Comparison of Two Approaches for GNSS Receiver Algorithms: Batch Processing and Sequential Processing Considerations

Frank van Graas, Andrey Soloviev, Maarten Uijt de Haag, Sandjeev Gunawardena, Michael Braasch,
Ohio University

BIOGRAPHY

Frank van Graas is a Fritz J. and Dolores H. Russ Professor of Electrical Engineering and Principal Investigator with the Avionics Engineering Center at Ohio University, Athens, OH. Dr. Van Graas is a Past President of The Institute of Navigation (98-99). In 1996, he received the Johannes Kepler Award from the Satellite Division of the ION. Dr. Van Graas' research interests center on all facets of GPS, including aircraft precision approach and landing, attitude determination, and system integration.

Andrey Soloviev is a research engineer at the Ohio University Avionics Engineering Center. He has received B.S. and M.S. in Applied Mathematics and Physics from Moscow University of Physics and Technology and a Ph.D. in Electrical Engineering from Ohio University. His research interests include various aspects of inertial navigation, GPS/INS integration, GPS transform domain receiver development, GPS carrier phase positioning, digital signal processing, and joint time-frequency data analysis.

Sanjeev Gunawardena is a Research Engineer with the Ohio University Avionics Engineering Center. His research interests include digital systems design, reconfigurable computing, and all aspects of real time GPS signal processing. Mr. Gunawardena received B.S. degrees in Engineering Physics and Electrical Engineering, and an M.S. in Electrical Engineering from Ohio University. He is currently pursuing a Ph.D. in Electrical Engineering.

Maarten Uijt de Haag is an Associate Professor of Electrical Engineering at Ohio University and a Principal Investigator with the Ohio University Avionics Engineering Center. He earned his Ph.D. from Ohio University and holds a B.S.E.E. and M.S.E.E. from Delft University of Technology, located in the Netherlands. He has been involved with GPS landing systems' research, advanced signal processing techniques for GPS receivers, GPS/INS integrated systems, and terrain-referenced navigation systems. The latter includes the development of terrain data base integrity monitors as an enabling technology for Synthetic Vision Systems and autonomous aircraft operations.

Michael S. Braasch is the Thomas Professor of Engineering in the School of Electrical Engineering and Computer Science at Ohio University. He has been performing navigation research with the Avionics Engineering Center at Ohio University since 1985 and has been conducting and directing GNSS software-defined receiver research since 1995. Mike is a licensed professional engineer in the State of Ohio, is an instrument-rated commercial pilot, and is the co-founder of GPSoft LLC.

ABSTRACT

Receiver design implementations are considered in terms of batch vs. sequential processing strategies. It is shown that the batch processing techniques improve performance characteristics of GNSS signal tracking for a number of application areas. Particularly, flight test results presented demonstrate that batch processing improves the GPS tracking margin by 8 dB as compared to sequential tracking for the deep GPS/INS integration mode that performs code and carrier phase tracking, and data bit recovery.

INTRODUCTION

Since the processing of digitized Global Navigation Satellite System (GNSS) signals has become feasible the number of terms used to describe different processing steps and different processing strategies has been increasing. Example terms include software-defined radio [1], software receiver [2], frequency-domain tracking [3], tracking loops [4], block processing [5], parallel FFT-based code search [6], parallel FFT-based frequency search [7], high sensitivity receivers [8], ultra-tight integration [9], deep integration [10], massive parallel correlator banks [11]. As the number of processing terms increases, their place in a framework for the GNSS receiver is becoming somewhat obscure. Hence, the first goal of this paper is to show the place of existing GNSS signal processing terms in the receiver design flow. This goal is achieved by representing the receiver design using sequential and batch processing approaches since the existing signal processing terms fit into sequential, batch or a combination of sequential and batch processing strategies. It is important to mention that sequential processing is well recognized while batch processing techniques remain underutilized despite the fact that for a number of practical applications the use of batch processing techniques efficiently overcomes limitations of the sequential processing approach. The second goal of this paper is therefore to demonstrate application areas for which batch processing is more powerful than a sequential approach. Example applications include tracking of signals with a low carrier-to-noise ratio (CNR) required to perform indoor navigation, navigation under foliage, and navigation in interference environments. The deep Global Positioning System (GPS)/Inertial Navigation System (INS) integration is used as a case study for these application examples.

Goals of this paper are thus formulated as follows:

- ❑ Show the place of existing processing terms in a framework for the GNSS receiver design.
- ❑ Provide new material that demonstrates efficient use of batch processing techniques for practical application areas.

The remainder of the paper is organized as follows. First, a simple filtering example is used to introduce basic concepts of sequential and batch processing. Basic sequential and batch processing concepts are then applied to consider generalized sequential and batch architectures of GNSS signal processing. The main result of this consideration is that the sequential and batch processing approaches have complementary features. Receiver design implementations that utilize combinations of sequential and batch processing techniques are therefore presented next. Finally, deep GPS/INS integration is used as a case study to demonstrate advantages of batch processing over sequential implementations.

INTRODUCTION TO SEQUENTIAL AND BATCH PROCESSING: FILTERING EXAMPLE

This section illustrates basic concepts of sequential and batch processing approaches using a filtering example. The example is formulated as follows: smooth 5 measurements given the measurement model defined by Equation (1):

$$\tilde{x}_k = a + n_k, k = 1,...,5 \qquad (1)$$

where:

$\tilde{x}_k, k = 1,...,5$ is the measurement;

$a = \text{const}$;

$n_k, k = 1,...,5$ is the measurement noise.

Figure 1 illustrates the estimation process of a sequential filter for the filtering example considered.

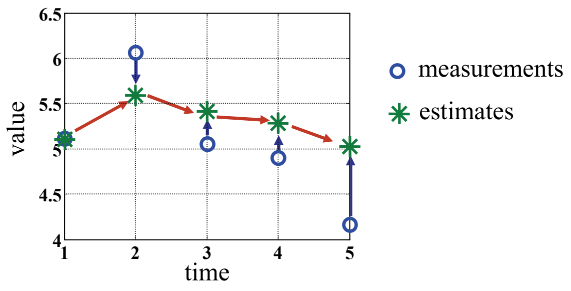

Figure 1. **Sequential filter estimation process**

The sequential filter computes its estimates on a sample by sample basis. An estimate is computed every time a new measurement arrives. Equation (2) is applied for the estimation computation:

$$\hat{x}_k = (1-\alpha_k)\cdot\hat{x}_{k-1} + \alpha_k\cdot\tilde{x}_k \qquad (2)$$
$$k = 1,...,5$$

where:

$\hat{x}_k, k = 1,...,5$ is the filter estimate;

$\alpha_k, k = 1,...,5$ is the estimation weight function, which is calculated using error minimization criteria such as e.g. the least mean square criterion.

