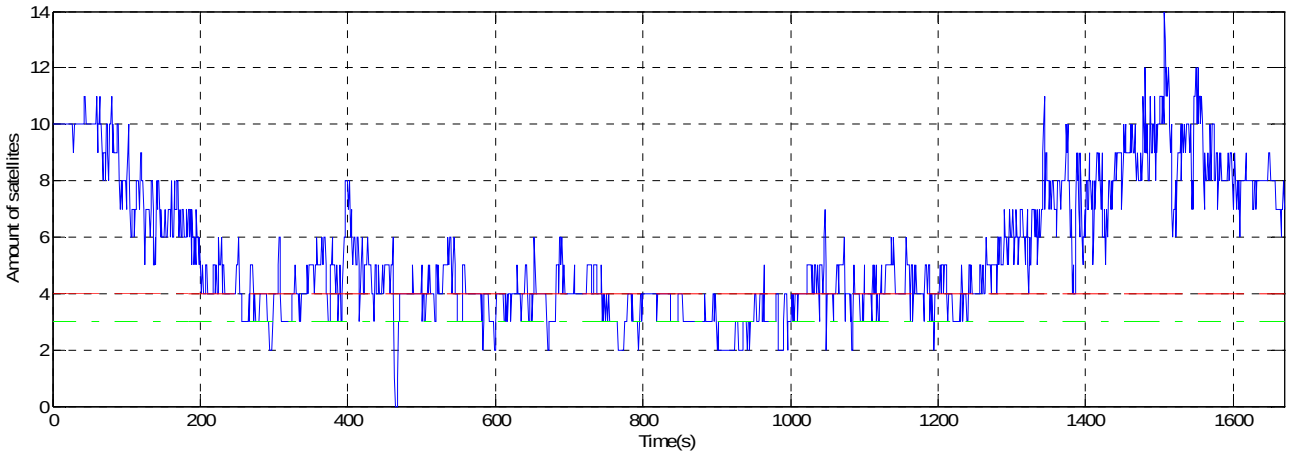


Figure 9. Amount of visible satellites in live data

In the middle part of fig 9 from about 200s to 1300s, time sections with less than 4 available satellites occur frequently during which traditional receivers with SLL will fail to navigate.

We start the VLL at 70s in the dataset using the tracking results, ephemeris and navigation solutions from SLL to initialize VLL. Fig 10 and fig 11 show the navigation results of SLL and VLL for the first 350s.

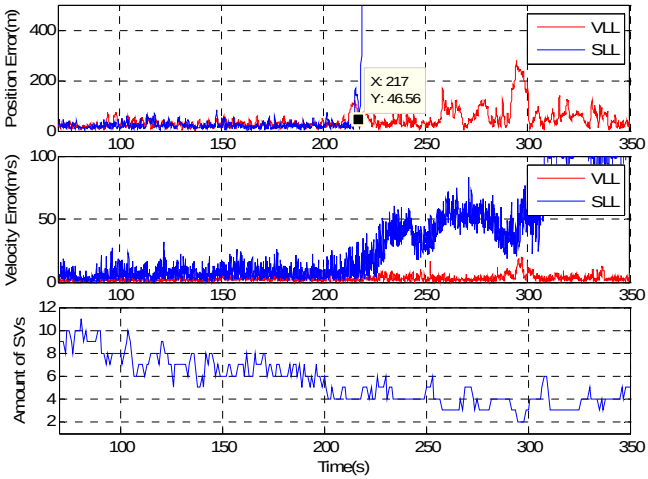


Figure 10. Navigation solution errors for the first 350s (Live data)

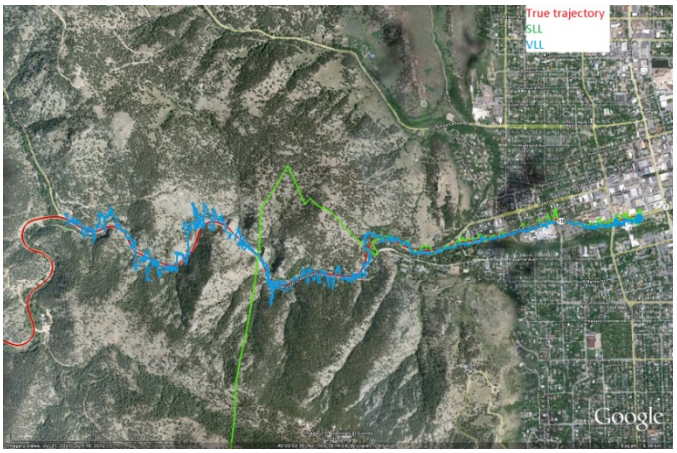


Figure 11. Position results of VLL and SLL on the map (Live data)

We can observe that at about 217s, SLL cannot provide accurate enough navigation solutions while VLL can still work. Although in fig 9, after 217s, there are still 4 available satellites that can be acquired, we should notice that to investigate how many satellites are available in every time section, we use 10ms for coherent integration time and 4 for non-coherent summations. However, to compare SLL and VLL based on the same conditions, we only use 1ms as pre-detection integration time for both of them, so weak signals may not be tracked by SLL even they can be acquired.

