Signal Processing Algorithms in GPS, Galileo, and GLONASS Integrated Receiver

Ondrej Jakubov, PhD Student, Pavel Kovar, Adviser

Abstract—Due to the increasing number of receiver-visible space vehicles (SVs), the thesis aims to develop advanced positioning, velocity, time (PVT) estimation algorithms that are of lower complexity and equivalent quality in comparison with the existing ones. The factor graph framework is proposed to be the underlying theory as several articles have been published concerning reduced-dimensionality and relatively precise localization in wireless communications. The key idea hidden there is to split vector vertices, of the factor graph modeling relations among vector variables, into scalar vertices and start iterative message passing in a cycle factor graph, thus avoiding matrix operations. The algorithms will be directly implemented and tested in universal software GNSS receiver, named the Witch Navigator (WNAV), which development is a parallel line of the PhD study.

Index Terms—factor graph, GNSS, multi-frequency reception, multipath mitigation, multi-system reception, NLOS positioning, PVT estimation, RAIM, software receiver, sum-product algorithm, vector tracking

I. INTRODUCTION

The upgrade of existing and construction of new GNSS systems offer position, velocity, time (PVT) estimation of higher precision and robustness. However, the increased number of visible and processed space vehicles (SVs) seriously boosts computational requirements on advanced signal processing algorithms mostly due to matrix operations, which complexity grows with second or third power of the number of visible SVs. These algorithms include PVT estimation/filtering [1]–[6], augmented by vector tracking [7]–[13], receiver autonomous integrity monitoring (RAIM) [2], multipath mitigation techniques and non-line-of-sight positioning [1], [2], [14]–[17], optionally integration of these algorithms.

The aim of the thesis is to develop equivalent algorithms that are of complexity directly related to the number of visible SVs or lower than of the existing ones. The factor graph framework is proposed to be the underlying theory as several articles have been published concerning reduced-dimensionality and relatively precise localization in wireless communications. The key idea hidden there is to split vector vertices, of the factor graph modeling relations among vector variables, into scalar vertices and start iterative message passing in a cycle factor graph, thus avoiding matrix operations.

The algorithms will be directly implemented and tested in universal software GNSS receiver, named the Witch Navigator (WNAV) [18], which development is a parallel line of the PhD study. The Witch Navigator is an open source project, the resulting algorithms will hence be available publicly on its website. Immediate verification, comments, ideas by WNAV users will be possible.

Organization of this document is as follows. First, we discuss the context beside which the research topic born, mostly speaking about WNAV. Lowering CPU and memory requirements, discussed in next section, is the motivation to the main problem. Then, we state in detail which algorithms are expected to be developed. Finally, we propose the underlying theory accompanied by a comprehensive example, and provide relevant references to the already resolved similar problems using this theory.

II. CONTEXT OF THE RESEARCH

In this section, we discuss the Witch Navigator project, as it motivated the research topic of the thesis. Its development is another crucial part of the study.

A. The Witch Navigator Project

The Witch Navigator (WNAV) is an open source project aiming to develop a low-cost high-performance GNSS software receiver capable to process most of the present and future GNSS signals [18].

The receiver consists of an ExpressCard receiver (Fig. 1), hosting two highly reconfigurable RF front-end channels [19] and FPGA-based universal correlators optimized to process majority of the GNSS signals [20], and PC or notebook. The high throughput and prompt communication over the PCI Express interface enables that the PC (or notebook) can serve as a control unit for the correlators, accomplish the acquisition, data decoding and PVT estimation. Optionally, the signal snapshots may be continuously transferred to the PC in real time, and fully processed there.

The flexible architecture enables layering ExpressCards, thus increasing the number of RF channels and number of available correlators. Undoubtedly, the Witch Navigator becomes a multi-system, multi-frequency, and multi-antenna GNSS receiver. The receiver further features connectors for inertial sensors (INS) and external frequency standard.

1) Open Source Philosophy: The open source philosophy of the project stems from the following facts. The ExpressCard receiver can be ordered on the internet websites where all the supporting source code is freely downloadable and home-editable. The communication driver and application programs are developed in C language under the Linux operating...
system with real-time patch. These and all other development tools, including the hardware part, are also free of charge.

