Received: 7 September 2020 | Revised: 16 December 2020

DOI: 10.1002/navi.412

ORIGINAL ARTICLE

WILEY ION

Deriving Accurate Time from Assisted GNSS Using Extended Ambiguity Resolution

Ryan Blay[1] | **Boyi Wang***[1,2] | **Dennis M. Akos**[1]

[1] Department of Aerospace Engineering Sciences, University of Colorado Boulder, Colorado, USA

[2] School of Electronic Information and Communications, Huazhong University of Science and Technology, Hubei, China

Correspondence:
Boyi Wang, Huazhong University of Science and Technology, 1037 Luoyu Rd., Wuhan, Hubei, China, 430074.
Email: boyiwang@hust.edu.cn

Abstract

The advent of assisted GPS/GNSS can reduce the time to first fix (TTFF) dramatically and still provide sufficient positioning accuracy. However, many applications utilize GNSS for timing and assisted GPS/GNSS does not always provide the accurate nanosecond synchronization. Accurate timing is challenging for assisted GNSS due to the lack of diversity in the solution over short time periods. The relatively short ambiguities of the GPS L1 C/A code signal, one millisecond and twenty milliseconds, are too coarse to resolve accurate time reliably. In this paper, the impact of different ambiguity times has been derived/computed and validated. Experimental results show that when the ambiguity time is 200 milliseconds or longer, the least squares process will reliably converge and the typical nanosecond-level timing can be achieved. An example shows how the GPS L1 C/A data parity word (provided every 600 milliseconds) can be used, demonstrating nanosecond-level timing from assisted GPS.

1 | INTRODUCTION

The Global Navigation Satellite System (GNSS) is widely used for positioning and timing services. However, the time to get the first positioning solution, i.e., the time to first fix (TTFF), could be relatively long due to low-rate ephemeris modulated on GNSS signals. To solve this problem, the assisted GNSS (A-GNSS) technique has been introduced. The basic idea of A-GNSS is: instead of receiving 40 seconds of data from satellites and decoding ephemeris information, it obtains ephemeris from external sources such as an internet server or it applies extended long-term ephemeris techniques. Then, the A-GNSS technique only needs to receive millisecond-level data to acquire the code phase offset to provide decent positioning services (Van Diggelen, 2009).

Several existing A-GNSS techniques have been proposed, and van Diggelen's technique proposed in van Diggelen and Abraham (2007) and Van Diggelen (2009) is the one that is representative and is being widely used. As far as timing goes, most A-GNSS techniques aim to achieve a coarse time estimate within 10 milliseconds until their receiver is able to synchronize with the navigation bits. Muthuraman et al. (2012) review two different time-recovery algorithms that achieve the 10 milliseconds accuracy under nominal signal conditions but with weak signal conditions (15 dB-Hz) could not achieve the same accuracy.

Besides, there are other papers that describe different approaches taken with A-GNSS. Weng et al. (2011) propose a method of self-generated assistance data for A-GNSS. Dovis et al. (2008) discuss the acquisition issue of indoor A-GPS. Huang and Akopian (2013) discuss the problem of A-GPS assistance network delay. Otaegui et al. (2006) propose a hybrid architecture of standalone and A-GNSS receiver. Cheong et al. (2014) propose a snapshot receiver with collective detection using MS-based A-GPS techniques. Syrjarinne (2001) explores the assistance that a communications network can provide to a GPS system. Li (2010, 2012) explores the positioning bias and how to eliminate it. Huang et al. (2014, 2015, 2016) measure and model network delays for MS-based assistance delivery. Bissig et al. (2017) detail an efficient collective detection (CD)

method to obtain a position fix using one millisecond of data. Vallina-Rodriguez et al. (2013) characterize the inefficiencies and costs in relying on network assistance for several A-GPS mobile devices. Sirola (2001) and Sirola and Syrjarinne (2002) propose a range-fitting cost function over the space of position and coarse time using an iterative approach. Akopian and Syrjarinne (2002) and Syrjarinne (2000) also formulate a time-recovery method but use optimization and extended Kalman filters. Glennon and Bryant (2005) propose two different approaches for accurate time recovery with a focus on the single-stage algorithm. However, the original and all these alternate methods are focused on a position solution and do not provide nanosecond-level time accuracy in all cases.

The motivation for this paper stems from a desire to obtain not just fine-time accuracy (less than 10 milliseconds error), but nanosecond-level accuracy in as little time as possible. In conventional A-GNSS techniques, achieving this accuracy is assumed to be solved when the navigation data bits transitions are known. Several papers discuss the algorithm to synchronize with the navigation bits. For example, the study proposed in Akopian and Syrjarinne (2009) uses navigation bit alignment for time recovery. Also, Agarwal et al. (2002) in their review approach the problem of time-stamp recovery in which they discuss several different approaches including the navigation-bit alignment. However, the papers highlight that without being able to synchronize with the navigation bits, the timing error could be incorrect by multiples of 20 milliseconds.

There has not been a discussion of what the alternative is when the data bit transitions cannot be synchronized accurately. Unlike some A-GNSS research that focuses on implementing external data assistance or investigating acquisition algorithms, this paper mainly focuses on deriving accurate time using A-GNSS. The contributions can be summarized as follows: (1) It provides the generalized methodology to use any ambiguity time in the attempt to achieve nanosecond-level accuracy. (2) It provides results that show the relationship between ambiguity time and likelihood of nanosecond-level accuracy. (3) It provides a method of finding an ambiguity time of 600 milliseconds by leveraging the GPS L1 C/A parity bits.

This problem will be defined throughout this paper as ambiguity resolution, i.e., the ability to correctly identify the ambiguity. For the GPS L1 C/A code, there is an obvious 1 millisecond resolution through the code period which is sufficient, with a series of constraints, to obtain a GNSS position solution. Again, for the GPS L1 C/A code, another level of resolution, 20 milliseconds, can be obtained through the data bit synchronization. This level improves the ability to resolve accurate time but is not always successful. It is recognized that the assistance server or external sources may be able to provide a broader time reference but for many field operations that level is nonexistent or too coarse. In GNSS signals, not all ambiguity times exist, but this paper will simulate the results as if several nonexistent ambiguity times did exist. The algorithm given will be general such that any ambiguity time can be used.

In this paper, the A-GNSS approach as described by van Diggelen will be reviewed. Next, the algorithm to investigate the ambiguity resolution impact will be proposed. Then, experiments will be performed to find the necessary ambiguity time for nanosecond-level accuracy in a majority of situations. Finally, an example of using a parity check algorithm to resolve ambiguity time for Global Positioning System (GPS) L1 CA signal with only the provided IF data stream will be presented that is sufficient for 99% of situations.

2 | REVIEW OF A-GNSS RECEIVER

Most often, the goal of A-GNSS receivers is to achieve faster time-to-first-fix (TTFF) and greater availability. Conventional GNSS receivers need to receive the data from the first three subframes of the 30 second navigation frame to solve for time and position. For the GPS L1 C/A signal, it can take as long as 36 seconds to receive these critical elements of the navigation data, which can be a significant delay for time-to-first-fix. With the A-GNSS technique, receivers can solve for position with only milliseconds of IF data, receiving the navigation data message via an alternate source. This means that the receiver can provide position without waiting to receive and decode the whole frame. The contents of the navigation data for the satellites, the precise orbital and clock parameters, also known as the assistance data, are provided via an alternate channel. For the mobile phone, this data is typically provided through the cellular network using the Secure User Plane Location (SUPL) protocol (Open Mobile Alliance, 2012). For other applications, this data can be provided by the Networked Transport of RTCM via Internet Protocol (NTrip) servers available through various servers on the internet (Weber et al., 2005). The use of this assistance data to determine coarse and accuracy position has been established (Van Diggelen, 2009) and that is presented next for review. Missing prior to this work is the ability to determine precise nanosecond-level time using this assistance data.

