SNACS – The Satellite Navigation Radio Channel Signal Simulator

F. M. Schubert[1,2,3] (*ION member*), R. Prieto-Cerdeira[2], P. Robertson[1], B. H. Fleury[3]

[1]German Aerospace Center (DLR), Institute of Communications and Navigation
[2]European Space Agency, ESA/ESTEC
[3]Aalborg University, Department of Electronic Systems

BIOGRAPHIES

F. M. Schubert studied Electrical Engineering and Information Technology at the University of Karlsruhe, Germany. Since 2007 he is member of the scientific staff at the Institute of Communications and Navigation, German Aerospace Center (DLR). During his studies, he was involved in a research project concerning inertial measurement unit (IMU) calibration at the Delft University of Technology, Delft, The Netherlands, and he developed a simulation software for a robust powerline communication system at the Massachusetts Institute of Technology, Cambridge, USA. Since 2007, F. M. Schubert is participating in the European Space Agency's Networking/Partnering Initiative (NPI) together with DLR and the University of Aalborg working towards his Ph.D. He is working on topics such as radio channel modeling for satellite navigation and the influences of harsh multipath environments on satellite navigation receiver performance.

R. Prieto-Cerdeira received his Telecommunications Engineering degree in 2002 from the University of Vigo, Spain, and followed postgraduate studies on Space Science and Radioastronomy in Chalmers University of Technology, Gothenburg, Sweden. Since 2004, he has been with the European Space Agency (ESA/ESTEC) in the Wave Interaction and Propagation Section where he is responsible of the activities related to radiowave propagation in the ionosphere and local environment for Global Navigation Satellite Systems (GNSS) such as Galileo and EGNOS projects, satellite mobile communications, and remote sensing and planetary radio science projects.

P. Robertson received a Ph.D. from the University of the Federal Armed Forces, Munich, in 1995. He is currently with DLR, where his research interests are navigation, sensor based context aware systems, signal processing, and novel systems and services in various mobile and ubiquitous computing contexts.

B. H. Fleury received the diploma in Electrical Engineering and Mathematics in 1978 and 1990 respectively, and the doctoral degree in Electrical Engineering in 1990 from the Swiss Federal Institute of Technology Zurich (ETHZ), Switzerland. Since 1997 B. H. Fleury has been with the Department of Electronic Systems, Aalborg University, Denmark, as a Professor in Communication Theory. He is the Head of the Section Navigation and Communications, one of the eight labs of this department. Since April 2006 he has also been affiliated as a Key Researcher with Forschungszentrum Telekommunikation Wien (FTW), Austria. B. H. Fleury's current areas of research include stochastic modelling and estimation of the radio channel, especially for multiple-input multiple-output (MIMO) applications and in fast time-varying environments, and iterative information processing with focus on efficient, feasible advanced receiver architectures.

ABSTRACT

A versatile GNSS signal simulation tool for time-domain simulations at sample-level with a focus on wave propagation effects, such as multipath or atmospheric disturbances, is introduced. The Satellite Navigation Radio Channel Signal Simulator (SNACS) uses a modular, object-oriented approach implemented in C++. The application is thoroughly designed in a multi-threaded way to provide highly accurate simulation results in a reasonable processing time.

SNACS not only fills the gap of a software tool using models of the GNSS channel from a GNSS signal generator as input for a GNSS software receiver, it also provides an implementation that spans the whole simulation chain from the transmitter to the receiver loops.

SNACS can be adopted and enhanced easily since it is published under an open source license.

1 INTRODUCTION

A challenge for researchers and developers is the ability to use realistic channel models in their full complexity for the simulation of GNSS signals. To further improve the performance of GNSS receivers in demanding real-life situations such as navigation in urban areas, it has become clear that realistic channel models have to be applied. In recent years, radio channel models were designed based on channel sounding measurements specifically for the GNSS use case. Up to now, models exist for the urban [7], sub-urban [9], and aeronautical [10] scenarios.

