Architecture, Design and Implementation of Carrier Phase Vector Tracking in GNSS RTK Receiver

Nikolay Mikhaylov[a], Valery Chistyakov[b] and David Oertel[a]

[a] *Robert Bosch GmbH, Hildesheim, Germany*
[b] *BORA Ltd, St. Petersburg, Russia*

Abstract

Navigation in GNSS denied environment is a great challenge for GNSS receivers because of such effects as signal blockage, multipath and non-line of sign reception. A promising approach for reducing these effects is vector tracking, which is well-known for its robustness against poor signal-to-noise levels, fast changing environments and temporary signal blockage. In this paper we consider architecture, design and implementation of carrier phase vector tracking in GNSS receiver able to receive GNSS measurements from a base station, i.e. a real-time kinematic (RTK) receiver. We discuss a concept of vector tracking in a differential phase lock loop (PLL), requirements to be satisfied for Vector PLL to start and how to fulfil these requirements in a high precision navigation engine. We provide architecture of software that implements a high precision navigation engine including Vector PLL. Kalman Filter-based algorithm of Vector PLL is described in detail. We also analyze the performance of carrier phase vector tracking. In the time of submission of the paper the testing was not yet completed. Nonetheless the first results obtained in a controlled environment provided promising results and indicate that the implemented vector PLL provides better results in terms of carrier phase tracking sensitivity and accuracy. More results are anticipated.

Keywords 1

GNSS, PLL, Vector PLL, Differential PLL, Extended Kalman Filter, Multipath Mitigation

1. Introduction

A known and promising approach to minimize the effect of GNSS multipath interference and to improve ground tracks is vector tracking loop (VTL), [1], [2], [3]. The approach assumes the replacement of lock loops used for code/carrier tracking in each channel with an extended Kalman Filter (EKF) that both tracks the GNSS signals and calculates the user position [4]. This approach is based upon the fact that all received signals are spatially correlated . The receiver motion projects onto received signals according to their corresponding locations. This property is used in VTL to enhance robustness [5].

The main drawback of the receivers using Vector DLL (VDLL) is that navigation solution is based on the code measurements only and, therefore, it has meters-level accuracy [5]. In order to increase the accuracy the phase measurements should be used. For pure Vector PLL (VPLL) to be viable, the errors that affect the carrier phase must be mitigated. For this reason VTL techniques for single frequency receivers use VDLL with scalar FLL/PLL and obtain navigation solution based on code measurements only. One of the ways to use carrier phase measurements in the navigation solution in that case is differential mode [1]. In this mode the base station measurements can be used in VPLL in the same way as they are used in scalar PLL of carrier phase differential GNSS receivers. Use of phase measurements of the base station allows obtaining a position solution directly in the VTL of a rover receiver with the same accuracy as in integer-resolved carrier phase differential GNSS receivers.

ICL-GNSS 2020 WiP Proceedings, June 02–04, 2020, Tampere, Finland
EMAIL: nikolay.mikhaylov@de.bosch.com (A. 1); vchistyakov@softnav.ru; david.oertel@de.bosch.com
ORCID: 0000-0001-9184-3465 (A. 1)

In this work we describe architecture, design and implementation of the differential vector PLL (DVPLL). The concept of DVPLL is presented in Section II. In section III a high-level software architecture is described. Section IV provides details of DVPLL navigation filter. Results of DVPLL testing are given in Section V. Finally, the conclusions are given in the last section. Appendices A and B contain mathematical details of common knowledge: model of GNSS measurements, relative positioning algorithm and system model used

2. DVPLL Concept

As stated above use of phase measurements of the base station allows obtaining a position solution directly in the VTL of a rover receiver with the same accuracy as in integer-resolved carrier phase differential GNSS receivers. The block diagram of such differential VPLL (DVPLL) is shown in Figure 1.

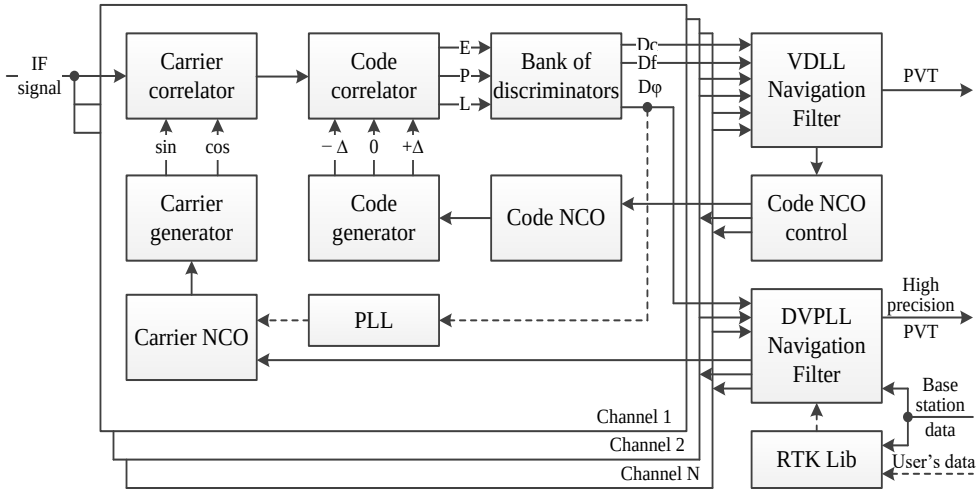

Figure 1. DVPLL Concept

Each channel in Figure 1 does not differ from a canonic scheme and includes code and carrier generators with associated NCOs, code and carrier correlators and discriminators. As in canonic scheme E, P and L in Figure 1 denote early, prompt and late correlator channel correspondingly. Dc, Df and Dϕ are outputs of the discriminators and denote differences between input and locally generated signals code, frequency and phase correspondingly.