As explained above, the TOT and time of reception are needed to calculate the pseudoranges. Without either time, the pseudoranges must be estimated and improved on, as described below.

2.1 | Five-state least squares approach to coarse-time estimation

To investigate this problem, the coarse-time error shall be defined as the difference between the true receiver time and the estimated receiver time. Looking at the effect of adding a coarse-time error, first consider the pseudorange residuals as used in the least squares algorithm:

$$\delta \boldsymbol{z}^{(k)} = \boldsymbol{z}^{(k)} - \hat{\boldsymbol{z}}^{(k)}, \tag{1}$$

where $z^{(k)}$ is the actual pseudorange for satellite k, and $\hat{z}^{(k)}$ is the predicted pseudorange and is given by

$$\hat{\boldsymbol{z}}^{(k)} = \left| \boldsymbol{x}^{(k)}(\hat{t}_{tx}) - \boldsymbol{x}_{xyz0} \right| - \delta_t^{(k)}(\hat{t}_{tx}) + b_0, \tag{2}$$

and $\hat{t}_{tx}$ is the estimated time of transmission of the satellite signal we are measuring, $\boldsymbol{x}^{(k)}(\hat{t}_{tx})$ is the calculated satellite position at time $\hat{t}_{tx}$, $\boldsymbol{x}_{xyz0}$ is the *a priori* receiver position, $\delta_t^{(k)}(\hat{t}_{tx})$ is the satellite clock error in units of length at time $\hat{t}_{tx}$, and b_0 is the *a priori* estimate of the common bias in units of length. However, the problem is that the time $\hat{t}_{tx}$ is in error, so the normal four-state least squares algorithm can not solve for the receiver position and common bias. When $\hat{t}_{tx}$ is incorrect, so is $\boldsymbol{x}^{(k)}(\hat{t}_{tx})$ and $\delta_t^{(k)}(\hat{t}_{tx})$. For each one second of error, the pseudoranges would be incorrect on the order of 800 m (Van Diggelen, 2009). There is hope though, as we know the velocities of the satellites, and thus the pseudorange rate. Below in Equation (3), the difference of estimated pseudorange at the correct and incorrect transmission time is shown. The second expression is true over short times with the assumption that the satellites moved in linear tracks.

$$\begin{aligned} \hat{\boldsymbol{z}}^{(k)}(\hat{t}_{tx}) - \hat{\boldsymbol{z}}^{(k)}(t_{tx}) &= \hat{\boldsymbol{z}}^{(k)}(\hat{t}_{tx}) - \hat{\boldsymbol{z}}^{(k)}(\hat{t}_{tx} + \delta_{tc}) \\ &= -\nu^{(k)}\delta_{tc} \end{aligned}, \tag{3}$$

where δ_{tc} is the coarse-time error, t_{tx} is the actual time of transmission, $\hat{t}_{tx}$ is the coarse-time estimate of the time of transmission, $\nu^{(k)} = (\boldsymbol{e}^{(k)} \cdot \boldsymbol{v}^{(k)} - \dot{\delta}_t^{(k)})$ is the pseudorange rate, $\boldsymbol{e}^{(k)}$ is the unit vector from the receiver to the satellite k, $\boldsymbol{v}^{(k)}$ is the satellite velocity vector, and $\dot{\delta}_t^{(k)}$ is the satellite-clock error rate in units of length/time. So now there is a term that takes care of the coarse-time error in the pseudoranges. The state-update vector is now

$$\delta \boldsymbol{x} = \begin{bmatrix} \delta_x \\ \delta_y \\ \delta_z \\ \delta_b \\ \delta_{tc} \end{bmatrix}, \tag{4}$$

and the pseudorange residuals for each satellite are

$$\delta \boldsymbol{z}^{(k)} = \boldsymbol{z}^{(k)} - \hat{\boldsymbol{z}}^{(k)} = -\boldsymbol{e}^{(k)} \cdot \delta \boldsymbol{x}_{xyz} + \delta_b + \nu^{(k)} \cdot \delta_{tc} + \varepsilon^{(k)}, \tag{5}$$

with $\varepsilon^{(k)}$ encapsulating all atmospheric and other errors. Stacking the pseudorange residuals for all satellites leads to the following:

$$\delta \boldsymbol{z} = \mathbf{H}\delta \boldsymbol{x} + \boldsymbol{\varepsilon}, \tag{6}$$

with

$$\mathbf{H} = \begin{bmatrix} -\boldsymbol{e}^{(1)} & 1 & \boldsymbol{\nu}^{(1)} \\ \vdots & \vdots & \vdots \\ -\boldsymbol{e}^{(K)} & 1 & \boldsymbol{\nu}^{(K)} \end{bmatrix}, \tag{7}$$

where K is the number of satellites. Since there are now five columns of the H matrix, there now must be at least five satellites to solve for $\delta \boldsymbol{x}$. Thus, a closed-form five-state solution has been developed to solve for the position without precise time. However, there is one issue that has yet to be addressed. The measured pseudoranges for each satellite will have a millisecond integer ambiguity error. Explaining and fixing this error will be discussed in the section below. There is also an *a posteriori* residual check to make sure that the computed position and time actually fit the given satellite positions and is usually not a concern if the *a priori* error is not too large.

2.2 | Fixing the millisecond ambiguity

Since the full pseudorange is not available because the subframes are not decoded, there is a problem with correctly identifying the number of milliseconds of each pseudorange. This is because there is uncertainty in the satellite positions, the common bias, and the satellite clock errors. The real problem occurs when the common bias causes an integer rollover. In Van Diggelen (2009), van Diggelen explores the benign and malign cases in depth. Conventional receivers do not have this problem as it relies on the navigation data to identify the time of transmit, whereas in A-GNSS, the data bits are not decoded. To solve for the millisecond ambiguity, the van Diggelen technique will be described.

The first step is to find the sub-millisecond pseudoranges given by the code phase measurements, either from acquisition or tracking. Next, a reference satellite (usually the strongest) is chosen. Using the ephemeris and the *a priori* state, the full pseudorange is estimated and the millisecond integer $\boldsymbol{N}^{(0)}$ is assigned. The zero represents the reference while k will represent any other

satellite. Thus, the reference satellite has full pseudorange $\boldsymbol{N}^{(0)} + \boldsymbol{z}^{(0)}$ where $\boldsymbol{z}^{(0)}$ is the sub-millisecond pseudorange. The assigned integer $\boldsymbol{N}^{(0)}$ will be used to reconstruct the the other satellites' integers in a way that removes the possibility of integer rollover. Keep in mind, the reconstructed full pseudorange has an implicit common bias and is related to the actual geometric range by the following:

$$\begin{aligned} \boldsymbol{N}^{(0)} + \boldsymbol{z}^{(0)} &= \boldsymbol{r}^{(0)} - \delta_t^{(0)} + b + \varepsilon^{(0)} \\ &= \hat{\boldsymbol{r}}^{(0)} - d^{(0)} - \delta_t^{(0)} + b + \varepsilon^{(0)} \end{aligned}, \tag{8}$$

where $\boldsymbol{r}^{(0)}$ is the unknown actual geometric range, $\hat{\boldsymbol{r}}^{(0)}$ is the estimated geometric range from the *a priori* position at the *a priori* time of transmission, $d^{(0)}$ is the error in $\hat{\boldsymbol{r}}^{(0)}$ caused by the error in the *a priori* position and time, $\delta_t^{(0)}$ is the satellite clock error, b is the common bias, and $\varepsilon^{(0)}$ is the measurement errors that includes atmospheric and thermal errors. The goal is to assign integers $N^{(k)}$ such that they share the same common bias. If this is possible, then for satellite k the full pseudorange would be