The GNSS channel models in general generate time-series of channel impulse responses (CIR) based on the tapped delay line method. The land-mobile channel model outlined in [7] uses a mixed deterministic and statistical approach that considers an artificial scenery. The computed CIRs consist of time-continuous Dirac-impulse-like echoes which have to be interpolated at the discrete instants considered in the simulation in order to correctly represent the temporal domain and motion of the receiver and the reflectors. This interpolation step makes shifts of the CIRs at sub-sampling time instants possible.

The samples of the interpolated CIR are then used as FIR (finite impulse response) filter coefficients in the complex-valued convolution with the generated GNSS signals. SNACS uses the multi-threading paradigm and decomposes the complex convolution into four real convolutions which can be calculated in parallel to yield a faster simulation speed.

Additionally, experimental CIRs obtained in channel sounding campaigns can also be fed into SNACS without the need of pre-processing, provided the sampling frequencies used to collect the CIRs and to perform the simulations match.

Unlike the fast-simulation approach which avoids the correlation operation and simulates GNSS signals in the correlation function domain, simulations in time-domain generate and process GNSS ranging signals sample by sample. Since the fast-simulation approach computes the number of correlation function samples (typically three) only for every correlation epoch, typically which equals 1 ms for the GPS C/A code, a time-domain simulator has to process 40000 samples within that period of time for a sampling frequency of 40 MHz, for example. CIRs can be used directly on the correlation function samples in the fast-simulation approach, whereas interpolation is needed for processing CIRs in a time-domain simulation.

Depending on the amount of impulses present in a CIR and the sampling frequency used for the time-domain simulation, the fast-simulation approach can be much faster than a simulation in time-domain. On the other hand, additive noise and interference have to be transformed into the correlation domain to be applicable for the fast-simulation method.

Navsim [6] is a representative implementing the fast-simulation approach whereas the Granada [5] and Juzzle Software Receivers [3] are examples for time-domain simulation tools.

The paper is organized as follows. Section 2 gives an overview of the simulation chain, consisting of GPS C/A code signal generation, channel data interpolation and convolution, noise generation, quantization, and the final processing with GNSS receiver algorithms: aquisition and signal tracking. A SNACS simulation example with GNSS channel model CIRs is demonstrated in Section 3.

2 IMPLEMENTATION

Fig. 1 shows the general structure of a typical simulation chain. The chain begins with a signal source which generates a GNSS signal sample by sample with sampling frequency f_s, resulting in the sampling interval $T = 1/f_s$. The samples are passed from one processing module to the next one in blocks of size K. Blocks are numbered from $m = 1...M$.

The total number of samples is thus $N = MK$ and the samples are numbered from $n = 1...N$. The elapsed simulation time instant relating to sample n is $t_n = nT$. The signal sink receives all samples at the end of the simulation chain.

Every link in the simulation chain works independently in parallel in its own thread and the data is passed in ring buffers suitable for concurrent access from one processing module to the next. The output signal of a processing module serves as input signal to the next module.

2.1 GPS Signal Generation

Up to now, only the GPS C/A code signal is implemented as GNSS signal source. The samples of the C/A code signal are generated for every time instant in baseband using

$$s_{\mathrm{GPS,prn}}(t_n) = \sqrt{2P_{\mathrm{CA}}} \cdot C_{\mathrm{CA,prn}}(t_n). \qquad (1)$$

Here, P_{CA} represents the signal power and $C_{\mathrm{CA}}(t_n)$ is the C/A code sample at time instant t_n. The parameter prn identifies the used C/A code through its pseudorandom noise (PRN) code number.

Usually a consecutive low-pass filter module is used to bandlimit the previously generated signal.

2.2 Channel data interpolation and convolution

The purpose of this module is to compute the time-discrete equivalent of the time-variant continuous-time CIR which are time-continuous in the delay variable τ. The variable t represents the simulation time and lasts typically from few seconds to minutes or hours. The variable τ represents the excess delay of the multipath resonse in the order of nanoseconds, typically. Fig. 2 shows an example of a time-variant CIR with the time variable t and the delay variable τ. The time-variant time-continuous CIR takes the form

$$h(t,\tau) = \sum_{d=1}^{D(t)} a_d(t)\delta(\tau - \tau_d(t)). \qquad (2)$$

The summands in 2 are called multipath component of the CIR. Each summand represents the contribution to the CIR due to one propagation path. At time t, $D(t)$ multipath components are present. Component d is characterized by its complex weight $a_d(t)$ and propagation delay $\tau_d(t)$.