DVPLL requires initialization. For this initialization a high precision relative position vector is needed. In Figure 1 dashed lines connect the blocks that are required to calculate the initial receiver position and form an initial high precision relative position vector. The calculation of the vector is done according to standard differential GNSS scheme. First, the receiver position is calculated with code measurements from VDLL (Doppler measurements can be used in the VDLL to improve the velocity and clock drift state estimates, like carrier smoothing or Doppler aiding in traditional scalar receivers). Then carrier phase measurements from the scalar PLL and carrier phase measurements from the base station are used to resolve phase ambiguities and to calculate the high precision relative position vector. During the initialization the scalar PLLs control the carrier NCOs and provide the carrier phase measurements in each channel. After the initialization, once the high precision relative position vector is calculated, DVPLL starts and takes over the control of NCO and production of carrier phase measurements. Initialization of DVPLL must be started every time when cycle slip or carrier phase tracking fault is detected. In this scheme VDLL drives the code NCO, and DVPLL drives the carrier NCO. Such approach prevents the less accurate code measurements from degrading the accuracy of the carrier phase based navigation solution.

3. Software Architecture

Development and testing of DVPLL is performed on the basis of SX3 receiver by IFEN. This is the software receiver which provides convenient API enabling the user to add, replace or expand the capabilities of the SX3 receiver (including integration of another navigation solution into SX3 software) via interface functions provided by SX3.

The core part is the High Precision Navigation Engine (HPNE), which implements VTL. HPNE SW architecture is defined by SX3 API provisions. The SX3 API concept is based on the usage of dynamic link library (DLL) files. Therefore part of HPNE modules is implemented as DLL. Such a library will be loaded seamlessly into SX3 at runtime and contain predefined function names to be called. Implementation of HPNE requires of using at least two SX3 API libraries: Baseband API and Navigation API. HPNE SW components, as well as main modules and main data flows are presented in Figure 2. Main call flows are noted by dashed lines, main C-functions are marked with ordinary "()" notation.

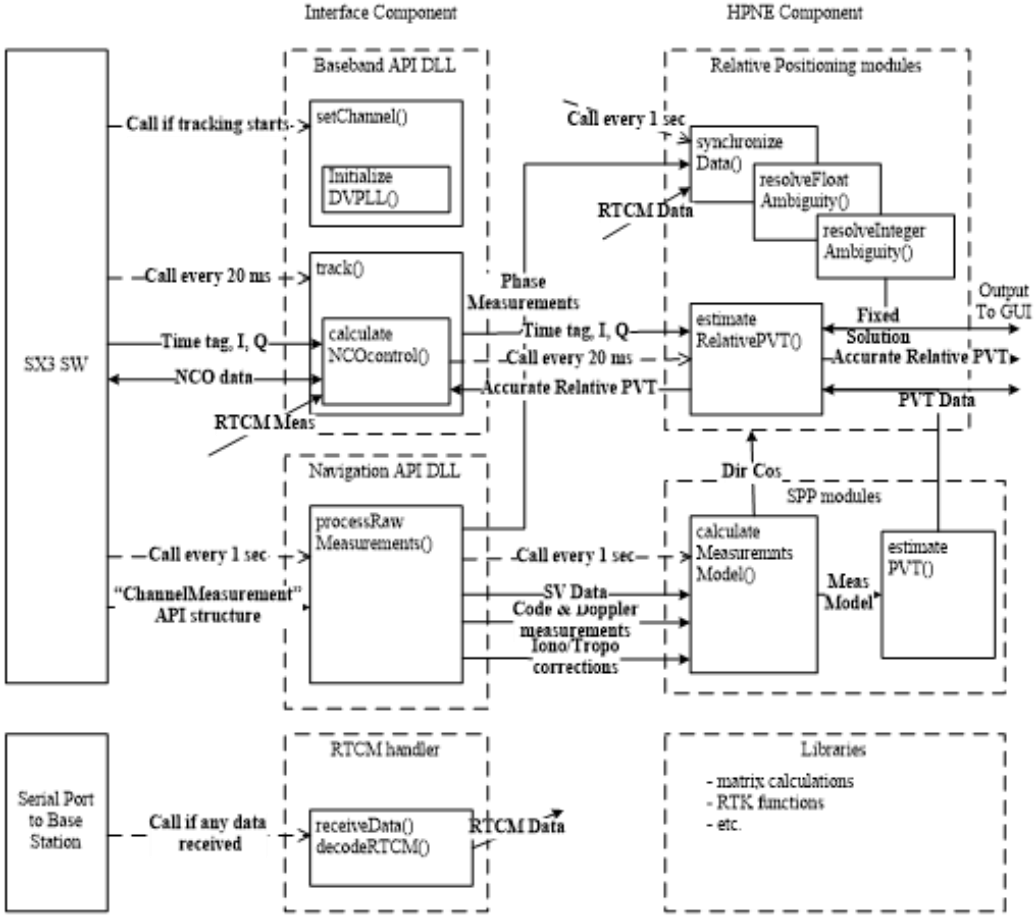

Figure 2. High-level software architecture

The software consists of two components: Interface and HPNE. The Interface component uses IFEN APIs to obtain Code and Doppler measurements from IFEN software receiver. The Code and Doppler measurements are used in single point positioning modules of HPNE denoted SPP in Figure 2 to calculate standard precision PVT.