$$\begin{aligned} \boldsymbol{N}^{(k)} + \boldsymbol{z}^{(k)} &= \boldsymbol{r}^{(k)} - \delta_t^{(k)} + b + \varepsilon^{(k)} \\ &= \hat{\boldsymbol{r}}^{(k)} - d^{(k)} - \delta_t^{(k)} + b + \varepsilon^{(k)} \end{aligned}, \tag{9}$$

where b is the same common bias as for the reference satellite. Subtracting satellite k's pseudorange by the reference satellite's, the following is found:

$$\begin{aligned} &(\boldsymbol{N}^{(k)} + \boldsymbol{z}^{(k)}) - (\boldsymbol{N}^{(0)} + \boldsymbol{z}^{(0)}) = \\ &\quad (\hat{\boldsymbol{r}}^{(k)} - d^{(k)} - \delta_t^{(k)} + b + \varepsilon^{(k)}) \\ &\quad - (\hat{\boldsymbol{r}}^{(0)} - d^{(0)} - \delta_t^{(0)} + b + \varepsilon^{(0)}) \end{aligned}, \tag{10}$$

and solving for $\boldsymbol{N}^{(k)}$:

$$\begin{aligned} \boldsymbol{N}^{(k)} = \\ &\boldsymbol{N}^{(0)} + \boldsymbol{z}^{(0)} - \boldsymbol{z}^{(k)} \\ &+ (\hat{\boldsymbol{r}}^{(k)} - d^{(k)} - \delta_t^{(k)} + b + \varepsilon^{(k)}) \\ &- (\hat{\boldsymbol{r}}^{(0)} - d^{(0)} - \delta_t^{(0)} + b + \varepsilon^{(0)}) \end{aligned}. \tag{11}$$

It is clear that the common bias b cancels exactly. And, if the magnitude of $(-d^{(k)} + \varepsilon^{(k)} + d^{(0)} - \varepsilon^{(0)})$ is less than 0.5 light-milliseconds (about 150 km), then the correct value of $\boldsymbol{N}^{(k)}$ is given by

$$\begin{aligned} \boldsymbol{N}^{(k)} = round(&\boldsymbol{N}^{(0)} + \boldsymbol{z}^{(0)} - \boldsymbol{z}^{(k)} + (\hat{\boldsymbol{r}}^{(k)} - \delta_t^{(k)}) \\ &- (\hat{\boldsymbol{r}}^{(0)} - \delta_t^{(0)})), \end{aligned} \tag{12}$$

where all variables are known, and no matter the value of $\boldsymbol{N}^{(0)}$, the values for $\boldsymbol{N}^{(k)}$ will have the same common bias. By solving for all millisecond ambiguities, the position can now be found. However, precise time has yet to be found as the one millisecond ambiguity is not large enough to distinguish the true time of transmission. Essentially, there is no information to calculate the true time of transmission given only the one millisecond ambiguities. The challenge is to find an integer which ensures all measurements have the same common bias. From Equation (12), we can see that the receiver can achieve this goal by calculating $\boldsymbol{N}^{(\boldsymbol{k})}$ as shown in Equation (12). For a more in-depth description and visualization of this equation development, refer to Van Diggelen (2009).

2.3 | Extended ambiguity time

As mentioned above, one millisecond of ambiguity time is not enough for accurate time recovery as the travel time from satellite to receiver is typically 60 to 90 milliseconds. Therefore, the ambiguity time should be extended if accurate timing service is required for A-GNSS. There are many GNSS constellations and signal structures that we can use to extend the ambiguity time. For example, GPS L2C has 20 milliseconds CM code and 1.5 seconds CL code. But even with the GPS L1 CA code, we can still resolve its ambiguity time to 20 milliseconds by applying the navigation bit synchronization algorithm. Therefore, the following discussions will be focused on using GPS L1 CA code to provide accurate timing service.

However, using the data bit transitions is more challenging. First, there has to be enough data to ensure that a data-bit transition is present. This can be upwards of 100 milliseconds to have a strong probability of transition. If the navigation bit transition is known, then the number of milliseconds from each satellite's bit transition is calculated. Every bit transition occurs at what the satellite thinks is a multiple of 20 milliseconds when it transmits the signal, i.e., at 1.02 second, 1.04 second, 1.06 second, etc. Therefore, the number of milliseconds from this bit transition gives the amount of milliseconds from a time with a modulus of 20 milliseconds. So, if for a satellite the number of milliseconds away from a bit transition was seven, then the possible transmission times could be 1.013 second, 1.033 second, 1.053 second, and likewise for every 20 milliseconds. The concern is how to pick the correct 20 millisecond time since it is ambiguous. Therefore, another constraint is that the five-state least squares algorithm's coarse time error from the true time is less than 10 milliseconds. This is not guaranteed for all geometries or situations, especially when atmospheric errors are not mitigated or the carrier-to-noise ratio is low (Muthuraman

et al., 2012). Assuming the error is less than 10 milliseconds, the time of transmission for satellite k is estimated as

$$\hat{t}_{tx}^{(k)} = \hat{t}_c - \left| \boldsymbol{x}^{(k)}(\hat{t}_{tx}) - \boldsymbol{x}_{xyz} \right| / c + \delta_t^{(k)}, \tag{13}$$

where $\hat{t}_c$ is the coarse time estimate returned by the five-state least squares algorithm, $\boldsymbol{x}^{(k)}(\hat{t}_{tx})$ is the satellite's position at time $\hat{t}_{tx}$, c is the speed of light, and $\delta_t^{(k)}$ is the satellite clock correction.

If the error from the true time is less than 10 milliseconds, then the following equation will round to the correct 20 millisecond:

$$t_{tx_{20ms}}^{(k)} = round(\hat{t}_{tx}^{(k)}/20ms + N^{(k)}/20) * 20ms, \tag{14}$$

where $\hat{t}_{tx}^{(k)}$ is the estimated transmission time given by Equation (13), and $N^{(k)}$ is the integer number of milliseconds from the satellite k's bit transition. With the correctly rounded 20 millisecond transmission time, the number of milliseconds $N^{(k)}$ can now be used to find the transmission of the current millisecond (i.e., without sub-millisecond correction) as

$$t_{tx_{1ms}}^{(k)} = t_{tx_{20ms}}^{(k)} - N^{(k)} \times 10^{-3}. \tag{15}$$

Finally, to find the transmission time with the sub-millisecond correction, the code phase is used as shown below:

$$t_{tx}^{(k)} = t_{tx_{1ms}}^{(k)} + (1 - z^{(k)}) \times 10^{-3}, \tag{16}$$

where $z^{(k)}$ is the code phase for the kth satellite in terms of a fraction of a chip. The calculated transmission time is now as close to the correct transmission time that can be achieved and is summarized in the equation below:

$$t_{tx}^{(k)} = round(\hat{t}_{tx}^{(k)}/20ms + N^{(k)}/20) * 20ms + (1 - z^{(k)} - N^{(k)}) \times 10^{-3}. \tag{17}$$

With this time of transmission, the error in the previously computed transmission time $\hat{t}_{tx}^{(k)}$ can be found for each satellite:

$$\delta_{t_{tx}}^{(k)} = t_{tx}^{(k)} - \hat{t}_{tx}^{(k)}. \tag{18}$$

Ideally, this value would be equal for all satellites, but because of slight errors in the code phase or other sources, they have slight variations. However, they should all be within one millisecond of each other. The average of $\delta_{t_{tx}}^{(k)}$ for all satellites is then added to the estimate of coarse time, as shown in the equation below:

$$t_c = \hat{t}_c + \delta_{t_{tx}}. \tag{19}$$

For even more accuracy, the normal GPS four-state least squares algorithm is then performed with the time estimate and associated satellite positions to further refine the accuracy of the four states (usually on the order of nanoseconds improvement). This results in the nanosecond-level accurate time estimate as comparable to full GPS processing.