2) Hardware: The block diagram of a single ExpressCard is depicted in Fig. 2, PCB layout in Fig. 3. Each card contains:

- two direct conversion receivers
- two dual channel 8-bit ADCs
- Spartan 6 FPGA with PCIe bus
- configuration flash
- highly stable quartz oscillator (20 MHz, 1 ppm)
- linear voltage regulators
- connector for interconnection of the receivers to the large system
- connector for external sensor or device.

The direct conversion receivers (MAX2120), can each be reconfigured from the PC. Their synthesizer’s frequency spans over the range of 925 ÷ 2175 MHz with 4 ÷ 40 MHz adjustable baseband filter. The integrated circuits feature voltage controlled amplifier with 75 dB dynamic range, and 20 dB reconfigurable baseband amplifier. Active antenna should be connected to the RF channel inputs (CH1, CH2) via MMCX connector.

The block diagram of the FPGA processor is depicted in Fig. 4. The universal correlators (UCorIP) communicate with the PC via the PCI express interface (PCIeIP, PCIe), clocked by phase lock loop (PLL125). The I2C controller serves as an interface between the direct conversion receivers and the PC, control and status data words are there wrapped into PCIe packets.

The universal correlator (UCorIP) is an economic implementation of the classical E-L correlator. It utilizes RAM based PRN generators of maximal length of 10230 chips. The signals that can be processed are listed in Table 1. The currently used FPGA (Spartan 6) has 24 of these E/L correlators. In other words, a single ExpressCard can process up to 24 BPSK signals in parallel using the FPGA. Another part of UCorIP is the signal capture unit which captured samples (=snapshots) are transmitted on the PCIe bus using direct memory access (DMA) at every time interval counter (TIC) event, occurring every 0.8 ms. After the DMA transfer finishes, the PC is interrupted by the FPGA and software handling begins. The situation is depicted in Fig. 5. The control words produced by the PC are send back on the bus at the end of the software handling, which is not illustrated in the figure.

3) Software: The PC software is primarily designed for the conventional architecture (=UCorIP in use). Transition to the pure software-defined-radio architecture is straightforward (=processing fully by the PC).

The requirements on low latency interrupt handling routing and open source project resulted in selection of the Linux operating system (OS) with real-time (RT) preemptive patch. RT preemptive patch changes a standard Linux kernel into a soft RT kernel. Soft RTOS is an OS that does not guarantee the RT behavior, however, with a high probability will behave like RTOS. The advantages it brings are
In case of an occasional outage, an error handling routine is implemented. The PC software diagram is depicted in Fig. 6. The developed PCI Express driver operates in the kernel space while other processes, including the FPGA handling process, acquisition process, and PVT process are accessible in the user space. The FPGA handling process is a real-time process with the highest priority. The interrupt handling routine is implemented as a blocking read cycle, which is being unblocked by the interrupt. The interface between the user and the running software is provided via a monitoring process with the highest priority. The interrupt handling routine is implemented.

The PC software has been designed to be easily extendable to the multi-system, multi-frequency version. GPS L2/C, Galileo E1b, Galileo FNAV, INAV modules are under construction, yet the acquisition and tracking modules of Galileo E1, E5 signals have been developed. The acquisition supports both serial, and parallel algorithms based on FFT. The tracking modules is implemented using separate DLL/FLL/PLL. The PVT estimation module has implemented the least squares (LS), weighted least squares (WLS), extended Kalman filter (EKF) algorithms, and a novel factor-graph-based (FG) filtering algorithm which is about to be published. The receiver supports cold and hot start, warm start is being implemented.

5) Future Plans: In future versions, the project is planned to undergo the following changes:

- reception of other GNSS signals, namely GPS L5C, COMPASS signals
- reception of SBAS signals, including EGNOS, WAAS, QZSS
- cooperation of multiple ExpressCards
- employment of a higher capacity FPGA
- development of an industrial version of the receiver (embedded PC)
- implementation of vector tracking
- implementation of RAIM
- implementation of multipath mitigation techniques, non-line-of-sight positioning techniques
- fast GPU-based acquisition
- release of program-stability, bug-removing patches
- release of patches increasing program’s efficiency (lowering CPU, memory load)
• at user space programs transition to an object-oriented programming language (C++, Python,...)
• join of other enthusiastic people, organizations cooperating on the development.

