THE JOURNAL OF NAVIGATION (2012), **65**, 363–375. © The Royal Institute of Navigation
doi:10.1017/S0373463311000737

Computational Efficiency Improvement for Unaided Weak GPS Signal Acquisition

Wen Zhang[1] and Mounir Ghogho[2]

[1] (*National University of Defense Technology, Changsha, China*)
[2] (*University of Leeds, United Kingdom; International University of Rabat, Morocco*)
(E-mail: zhangwendaisy@hotmail.com)

Acquisition of unaided weak Global Positioning System (GPS) signals requires long coherent integration time and thus all the possible navigation data bit combination paths have to be searched. In this paper, to improve the computational efficiency, the Improved Fast Modified Double-Block Zero Padding (IFMDBZP) algorithm using the *Optimal Path Search Method (OPSM)* is proposed instead of the FMDBZP algorithm using the *All Paths Search Method (APSM)*. The proposed method consists of unlikely data bit combination path elimination by applying the Viterbi algorithm during each coherent integration step to improve the FMDBZP algorithm. The analyses show that the proposed OPSM can reduce the computation calculations and save memory space without suffering any loss compared to the APSM. And the longer the coherent integration time is, the more benefit one can gain from the proposed method. The simulation results also show that the IFMDPZP algorithm using the proposed OPSM has the same acquisition performance as the APSM.

KEY WORDS

1. GPS. 2. Acquisition. 3. Weak Signal. 4. Viterbi Algorithm.

1. INTRODUCTION. GPS is widely used nowadays but still faces limitations in indoor and dense-urban environments because of the weak received signals. Acquisition of unaided weak GPS signals requires long coherent integration time and more noncoherent integration steps to boost the post-correlation Signal-to-Noise Ratio (SNR). When the coherent integration time is a multiple of one navigation data bit, a search over all possible data bit combinations is required. Since the data bit duration is 20 ms and the data bit edge is synchronized with the 1 ms period Pseudo Random Noise (PRN) code, there are twenty possible bit edge positions. Thus, for weak signals, we have to search over both possible data bit combinations and possible bit edge positions, besides the search of visible satellites, Doppler shifts and code phases.

Many acquisition approaches have been developed. The conventional hardware approach sequentially searches for visible satellites at each possible code phase and Doppler frequency shift bin (Parkinson and Spilker, 1996; Misra and Enge, 2001);

https://doi.org/10.1017/S0373463311000737 Published online by Cambridge University Press

a simple but time-consuming method. Then circular correlation method using Fast Fourier Transform (FFT) is introduced to reduce the processing time (Tsui, 2004; Van Nee and Coenen, 1991). Calculation operations are reduced because the correlation at all code phases (or frequency bins) can be computed by one FFT operation, for each Doppler bin (or code phase). In (Tsui, 2004; Lin and Tsui, 2000; Lin et al., 1999), an approach that processes long data coherently with fewer operations, called Double-Block Zero Padding (DBZP), is discussed. It uses a frequency domain method to search over a delay-Doppler space and acquires the GPS signal, using coherent integration time that is a multiple of the 20 ms data bit period. DBZP is especially suitable for unaided software-defined GPS receivers. But the presence of the 50 Hz data bit message and the unknown satellite Doppler shift limit its ability to detect weaker signals through long coherent and noncoherent integrations. Ziedan and Garrison (Ziedan and Garrison, 2004) first modified the DBZP algorithm to solve this problem. The Modified DBZP (MDBZP) algorithm uses a coherent integration time that is a multiple of one data bit, without assuming the availability of any bit edge assisting information. Furthermore, it circumvents DBZP's limitation resulting from the unaccounted effect of Doppler shift on the code duration. It also introduces a frequency bin elimination method after one or several coherent integration steps. Then Heckler and Garrison (Heckler and Garrison, 2009) introduced the FMDBZP algorithm based on MDBZP algorithm by eliminating the redundant FFT computations. The number of FFT operations in FMDBZP is found to scale linearly with the coherent integration time, in contrast to the geometric growth experienced in MDBZP. Although FMDBZP can save more operations and memory space than MDBZP, it also can be improved during the coherent integration steps. In this paper, the *Optimal Path Search Method (OPSM)* is proposed to instead of the *All Paths Search Method (APSM)* for the FMDBZP algorithm. It consists of unlikely data bit combination path elimination during each coherent integration step to improve the FMDBZP algorithm. It is based on the Viterbi algorithm theory, and aims to reduce the computational operations and save memory space further.

The rest of the paper is organized as follows. In Section 2, the implementation of FMDBZP is illustrated using matrices and figures; in particular, the APSM is introduced. In Section 3, the OPSM during each coherent integration step is proposed based on the Viterbi algorithm. In Section 4, the complexity analysis and comparison between the APSM and the proposed OPSM is analysed. In Section 5, the example testing of the OPSM is presented. In Section 6, conclusions are drawn.