The estimation process of a batch filter is shown in Figure 2.

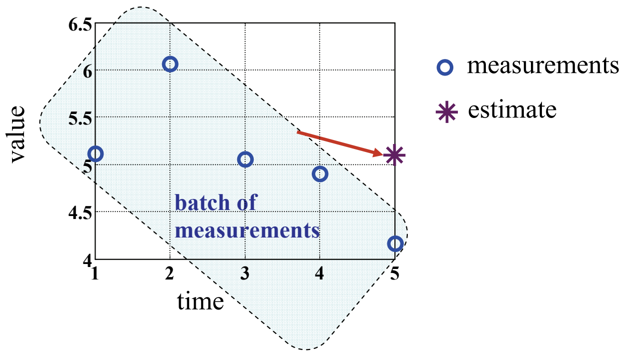

Figure 2. **Batch filter estimation process**

The batch filter waits until all five measurements are available and then performs all the computations at once by computing its only estimate based on the batch of five measurements:

$$\hat{x}_5 = \sum_{k=1}^{5} \alpha_k \cdot \tilde{x}_k \qquad (3)$$

Performance of the sequential and batch filters differs in two main aspects: computational expenses and signal observability. With respect to the computational expenses, the sequential filter requires less memory and needs fewer operations to compute an estimate as compared to the batch filter. On the observability aspect, the batch filter delivers better signal observability. A more detailed comparison of the sequential and batch filter implementations for the above example follows.

Sequential filter vs. batch filter: computational expenses

As can be inferred from Equation (1), the sequential filter needs to keep 3 elements in memory (a current measurement, a previous estimate, and a current value of the weight function) to compute an estimate. The batch filter requires 10 elements in memory (5 measurements and 5 values of the weight function) for the estimate computation formulated by Equation (3). The sequential filter therefore requires less memory.

The sequential filter needs 20 operations (2 additions and 2 multiplications per estimate, where the total number of estimates is 5) to compute the values of all estimates. However, these operations are spread over time and every time an estimate is computed only 4 operations are needed. The batch filter does all the computations at once (cannot spread computations over time) and needs 10 operations to compute an estimate (5 additions and 5 multiplications). Because the sequential filter is capable of spreading computations over time it requires fewer operations per estimate as compared to the batch filter.

Sequential filter vs. batch filter: signal observability

To obtain initial insight into the signal observability aspect, a smoothing example is given with measurement set that includes a noise spike. A corresponding measurement set is represented by Figure 3.

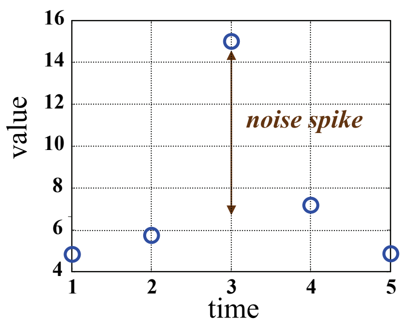

Figure 3. **Measurement set with a noise spike**

As stated above, the sequential filter performs estimations on a sample by sample basis. As a result, sequential filter observation is limited to a measurement with the noise spike and a previous estimate as illustrated by Figure 4. Limited observability of the sequential filter limits the filter's ability to identify the noise spike and adjust the estimation weight function accordingly. The batch filter observation ability is based on the entire data batch, which improves the noise spike visibility significantly as compared to the sequential filter case.

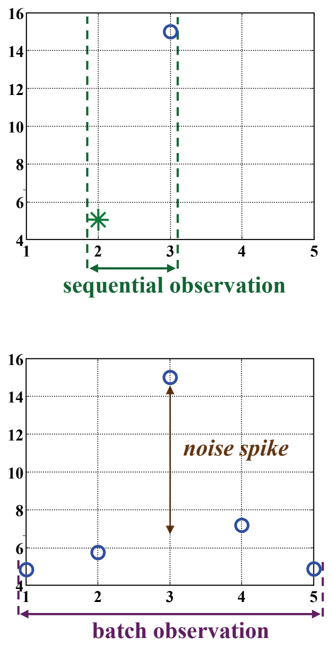

Figure 4. **Filter observations: sequential vs. batch**

This example illustrates the signal observability enhancement achieved through the use of batch processing.

SEQUENTIAL AND BATCH PROCESSING OF GNSS SIGNALS

The next step is to apply sequential and batch processing concepts discussed in the previous section to the processing of GNSS signals. It is important to point out that the consideration provided is independent of the receiver implementation platform (i.e. Field Programmable Gate Array (FPGA), Application Specific Integrated Circuit (ASIC), software receiver, software radio, Digital Signal Processor (DSP)). The choice of the implementation platform is application specific and generally depends on system requirements such as weight, power, cost, processing speed, and user friendliness. The use of a particular implementation platform, however, does not influence performance characteristics of GNSS signal processing. What does influence the signal processing performance is whether the sequential or batch processing approach is used.

Sequential Acquisition

Figure 5 shows a generalized structure of the sequential acquisition. The incoming signal code phase and carrier Doppler frequency shift are searched in sequential steps [4]. For each search step, the incoming signal is sequentially correlated with replica signals that are generated by Numerically Controlled Oscillators (NCOs) using values of the code phase and Doppler shift that correspond to the search step. The acquired code phase and carrier Doppler frequency shift are the results of the sequential acquisition search.

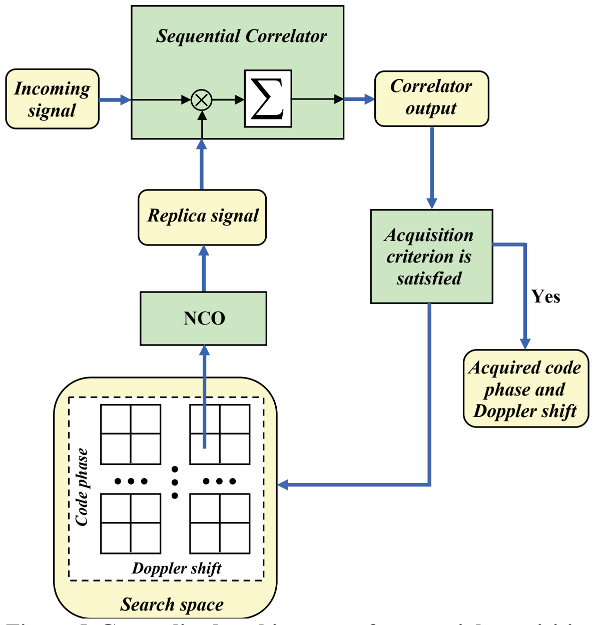

***Figure 5.* Generalized architecture of sequential acquisition**

Sequential Tracking

The acquired code phase and carrier Doppler shift are used to initialize sequential tracking loops. Figure 6 shows a generalized architecture of the sequential tracking stage.