The main application area of the Witch Navigator lies in education, research, and scientific small scale applications. The project does not aim to compete with commercial companies, but will likely provide materials useful for the development of their projects. The project is not funded by any institution and is politically independent.

B. Student’s Project Tasks and Responsibilities

The proposing student has the following responsibilities and tasks concerning the Witch Navigator project:
• RF front-end control
• frame synchronization
• data decoding, deinterleaving
• navigation data storage
• PVT estimation/filtering
• cooperation on the selection of RTOS
• cooperation on the interprocess communication (IPC)
• others (web design, templates, logos,...).

The PhD student shall involve university undergraduate students to accomplish easier tasks in order to speed up the development process.

III. MOTIVATION

The main topic of thesis, as stated in the introduction, will be the development of low-complexity signal processing algorithms with respect to the increasing number of visible SVs. The problem lies in the fact that most of the algorithms employs matrix manipulations, resulting in second or third power complexity. When reception of GPS, GLONASS, Galileo, GLONASS signals, the number of processed SVs can reach up to 40. Adding SBAS SVs, pseudolites or other radio beacons, the number can be even higher.

Fig. 7 illustrates the computational complexity of LS, WLS, EKF algorithms for PVT estimation/filtering in terms of the number of floating point operations (flops). Suppose, for simplicity, 1 GHz CPU with a single flop taking one CPU cycle. The extended Kalman filter for 40 SVs would take 0.3 ms, for 60 SVs 1 ms, and for 80 SVs 3 ms. These time relations appear to be acceptable, but using them in a 100 MHz CPU or lower might be limiting. In addition to it, considering more advanced signal processing methods, the elapsed time will likely be much higher.

Fig. 7. Complexity of LS, WLS, EKF algorithms for PVT estimation/filtering

IV. DETAILED DESCRIPTION OF THE RESEARCH, DESIGN, EXPECTED OUTCOMES

The detailed description of the research, design and expected outcomes may be summarized as follows
• detailed study of the existing advanced PVT estimation/filtering methods including multi-system, multi-frequency reception, vector tracking, RAIM, multipath mitigation techniques, NLOS positioning, and mutual integration of these methods
• precision, robustness\(^1\), complexity analysis of the methods
• detailed study of similar existing non-GNSS FG-based methods
• design of new methods that are of “equivalent quality” and lower complexity with respect to the number of visible SVs or other radio beacons, splitting this into the following steps

1. find a factor graph modeling relations between the vector variables, and the message passing algorithm, resulting in the known algorithm
2. split the vector variable nodes into scalar ones, accordingly do with the factor nodes
3. if cycles appear in the FG, find an appropriate probability-density-function (PDF) representation, and iterative message passing algorithm
4. investigate convergence, robustness, precision, and complexity of the algorithm
5. investigate if the algorithm is easily extendable with respect to the number of SVs, time references
6. investigate if the algorithm is distributed in order to split the computational power into parallel processing entities

\(^1\)the term robustness refers to the availability, continuity, and integrity of the algorithm
investigate if the algorithm may be easily integrated with other algorithms.

8) if points 3-7 do not fulfill the requirements, try to modify the factor graph, PDF representation, or message passing algorithm

- use the Witch Navigator receiver as the main verification tool
- publish the source code of the algorithms freely on its website, after release in a journal, conference proceedings etc.

V. FACTOR GRAPHS, THE SUM-PRODUCT ALGORITHM, AND LOCALIZATION

In this section, we sum up the adoption of factor graph framework to the localization in wireless communications, overview the general theory of factor graphs and the sum-product algorithm, and deliver an illustrative example for a 2-D position estimation using time-of-arrival (TOA) measurements.