For implementation convenience, the duration of one data block with block size K has to equal the update inter-

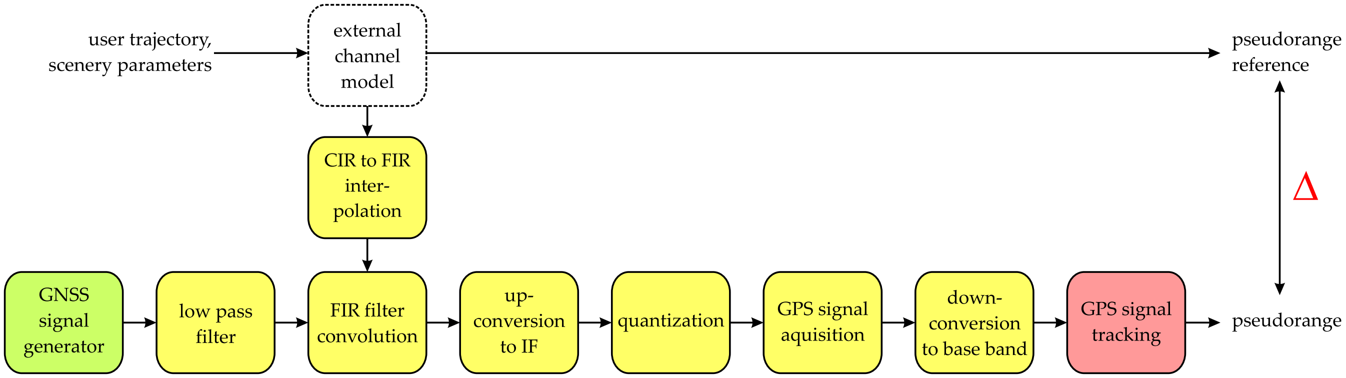

Figure 1. Software simulator structure. The signal source is shown in green, the processing modules are depicted in yellow and the signal sink is displayed as a red box.

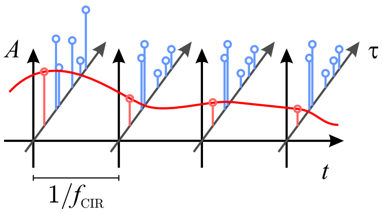

Figure 2. An example of a time-variant CIR sampled in time and delay. A denotes the multipath components' amplitude.

val $1/f_{\mathrm{CIR}}$ at which the temporal samples of the CIR are queried:

$$\frac{K}{f_{\mathrm{s}}} = \frac{1}{f_{\mathrm{CIR}}} \tag{3}$$

For instance if the channel model is queried at $f_{\mathrm{CIR}} = 1\,\mathrm{kHz}$ and the simulation sampling frequency is set to $f_{\mathrm{s}} = 100\,\mathrm{MHz}$, the data block size is $K = 100000$ samples.

The input-output of a time-variant channel with CIR $h(t,\tau)$ reads [11]

$$y(t) = \int_{-\infty}^{\infty} h(t,\tau)x(t-\tau)d\tau. \tag{4}$$

Invoking the Sampling Theorem, provided $x(t)$ is bandlimited to the bandwidth, say B, $h(t,\tau)$ in (4) can be replaced by the sampled version

$$h_{\mathrm{discr}}(t_m,\tau)|_{\tau=[0\ldots(F-1)]T} = \sum_{d=1}^{D(t_m)} \sum_{i=0}^{F-1} a_d(t_m)\mathrm{sinc}[2B(iT-\tau_d(t_m))]. \tag{5}$$