The Interface Component also receives signal accumulations in sinphase and quadrature channels denoted as I and Q in Figure 2. It calculates phase measurements and provides them to HPNE. The Interface Component calculates carrier NCO too. It should be noted that on the first step of this project code NCO is controlled by IFEN Software.

RTCM handler, also a part of the Interface Component does a routine processing of RTCM data.

HPNE includes library modules, SPP modules to provide standard precision PVT and directive cosines to relative positioning modules. The relative positioning modules implement float and integer

ambiguity resolution and compute accurate PVT based on the single differences between phase measurements of the rover receiver and those of the base station. The accurate PVT is used by the Interface Component to control NCOs in each channel.

4. DVPLL Algorithm

DVPLL navigation filter estimates relative PVT, which is used to predict carrier phase for each tracked signal. Initialization of the relative PVT is performed on the basis of fixed (relative position and clock bias) and float (relative velocity and clock drift) solutions.

DVPLL navigation filter is implemented as an error state Kalman filter (ErKF), which estimates the error in the states, rather than the states themselves. In our case the filter estimates the errors in the relative coordinates, relative clock bias and corresponding derivatives. Error state vector is modelled as zero mean Gaussian process. This has the effect of resetting the predicted error at each time instant, i.e. the instantaneous error is more important than error history.

In Appendix A one can find the algorithm for relative positioning. It is shown there that the single differences of phase measurements are given by

$$\boldsymbol{\varphi}_{ub} = \frac{1}{\lambda}(-\boldsymbol{A}_{M\times 3}\boldsymbol{X}_{ub} + cb_{ub}\boldsymbol{I}_{M\times 1}) + \boldsymbol{N}_{ub} + \boldsymbol{\varepsilon}_{\varphi,ub}, \quad (1)$$

where

$\boldsymbol{\varphi}_{ub}$ – difference of carrier measurements between user u and base station b, m
λ – carrier wavelength, m
$\boldsymbol{A}_{M\times 3}$ – matrix of directive cosines (M is the number of measurements)
$\boldsymbol{X}_{ub}$ – $\boldsymbol{X}_{ub} = (x_{ub}\ y_{ub}\ z_{ub}\ cb_{ub})^T$ State vector consisting of differences of position components and clock bias between user u and base station b, m
c – velocity of light, m/s
b_{ub} – user clock bias relatively base station receiver, s
$\boldsymbol{I}_{M\times 1}$ = $(1\ 1\ \dots\ 1)^T$
$\boldsymbol{N}_{ub}$ – vector of integer ambiguities
$\boldsymbol{\varepsilon}_{\varphi,ub}$ – noise of the difference of carrier phase measurements between user u and base station b, cycles

The equation (2) below can be obtained by differencing (1), connects errors of the carrier phase single differences with error in relative position:

$$\Delta\boldsymbol{\varphi}_{ub} = \frac{1}{\lambda}(-\boldsymbol{A}_{M\times 3}\Delta\boldsymbol{X}_{ub} + c\Delta b_{ub}\boldsymbol{I}_{M\times 1}) + \boldsymbol{\varepsilon}'_{\varphi,ub}, \quad (2)$$

where

$\Delta\boldsymbol{X}_{ub}$ – relative position error,
Δb_{ub} – relative clock bias error,
$\boldsymbol{\varepsilon}'_{\varphi,ub}$ – noise of the carrier phase single differences.

$$\Delta\boldsymbol{\varphi}_{ub} = \boldsymbol{\varphi}_{ub} - \widehat{\boldsymbol{\varphi}}_{ub} = \boldsymbol{\varphi}_u - \boldsymbol{\varphi}_b - (\widehat{\boldsymbol{\varphi}}_u - \widehat{\boldsymbol{\varphi}}_b) = \boldsymbol{\varphi}_u - \widehat{\boldsymbol{\varphi}}_u - (\boldsymbol{\varphi}_b - \widehat{\boldsymbol{\varphi}}_b), \quad (3)$$

where $\widehat{\boldsymbol{\varphi}}_b$ and $\widehat{\boldsymbol{\varphi}}_u$ denote predictions of the carrier phase in base station and in rover receiver.

Taking into account that carrier phase error for base station is significantly less than for rover receiver, the equation (2) can be rewritten as:

$$\lambda\Delta\boldsymbol{\varphi}_u = -\boldsymbol{A}_{M\times 3}\Delta\boldsymbol{X}_{ub} + c\Delta b_{ub}\boldsymbol{I}_{M\times 1} + \boldsymbol{\varepsilon}'_{\varphi,ub}, \quad (4)$$

where

$\Delta\boldsymbol{\varphi}_u$ carrier phase error of the rover receiver (PLL discriminator).