Unfortunately, there is a low likelihood that the accuracy of the five-state least squares is within 10 milliseconds (refer to Figure 2). Therefore, a different ambiguity time is required to be a benchmark for the satellite transmission times. All that needs to change for the corrected time of transmission equation is to replace the 20 millisecond modulus time with a different ambiguity time as shown below:

$$t_{tx}^{(k)} = round(\hat{t}_{tx}^{(k)}/t_{mod} + N^{(k)}/(t_{mod}/10^{-3})) * t_{mod} + (1 - z^{(k)} - N^{(k)}) \times 10^{-3}, \tag{20}$$

where t_{mod} is the ambiguity time in seconds, and $N^{(k)}$ is the number of milliseconds from that ambiguity. The experiments described later will investigate the ambiguity time where accurate timing can most likely be achieved.

3 | IMPACT OF DIFFERENT AMBIGUITY TIMES

Even though the five-state least squares converges and gives a good coarse time, the timing can still be tens of milliseconds off depending on the mitigation of error sources. The problem is that it is very difficult to ascertain if the satellites sent their signals at one millisecond or the next since the pseudorange changes on the order of meters. So the least squares will converge but because of the other errors, it is most likely not the true time.

The big question is: how much ambiguity time is required to ensure that the A-GPS algorithm can find time with nanosecond-level accuracy? In order to investigate this, we discuss how different ambiguities impact timing error accuracy in an experimental approach. As the GPS L1 CA signal has been operating stably and is widely used, the main experiment was performed where the accuracy of the timing was examined with several GPS L1 CA signal data sets.

3.1 | Experiment setup

The setup for the entire code structure consists of a GPS L1 software-defined receiver (SDR) (Borre et al., 2007) and an assisted GPS (A-GPS) L1 SDR. For convenience, the traditional SDR will be called the full SDR in the following discussion. A data file is first passed to the full SDR and goes through acquisition, tracking, and position estimation. Importantly, this implies that the file should be longer than 36 seconds as it is the minimal required time length to decode the necessary ephemeris information. After the file is analyzed, the relevant navigation solutions are saved. If time accuracy is not a worry, then only the decoded ephemeris parameters, the acquired satellites, the position, and the first sample time of the file are saved. The number of milliseconds from the start of the file to the first subframe start for each satellite is also saved. This is explained in Section 2.

The saved parameters are then passed to the A-GPS SDR. A detailed description of this interaction is presented in Wang et al. (2019). The position and time are the *a priori* guesses for the A-GPS algorithm (subject to error offsets). Then, the PRNs given by the full SDR are acquired and the five-state least squares A-GPS algorithm is implemented. The signals are tracked for only three seconds and then the time-corrected five-state least squares algorithm is used.

3.2 | Ambiguity time experiments

In order to answer the above-mentioned problem, we present the experimental test results in this section. Basically, it is the search for how much ambiguity time there must be to then converge the time solution within nanoseconds. As an analogy, imagine you go on a walk with signs at specific intervals and a step-counter you reset at every sign. If the signs were a mile apart, then it would be easy to know how far you have walked since you know easily whether you are on your first or second mile and also know how many steps you are after the mile marker. However, it is quite different if the signs were 200 feet apart. It would be very hard to distinguish how many signs you passed but you still know exactly how many steps you are from the last sign. There is a chance of guessing your distance walked correctly, but it is low. With a sign every foot, it would be almost impossible to guess the distance traveled. For A-GPS, the signs are the ambiguity times, and the step counter is represented by the number of milliseconds until the ambiguity time. That is to say: *the longer the ambiguity time is, the more probable it is to get accurate time recovery*.

In the GPS L1 C/A signal, there are three main ambiguity times built in. They are the millisecond code period, the 20 millisecond navigation bit, and the 6 second navigation message subframe. For this experiment, the milliseconds to the subframe start for each satellite are passed to the A-GPS code. With that, it can be modulated down to any ambiguity time that is a factor of 6,000 milliseconds. This is so that at the six second subframe start, every other ambiguity time is also at that time. Then, an analysis of time accuracy as a function of ambiguity time can be tested.

In order to look at the time accuracy with respect to ambiguity time, the A-GPS SDR results are compared with the full SDR results. It is known that the A-GPS needs external assistance data. In order to achieve this purpose, a two-step approach is proceeded: the IF data is first processed with the conventional GPS SDR to generate assistance data. Then, the same IF data and generated assistance data are processed with the A-GPS SDR. The exact processing steps can be explained as follows: data is first collected and processed with the full SDR. The necessary information for A-GPS is collected as well as the full SDR's first sample time. This "first sample time" refers to the receiver's time of collection of the first sample of IF data, a quantity which can easily be time stamped within an A-GPS system. Also collected are the corrected sampling frequency and the number of milliseconds to the first subframe for each satellite. The identical IF data is then processed with the A-GPS SDR and the algorithm described in Section 2.3 is implemented for the desired ambiguity time. This is repeated for every ambiguity time that is being tested. Further, this is then repeated 200 times with new data collected every 15 minutes to cover as many satellite geometric conditions as possible. An important note is that the first sample time and corrected sampling frequency calculated by the full SDR is calculated using only the results from the first three seconds of data so that it is a better comparison to the A-GPS code.

In Figure 1, the first sample time errors are shown with respect to ambiguity time. The first sample time error refers to the error between the time of the first sample calculated by the full SDR and the time calculated by the A-GPS SDR. Each color represents a single trial.

As shown, the ambiguity time chosen makes a large difference in the accuracy of the time estimate with many errors on the order of seconds. As ambiguity time increases, more trials reach nanosecond-level accuracy with 500 milliseconds being enough time for all trials. Interestingly, the error increases in some cases as ambiguity time increases. This is likely due to the transmission time rounding to the incorrect time, which is a bigger mistake the larger the ambiguity. Though as soon as there is a large enough ambiguity time for that data, the error then converges to nanosecond-level error.

In this context, convergence refers to the iterative least squares of the A-GPS SDR reaching nanosecond-level

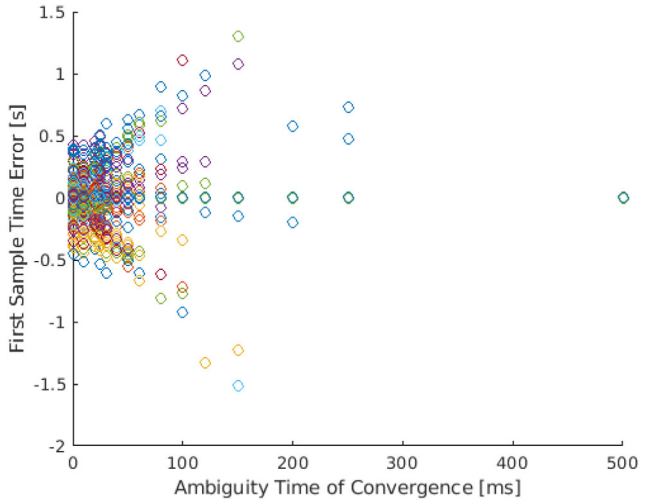

FIGURE 1 First sample time error w.r.t ambiguity time [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

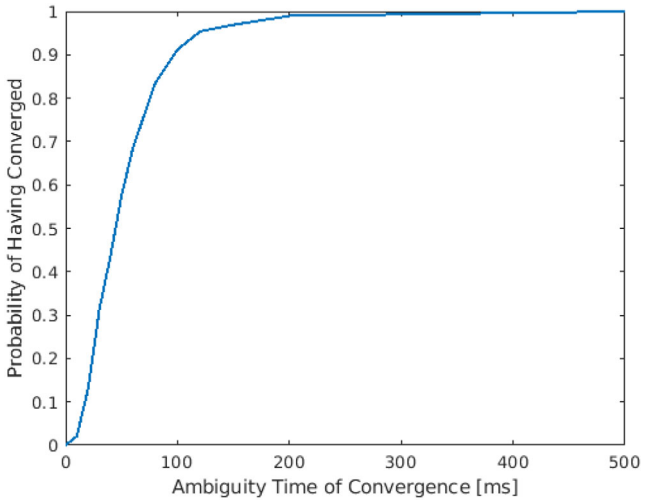

FIGURE 2 Cumulative distribution function of ambiguity time of convergence [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

error as compared to the full SDR. Note that in this paper, the convergence for timing is defined such that the timing error between A-GPS and full SDR is less than two nanoseconds. In Figure 2, the cumulative distribution is shown that gives a good visualization of convergence. As can be seen, at 100 milliseconds, it is 91% likely to converge while at 200 milliseconds it is 99% likely.