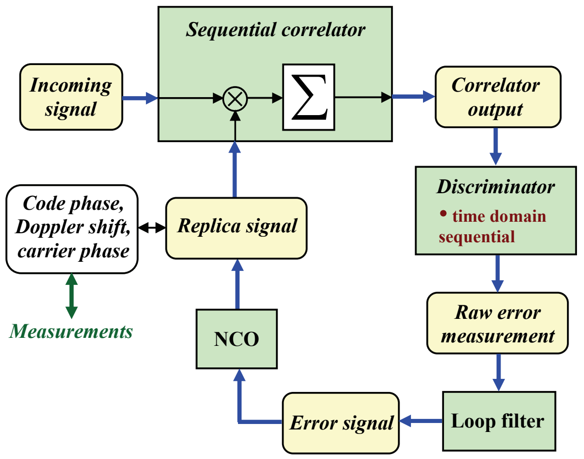

***Figure 6.* Generalized architecture of sequential tracking**

The sequential tracking stage performs sequential correlation of the incoming and replica signals. Correlator outputs are used to compute a raw error signal, which is first smoothed by a loop filter and then used to adjust the parameters of the NCOs that generate replica signals. The replica code phase, carrier Doppler shift and carrier phase serve as the actual tracking loop measurements. As depicted by Figure 6, the sequential tracking approach relies on a closed loop tracking architecture.

Batch Processing

Figure 7 shows a generalized architecture of batch processing.

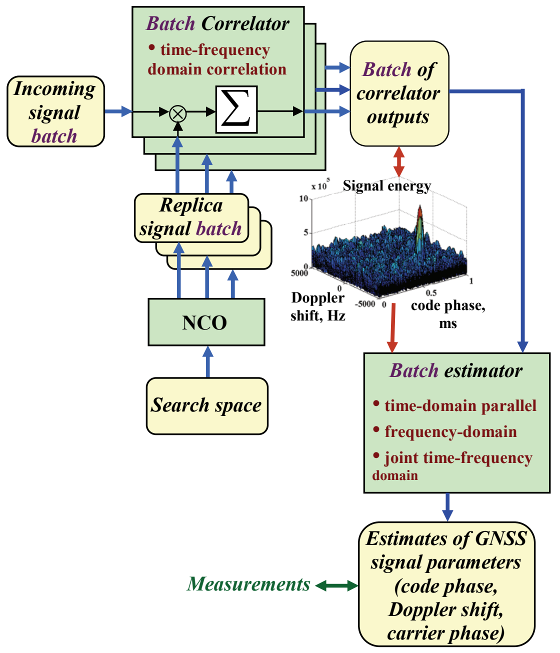

***Figure 7.* Generalized architecture of batch processing**

The batch processing approach does not separate acquisition and tracking stages. A batch processor obtains the entire image of the signal by acquiring the signal for every data batch. To do so, the input signal batch is correlated with batches of replica signals that are constructed based on the code and carrier search space. The correlation is a batch-based correlation that uses joint time-frequency domain techniques to allow parallel correlation computations. E.g. the parallel computation of correlation coefficients for all code shifts from the code search space using a direct and inverse Fast Fourier Transform (FFT) [6]. A result of the batch-based correlation is a 3 dimensional (3D) signal image for the current data batch where the image dimensions are code shift, Doppler shift, and signal energy as shown in Figure 7. A batch estimator computes estimates of signal parameters based on a batch of correlator outputs employing parallel estimation techniques to improve the signal observability. Contrary to the sequential closed loop tracking, batch processing maintains an open loop tracking architecture.

Measurement Exploitation for the Batch Processing Approach

To obtain the entire image of the signal for the current data batch, one of the following search options can be used:

- ❑ Full search for which there is no measurement feedback.
- ❑ Local search where signal parameter estimates from previous batches are used to reduce the search space for the current data batch.

It is important to note that measurements of signal parameters that correspond to different data batches are independent for both the full search and the local search. Different data batches are processed independently from each other thus providing independent measurements if the full search option is chosen. For the local search case, previous estimates are applied to reduce the search space for the current data batch. The estimation of signal parameters for the current data batch is, however, statistically independent from previous estimates. Figure 8 illustrates the use of previous estimates for a code phase estimation example.

A full code search is performed for the first data batch. The first estimate of the code phase is then applied to define the local search space for the second data batch. Thus only the first estimate herein defines the local search space. The code phase estimate for the second data batch is statistically independent from the first estimate. Batch processing thus provides independent measurements at different data batches regardless of the search option used.

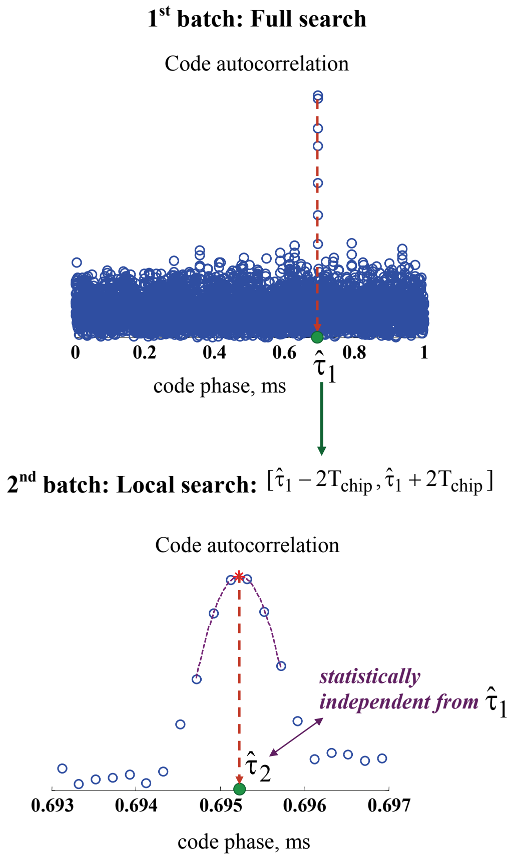

***Figure 8.* Example of the use of previous estimates for the batch processing local search**

FEATURES OF BATCH AND SEQUENTIAL PROCESSING

Batch Processing Features

Main features of the batch processing approach are summarized as follows:

- ❑ Improved signal observability as compared to sequential processing;
- ❑ Capability of parallel computations;
- ❑ Improved tracking robustness as compared to closed loop sequential tracking.