Equation for the error in relative velocity can be obtained by differentiating (4):

$$\lambda\Delta\boldsymbol{f}_u = -\boldsymbol{A}_{M\times 3}\Delta\dot{\boldsymbol{X}}_{ub} + c\Delta\dot{b}_{ub}\boldsymbol{I}_{M\times 1} + \boldsymbol{\varepsilon}'_{f,ub}, \quad (5)$$

where

$\Delta\boldsymbol{f}_u$ carrier frequency error of the rover receiver (FLL discriminator).

Let's define state vector and dynamic model as follows:

$\Delta \boldsymbol{X}_k = (\Delta x_{ub}^k\ \Delta y_{ub}^k\ \Delta z_{ub}^k\ c\Delta b_{ub}^k\ \Delta \dot{x}_{ub}^k\ \Delta \dot{y}_{ub}^k\ \Delta \dot{z}_{ub}^k\ c\Delta \dot{b}_{ub}^k)^T$, (6)

$\Delta \boldsymbol{X}_{k,k+1} = \boldsymbol{F}_{k,k+1}\Delta \boldsymbol{X}_{k,k}$, (7)

where $\boldsymbol{F}_{k,k+1} = \begin{pmatrix} \boldsymbol{0}_{4\times4} & \Delta t_k \boldsymbol{E}_{4\times4} \\ \boldsymbol{0}_{4\times4} & \boldsymbol{0}_{4\times4} \end{pmatrix}$, $\Delta t_k = t_{k+1} - t_k$,

$\Delta \boldsymbol{X}_{k,k}$ denotes state vector in time t_k, and $\Delta \boldsymbol{X}_{k,k+1}$ denotes state prediction from t_k to t_{k+1}.

Measurements update for the DVPLL navigation filter is performed at the end of integration period in each channel. If navigation symbol location is found then integration interval is usually equal to the navigation symbol length. Neglecting the impact of Doppler on the code frequency we will assume that integration intervals for all channels are the same. However the ends of the integration intervals for different channels generally do not match. Because the range between a satellite and a user varies from satellite to satellite, different channels receive the same navigation symbol at different time.

Let's define the measurements vector and measurement matrix as follows:

$\boldsymbol{Z}_k = \lambda(\Delta\varphi_k\ \Delta f_k)^T$,

$\Delta\varphi_k = \frac{1}{2\pi}\mathrm{arctg}\left(\frac{Q_k}{I_k}\right)$,

$\Delta f_k = \frac{1}{2\pi\Delta t}\mathrm{arctg}\left(\frac{Q_k I_{k-N} - I_k Q_{k-N}}{I_k I_{k-N} + Q_k Q_{k-N}}\right)$,

$\boldsymbol{H}_k = \begin{pmatrix} -a_x^k & -a_y^k & -a_z^k & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -a_x^k & -a_y^k & -a_z^k & 1 \end{pmatrix}$, where

Δt – I&Q accumulation time (20 ms); a_x^k, a_y^k, a_z^k are the directive cosines for the satellite on the current channel.

Note that $\boldsymbol{Z_k}$ corresponds to the difference between received signal and predicted one based on the state vector at beginning of kth integration interval:

$\boldsymbol{Z}_k = \boldsymbol{H}_k\left(\boldsymbol{X}_{k,k} - \boldsymbol{F}'_{k-N,k}\boldsymbol{X}_{k-N,k-N}\right)$, where (8)

$\boldsymbol{F}'_{k-N,k} = \boldsymbol{E} + \boldsymbol{F}_{k-N,k}$ is the matrix of state transition from state $\boldsymbol{X}_{k-N,k-N}$ at time t_{k-N} to state prediction $\boldsymbol{X}_{k,k}$ at time t_k.

However, for implementation of the DVPLL navigation filter in a standard form it is required to have measurement in the following form:

$\boldsymbol{Z}_k^* = \boldsymbol{H}_k\left(\boldsymbol{X}_{k,k} - \boldsymbol{F}'_{k-1,k}\boldsymbol{X}_{k-1,k-1}\right)$. (9)

These measurements are related as follows:

$\boldsymbol{F}'_{k-1,k}\boldsymbol{X}_{k-1,k-1} =$

$\boldsymbol{F}'_{k-1,k}\left(\boldsymbol{X}_{k-2,k-1} + \Delta \boldsymbol{X}_{k-1,k-1}\right) = \boldsymbol{F}'_{k-2,k}\boldsymbol{X}_{k-2,k-2} + \boldsymbol{F}'_{k-1,k}\Delta \boldsymbol{X}_{k-1,k-1} = \cdots$

$\boldsymbol{F}'_{k-\mathrm{N},k}\boldsymbol{X}_{k-N,k-N} + \boldsymbol{F}'_{k-\mathrm{N}+1,k}\Delta \boldsymbol{X}_{k-N+1,k-N+1} + \cdots + \boldsymbol{F}'_{k-1,k}\Delta \boldsymbol{X}_{k-1,k-1}$. (10)

In (10) we used the equation for updated state estimate.