In this expression $\mathrm{sinc}(x) = \sin(\pi x)/(\pi x)$ and F is the number of discrete-time samples. These samples are used as FIR filter coefficients in the implementation. The maximum delay

$$\tau_{d,\max} = \max \tau_d(t) : t \in [0,t_{\max}], d \in 1,\ldots,D(t) \tag{6}$$

of the time-variant CIR over the simulation interval $[0,t_{\max}]$ determines the maximum number of filter coefficients F. Also, the parameter F has to be chosen large enough to let the sidelobes of the sinc-functions decay sufficiently. The bandwidth B can be the maximum bandwidth allowed by the sampling theorem $B_{\max} = f_{\mathrm{s}}/2$. In the case of the GPS C/A code signal, it can be set to $10.23\,\mathrm{MHz}$ to consider the C/A signal's bandwidth.

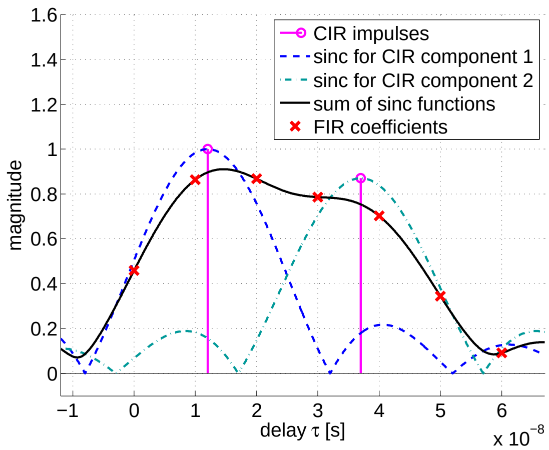

Figure 3. Example of an interpolation of a CIR with two components. The dashed curves relate to the right-hand side of (5). Their sum is shown as the black curve.

Fig. 3 shows an example of a time-invariant CIR consisting of $D = 2$ multipath components with delays $\tau_1 = 1.2 \cdot 10^{-8}\,\mathrm{s}$ and $\tau_2 = 3.7 \cdot 10^{-8}\,\mathrm{s}$, and complex weights $a_1 = 1$ and $a_2 = 0.87e^{j\cdot\pi/4}$, respectively. The sampling frequency is $100\,\mathrm{Mhz}$, so the interpolation bandwidth is set to $50\,\mathrm{Mhz}$. The two summands in the right-handed side of (5) and their sum are depicted in the figure. The resulting $F = 7$ coefficients of the equivalent FIR filter which results from sampling the time-continuous CIR are shown as red crosses.

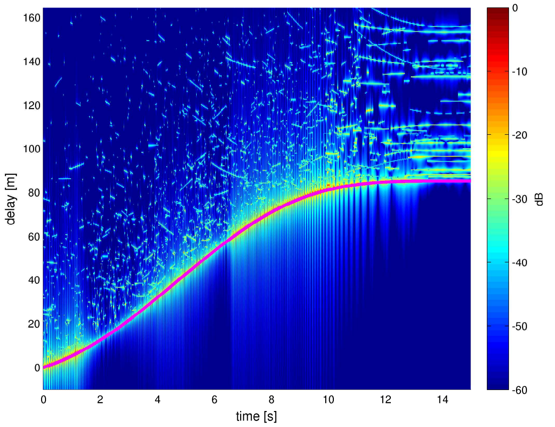

Figure 4. Interpolated CIR generated with the DLR LMS channel model. The magenta line shows the reference range, i.e. the exact absolute delay of the direct path.

Fig. 4 shows an example of a generated CIR generated using the DLR LMS urban channel model [7]. The CIR is interpolated using a bandwidth of $B = 400\,\text{MHz}$. It demonstrates the fast time-variance of the channel in an urban environment.