First, examining a batch of data improves the signal observability as compared to signal processing on a sample by sample basis. More specifically, the signal observability is improved by applying parallel estimation techniques such as parallel time domain, frequency-domain, and joint time frequency-domain estimation computations. E.g., a Short-Time Fourier Transform (STFT) can be applied for the estimation of the carrier frequency in high-dynamic environments. In a high dynamic environment at nominal GPS signal strength a full search is performed on a short sample set (approx. 1 ms) to find an initial estimate of the frequency and code-phase. These estimates are then used to refine the code-rate estimate from a long sample set (approx. 1 second) using a non-linear (median) filter and a least squares estimator. This same sample set is used to estimate the precise frequency as a function of time by applying the STFT to the raw signal for the local search window. Figure 9 shows an example result of this operation for an 8 g acceleration along the line-of-sight.

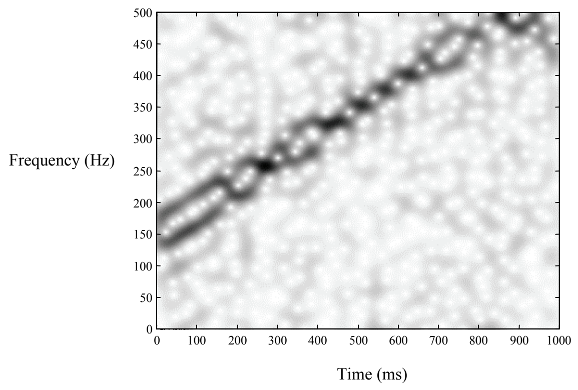

***Figure 9*. Time-Frequency tiling for a GPS signal with a 400Hz/second frequency change.**

Second, the use of batch processing allows parallel computations. Particularly, parallel correlation computations can be performed by using frequency-domain correlation techniques. E.g. computation of correlation coefficients for all Doppler shifts from the Doppler shift search space can be performed in parallel using a FFT [7]. Parallel correlation computation for the entire GPS Coarse Acquisition (CA) code search space using direct and inverse FFT [6] serves as another example.

Third, batch processing uses an open loop tracking architecture thus providing more robust tracking as compared to the sequential closed loop tracking approach. Particularly, batch processing computes the entire image of the signal, which allows immediate tracking recovery after a temporary signal loss: the signal is “seen” immediately after it “reappears” following a temporary signal loss. In addition, a batch processor does not have to follow the motion dynamic within the bandwidth of a tracking loop filter. Instead, the motion dynamic needs to be followed within a correlator bandwidth, which is inversely proportional to a correlation interval and is generally significantly wider than a loop filter bandwidth. As a result, batch processing is less sensitive to high dynamic stress conditions.

Why Still Use Sequential Processing?

Having the above powerful features of batch processing why would one still need to use the sequential processing approach? The answer is given by the computational efficiency of sequential processing. Sequential processing allows minimization of memory and computational resources. Contrary to batch processing, it does not require additional computational resources to increase resolution of signal parameters. For instance, in order to decrease the correlator spacing for the GPS CA code from 1 chip to 0.25 chip, a batch processor that performs an FFT-based parallel code search needs to substitute a 1024 point FFT by a 4096 point FFT. In terms of computational resources involved, this substitution is equivalent to using 320 instead of 100 sequential correlators if a floating point FFT is performed. Sequential processing can rely on only 3 correlators (early, prompt, and late) for both the 1 chip and 0.25 chip cases by adjusting the correlator spacing accordingly. As a result, sequential processing is computationally cheaper for high-accuracy, wide-bandwidth tracking.

RECEIVER DESIGN UTILIZING A COMBINATION OF BATCH AND SEQUENTIAL PROCESSING

The above comparison of batch and sequential processing shows that these processing techniques have complementary features. A receiver can therefore be designed utilizing a combination of the batch and sequential processing approaches. Based on the generalized sequential and batch processing architectures discussed earlier, key components of GNSS signal processing include:

- Correlation;
- Measurement computation;
- Measurement exploitation.

Table 1 summarizes realizations of these key components with batch and sequential processing techniques. Correspondingly, the correlation component for a combined batch/sequential correlation applies batch correlation to obtain a coarse image of the entire signal for every data batch (coarse signal zoom), while sequential correlation is used to achieve a fine signal zoom in a computationally efficient manner. The measurement computation component can combine batch-based parallel estimation techniques with time-domain sequential measurement computations. For instance, code phase estimation is significantly less sensitive to high dynamic conditions than estimation of the Doppler shift. A STFT can thus be applied to improve dynamic performance of a carrier estimator while for a code phase estimator it is sufficient to use a sequential discriminator (e.g. early minus later discriminator).

Table 1. **Realization of key processing components using sequential and batch approaches**

	Sequential	Batch	Combination of batch and sequential
Correlation	• Sequential correlation (time-domain)	• Batch parallel correlation (joint time-frequency domain)	• Batch-based parallel correlation for coarse signal zoom combined with sequential correlation for fine signal zoom.
Measurement computation	• Time-domain sequential	• Time-domain parallel; • Frequency -domain; • Joint time frequency domain.	• Combination of time-domain sequential estimation of signal parameters and batch-based parallel estimation of signal parameters. E.g.: time-frequency-domain estimation of the Doppler shift with sequential estimation of the code phase
Measurement exploitation	• Measurement feedback via a closed tracking loop	• No feedback; • Define local search space	• Batch-based measurements aid sequential correlators via: *a) multiple passes through the same data batch* or *b) feed-forward aiding*.

The measurement exploitation component employs results of the batch-based correlation that provide a coarse signal localization, to aid sequential correlators. Here, sequential correlation can be either applied to the same batch of data (multiple passes through the same data batch) or the results of the batch-based correlation for the current data batch can be used to set up sequential correlators for the next batch of data (feed-forward aiding).

Figure 10 exemplifies the use of combined batch and sequential processing for tracking of the GPS Link 1 (L1) CA code signal. The combined batch/sequential architecture shown herein does not take into account the presence of navigation data bits. Consideration of the data-less case allows for a demonstration of the main features of combined batch/sequential tracking without obscuring this conceptual illustration by additional details. The use of combined batch/sequential tracking can be extended for a more general case where navigation data bits are taken into account.