The results for key tested ambiguities can be seen in Table 1. It is also important to point out that using the navigation bit ambiguity time of 20 milliseconds only results in 13 % probability of convergence.

These results lead to the next question: what causes the least squares to converge at different ambiguity times? One

TABLE 1 Probability of having converged given ambiguity time

Ambiguity Time (millisecond)	10	20	50	100	150	200
Convergence Prob. (%)	2.1	13.0	57.5	91.2	96.9	99.0

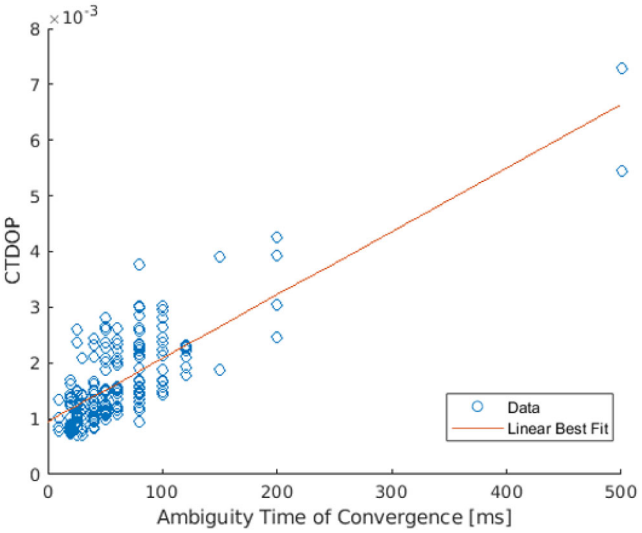

FIGURE 3 Coarse time DOP w.r.t ambiguity time of convergence [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

possible metric is the dilution of precision of the fifth state, coarse time, as explained by Muthuraman et al. (2012). Let $\boldsymbol{D} = (\boldsymbol{H}^T\boldsymbol{H})^{-1}$ be the dilution of the precision matrix with $\boldsymbol{H}$ being the 5x5 linearized transformation matrix described earlier. The coarse-time dilution of precision (CTDOP) is then

$$CTDOP = \sqrt{\boldsymbol{D}_{5,5}}. \tag{21}$$

Muthuraman et al. (2012) examined the relationship between CTDOP and TOW time error without using the ambiguity convergence technique shown in this paper. With this technique, however, CTDOP can give insights into how long the ambiguity time must be for convergence. Below in Figure 3 is a plot of the coarse time dilution of precision (CTDOP) with respect to the ambiguity time of convergence.

Above, a linear best fit is plotted in red. Although the data is wide spread, the CTDOP does explain the ambiguity time of convergence. With only 10 milliseconds required, the CTDOP is very small and when 500 milliseconds is required, the CTDOP is very large. The CTDOP is more clearly visualized as a 2-D histogram in Figure 4 below. Most of the distribution is within 200 milliseconds and a CTDOP of 0.003.

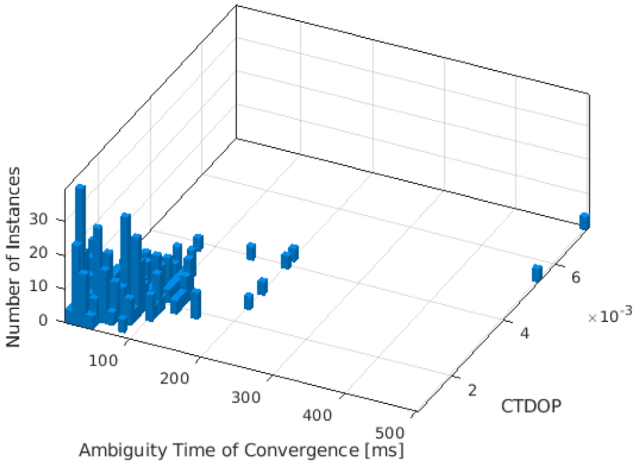

FIGURE 4 2-D histogram of coarse time DOP and ambiguity time of convergence [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

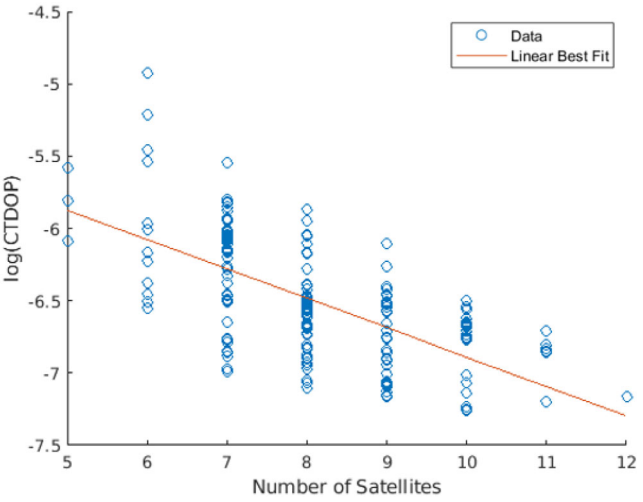

FIGURE 5 The logarithm of CTDOP w.r.t number of satellites [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

Another possible metric for predicting the ambiguity time of convergence is the number of satellites. However, it might be important to see the relationship between the number of satellites and CTDOP first. Looking at Figure 5, the logarithm of the CTDOP is plotted with respect to the number of satellites with a linear best fit plotted in red:

This shows that there is an exponential decay in CTDOP as the number of satellites increases. This is an interesting but reasonable result, since the geometry of the satellites directly impacts the coarse time result.

As an important note, it is necessary to see the distribution of the number of satellites that the experiment's data contains. In Figure 6, the number of satellites for the experiment is shown.

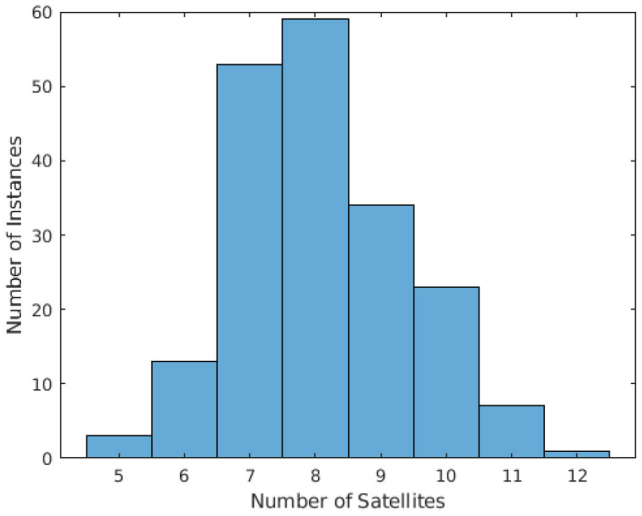

FIGURE 6 Histogram of number of satellites [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

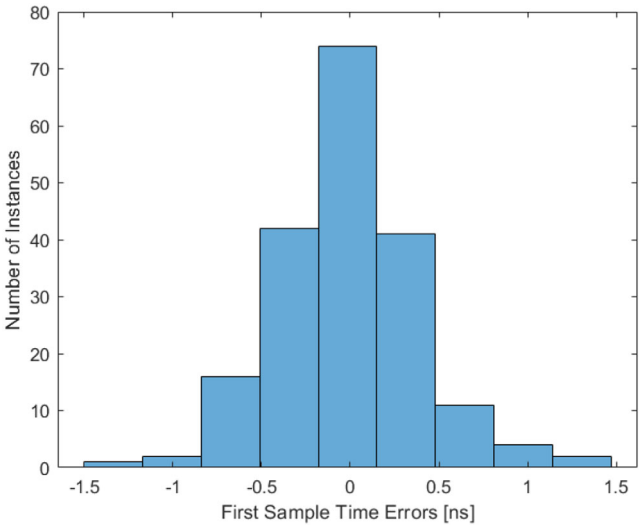

FIGURE 7 Probability distribution of first sample time error [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

This is significant in the fact that most of the cases had a good amount of satellites, and thus, the data does not represent the cases with five or six satellites well.