The signal at the output of this processing module is the complex discrete-time convolution of the input signal with the time-discrete CIR:

$$s_{\text{c,o}}(t_n) = s_{\text{i}}(t_n) * h_{\text{discr}}(t_m, t_n) \tag{7}$$

2.3 Noise generation

The thermal noise is represented by additive white Gaussian noise (AWGN). The carrier-to-noise-ratio (CNR) is an input parameter given in dB. It is converted to a linear value according to

$$\text{CNR}_{lin} = 10^{\text{CNR}_{dB}/10} \tag{8}$$

and used to calculate the noise sample variance for each in-phase and quadrature noise component according to

$$\sigma_n^2 = \frac{f_{\text{s}}}{2} \cdot \frac{1}{\text{SNR}_{lin}}. \tag{9}$$

The zero-mean unit-variance Gaussian random complex sequence $s_{\text{rnd}}(t_n)$ (generated by the polar algorithm [11]) is scaled with σ_n and added to the inphase and quadrature components of the incoming signal:

$$s_{\text{n,o}}(t_n) = s_{\text{i}}(t_n) \; + \sigma_n \cdot s_{\text{rnd}}(t_n) \tag{10}$$

Since the input signal of the noise processing module is assumed to have unit power the resulting signal $s_{\text{n,o}}(t_n)$ exhibits the desired CNR.

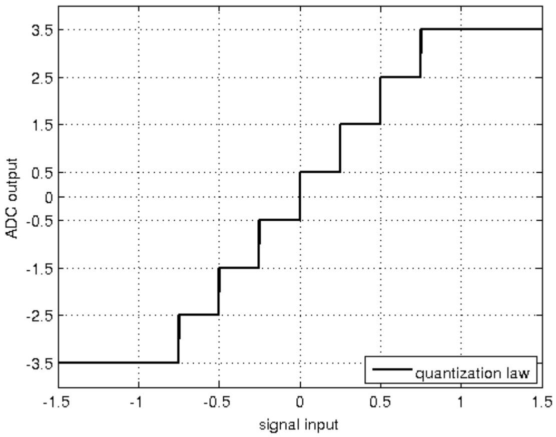

Figure 5. Example of an ADC quantization function for 3-bit quantizer.

2.4 Quantization

The signal generated by above described modules can be processed by an optional variable-gain amplifier (VGA) before it enters the quantization process; The VGA emulates the behavior of an ideal automatic gain control (AGC). This operation ensures that the full range of the following analog-to-digital converter (ADC) is equally excited. The VGA computes the maximum absolute sample within the currently processed block:

$$a_{\max}(m) = \max\{|s(t_n)|\}|_{n=mK\ldots(m+1)K-1} \tag{11}$$

and scale all samples in the current signal data block by this value:

$$s_{\text{VGA}}(t_n)(m) = \left.\frac{s_{\text{i}}(t_n)}{a_{\max}(m)}\right|_{n=mK\ldots(m+1)K-1} \tag{12}$$

In addition, the maximum and minium values of $a_{\max}(m)$ can be limited.

Then, the ADC up-converts either $s_{\text{VGA}}(t_n)$ or $s_{\text{i}}(t_n)$ at the intermediate frequency f_{IF}:

$$s_{\text{up}}(t_n) \quad = \quad s_{\text{i}}(t_n) \; \cdot e^{j2\pi f_{\text{IF}} t_n} \tag{13}$$

A one-bit quantizer considers the polarity of $s_{\text{up}}(t_n)$ only:

$$s_{\text{ADC,out}}(t_n) = \begin{cases} +1, & \text{if } s_{\text{up}}(t_n) = 1 \\ -1, & \text{if } s_{\text{up}}(t_n) = 1. \end{cases} \tag{14}$$

For the case of higher order bit quantization, a non-centered quantization law is applied. Fig. 5 shows the quantization function for 3-bit quantization.

2.5 Acquisition

The code-phase and frequency-offset aquisition is performed using the parallel code-phase search method as outlined in

[4]. At first, the incoming signal is multiplied with the inphase and quadrature components of a locally generated carrier at f_{IF}. To search for a Doppler shift of ±4 kHz in 500 Hz steps, $8\,\text{kHz}/500\,\text{Hz} = F_{\text{B}} = 16$ frequency bins have to be searched. The incoming signal $s_{\text{i}}(t_n)$ is down-converted using the F_{B} different carrier frequencies:

$$\left[s_{\text{i, fbins}}(t_n) = s_{\text{i}}(t_n)\ \cdot e^{j2\pi f_{\text{aq,i}}\cdot t_n}\right]_{i=1\dots F_{\text{B}}} \qquad (15)$$