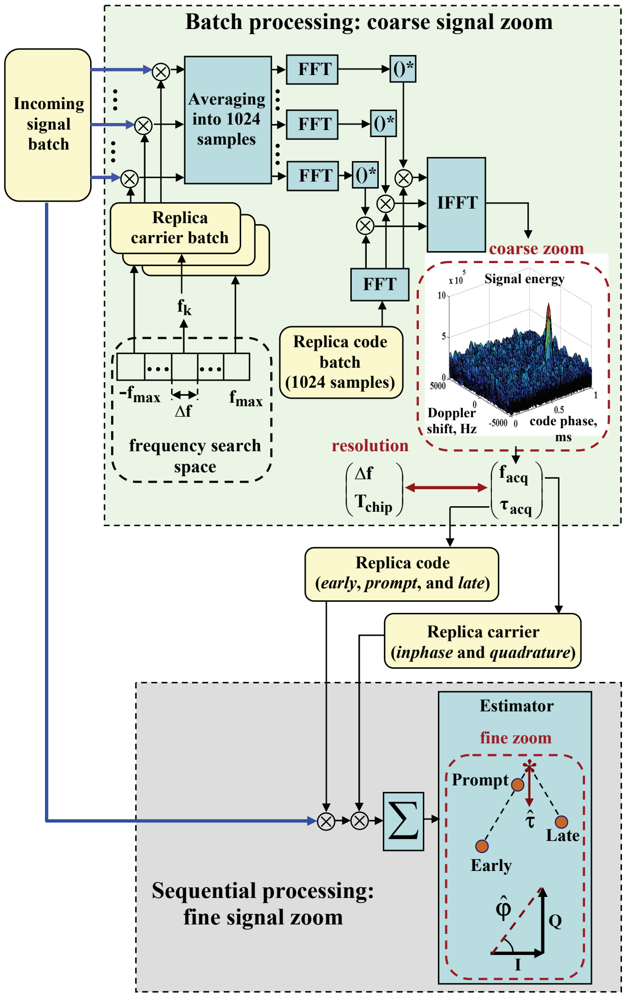

Figure 10. **Combined use of sequential and batch processing for GPS L1 CA code and carrier phase tracking**

As represented by Figure 10, batch processing is used for the coarse signal localization. The frequency is localized within a bin of a frequency search space via a time-domain frequency search. The code phase is localized within a CA code chip through a parallel time-frequency domain code search that is based on 1024-point direct and inverse FFTs. Note that for those cases where the batch duration exceeds 1 ms (CA code period) the parallel code search requires the use of batch addition techniques (as described e.g. in [5]). Results of the coarse code and carrier localization (τ_{acq} and f_{acq} , correspondingly) are handled to sequential correlators that perform a fine signal zoom by using early, prompt, and late correlation to achieve a fine code zoom and in-phase and quadrature correlation for a fine carrier zoom.

Early (E), prompt (P), and late (L) replica code samples are constructed as follows for the sequential correlation:

$$PRN_{E,P,L}(t_n) = PRN\left(t_n - \tau_{acq} - \tau_{E,P,L}\right) \qquad (4)$$

where:

PRN is the CA-code pseudo-random noise for the satellite being tracked;

$\Delta\tau_E = -\Delta\tau, \Delta\tau_P = 0, \Delta\tau_L = \Delta\tau$; and $\Delta\tau$ is the CA code correlator spacing;

$t_n = n \cdot \Delta t$; n is the number of a signal samples within the current data batch; and Δt is the time interval between GPS signal samples.

Equation (5) is applied to compute samples of inphase replica carrier (SIN) and quadrature replica carrier (COS):

$$\begin{aligned} SIN(t_n) &= \sin\left(2\pi \cdot (f_{IF} + f_{acq}) \cdot t_n + \hat{\varphi}_{frac-1}\right) \\ COS(t_n) &= \cos\left(2\pi \cdot (f_{IF} + f_{acq}) \cdot t_n + \hat{\varphi}_{frac-1}\right) \end{aligned} \quad (5)$$

where:

f_{IF} is the down-conversion frequency;

$\hat{\varphi}_{frac-1}$ is the fractional carrier phase estimated for the previous data batch that is used herein to maintain consistency of carrier phase tracking.

Signal accumulation over the duration of a data batch is performed as formulated by Equation (6):

$$\begin{aligned} I_{E,P,L} &= \sum_{n=1}^{N_{batch}} s(t_n) \cdot SIN(t_n) \cdot PRN_{E,P,L}(t_n) \\ Q_{E,P,L} &= \sum_{n=1}^{N_{batch}} s(t_n) \cdot COS(t_n) \cdot PRN_{E,P,L}(t_n) \end{aligned} \quad (6)$$

where:

s is a down-sampled incoming GPS signal;

N_{batch} is the total number of signal samples within the data batch.

Equation (7) is applied to estimate the code phase ($\hat{\tau}$, fine code zoom):

$$\begin{aligned} \hat{\tau} &= \tau_{acq} + \frac{R_L - R_E}{R_L + R_E} \cdot T_{chip} \\ R_{E,L} &= \sqrt{I_{E,L}^2 + Q_{E,L}^2} \end{aligned} \quad (7)$$

where

T_{chip} is the CA code ship length.

The accumulated Doppler for the current data batch ($\hat{\varphi}_{acm}$, fine carrier zoom) is estimated as follows. A temporary carrier phase value ($\hat{\varphi}_{temp}$) is first computed:

$$\begin{aligned} \delta\hat{\varphi} &= \arctan\left(Q_P, I_P\right) \\ \hat{\varphi}_{temp} &= \hat{\varphi}_{frac-1} + \delta\hat{\varphi} + 2\pi \cdot f_{acq} \cdot N_{acm} \cdot \Delta t \end{aligned} \quad (8)$$

where:

$\arctan\left(Q_P, I_P\right)$ is a 4-quadrant arctangent function illustrated by Figure 11.