Looking at the first sample time, Figure 7 shows the distribution of first sample time errors as compared to the full SDR for all trials.

As can be seen, more than 95% of errors fall under a magnitude of one nanosecond error with none exceeding two nanoseconds. An important note is that these first sample time errors are the errors between the A-GPS SDR and the full SDR. These are the errors once converged, and they do not change as ambiguity time is increased further. Muthu-

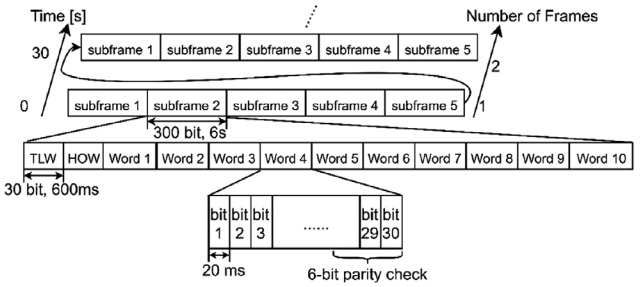

FIGURE 8 GPS navigation message structure

raman's method 1 in Muthuraman et al. (2012) can achieve similar accuracy some of the time with an ambiguity time of 20 milliseconds. However, in cases where the data bit knowledge is not enough, the method produces errors in multiples of 20 milliseconds.

Clearly, the choice of ambiguity time makes a large impact on the accuracy of the time estimate. These results show that nanosecond-level accuracy can be achieved reliably with an ambiguity time larger than 200 milliseconds.

4 | SAMPLE SOLUTION

In the previous section, experiments show that once the ambiguity time can be resolved to larger than 200 milliseconds, nanosecond-level time recovery can be achieved absolutely. Some use cases may allow such levels of timing accuracy to be obtained via the data network. This paper has developed and presented the timing and probability levels of success. However, many field experiments rely on data links with low bandwidth and variable latencies. Thus, sufficiently large ambiguity times must be obtained from the IF data stream itself. It has been shown that the 20 millisecond navigation data bit has a low probability of resolving the desired nanosecond time resolution. It is obvious that data segments of greater than 6s can be used to obtain much better ambiguity times using the decoded frame synchronization. Yet, 6s of IF data could be considered extreme for an assisted solution. Therefore, we have identified an example characteristic that can be used to find a sufficient ambiguity time while minimizing data volume. This example utilizes a parity check algorithm to resolve GPS L1 CA code ambiguity time to 600 milliseconds. Then, we discuss briefly how all GNSS signals have similar structures, which can be exploited.

The GPS NAV navigation message is transmitted in five 300-bit subframes (Kaplan, 2017), as shown in Figure 8. From Figure 8, we can see that each subframe is composed of ten words, while each word is composed of 30 bits. Note that the six-bit parity check bits exist at the end of each word. Since the location of the parity bits are known, the amount of time from the start of the word can be calculated if the location of the parity bits are found. This characteristic leads us to consider using the parity check bits to narrow down the ambiguity time to 600 milliseconds (length of time for a word) for an assisted GPS receiver.

4.1 | Introduction of regular parity checking algorithm

In a conventional receiver, a Hamming code [32,26] is employed for parity checking and error detection (GPS.gov, 2010). The six-bit parity check bits are obtained by using 24 bits of the current word and the last two bits of the previous word by using the following equations:

$$[D_{25}D_{26}D_{27}D_{28}D_{29}D_{30}]^T = \boldsymbol{H} \times [D_{29}^{-}D_{30}^{-}d_1d_2d_3 \ldots d_{23}d_{24}]^T, \tag{22}$$

where the H matrix is

$$\boldsymbol{H} = \begin{bmatrix} 101110110001111100110100 10 \\ 010111011000111110011010 01 \\ 101011101100011111001101 00 \\ 010101110110001111100110 10 \\ 011010111011000111110011 01 \\ 100010110111101010001001 11 \end{bmatrix}. \tag{23}$$

The elements '1' and '0' in the matrix H represent if the corresponding bit is used or is not used in the calculation of that row's parity bit. For example, the parity bit D_{25} is the sum of all the bits with a "1" in the first row of the H matrix and likewise for the five remaining parity bits. Therefore, by checking the consistency between the calculated parity check bits and the decoded parity check bits, the receiver can claim whether or not the parity check passed. If the parity check passed, then it is flagged.

4.2 | Proposed approach

Using the above parity check method, we introduce a new approach that can find a large enough ambiguity time by tracking a short period of GPS L1 CA signal. This approach is achieved by finding flags denoting that the parity check passed. As discussed above, the parity check flag repeats every 30 bits. Therefore, by tracking a GPS L1 CA signal for a short period of time, and using this repeating pattern, we can find the exact position of every parity checking bit. Figure 9 shows the schematic of proposed algorithm.

From Figure 9, we can see that the proposed algorithm starts with signal acquisition and tracking procedures. Next, if a phase lock loop (PLL) is used then the prompt

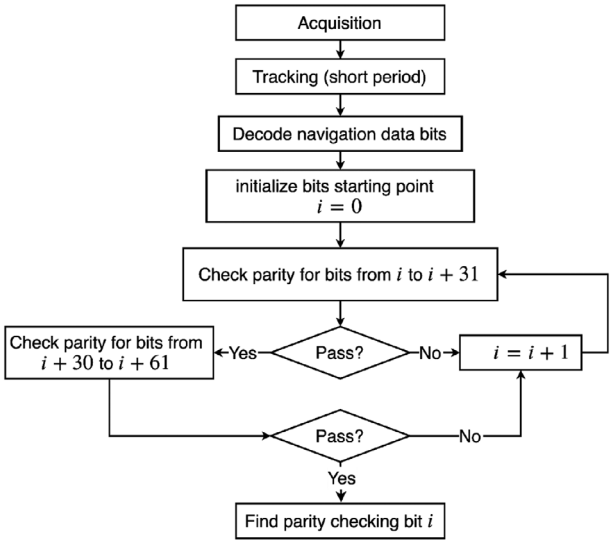

FIGURE 9 Proposed algorithm

output of the I channel can be used to decode the navigation data bits. If a frequency lock loop (FLL) is used, differential bit decoding algorithms can be adopted (Natali, 1986). Then, the decoded bits are sent to the parity checking module. The detailed logic of the parity checking module is illustrated in Figure 9. To summarize, only when two parity check flags are found spaced by 30 bits, i.e., i and i+30, the algorithm claims that parity check bit i is found. Knowing that a bit is the parity checking bit is enough information to calculate the offset from an ambiguity time of 600 milliseconds (since every word is 600 milliseconds). As shown in the previous section, this is more than enough time to reach nanosecond-level accuracy reliably.