A. Factor Graphs in Localization

Factor graph (FG) is a bipartite graph that represents relations among variables of a system [21], [22]. The system is assumed to be described by a complicated global function that factors into simpler local functions, each of which having arguments from a subset of the system variables. The sum-product algorithm (SPA) is a generic message-passing (MP) algorithm which operates in a factor graph and attempts to compute various marginal functions associated with the global function.

A wide variety of algorithms developed in artificial intelligence, signal processing, and digital communications can be derived as specific instances of the sum-product algorithm, operating in an appropriately chosen factor graph [21], [22]. Examples include iterative algorithms for decoding low-density-parity-check codes (LDPC) and turbo codes, the Viterbi algorithm, the Kalman filter, and certain fast Fourier transform (FFT) algorithms.

Factor graphs were first adopted to localization of a mobile station (MS) in wireless communication systems in 2003 [23], [24]. The MS therein estimates signal time of arrival (TOA) from base stations (BS) with standard radius of 5 km. After a simple initialization, 2-D position is iteratively estimated using a simple easy-to-implement MP algorithm with relatively high precision in comparison with the complex maximum likelihood (ML) algorithm. Later contributions incorporate similar approaches to time-difference-of-arrival (TDOA) or angle-of-arrival (AOA) localization [25], [26].

In [27], the authors propose a method for non-line-of-sight (NLOS) positioning in wireless communications using factor graph. The messages are therein represented by samples of the estimated probability density function (PDF). The update rules for the vertices are accomplished using importance sampling.

A universal algorithm based on FG, called SPAWN, has been developed for cooperative localization in wireless networks [28]. This promising algorithm takes into account the history of the measurements, motion model and is fully distributed. The tests were conducted using ultra-wideband (UWB) radios indoor buildings. A hybrid GNSS/UWB extension for cooperative localization is presented in [29].

B. Factor Graph and the Sum-Product Algorithm

In this section, we briefly define the factor graph and describe the sum-product algorithm. Here, we restrict factorization and marginalization to multiplication and integration, respectively. The global and local functions will be represented by the probability density functions (PDF). Although the factor graph framework applies to more general problems, such constraint will make our approach more illustrative.

Let’s assume that global function \( p(X) \geq 0 \) of \( K \) system variables \( X = \{x_1, \ldots, x_K\} \) where \( \forall k \in \{1, \ldots, K\} : x_k \in \mathcal{A}_k \subseteq \mathbb{R} \) factors into a product of local functions \( \{p_j(X_j) : j \in J \land p_j(X_j) \geq 0\} \)

\[
p(X) = \prod_{j \in J} p_j(X_j)
\]

for \( X_j \) being a subset of the global function arguments \( X_j \subseteq X \) and \( j \) denoting the index of the corresponding local function in set \( J \) of such indices. Factor graph is a bipartite graph that visualizes the factorization using three types of components: variable nodes - each representing the system variable \( x_k \), factor nodes - each representing the local function \( p_j(X_j) \), and edges - connecting the variable nodes \( x_k \) and factor nodes \( p_j(X_j) \) if and only if \( x_k \in X_j \). An example FG where global function \( p(x_1, x_2, x_3, x_4) \) factors into local functions \( p_A(x_1, x_2, x_3) \), \( p_B(x_3, x_4) \) is in Fig. 8.

![Figure 8. Example FG with MP for global function \( p(x_1, x_2, x_3, x_4) \) that factors into local functions \( p_A(x_1, x_2, x_3) \), \( p_B(x_3, x_4) \) in the following manner \( p(x_1, x_2, x_3, x_4) = p_A(x_1, x_2, x_3) \cdot p_B(x_3, x_4) \). The direction of the messages is denoted by arrow.](image)

The marginal function \( p_i(x_i) \) is defined as a summation of the global function \( p(x_1, \ldots, x_K) \) over all arguments except \( x_i \), denoted by \( \sim x_i \),

\[
p_i(x_i) \triangleq \int_{\sim x_i} p(x_1, \ldots, x_K) \, dx_1 \ldots dx_K
\]

\[
\triangleq \int_{\sim x_i} p(X) \, dX.
\]