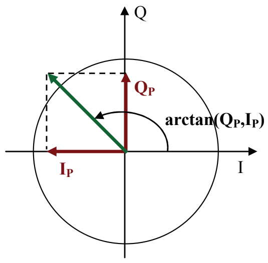

***Figure 11.* 4-quadrant arctangent function**

The fractional carrier phase estimate ($\hat{\varphi}_{frac}$) and the integer number of accumulated carrier cycles (N_φ) are then updated:

$$\begin{aligned} \hat{\varphi}_{frac} &= \text{fractional part of } \hat{\varphi}_{temp} \\ N_\varphi &= N_{\varphi_{-1}} + \text{whole part of } \hat{\varphi}_{temp} \end{aligned} \quad (9)$$

Finally, the accumulated Doppler estimate is computed as follows:

$$\hat{\varphi}_{acm} = 2\pi \cdot N_\varphi + \hat{\varphi}_{frac} \quad (10)$$

CASE STUDY: DEEP GPS/INS INTEGRATION

This section uses deep GPS/INS integration [12] as a case study to demonstrate advantages of using batch processing techniques over implementations that are restricted to sequential tracking. The deep integration combines GPS and inertial data at the earliest data processing stage possible by integrating GPS signal samples with measurements from an Inertial Measurement Unit (IMU). The goal of the deep integration is to significantly increase the signal integration time as compared to an unaided GPS receiver. It is important to mention that implementation of a deeply integrated GPS/INS system is primarily defined by a deep integration mode chosen. There are four integration modes possible:

- Deep integration for the code phase tracking only;
- Deep integration for the code phase and carrier frequency tracking with known navigation data bits;
- Deep integration for the code phase and carrier phase tracking with known navigation data bits;
- Deep integration for the code phase and carrier phase tracking with navigation data bit recovery.

The above modes are listed in ascending order of the implementation complexity. Particularly, the deep integration for the code phase tracking only is the least complex of the possible integration modes. This deep integration mode generally relies on non-coherent signal integration (for integration intervals that exceed 20 ms) [13] and does not require the use of inertial measurements unless severe dynamic conditions are involved (motion acceleration on the order of 1000 g and higher, where g is the acceleration due to gravity). Deep integration for the code phase and carrier phase tracking, and the data bit recovery is the most challenging integration mode. Coherent signal integration and a 1-cm/s accurate inertial aiding are required to operate in this integration mode [10].

The reminder of this section illustrates the use of sequential and batch processing techniques for the deep GPS/INS integration for the CA code phase and carrier phase tracking with and without data bit recovery. The deep integration is performed herein to improve the GPS tracking margin by 17 dB as compared to a 32 dB-Hz tracking threshold of an unaided GPS receiver.

Sequential Deep Integration

Figure 12 shows a deep GPS/INS integration scheme that relies on sequential tracking.

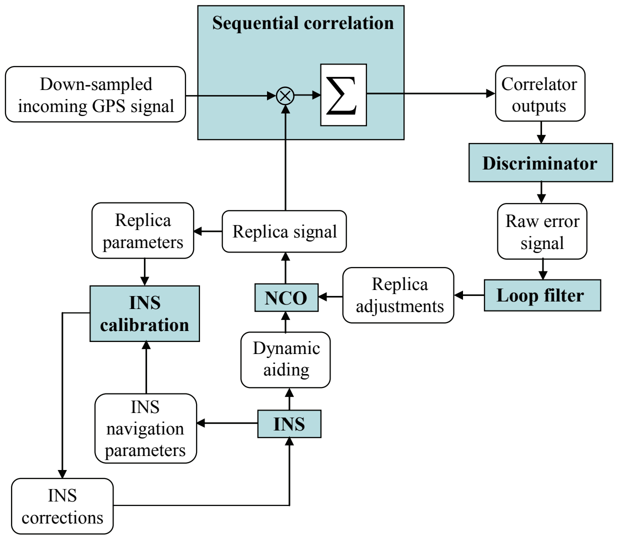

Figure 12. **Deep GPS/INS integration utilizing the sequential tracking approach**

Accordingly, inertialy aided signal integration is comprised of the correlation integration followed by the loop filter averaging. The correlation step needs to pull the signal out of the noise floor with the additional smoothing performed by the loop filter. A 0.1-s correlation interval is required to achieve a 17 dB increase in the GPS tracking margin. This correlation interval exceeds a 20-ms duration of navigation data bits, which makes impossible the data bit recovery for the sequential approach. The loop filter needs to average correlator outputs over a 1 s interval.

Limitations of the sequential approach for the deep integration are primarily imposed by extremely tight pull-in ranges of sequential tracking loops. Figure 13 illustrates pull-in ranges of the sequential deep integration.

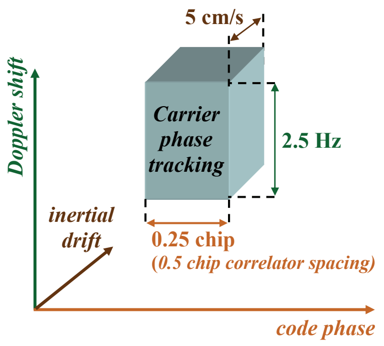

Figure 13. **Pull-in ranges of sequential tracking loops for deep GPS/INS integration**

A pull-in range of the code phase tracking loop is determined by the code correlator spacing. For instance, the pull-in range is 0.25 chip for the case where a 0.5 chip code correlator spacing is used. A pull-in range of the carrier phase tracking loop is inversely proportional to the correlation interval and is equal to 2.5 Hz for a 0.1-s correlation. The inertial aiding accuracy within 5 cm/s is required to aid the carrier phase loop filter over 1-s averaging intervals.

As stated previously, extremely tight pull-in ranges of sequential tracking loops significantly limit capabilities of sequential tracking. Figure 14 shows a tracking state graph for the deep GPS/INS integration based on sequential tracking.

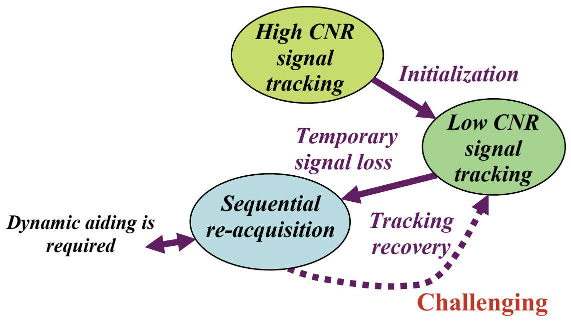

Figure 14. **Tracking state graph for the deep GPS/INS integration utilizing sequential tracking**

A high CNR signal is required for the system initialization in order to limit uncertainties of signal parameters to pull-in ranges of sequential tracking loops. Once in the low CNR tracking state, if the signal is temporary lost due to e.g. a CNR drop, the tracking status can be recovered through a dynamically aided sequential re-acquisition only. The sequential re-acquisition needs to get the signal back into the tight pull-in ranges of sequential tracking loops, which is challenging. E.g., code phase can jump over a few code chips due to switching between a multipath and a direct signal for indoor tracking scenarios. As a result, sequential search can miss the signal. Another example is unreliability of sequential search for those cases where low-cost inertial measurements are used and a significant carrier frequency drift (on the order of 2 Hz/s if GPS updates are unavailable for more than 10 s) is thus present.