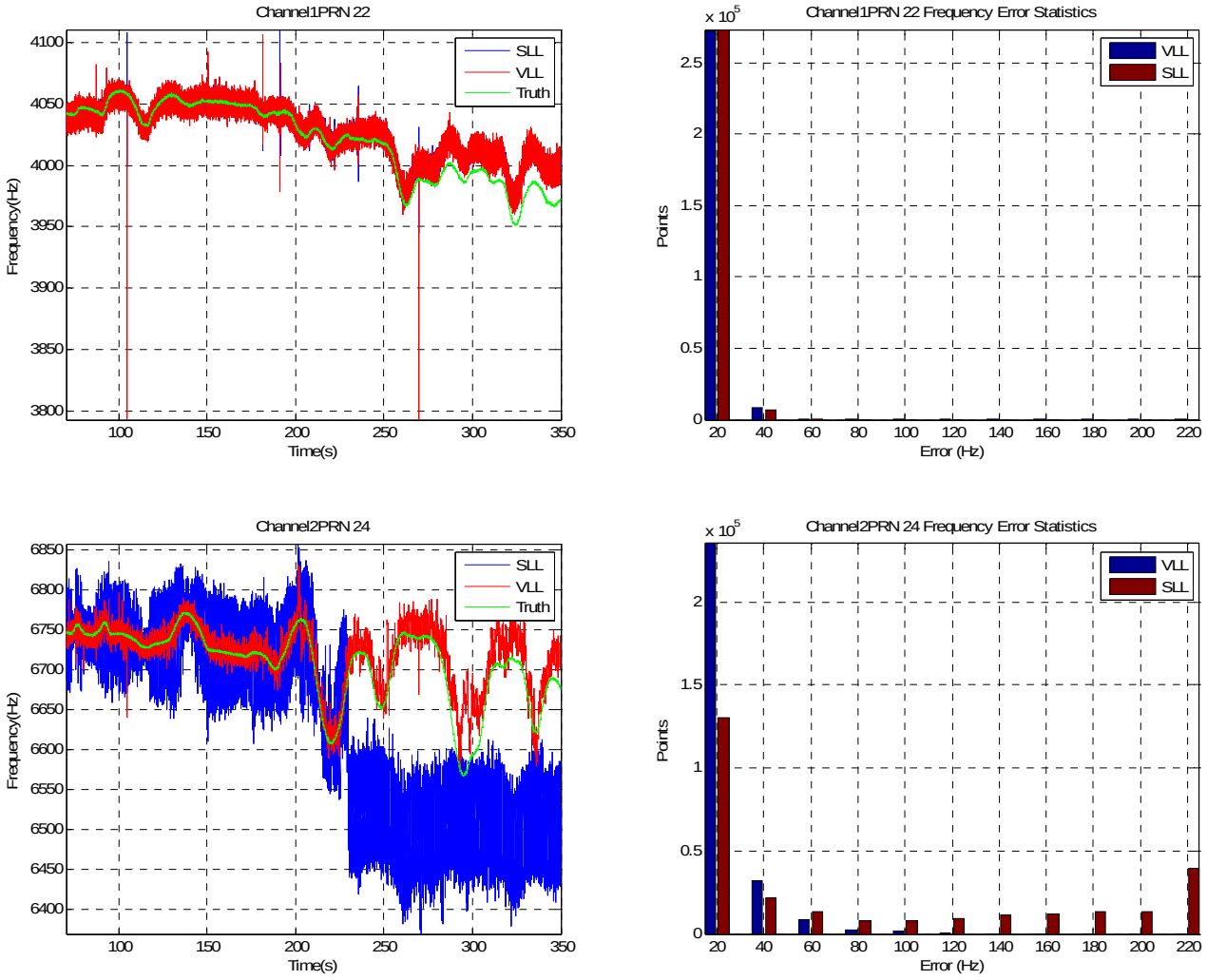


Figure 12. Frequency results of PRN22 and PRN24 (live data)