In the given example, e.g. the marginal function \( p_3(x_3) \) would be computed as

\[
p_3(x_3) = \int_{x_1 \in A_1} \int_{x_2 \in A_2} \int_{x_4 \in A_4} p(x_1, x_2, x_3, x_4) \, dx_1 \, dx_2 \, dx_4.
\]

The key property that multiplication distributes over summation is adopted in FG to distributively compute the resulting...
marginal function \( p_i(x_i) \) from marginalized local functions \( \int_{x_i} p_j(X_j) \) : 
\[
p_i(x_i) = \int_{x_i} p(X) \, dX = \int_{x_i} \prod_{j \in J} p_j(X_j) \, dX = \prod_{j \in J} \left( \int_{x_i} p_j(X_j) \, dX_j \right).
\]

In the given example
\[
p_3(x_3) = \left( \int_{x_1 \in A_1} \int_{x_2 \in A_2} p_A(x_1, x_2, x_3) \, dx_1 \, dx_2 \right) \cdot \left( \int_{x_4 \in A_4} p_B(x_3, x_4) \, dx_4 \right).
\]

Computation of a single marginal function in a tree FG is then a “bottom-up” procedure that starts at the leaf vertices. The leaf vertices send trivial identity messages to their parents which wait for messages from all their children before they start calculation of the local marginal functions. When the marginalization is completed in a vertex, it becomes a child and sends the results to its parents. The algorithm continues until the target marginal function is obtained.

In our example (Fig. 8), in order to get the resulting marginal function \( p_3(x_3) \), variable nodes \( x_1, x_2 \) first send simple identity messages \( \lambda_{x_1 \rightarrow A}(x_1) \), \( \lambda_{x_2 \rightarrow A}(x_2) \) to factor node \( A \), and so sends the variable node \( x_4 \) message \( \lambda_{x_4 \rightarrow B}(x_4) \) to factor node \( B \). The local marginals, the expressions in parenthesis in (1), are evaluated and information about the results is send in messages \( \mu_{A \rightarrow x_3}(x_3) \), \( \mu_{B \rightarrow x_3}(x_3) \) to variable node \( x_3 \). According to (1), the resulting marginal function is a product of these local marginals, symbolically,
\[
p_3(x_3) = \mu_{A \rightarrow x_3}(x_3) \cdot \mu_{B \rightarrow x_3}(x_3).
\]

We use the term “symbolically”, since the messages passed in a FG represent information about a PDF, not necessarily the PDF itself. However, it is instructive to use these messages to denote the corresponding marginal PDFs.

The sum-product algorithm is a generalization of the MP algorithm for efficient calculation of all the marginal functions associated with the global function. The summary is given in Algorithm 1. Messages from factor node to variable node are denoted with \( \lambda \), whereas messages from variable node to factor node with \( \mu \).

In a tree factor graph, the beliefs truly represent the marginals of the global function. However, when cycles appear in the graph, the sum-product algorithm causes deadlocks. It follows from the fact that vertices in a cycle infinitely wait for results from each other. A solution to this problem might be to initialize the messages and let the sum-product algorithm iterate on the FG. The convergence is mostly difficult to prove. Nonetheless, “appropriate” initial conditions usually succeed.

C. Example of 2-D Position Estimation from TOA Measurements Using FG

In this subsection, we give an illustrative example of 2-D position estimation from TOA measurements using FG. The

Algorithm 1 The sum-product algorithm

Let \( x \) denote a variable node, \( F \) a factor node, and \( n(x), n(F) \) the sets of neighbors of variable node \( x \) and factor node \( F \), respectively. The messages sent in a FG are recursively computed according to the following algorithm [22]:

**Variable Node to Factor Node:**
\[
\lambda_{x \rightarrow F}(x) = \prod_{G \in n(x) \setminus \{F\}} \mu_{G \rightarrow x}(x)
\]