Batch Deep Integration

Figure 15 represents a deeply integrated GPS/INS system that is based on the batch tracking approach.

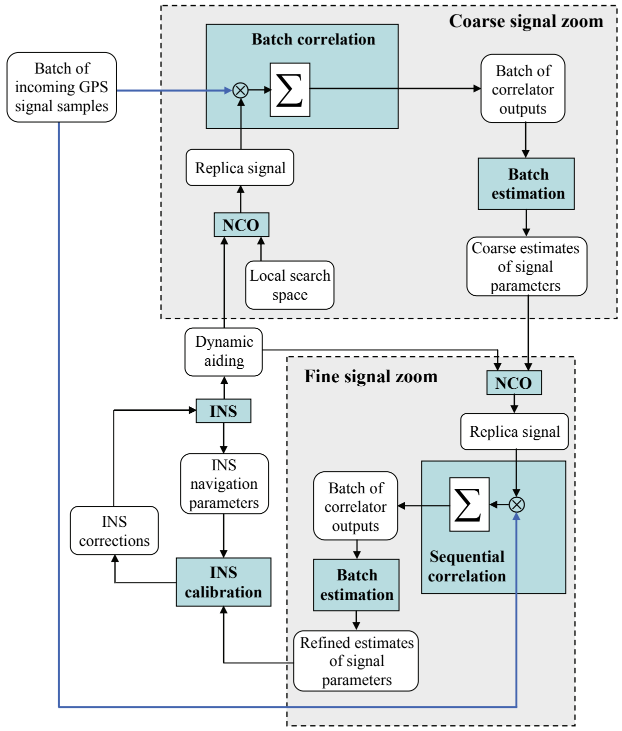

Figure 15. **Deep GPS/INS integration utilizing the batch tracking approach**

The signal is integrated over 1 s using inertialy-aided correlators. No knowledge of navigation data bits is required if the signal integration is augmented by a batch-based bit estimation algorithm, which employs a time-domain parallel search for the bit combination that maximizes the signal energy over the integration interval [10]. Tracking pull-in ranges are defined by the local search space as illustrated in Figure 16. The full code search is performed, so no a priori knowledge of the code phase is therefore required. Frequency search space can be in the range from 10 Hz (which is currently feasible in real-time for satellite constellations that include 6 satellites or fewer as it can be inferred from real-time performance figures given in [14]) to the full frequency search within a 10-kHz search space.

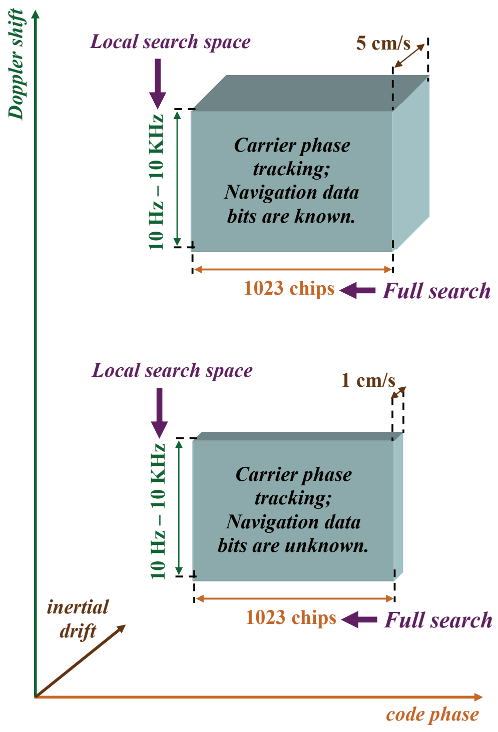

Figure 16. **Pull-in ranges for the batch-based deep GPS/INS integration**

Inertial drift needs to be maintained within a 5 cm/s boundary to aid 1-s signal integration for the carrier phase tracking for the case where navigation data bits are known. If the navigation data bits are unknown and the batch-based deep integration performs the data bit recovery, the maximum allowable inertial drift is reduced to a 1 cm/s level in order to maintain sign consistency of bit sequences estimated by the batch-based algorithm introduced in [10].

Figure 17 shows a tracking state graph for the batch deep integration.

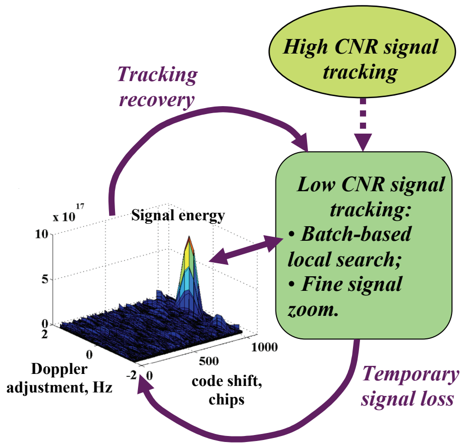

Figure 17. **Tracking state graph for the deep GPS/INS integration utilizing batch tracking**

A high CNR signal can be used for the system initialization in order to reduce the size of the local search space. The use of a high CNR signal is, however, not required. The batch processing based deeply integrated GPS/INS can start in the low CNR tracking state if a sufficient size of the local search space is used. Since the batch processor always computes the entire signal image, the tracking status can be immediately recovered after a temporary signal loss as illustrated in Figure 17.

Flight Test Results: Sequential vs. Batch Deep Integration

Flight test results are presented below to demonstrate benefits of using batch processing techniques for the deep GPS/INS integration. The code and carrier phase tracking with navigation data bit recovery is implemented for the deep integration mode. A low-cost AGNC coremicro® IMU [15] provides inertial measurements. Figure 18 shows a flight test trajectory that incorporates significant motion dynamic including a 90-deg turn.

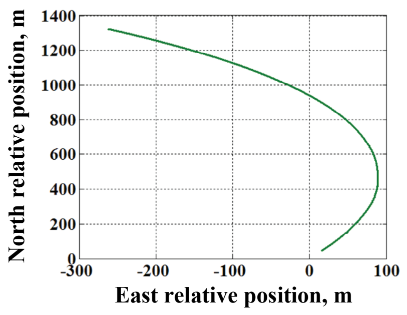

Figure 18. **Flight test trajectory**

Figure 19 through Figure 21 exemplify flight test results for the sequential and batch deep integration approaches.