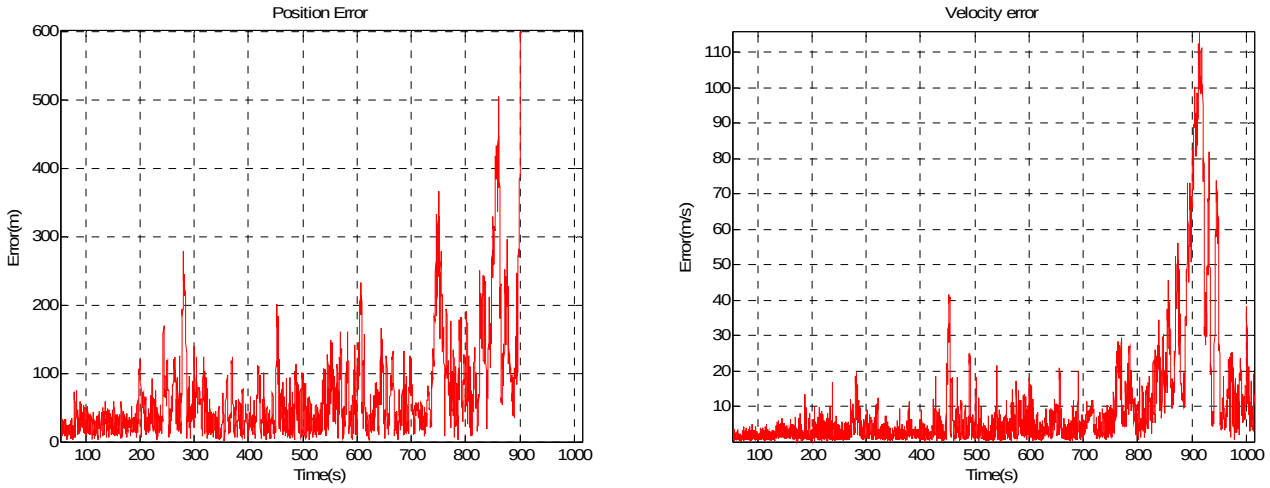


Figure 13. VLL navigation solutions of 70s~1030s (live data)

Figure 14. VLL navigation solutions of 70s~1030s on Google map

Actually, VLL can continue to keep lock and provide navigation solutions longer than fig 11 shows. Fig 13 and fig 14 show the VLL navigation results from 70s to 1030s. The results begin to diverge since about 900s in the dataset. Recall fig 9, between 901s and 925s, there are only 2 available satellites, although similar situations happen before 900s, the longest time they last is only 13s (around 760s in fig 9), which is much shorter than this time. During this relatively long period with only 2 available satellites, there is too little new information from satellite signals to update the navigation filter, so it updates the states, only depending on the process model which is not accurate enough. Errors will accumulate with time growing and finally the filter fails to converge back to the correct values.

Anyhow, fig 13 and fig 14 show that VLL maintains providing navigation solutions longer than SLL in the time period of 217s to 900s, during which the amount of available satellites varies from 8 to 0. This result demonstrates that VLL has the ability to work with less than 4 satellites during a short time of signal outage.

6. CONCLUSION & FUTURE WORK

In this paper, we illustrate the principle of the vector lock loop, discuss its advantages over SLL and reveal many details on designing and implementing VLL. An open source VLL code based on an open source software-defined GNSS receiver is presented. Kalman filtering is employed to accomplish both tracking and navigation tasks in the VLL. Considerations on filter tuning are given empirically. Three different datasets including two simulated and one live data are used to verify the performance of VLL. Experiment results show that VLL can work under lower C/N0 environments than SLL. How much VLL outperforms SLL depends on many factors including receiver maneuvering, the amount of available satellites, etc. In the first dataset of our experiments where all satellites have decreasing signal power, VLL can provide over 4dB better performance than SLL and the accuracy of navigation solutions can be improved. Testing the second dataset, we choose a 4 satellites combination inside which two satellites have decreasing signal powers. VLL provides over 7dB better performance for one satellite with decreasing power and navigation solutions with smaller errors than SLL. The last experiment uses live data collected from a canyon area in Boulder where satellite signals are blocked periodically. The proposed VLL outperforms the SLL in both accuracy in navigation solutions and ability to cover the GPS signal outage.

Our future work includes:

1) We only consider a fixed amount of available satellites that can be tracked by VLL. But in real world, due to user movement or change of environment, some blocked signals may reappear. How to leverage their information effectively would be useful for better performance.

2) The open source VLL code works on a post-processing basis. To modify it into a real-time working mode would be more applicable.

3) The criteria to indicate whether VLL is working normally should be established so that when VLL cannot work, the receiver could turn into SLL mode or even re-acquisition mode to avoid incorrect navigation results.

ACKNOWLEDGMENTS

The authors are grateful to Tianxing Chu for his help to collect and play back the live dataset. The first author's visit to the University of Colorado to complete this work is funded by Tsinghua University, China.

REFERENCES

[1] E. D. Kaplan, C. J. Hegarty, "Understanding GPS: Principles and Applications, 2nd Ed", Artech House Inc, MA, 2006

[2] B. W. Parkinson, J. J. Spilker Jr, "Global Positioning System: Theory and Applications", American Institute of Aeronautics and Astronautics, Washington DC, 1996.

[3] M. Petovello, G. Lachapelle, "Comparison of Vector-based Software Receiver Implementations with Application to Ultra-tight GPS/INS Integration", Proc. of Institute of Navigation GPS/GNSS Conference, Fort Worth, TX, 2006

[4] M. Lashley, "Modeling and Performance Analysis of GPS Vector Tracking Algorithms", PhD Dissertation, Auburn University, Alabama, 2009

[5] K. Borre, D. M. Akos, et al. "A Software-Defined GPS and Galileo Receiver: A Single-Frequency Approach", Springer, New York, 2007