Therefore,

$\boldsymbol{Z}_k^* = \boldsymbol{Z}_k - \boldsymbol{H}_k \sum_{m=k-N+1}^{k-1} \boldsymbol{F}'_{m,k}\Delta \boldsymbol{X}_{m,m}$. (11)

Note that equation (11) was obtained for the Channel 1. Let's determine adjustment vector $\boldsymbol{e}_i$ for ith channel as follows:

$\boldsymbol{e}_i = \sum_{m=k+i-N}^{k+i-2} \boldsymbol{F}'_{m,k}\Delta \boldsymbol{X}_{m,m}$. (12)

Then,

$\boldsymbol{Z}_{k+i-1}^* = \boldsymbol{Z}_{k+i-1} - \boldsymbol{H}_{k+i-1}\boldsymbol{e}_i$. (13)

Computation of the adjustment vectors is performed iteratively each time when new measurement is available. After adjustment is applied the corresponding vector is zeroed.

Let's define estimated relative vector as follows:

$\boldsymbol{X}_k = (x_{ub}^k\ y_{ub}^k\ z_{ub}^k\ cb_{ub}^k\ \dot{x}_{ub}^k\ \dot{y}_{ub}^k\ \dot{z}_{ub}^k\ c\dot{b}_{ub}^k)^T$.

The algorithm of navigation filter consists of the following steps:

Step 1 (initialization)
$\boldsymbol{X}_0 = \boldsymbol{X}_{fixed}; \Delta\boldsymbol{X}_0 = 0, \boldsymbol{P}_0 = \boldsymbol{P}_{float}; \boldsymbol{e}_i = \boldsymbol{0}, i = 1 \dots N$

Step 2 (prediction)
$\boldsymbol{X}_{k-1,k} = \boldsymbol{X}_{k-1,k-1} + \boldsymbol{F}_{k-1,k}\boldsymbol{X}_{k-1,k-1},$

$\Delta\boldsymbol{X}_{k-1,k} = \boldsymbol{F}_{k-1,k}\Delta\boldsymbol{X}_{k-1,k-1},$
$\boldsymbol{P}_{k-1,k} = \boldsymbol{F}_{k-1,k}\boldsymbol{P}_{k-1,k-1}\boldsymbol{F}_{k-1,k}^T + \boldsymbol{Q}_k$, $\boldsymbol{Q}_k$ – see Appendix B,
$\boldsymbol{e}_i = \boldsymbol{F}'_{k-1,k}\boldsymbol{e}_i, i = 1 \dots N.$

Step 3 (measurement update)
$\boldsymbol{G}_k = \boldsymbol{P}_{k-1,k}\boldsymbol{H}_k^T(\boldsymbol{H}_k\boldsymbol{P}_{k-1,k}\boldsymbol{H}_k^T + \boldsymbol{R}_k)^{-1}$, $\boldsymbol{R}_k = \begin{pmatrix} \sigma_\varphi^2 & 0 \\ 0 & \sigma_f^2 \end{pmatrix}$,
$\boldsymbol{Z}_k^* = \boldsymbol{Z}_k - \boldsymbol{H}_k\boldsymbol{e},$
$\Delta\boldsymbol{X}_{k,k} = \Delta\boldsymbol{X}_{k-1,k} + \boldsymbol{G}_k(\boldsymbol{Z}_k^* - \boldsymbol{H}_k\Delta\boldsymbol{X}_{k-1,k}),$
$\boldsymbol{P}_{k,k} = (\boldsymbol{E} - \boldsymbol{G}_k\boldsymbol{H}_k)\boldsymbol{P}_{k-1,k},$
$\boldsymbol{e} = \boldsymbol{0},$ where $\boldsymbol{e}$ is the corresponding adjustment vector
$\boldsymbol{e}_i = \boldsymbol{e}_i + \Delta\boldsymbol{X}_{k,k}, i = 1 \dots N$, excepting the channel corresponding to $\boldsymbol{e}$.

$\boldsymbol{X}_{k,k} = \boldsymbol{X}_{k-1,k} + \Delta\boldsymbol{X}_{k,k}.$

Step 4
Propagate $\boldsymbol{X}_{k,k}$ to the end of next integration period and compute predicted carrier frequency for the corresponding channel.

5. Performance Analysis

At the moment only the first test results are obtained. The results basically show that the DVPLL implementation provides anticipated results.

Figure 1 below shows a carrier-to-noise density for a static user. Figure 2 demonstrates that vector PLL has higher accuracy than scalar PLL, especially for low levels of signal power. One can see from Figure 3 that the accuracy of vector PLL for the simulated scenario, i.e. down to about 24 dBHz remains in ±1Hz range. Further tests will follow soon.