The result is Fourier-transformed and multiplied with the complex conjugate of the Fourier-transformed PRN code sequence:

$$S_{\text{i,aq}}(f_n) = \mathcal{F}\{s_{\text{i,fbins}}(t_n)\}\cdot\mathcal{F}^*\{s_{\text{GPS,prn}}(t_n)\} \qquad (16)$$

To obtain the final aquisition result, the magnitude of the inverse Fourier transforms of $s_{\text{i,fbins}}(t_n)$ is computed:

$$s_{\text{i,res}}(t_n) = \left|\mathcal{F}^{-1}\{S_{\text{i,aq}}(f_n)\}\right| \qquad (17)$$

The index of the largest value of all $s_{\text{i,res}}(t_n)$

$$i_{\text{aq, max1}} = \arg\max\{s_{\text{i,res}}(t_n)|_{i=1\dots F_{\text{B}}}\} \qquad (18)$$

and the index of the second largest value

$$i_{\text{aq, max2}} = \arg\max\{s_{\text{i,res}}(t_n)|_{i=1\dots F_{\text{B}}, i\neq i_{\text{aq, max1}}}\} \qquad (19)$$

are subsequently determined. If the ratio

$$r_{\text{aq}} = \frac{s_{i,res}(t_n)|_{i=i_{\text{aq, max1}}}}{s_{i,res}(t_n)|_{i=i_{\text{aq, max2}}}} \qquad (20)$$

exceeds a certain threshold, for example $r_{\text{aq}} = 2.5$, the signal Doppler shift and code-phase is considered to be found and the controller starts the software receiver's tracking loops.

2.6 Carrier phase and code-phase tracking

The tracking loop module employs a phase-lock-loop (PLL) and a delay-lock-loop (DLL) as described in [4]. The incoming signal $s_{\text{i}}(t_n)$ is down-converted using

$$s_{\text{down}}(t_n) \quad = \quad s_{\text{i}}(t_n)\ \cdot e^{j2\pi f_{\text{c}} t_n} \qquad (21)$$

In the standard configuration, SNACS correlates the down-converted signal with three local GPS C/A PRN code replicas to obtain the output of the early, prompt, and late inphase and quadrature correlation branches I_e, Q_e, I_p, Q_p, I_l, Q_l, respectively. In addition, the simulation package can also be configured to output further correlation results at equally spaced chip spacings which could be used as input for further multipath mitigation algorithms. This has been demonstrated in [8] with a particle-filtering multipath mitigation algorithm.

The PLL is implemented as a Costas loop using the prompt inphase and quadrature with $\varphi = \arctan(Q_p/I_p)$ as its discriminator. Both PLL and DLL are implemented as second-order control loops. The PLL loop adjusts f_{c} for the down-conversion step using a numerically controlled oscillator (NCO). The PLL is initialized with the Doppler frequency calculated during the aquisition step.

As for the DLL, the normalized early-minus-late and the dot-product discriminators can be chosen. The DLL loop filter adjusts the advancement or the lag of the PRN code-phase using an NCO, too. The value of the initial code phase has been computed in the aquisition step.

The block length of the received data of the 1 ms long GPS C/A code varies due to the simulated receiver movement. Hence, the incoming signal samples are serialized before they enter the aquisition and tracking module. The tracking algorithm computes the expected block length of the incoming C/A code period based on the current DLL NCO output.

The resulting pseudorange is calculated by determining the travel time of the signal from transmitter to the receiver. The absolute time t_n of the current sample with number n serves as time reference. The number of the actually processed samples per code is related to this time reference by multiplying it with the length of a whole PRN sequence. A residual code-phase can be computed at the end of each processed C/A code period. This code-phase remainder is used to further increase the accuracy of the pseudorange.