The duration of tracking in order to apply the proposed method depends on the starting points and parity check threshold. From the previous analyses, we know that to detect one parity check passed flag, we need to have a data record that includes D_{29}^- and D_{30}^-, as well as $[d_1 d_2 d_3 \ldots d_{23} d_{24}, D_{25}, \ldots, D_{30}]$. Therefore, the worst case is in the case that data bit starts with D_{30}^-. However, the false alarm probability of detecting one parity check passed flag could be 1.5 % to 3%. This false alarm probability is relatively high when considering a robust application. A detailed analysis for the false alarm probability is presented in the next section. Therefore, to enhance the robustness of the proposed algorithm, we set the threshold of parity check passed flag to two, which means that the algorithm claims that the parity check bits are found when two continuous parity check passed flags are detected.

Figure 10 shows the schematic of the proposed method. As can be seen from the blue line, given a random data starting point, we need at most 61 bits to find one parity check bit. Similarly, to find two contentious parity check flags, we need at most 91 bits of data. When considering the

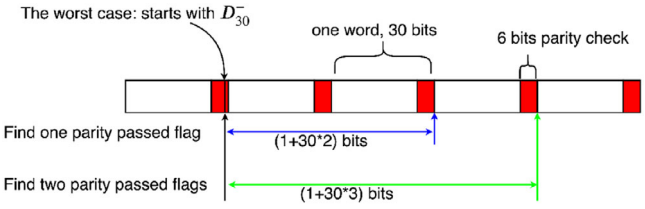

FIGURE 10 Schematic of the proposed method [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

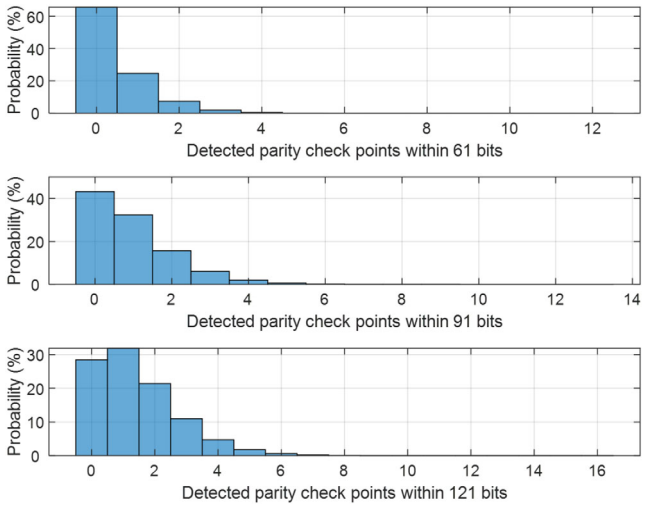

FIGURE 11 Detected parity check points within 61 bits, 91 bits, and 121 bits, obtained by Monte-Carlo simulation with random data [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

starting bits of the GPS signal might be wrong due to the pull-in process of signal tracking, it would be more robust to get 92 to 94 bits of decoded navigation data for processing, which is 1,840 to 1,880 milliseconds for the GPS L1 CA signal.

4.3 | Monte Carlo analyses and live data tests

Though the parity check algorithm used in a GPS C/A signal receiver is specifically designed for detecting bit errors, some other undesired bit strings can still pass the parity checking. Therefore, we used both Monte Carlo analysis and live data testing approaches to evaluate the probability of random bits or real navigation bits passing the parity check.

The probabilities for detecting parity check points with 61 bits, 91 bits, and 121 bits are shown in Figure 11. As seen, the probability of detecting one parity check point and two parity check points are relatively high. Note that in

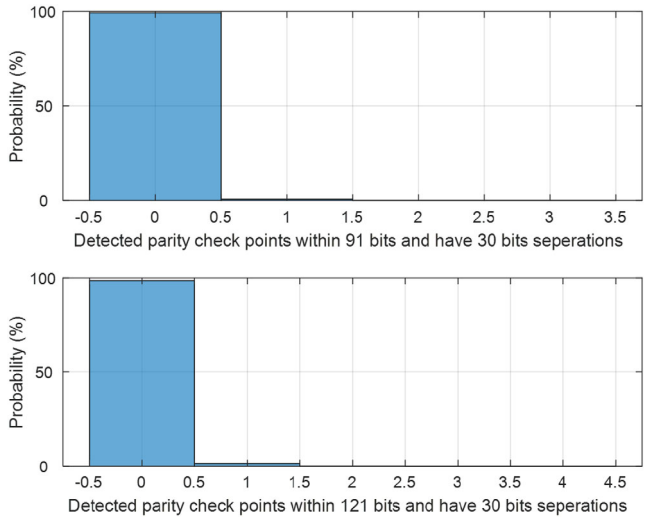

FIGURE 12 Detected parity check points within 91 and 121 bits and have 30 bits separations, obtained by Monte-Carlo simulation with random data [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

TABLE 2 False alarm probability obtained by MC simulation when considering 30 bits separations

Number of detected groups	1	2	3 or more
91 random bits	0.70 %	0.01 %	0.01 %
121 random bits	1.37 %	0.04 %	0.04 %

this simulation, the separations between two parity check points are not considered.

When considering the separations between parity check points, the false alarm probability decreases dramatically, as it can be seen in Figure 12. The false alarm probabilities for 91 bits and 121 bits are shown in Table 2.

Table 3 shows the false alarm probability for parity checking obtained by a three-hour GPS L1 CA signal data set. As it can be seen, the false alarm probability for finding one parity check passed flag ranges from 2.22 % to 2.91 % within three hours, which is relatively high when considering a robust assisted GPS application. However, the probability for finding two continuous flags with 30-bit spacing is very low within three hours, ranging from 0 % to 0.08 %.

TABLE 3 False alarm probability obtained by live data test

PRN	19	12	2	6	24
FA prob. for finding 1 flag	2.89 %	2.68 %	2.91 %	2.22 %	2.60 %
FA prob. for finding 2 cont. flags	0 %	0.01 %	0.08 %	0 %	0 %
FA prob. for finding 3 cont. flags	0 %	0 %	0 %	0 %	0 %

Ultimately, the parity check algorithm shows the ability to achieve an ambiguity time large enough without the expense of excessive data volumes. This allows for reliable nanosecond-level accuracy with just under two seconds of GPS data given an *a priori* estimate.

5 | CONCLUSION

In this paper, a means to obtain nanosecond-level accuracy was developed and presented. In parallel, the requirement of the ambiguity time to reach nanosecond-level accuracy was investigated and presented.

Experiments were then run to test the performance of the first sample time error, a common time marker in field hardware, as a function of ambiguity time. It was found that an ambiguity time of 200 milliseconds converged to nanosecond-level accuracy 99 % of the time. This conclusion can be enlightening for future GNSS signal design. Especially when designing the navigation message structure, taking this into consideration would be beneficial for consumer electronic devices.

Interestingly, the coarse time dilution of precision had a linear relationship with the ambiguity time of convergence (the ambiguity time that achieves nanosecond-level accuracy) and decayed exponentially with an increase in the number of satellites. Most first sample time errors were within one nanosecond while none exceeded two nanoseconds.

Finally, an example using the 600 milliseconds GPS L1 CA signal parity check in each GPS word was described and confirmed to be a reliable and consistent ambiguity time. This allows for reliable nanosecond-level accuracy with a relatively small amount of data. The live data results show that with just under two seconds of data, the algorithm will have a 99% probability to resolve the full-time ambiguity.

With this technique, field applications that require real-time nanosecond-level timing can be accomplished robustly using an adapted assisted GPS/GNSS approach presented within this paper. That approach involves: (a) leveraging the expanded algorithm to resolve nanosecond-level time and (b) utilizing a few seconds of IF data as opposed to a few milliseconds (A-GNSS receiver), which is still more than an order of magnitude less time than required for decoding the required broadcast navigation parameters (conventional GNSS receiver). Future work can be focused on utilizing new characteristics of other systems/signals (such as Galileo E1 OS) to resolve the assisted GNSS time ambiguity.