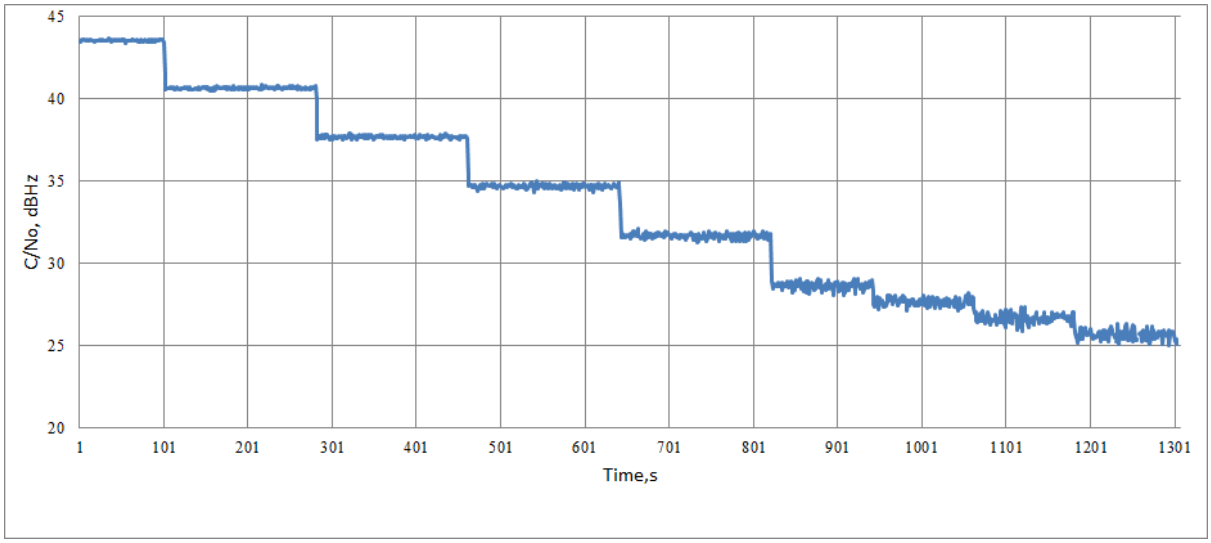

Figure 1 Carrier-to-noise density as a function of time in a static user scenario

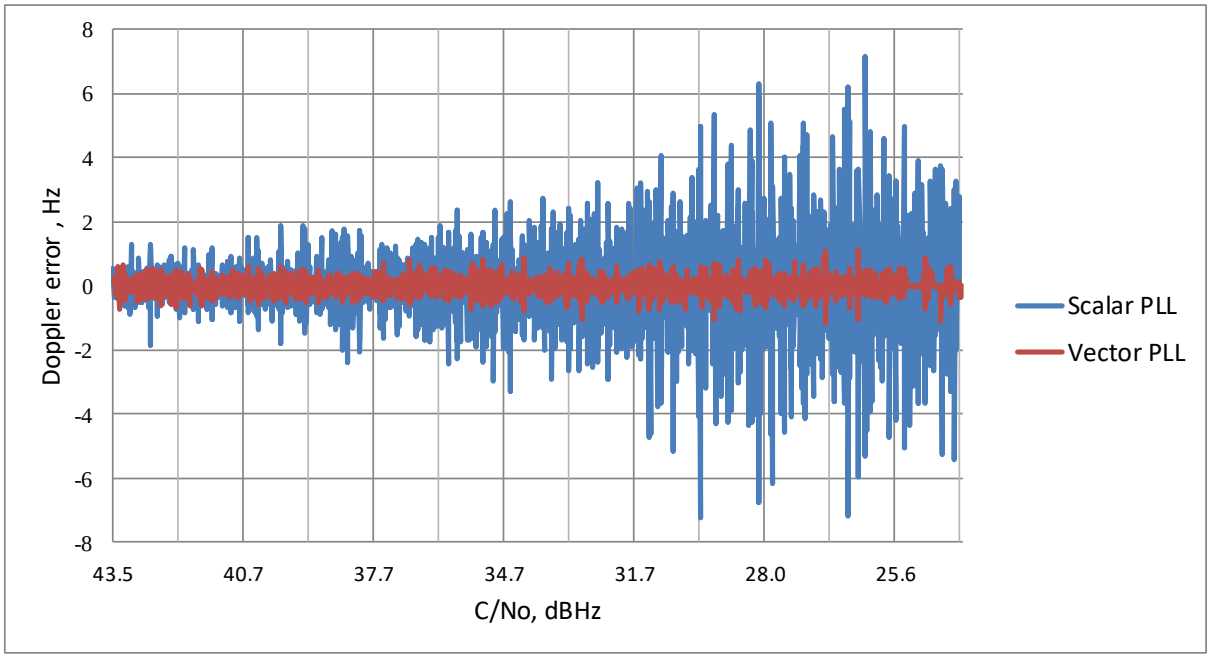

Figure 2 Doppler frequency error as a function of carrier-to-noise density for scalar and vector PLLs