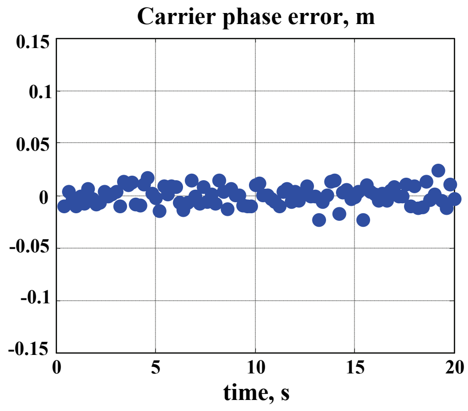

***Figure 19*. Sequential tracking: Carrier phase tracking performance for SV 30 (CNR range is [23, 26] dB-Hz)**

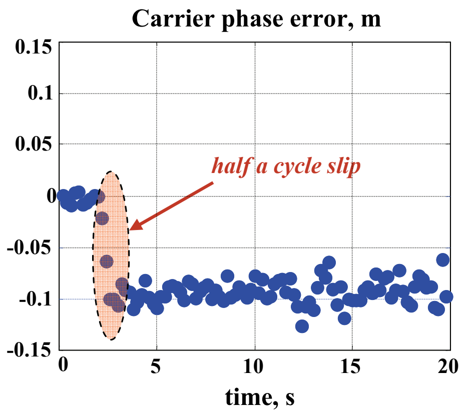

***Figure 20*. Sequential tracking: Carrier phase tracking performance for SV 30 (CNR range is [21, 24] dB-Hz)**

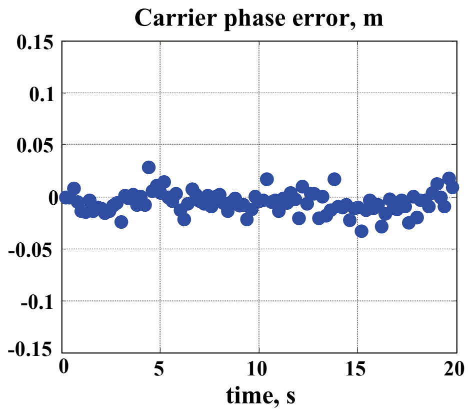

***Figure 21*. Batch tracking: Carrier phase tracking performance for SV 30 (CNR range is [15, 18] dB-Hz)**

The results presented demonstrate that the sequential tracking approach maintains continuous carrier phase tracking for the case where the CNR stays in the range from 23 to 26 dB-Hz. Sequential tracking is however not reliable if the CNR is reduced to the 21 dB-Hz level: half a cycle slips is present in the carrier phase error plot represented by Figure 20. Batch processing maintains consistent carrier phase tracking without the knowledge of navigation data bits for the CNR in the range from 15 to 18 dB-Hz as shown in Figure 21. Flight test results shown thus demonstrate that batch processing improves the tracking margin of deep integration by 8 dB as compared to sequential processing.

CONCLUSIONS

This paper demonstrates that in order to optimize performance characteristics of GNSS signal processing the receiver design needs to be considered not in terms of its implementation platform (i.e. FPGA, ASIC, software receiver, software radio, DSP) but in terms of batch vs. sequential processing strategies. Advantages of signal processing architectures that utilize batch techniques over receiver design implementations restricted to sequential tracking include improved signal observability and tracking robustness. For practical application areas, advantages of batch processing are shown using deep GPS/INS integration as a case study. Particularly, flight test results presented demonstrate that batch processing increases the GPS tracking margin by 8 dB as compared to the sequential processing approach.

ACKNOWLEDGEMENTS

The research reported in this paper was supported, in part, by Federal Aviation Administration Aviation Research Cooperative Agreement 98-G-002 (Local Area Augmentation System Research and Development).

REFERENCES

[1] Akos, D. M., *A Software Radio Approach to Global Navigation Satellite System Receiver Design*, Ph.D. Dissertation, Ohio University, August 1997.

[2] MacGougan, G., Normark, P.L., and Stahlberg, C., “Satellite Navigation Evolution: The Software GNSS Receiver,” GPS World, Vol. 15, No. 1, January 2005.

[3] Yang, C., “Tracking of GPS Code Phase and Carrier Frequency in the Frequency Domain,” Proceedings of the ION GPS-2003, September 2003.

[4] Kaplan, E.D., Editor, *Understanding GPS – Principles and Applications*, Artech House Publishers, Boston, 1996.

[5] Uijt de Haag, M., *An Investigation into the Application of Block Processing Techniques for the Global Positioning System*, Ph.D. Dissertation, Ohio University, August 1999.

[6] Coenen, A. J. R. M., and van Nee D. J. R., “Novel Fast GPS/GLONASS Code-Acquisition Technique Using Low Update-Rate FFT,” Electronic Letters, Vol. 28, No. 9, April 1992.

[7] Cheng, U., and Hurd, W. J., and Statman, J. I., “Spread spectrum code acquisition in the presence of Doppler shift and data modulation,” IEEE Transactions on Communications, Vol. 38, February 1990.

[8] MacGougan, G., Lachapelle, G., Klukas, R., Siu, K., Garin, L., Shewfelt, J., and Cox, G. “Degraded GPS Signal Measurements With A Stand-Alone High Sensitivity Receiver,” Proceedings of the Satellite Division of the Institute of Navigation National Technical Meeting, January 2002.

[9] Alban, S., Akos, D. M., and Rock S. M. “Performance Analysis and Architectures for INS-Aided GPS Tracking Loops,” Proceedings of the Satellite Division of the Institute of Navigation National Technical Meeting, January 2003.

[10] Soloviev, A., Gunawardena, S., and van Graas, F. “Deeply Integrated GPS/Low-Cost IMU for Low CNR Signal Processing: Flight Test Results and Real Time Implementation,” Proceedings of the ION GNSS-2004, September 2004.

[11] van Diggelen, F., “Indoor GPS,” ION GPS-2001 Tutorial, September 2001.

[12] Phillips, R. E., Schmidt G. T, “GPS/INS Integration,” AGARD Lecture Series on Systems Implications and Innovative Applications of Satellite Navigation, Paris, France, July 1996.

[13] Gustafson, D., Dowdle, J., and Flueckiger, K., “A High Anti-Jam GPS-Based Navigator,” Proceedings of the Satellite Division of the Institute of Navigation National Technical Meeting, January 2000.

[14] Gunawardena, S., Soloviev, A., and van Graas, F. “Real Time Implementation of Deeply Integrated Software GPS receiver and Low Cost IMU for Processing Low-CNR GPS Signals,” Proceedings of the 60th Annual Technical Meeting of the Institute of Navigation, June 2004.

[15] American GNC coremicro® IMU specifications, available at www.americangnc.com.