ORCID

*Boyi Wang** https://orcid.org/0000-0002-4598-9890

REFERENCES

Agarwal, N., Basch, J., Beckmann, P., Bharti, P., Bloebaum, S., Casadei, S., Chou, A., Enge, P., Fong, W., Hathi, N., Mann, W., Sahai, A., Stone, J., Tsitsiklis, J., & Van Roy, B. (2002). Algorithms for GPS operation indoors and downtown. *GPS Solutions*, *6*(3), 149–160. https://doi.org/10.1007/s10291-002-0028-0.

Akopian, D., & Syrjarinne, J. (2002, April). A network aided iterated LS method for GPS positioning and time recovery without navigation message decoding. *2002 IEEE Position Location and Navigation Symposium*, Palm Springs, CA, 77–84. https://doi.org/10.1109/PLANS.2002.998892

Akopian, D., & Syrjarinne, J. (2009). A fast positioning method without navigation data decoding for assisted GPS receivers. *IEEE Transactions on Vehicular Technology*, *58*(8), 4640–4645. https://doi.org/10.1109/TVT.2009.2019073

Bissig, P., Eichelberger, M., & Wattenhofer, R. (2017, April). Fast and robust GPS fix using one millisecond of data. *2017 16th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, Pittsburgh, PA, 223–234.

Borre, K., Akos, D. M., Bertelsen, N., Jensen, S. H., & Rinder, P. (2007). *A software-defined GPS and Galileo receiver: A single-frequency approach*. Birkhauser Boston.

Cheong, J. W., Wu, J., & Dempster, A. (2014). Dichotomous search of coarse time error in collective detection for GPS signal acquisition. *GPS Solutions*, *19*(1), 61–72. https://doi.org/10.1007/s10291-014-0365-9

Dovis, F., Lesca, R., Margaria, D., Boiero, G., & Ghinamo, G. (2008, May). An assisted high-sensitivity acquisition technique for GPS indoor positioning. *2008 IEEE/ION Position, Location and Navigation Symposium*, Monterey, CA, 1350–1361. https://doi.org/10.1109/PLANS.2008.4570123

Glennon, E. P., & Bryant, R. C. (2005, September). Solution of timing errors for AGPS. *Proceedings of the 18th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2005)*, Long Beach, CA, 136–141.

GPS.gov. (2010). *Global Positioning System Wing (GPSW) systems engineering & integration interface specification IS-GPS-200 revision E*. https://www.gps.gov/technical/icwg/

Huang, G., & Akopian, D. (2013, September). A-GPS assistance network delay modeling and estimation over mobile networks. *Proceedings of the 26th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2013)*, Nashville, TN, 1944–1950. https://www.ion.org/publications/abstract.cfm?articleID=11233

Huang, G., Akopian, D., & Chen, C. L. P. (2014). Measurement and modeling of network delays for MS-based A-GPS assistance delivery. *IEEE Transactions on Instrumentation and Measurement*, *63*(8), 1896–1906. https://doi.org/10.1109/TIM.2014.2302238

Huang, G., Akopian, D., & Chen, C. L. P. (2015). Network delay modeling for assisted GPS. *IEEE Transactions on Aerospace and Electronic Systems*, *51*(1), 52–64. https://doi.org/10.1109/TAES.2014.120686

Huang, G., Miller, M. M., & Akopian, D. (2016). Inference of network delays for SUPL 3.0-based assisted GNSS. *GPS Solutions*, *21*, 651–661. https://doi.org/10.1007/s10291-016-0549-6

Kaplan, E. (2017). *Understanding GPS/GNSS: Principles and applications*. Artech House.

Li, J. (2010, February). Eliminating abnormal positioning bias with translation technology in AGPS method. *2010 The 2nd International Conference on Computer and Automation Engineering (ICCAE)*, Singapore, 570–574. https://doi.org/10.1109/ICCAE.2010.5451818

Li, J. (2012, April). The analysis of positioning bias without current navigation data. *2012 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet)*, Yichang, China, 1369–1372. https://doi.org/10.1109/CECNet.2012.6201886

Muthuraman, K., Brown, J., & Chansarkar, M. (2012, January). Coarse time navigation: Equivalence of algorithms and reliability of time estimates. *Proceedings of the 2012 International Technical Meeting of The Institute of Navigation*, Newport Beach, CA, 1115–1138. www.ion.org/publications/abstract.cfm?articleID=10010

Natali, F. (1986). Noise performance of a cross-product AFC with decision feedback for DPSK signals. *IEEE Transactions on Communications*, *34*(3), 303–307. https://doi.org/10.1109/TCOM.1986.1096521

Open Mobile Alliance. (2012). *Secure user plane location architecture* (OMA-AD-SUPL-V2_0-20120417-A). https://www.openmobilealliance.org/release/SUPL/V2_0-20120417-A/OMA-AD-SUPL-V2_0-20120417-A.pdf

Otaegui, O., Lucas, N., & Rohmer, G. (2006, September). A hybrid architecture for high sensitivity standalone and assisted Galileo/GPS receivers. *Proceedings of the 19th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2006)*, Fort Worth, TX, 2361–2369. https://www.ion.org/publications/abstract.cfm?articleID=6982

Sirola, N. (2001). *A method for GPS positioning without current navigation data* (Master's thesis). Tampere University of Technology, Tampere.

Sirola, N., & Syrjarinne, J. (2002, September). GPS position can be computed without the navigation data. *Proceedings of the 15th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GPS 2002)*, Portland, OR, 2741–2744.

Syrjarinne, J. (2000, July). Time recovery through fusion of inaccurate network timing assistance with GPS measurements. *Proceedings of the Third International Conference on Information Fusion*, Paris, France. https://doi.org/10.1109/IFIC.2000.859864

Syrjarinne, J. (2001). Wireless-assisted GPS: Keeping time with mobiles. *GPS World*, *12*(1), 22–31.

Vallina-Rodriguez, N., Crowcroft, J., Finamore, A., Grunenberger, Y., & Papagiannaki, K. (2013). When assistance becomes dependence: Characterizing the costs and inefficiencies of A-GPS. *Mobile Computing and Communications Review*, *17*, 3–14.

Van Diggelen, F., & Abraham, C. (2007, September). Coarse-time AGPS; Computing TOW from pseudorange measurements, and the effect on HDOP. *Proceedings of the 20th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2007)*, Fort Worth, TX, 357–367.

Van Diggelen, F. S. T. (2009). *A-GPS: Assisted GPS, GNSS, and SBAS*. Artech House.

Wang, B., Ruane, L., Blay, R., & Akos, D. M. (2019, September). Assisted GNSS: An open source SDR-based approach. *Proceedings of the 32nd International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2019)*, Miami, FL, 2940–2949. https://doi.org/10.33012/2019.17015https://doi.org/10.33012/2019.17015

Weber, G., Dettmering, D., Gebhard, H., & Kalafus, R. (2005, September). Networked transport of RTCM via internet protocol (Ntrip)…IP-streaming for real-time GNSS applications. *Proceed-*

ings of the 18th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2005), Long Beach, CA, 2243–2247.

Weng, C., Chien, Y., Chen, C., Huang, K., & Yau, W. (2011, September). An efficient method of self-generated assistance data for fast-fix applications. *Proceedings of the 24th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2011)*, Portland, OR, 3967–3974. https://www.ion.org/publications/abstract.cfm?articleID=9954

How to cite this article: Blay R, Wang B, Akos DM. Deriving Accurate Time from Assisted GNSS Using Extended Ambiguity Resolution. *NAVIGATION*. 2021;68:217–229. https://doi.org/10.1002/navi.412