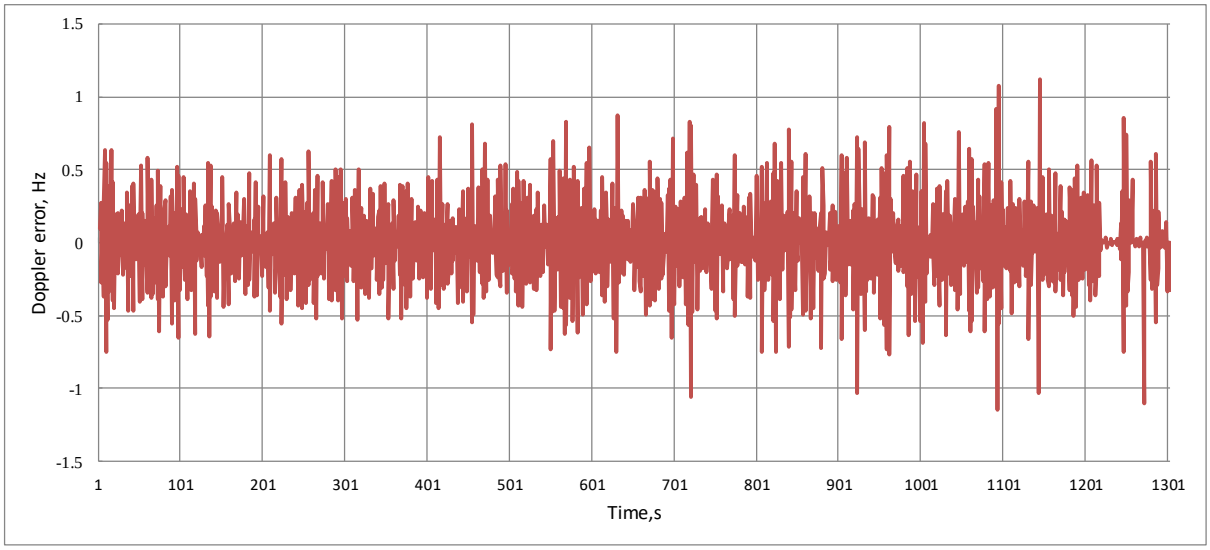

Figure 3 Doppler frequency error as a function of carrier-to-noise density for vector PLL

6. Conclusions

The paper provides a detail analysis of a DVPLL including, concept, design, implementation and first test results. It is planned to present the results of full DVPLL testing in ION conference in September 2020.

7. References

[1] J. Brewer and J. Raquet, "Differential Vector Phase Locked Loop," *IEEE Transactions on Aerospace and Electronic Systems*, pp. 1046-1055, June 2016.

[2] S. M. Martin, D. M. Bevly, R. G. Keegan and S. F. Rounds, "RTK Vector Phase Locked Loop Architecture". USA Patent 20190120973, 11 10 2019.

[3] A. Shafaati, T. Lin, A. Broumandan and G. Lachapelle, "Design and Implementation of an RTK-based Vector Phase Locked Loop," *Sensors*, March 2018.

[4] M. Lashley and D. M. Bevly, "Comparison in the Performance of the Vector Delay/Frequency Lock Loop and Equivalent Scalar Tracking Loops in Dense Foliage and Urban Canyon," in *Proceedings of the 24-th ITM of the Satellite Division of the Institute of Navigation*, Portland, OR, 2011.

[5] M. Lashley, "Modelling and Performance Analysis of GPS Vector Tracking Algorithms. Ph.D. Thesis.," Auburn University, Auburn, 2009.

[6] P. Teunissen, "A least-squares ambiguity decorrelation adjustment: a method for fast GPS integer ambuity estimation," *Journal of Geodesy*, pp. 65-82, 1985.

[7] Y. Bar-Shalom, R. L. X. and T. Kirubarajan, Estimation with Application to Tracking and Navigation: Theory Algorithms and Software, New York: John Wiley & Sons, 2001.

[8] M. Scott, "GPS Carrier Phase Tracking in Difficult Environments Using Vector Tracking for Precise Positioning and Vehicle Attitude Estimation Tracking in GNSS. Ph.D. Thesis," Auburn University, Auburn, USA, 2017.

8. Appendix A: Relative Positioning Algorithm

For the basic relative positioning algorithm, the pseudorange and carrier phase measurements are modelled as:

$$\boldsymbol{\rho_u^k = r_u^k + I_u^k + T_u^k + M_u^k + c\left(b_u + b^k\right) + \varepsilon_{\rho,u}^k},$$

$$\varphi_u^k = \frac{1}{\lambda}\left(r_u^k - I_u^k + T_u^k + M_u^k + c\left(b_u + b^k\right)\right) + N_u^k + \varepsilon_{\varphi,u}^k,$$

where the notation is analogue to the one given in section **Error! Reference source not found.** and additionally:

I_u^k – ionospheric delay, m
T_u^k – tropospheric delay, m
M_u^k – multipath error, m.

Neglecting the multipath error we use the measurement differences $\boldsymbol{\varphi}_{ub}$ from eq. (1) and the pseudorange differences

$$\rho_{ub}^k = r_{ub}^k + cb_{ub} + \varepsilon_{\rho,ub}^k$$

in a Kalman filter to, at first, estimate the state vector

$$\boldsymbol{X} = (x_{ub}\ y_{ub}\ z_{ub}\ cb_{ub}\ \dot{x}_{ub}\ \dot{y}_{ub}\ \dot{z}_{ub}\ c\dot{b}_{ub}\ N_{ub}^1 \dots N_{ub}^M)^T.$$