**Factor Node to Variable Node:**
\[
\mu_{F \rightarrow x}(x) = \int_{x} p_F(X) \prod_{y \in n(F) \setminus \{x\}} \lambda_{y \rightarrow F}(y) \, dx
\]

where \( X \) is a set of all arguments of \( p_F \). The marginalized version of the global function with argument \( x \) is named belief and is evaluated as
\[
B(x) = \prod_{G \in n(x)} \mu_{G \rightarrow x}(x).
\]

example was inspired by [23], [24]. Suppose that the user position is denoted as \( x = [x_1 y]^T \), position of \( i \)th SV as \( x_{SV,i} = [x_{SV,i} y_{SV,i}]^T \), the true distance between the user and \( i \)th SV is then
\[
d_i = \sqrt{(x - x_{SV,i})^2 + (y - y_{SV,i})^2} (2)
\]
where
\[
\begin{bmatrix}
g_1(x) \\
\vdots \\
g_I(x)
\end{bmatrix} \triangleq \begin{bmatrix} d_1 \\
\vdots \\
d_I
\end{bmatrix} = \begin{bmatrix}
\| x - x_{SV,1} \| \\
\vdots \\
\| x - x_{SV,I} \|
\end{bmatrix}
\]

we can model the measured distance \( z \) as
\[
z = g(x) + \mathbf{w}
\]
where \( \mathbf{w} \) is white Gaussian noise with zero mean \( \mu[\mathbf{w}] = 0 \) and known diagonal covariance matrix
\[
\mathbf{C}_w = \text{diag}(\sigma_{d_1}^2, \ldots, \sigma_{d_I}^2).\]

Let \( \mathbf{1}_i \) denote unit line-of-sight vector between the user and \( i \)th SV
\[
\mathbf{1}_i \triangleq \frac{x - x_{SV,i}}{\| x - x_{SV,i} \|}
\]
and matrix \( \mathbf{G} \) the geometry (cosine) matrix, calculated about the linearizing position \( \mathbf{x} \)
\[
\mathbf{G} \triangleq \begin{bmatrix}
\frac{\partial g(x)}{\partial \mathbf{x}} |_{x=\mathbf{x}} \\
\vdots \\
\frac{\partial g(x)}{\partial \mathbf{x}} |_{x=\mathbf{x}}
\end{bmatrix} = \begin{bmatrix}
\mathbf{1}_1^T \\
\vdots \\
\mathbf{1}_I^T
\end{bmatrix}
\]

Since the measured distances are assumed to be uncorrelated, the vector likelihood \( p(\mathbf{z} | \mathbf{d}) \) factors into scalar likelihoods
The factor graph for global function on the right side of (9) is in Fig. 8 (left). This is the vector case. In the scalar case, the estimates of user position elements can be written as

\[
\hat{x} = \mathbb{E}_x [p(x|z)] = \arg\max_x p(x|z)
\]

\[
= \arg\max_x \frac{p(z|x)p(x)}{p(z)}
\]

\[
= \arg\max_x p(z|x)p(x).
\]

Further factorizing the last equation, we get

\[
\hat{x} = \arg\max_x \prod_{i=1}^l p(z_i|d_i) p(d|x) p(x) 
\]

\[
= \arg\max_x \int_{\{y\}} p(d|x, y) p(y) dy.
\]

Similarly, the estimate of \( y \) is

\[
\hat{y} = \arg\max_y \prod_{i=1}^l p(z_i|d_i) p(y) 
\]

\[
\int_{\{x\}} p(d|x, y) p(x) dx.
\]

The corresponding factor graph is in Fig. 8 (right). All the PDFs denoted in Fig. 8 (left, right) complies with the factor functions

\[
p_{A_1}(d_1) = p(z_1|d_1)\]

\[
p_{A_i}(d_i) = p(z_i|d_i)\]

\[
p_{A_1}(x) = p(x)\]

\[
p_{C_x}(x) = p(x)\]

\[
p_{C_y}(y) = p(y)\]

except for nodes \( B, B_1, \ldots, B_I \) where a linearization must be adopted

\[
p_B(d|x) = \delta\left( (d - \hat{d}) - G(x - \hat{x}) \right)
\]

\[
p_{B_i}(d_i, x, y) = \delta\left( (d_i - \hat{d}_i) - g_{i,1}(x - \hat{x}) - g_{i,2}(y - \hat{y}) \right)
\]