3 SIMULATION EXAMPLE

An example simulation with the DLR LMS channel model is shown in Fig. 6. The model input parameters for the artificial scenery are the same as given in the example configuration file which is included in the channel model implementation [1]. A slowly accelerating and decelarating trajectory was chosen as movement parameter for the receiving vehicle. In the SNACS simulation the sampling frequency is selected to be $f_{\text{s}} = 40\,\text{Mhz}$ and the DLL discriminator is set to a spacing of 1 chip. The DLL and PLL noise bandwidths are set to 5 Hz and 25 Hz, respectively. The noise generator's CNR_{dB} is adjusted to 45 dB-Hz. The rms error for this simulation is 8.80 m.

SNACS also offers a graphical user interface to monitor running simulations. Each processing module visualizes its processed data in columns which can be arranged individually. A screenshot of one simulation with the DLR LMS channel model is shown in Fig. 7. The three columns represent the CIR to FIR interpolation, the up-conversion and quantization, the correlation function and pseudorange result, respectively.

The pseudorange determined by the tracking loop can be compared to the channel model's reference pseudorange output. Matlab scripts are available for the analysis of radio channel data input and the SNACS simulation results.

4 CONCLUSION AND OUTLOOK

SNACS has been introduced as an accurate and fast software simulator for applying frequency-selective time-variant radio propagation channels in time-domain in the com-

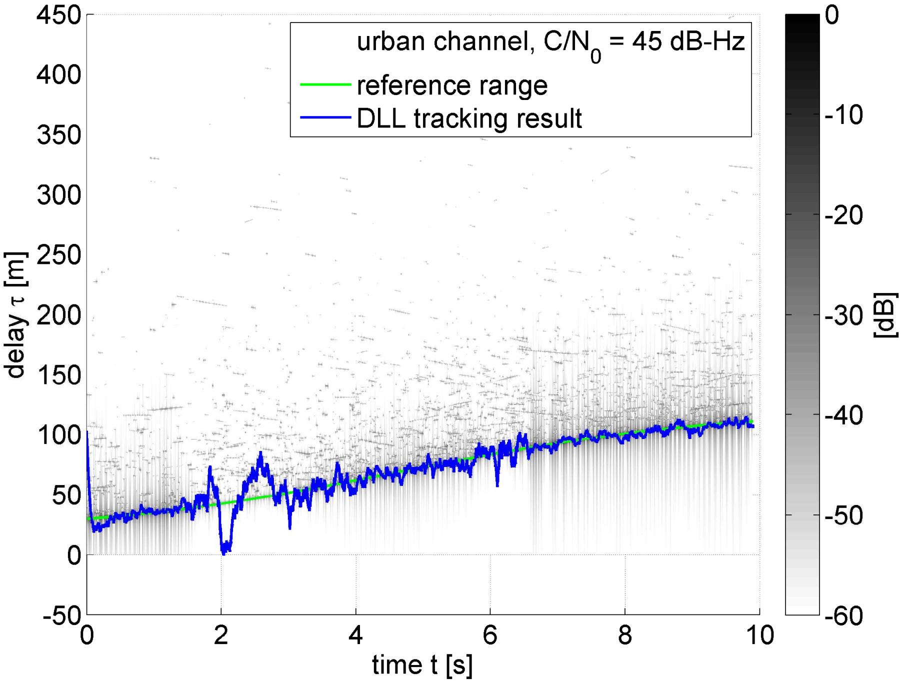

Figure 6. SNACS example simulation run with DLR land-mobile urban channel model.