Here, the ambiguities N_{ub}^i are contained with relaxed float constraints. Using a system transition model as in eq. (7) with additional identity transition on the ambiguities, the measurement vector $\boldsymbol{Z}$, observation matrix $\boldsymbol{H}$ and measurement noise matrix $\boldsymbol{R}$ for this filter are as follows:

$$\boldsymbol{Z} = \left(\rho_{ub}^{1}\ \ \rho_{ub}^{2} \ldots \rho_{ub}^{M}\ \lambda\varphi_{ub}^{1}\ \lambda\varphi_{ub}^{2} \ldots \lambda\varphi_{ub}^{M}\right)^{T}$$

$$\boldsymbol{H} = \begin{pmatrix} -\boldsymbol{A}_{M\times 3}\ \boldsymbol{I}_{M\times 1}\ \boldsymbol{0}_{M\times 4}\ \ \boldsymbol{0}_{M\times M} \\ -\boldsymbol{A}_{M\times 3}\ \boldsymbol{I}_{M\times 1}\ \boldsymbol{0}_{M\times 4}\ \lambda\boldsymbol{E}_{M\times M} \end{pmatrix}.$$

$$\boldsymbol{R} = \mathrm{diag}(R_1, \ldots, R_{2M}),\ R_i = \begin{cases} \sigma_{\rho_u^i}^2 + \sigma_{\rho_b^i}^2,\ i \le M \\ \sigma_{\varphi_u^i}^2 + \sigma_{\varphi_b^i}^2,\ i > M \end{cases}.$$

Variances σ_ρ^2 and σ_φ^2 are determined for each channel on the basis of signal to noise ratio and DLL/PLL parameters. The default Kalman filter equations apply for generating a float ambiguity solution.

In a follow-up step, the float solution is refined using the LAMBDA method [6] for an integer ambiguity resolution.

The known integer ambiguities $\boldsymbol{N}_{ub}$ are then forwarded to estimate $\boldsymbol{X}_{ub}$ through the equations:

$$\boldsymbol{H_r X}_{ub} = \lambda(\boldsymbol{\varphi}_{ub} - \boldsymbol{N}_{ub}),$$

$$\boldsymbol{H_r} = \begin{pmatrix} -a_x^1 & -a_y^1 & -a_z^1 & 1 \\ & \vdots & & \\ -a_x^M & -a_y^M & -a_z^M & 1 \end{pmatrix}.$$

Here, an ordinary least-squares estimation yields the fixed solution, i.e. the solution with resolved integer ambiguities providing the foundation and initialization for the DVPLL computations.

9. Appendix B: System Dynamic Model

The system dynamics are based on [7] which provides a detailed discussion and deduction including the transformation of the continuous to the discrete system representation. In short, a continuous white noise acceleration model (CWNA) transformed into discrete time is employed in this work. The resulting system transition is given in equation (7). The respective noise matrix $\boldsymbol{Q}_k$ as a result from the discretization is composed of a motion dynamic part $\boldsymbol{Q}_{dyn}$ and a clock dynamic influence $\boldsymbol{Q}_{clk}$:

$$\boldsymbol{Q} = \boldsymbol{Q}_{dyn} + \boldsymbol{Q}_{clk},$$

$$\boldsymbol{Q_{dyn}} = \begin{pmatrix} \sigma_x^2\frac{\Delta t^3}{3} & 0 & 0 & 0 & \sigma_x^2\frac{\Delta t^2}{2} & 0 & 0 & 0 \\ 0 & \sigma_y^2\frac{\Delta t^3}{3} & 0 & 0 & 0 & \sigma_y^2\frac{\Delta t^2}{2} & 0 & 0 \\ 0 & 0 & \sigma_z^2\frac{\Delta t^3}{3} & 0 & 0 & 0 & \sigma_z^2\frac{\Delta t^2}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \sigma_x^2\frac{\Delta t^2}{2} & 0 & 0 & 0 & \sigma_x^2\Delta t & 0 & 0 & 0 \\ 0 & \sigma_y^2\frac{\Delta t^2}{2} & 0 & 0 & 0 & \sigma_y^2\Delta t & 0 & 0 \\ 0 & 0 & \sigma_z^2\frac{\Delta t^2}{2} & 0 & 0 & 0 & \sigma_z^2\Delta t & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\boldsymbol{Q_{clk}} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_b^2 \Delta t + \sigma_d^2 \frac{\Delta t^3}{3} & 0 & 0 & 0 & \sigma_d^2 \frac{\Delta t^2}{2} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_d^2 \frac{\Delta t^2}{2} & 0 & 0 & 0 & \sigma_d^2 \Delta t \end{pmatrix}$$

The noise constants need to reflect the respective motion behavior of the receiver. In this work, the acceleration parameters are fixed to

$\sigma_x = \sigma_y = \sigma_z = 10$.

The terms σ_b^2 and σ_d^2 in $\boldsymbol{Q_{clk}}$ can be approximated by [4] :

$\sigma_b^2 = c^2 \frac{h_0}{2}$,

$\sigma_d^2 = c^2 2\pi^2 h_{-2}$,

where the parameters h_0 and h_{-2} are the power spectral density coefficients for the clock's oscillator:

$S_\phi(f) = N^2 \left(h_0 + \frac{h_{-1}}{f} + \frac{h_{-2}}{f^2} + \frac{h_{-3}}{f^3} + \frac{h_{-4}}{f^4} \right)$.

N is the ratio the frequency of interest (for example, GPS L1) to the nominal frequency of oscillator. Power spectral density coefficients can be obtained using polynomial fit of the data from the oscillator specifications. For TCXO the following coefficients can be used [4]:

h_0 = 2 × 10-19, h_{-1} = 7 × 10-21, h_{-2} = 2 × 10-20.