where \( \hat{d} = g(\hat{x}) \). It is because the transformation of the user position \( x \) into the true distance \( d_i \) does not preserve Gaussianity. In the vector case, update rules for factor graphs with Gaussian PDFs are summarized in [30]. In the scalar case, the update rules are in Algorithm 2. The final update rules and message passing algorithms are in Algorithm 3 (the vector case), Algorithm 4 (the scalar case). The variances of initial position estimation \( \sigma_{x_0}^2, \sigma_{y_0}^2 \) were set to infinity. The MMSE estimate is then identical to the maximum likelihood (ML) estimate, and the weighted least squares estimate (WLS).

Note that both vector and scalar algorithm are iterative. The reason for iterations in the vector case is due to the non-linear character. In the scalar case, the other reason is cycles in
the FG. The number of necessary iteration may hence differ. However, results in [23], [24] demonstrate that 4 iterations are enough in the scalar case for an estimate in the area of 10-km radius. Similar precision between the vector and scalar case was obtained therein. The complexity of the scalar case algorithm is in direct relation with the number of visible SVs. The algorithm is distributed.

Algorithm 2 Update rules for vertices with Gaussian PDFs

Variable Node to Factor Node (VN→FN):

\[ \sigma_b^2 = \frac{1}{\sum_{i=1}^{k} \sigma_{b_i}^2} \]

\[ m_b = \sigma_b^2 \sum_{i=1}^{k} \frac{m_{b_i}}{\sigma_{b_i}^2} \]

Factor Node to Variable Node (δFN→VN):

\[ \sigma_b^2 = \sum_{i=1}^{k} h_i^2 \sigma_{a_i}^2 \]

\[ m_b = - \sum_{i=1}^{k} h_i m_{a_i} \]

### Algorithm 3 Position estimation algorithm - vector case

1. \( \mu_{A,x}(d) = \lambda_{d-B}(d) = \{x, C_w\} \)
2. \( \mu_{C,x}(x) = \{x_0, C_{x_0}\} \)
3. set initial estimate \( \hat{x} = x_0 \)
4. set linearizing point \( \bar{x} = \hat{x} \)
5. calculate \( G \) using (8)
6. get the covariance matrix of the estimate \( C_x = (G^T W G)^{-1} \)
7. get new position estimate, \( W = C_w^{-1} \)
8. if \( ||\hat{x} - \hat{x}\|| \) is not sufficiently low, iterate 4-7
9. \( \mu_{B,x}(x) = \{\hat{x}, C_x\} \)

VI. CONCLUSION

The document proposed a PhD-thesis topic dealing with complexity reduction of the advanced signal processing algorithms in GNSS. It was shown that the increasing number of visible SVs may prevent the algorithms from implementation in low cost devices. Thus, an underlying theory, the factor graph framework, was proposed to efficiently modify the existing algorithms in order to get algorithms of lower complexity and equivalent quality. The procedures for deriving new algorithms were provided as well as an illustrative example for 2-D localization using TOA measurements. A testing device, the Witch Navigator receiver, was introduced. It was shown that it can adopt most of the advanced algorithms respecting various GNSS signals and systems.

REFERENCES


Ondrej Jakubov was born in Liberec, the Czech republic. He received his MSc in electrical engineering from the CTU in Prague in 2010.

He is a postgraduate student at the same university at the Department of Radio Engineering. His research interests include GNSS signal processing algorithms and receiver architectures. His research focuses on advanced positioning and tracking methods. He is an active member of the team developing multi-system, multi-frequency, and multi-antenna GNSS receiver, named the Witch Navigator.

Pavel Kovar was born in Uherske Hradiste in 1970. He received his MSc and PhD degree at the CTU in Prague in 1994 and 1998. Since 1997 to 2000 he worked in Mesit Instruments as a designer of avionics instruments. Since 2000 he is with the Faculty of Electrical Engineering of the CTU in Prague as an Assistant Professor and since 2007 as an Associate Professor (Doc.). His interests are receiver architectures, satellite navigation, digital communications and signal processing.