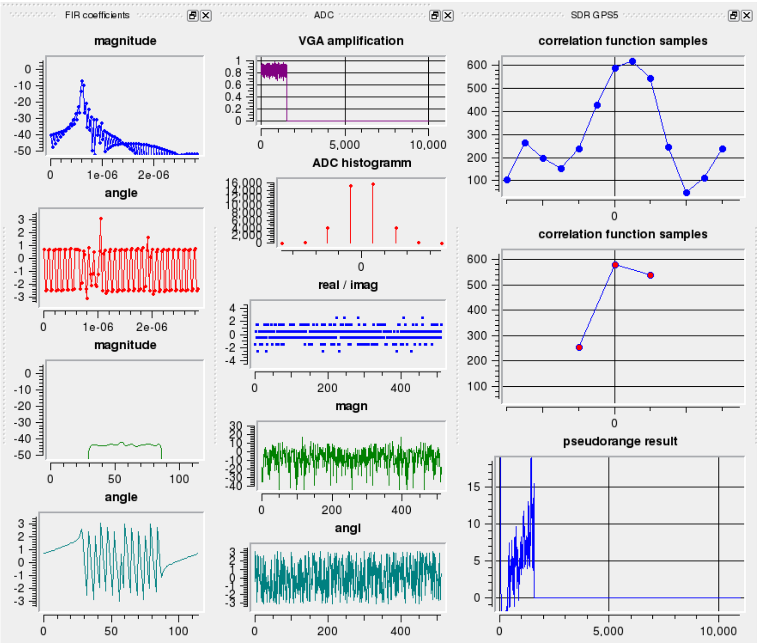

Figure 7. Screenshot of a running simulation. The three columns show the output of the FIR coefficients interpolation module, the quantization module, and the tracking loops module, respectively.

plete GNSS signal chain. The software tool is written using an object-oriented approach as well as parallel-processing. It provides a sound basis for the easy adoption of existing components or the development of new source, sink, or processing modules.

It is planned that the various Galileo signals and receiver architectures will be implemented in the future.

Since the main focus of SNACS lies within the pseudorange tracking domain, data bit demodulation and computation of a position solution has not been implemented so far. But the modular architecture of SNACS allows for an implementation of simulations consisting of more than one satellite link in the future.

SNACS is open source and licensed under the GNU Public Licence (GPL). The source, binary distributions, and the documentation can be downloaded from the project website [2].

REFERENCES

[1] Implementation of the DLR land mobile channel model:
http://www.kn-s.dlr.de/satnav.

[2] The SNACS project web page:
http://snacs.sourceforge.net.

[3] G. Artaud, G. Menard, L. Ries, J. Dantepal, and J.-L. Issler. Juzzle software receiver. In *Proceedings of Navitec 2006*, Noordwijk, The Netherlands, 2006.

[4] K. Borre, D.M. Akos, N. Bertelsen, P. Rinder, and S.H Jensen. *A Software-Defined GPS and Galileo Receiver, A Single-Frequency Approach*. Birkhäuser, Boston, USA, 2007.

[5] A. Fernandez, J. Diez, L. Marradi, and V. Gabaglio. Galileo receiver performance under GPS interference and multipath with the GRANADA software receiver. In *Proceedings of the 17th International Technical Meeting of the Satellite Division of the Institute of Navigation*, Long Beach, California, USA, 2004.

[6] J. Furthner, E. Engler, A. Steingass, M. Angermann, J. Hahn, A. Hornbostel, R. Krämer, H. P. Müller, T. Noack, P. Robertson, S. Schlüter, and J. Selva. Realization of an end-to-end software simulator for navigation systems. *International Journal of Satellite Communications, Special Issue: Satellite Navigation and Positioning*, 2000.

[7] A. Lehner and A. Steingass. A channel model for land mobile satellite navigation. In *Proceedings of the 18th International Technical Meeting of the Institute of Navigation Satellite Division*, Long Beach, California, USA, 2005.

[8] F. M. Schubert and B. Krach. Simulation of high-realistic multipath environments: Developments and applications. In *Proceedings of the 2nd EuNavTec Positions Congress 2008*, Dresden, Germany, 2008.

[9] A. Steingass and A. Lehner. Navigation in multipath environments for suburban applications. In *Proceedings of the 20th International Technical Meeting of the Institute of Navigation Satellite Division*, Fort Worth, Texas, USA, 2007.

[10] A. Steingass, A. Lehner, F. Pérez-Fontán, E. Kubista, and B. Arbesser-Rastburg. Characterization of the aeronautical satellite navigation channel through high-resolution measurement and physical optics simulation. *International Journal of Satellite Communications and Networking*, 26, 2008.

[11] W. H. Tranter, K. S. Shanmugan, T. S. Rappaport, and K. L. Kosbar. *Principles of Communication Systems Simulation*. Prentice Hall, 2004.