2. IMPLEMENTATION OF FMDBZP.

After double-block processing of the received GPS baseband data, zero padding processing of the local code replica and circular correlation between them, an important primary integration matrix M_{pi} will be obtained (Ziedan and Garrison, 2004; Heckler and Garrison, 2009; Ziedan, 2006). The matrix has size of $(N_b + N_b/N_{db}) \times N_{ms}$, where N_b is the number of possible Doppler frequency bins, N_{ms} is number of possible code phase, N_{db} is the number of navigation data bits in the coherent integration time T_I. The submatrix of size $N_b \times N_{ms}$ corresponds to the integration matrix of T_I data. The remaining extra submatrix of size $(N_b/N_{db}) \times N_{ms}$ corresponds to the integration matrix of the next one data bit long data. It will be used later when the possible bit edges are taken into account.

https://doi.org/10.1017/S0373463311000737 Published online by Cambridge University Press

The process is as follows. Divide each one of the N_{ms} columns into $(N_{db}+1)N_{be}$ vectors of length $N_b/(N_{db}N_{be})$, where N_{be} is the number of possible data bit edges. Extract each vector from the original column. For each of the 1st to $N_{db}N_{be}$th vectors, remember its original position and then zero pad it into a new vector of length N_b. For the $(N_{db}N_{be}+1)$st to $((N_{db}+1)N_{be}-1)$st vectors, zero padding them using the same method as the 1st to $(N_{be}-1)$st vectors, respectively. The $(N_{db}+1)N_{be}$th vector is discarded. Then $((N_{db}+1)N_{be}-1)$ vectors of length N_b are obtained. The divide and zero padding process is shown in Figure 1. To transform the data to the frequency domain, perform N_b-point zoom FFT (Wild et al., 1987; Sorensen and Burrus, 1993; He and Torkelson, 1996; Yang, 2001) on the $((N_{db}+1)N_{be}-1)$ vectors and save the resulted vectors as $X_1, X_2, ..., X_{(N_{db}N_{be}+1)}, X_{(N_{db}N_{be}+2)}, ..., X_{((N_{db}+1)N_{be}-1)}$. Each member is a vector of length N_b.

Then, take into account the N_{be} possible navigation data bit edges, and rearrange the $X_1, X_2, ..., X_{(N_{db}N_{be}+1)}, X_{(N_{db}N_{be}+2)}, ..., X_{((N_{db}+1)N_{be}-1)}$ vectors into a matrix of size $(N_{db}N_{be})N_{be}$. The matrix is named m_{ti} and shown in Figure 2.

Because every N_{be} data belong to the same data bit, add every N_{be} vectors of each column of matrix m_{ti} in Figure 2. The resulted matrix m_{bei} has size $N_bN_{db}N_{be}$ and is shown in Figure 3. Each member Y_{ij} is a vector of length N_b. The additions are as follows:

$$
\begin{array}{c}
Y_{11} = X_1 + X_2 + \cdots + X_{N_{be}} \\
Y_{21} = X_{(N_{be}+1)} + X_{(N_{be}+2)} + \cdots + X_{2*N_{be}} \\
\vdots \\
Y_{N_{db}1} = X_{((N_{db}-1)N_{be}+1)} + X_{((N_{db}-1)N_{be}+2)} + \cdots + X_{N_{db}*N_{be}} \\
\hline
Y_{12} = X_2 + X_3 + \cdots + X_{(N_{be}+1)} \\
Y_{22} = X_{(N_{be}+2)} + X_{(N_{be}+3)} + \cdots + X_{(2*N_{be}+1)} \\
\vdots \\
Y_{N_{db}2} = X_{((N_{db}-1)N_{be}+2)} + X_{((N_{db}-1)N_{be}+3)} + \cdots + X_{(N_{db}*N_{be}+1)} \\
\hline
\vdots \\
\hline
Y_{1N_{be}} = X_{N_{be}} + X_{(N_{be}+1)} + \cdots + X_{(2*N_{be}-1)} \\
Y_{2N_{be}} = X_{(2*N_{be})} + X_{(2*N_{be}+1)} + \cdots + X_{(3*N_{be}-1)} \\
\vdots \\
Y_{N_{db}N_{be}} = X_{N_{db}N_{be}} + X_{(N_{db}N_{be}+1)} + \cdots + X_{((N_{db}+1)*N_{be}-1)}
\end{array}
\tag{1}
$$

To reduce the calculation operations of Equation Set (1), first calculate $Y_{11} = X_1 + X_2 + \cdots + X_{N_{be}}$, so $Y_{12} = Y_{11} - X_1 + X_{(N_{be}+1)}$. Then apply the same method to calculate Y_{13} to $Y_{N_{db}N_{be}}$.

Finally, take into account all the $2^{(N_{db}-1)}$ possible navigation data combination paths when calculate the coherent integration. Take the first column of matrix m_{bei} in Figure 3 as example, the coherent integration operation method on the $2^{(N_{db}-1)}$ paths is shown in Figure 4. After all the N_{be} columns of matrix m_{bei} in Figure 3 are processed

https://doi.org/10.1017/S0373463311000737 Published online by Cambridge University Press

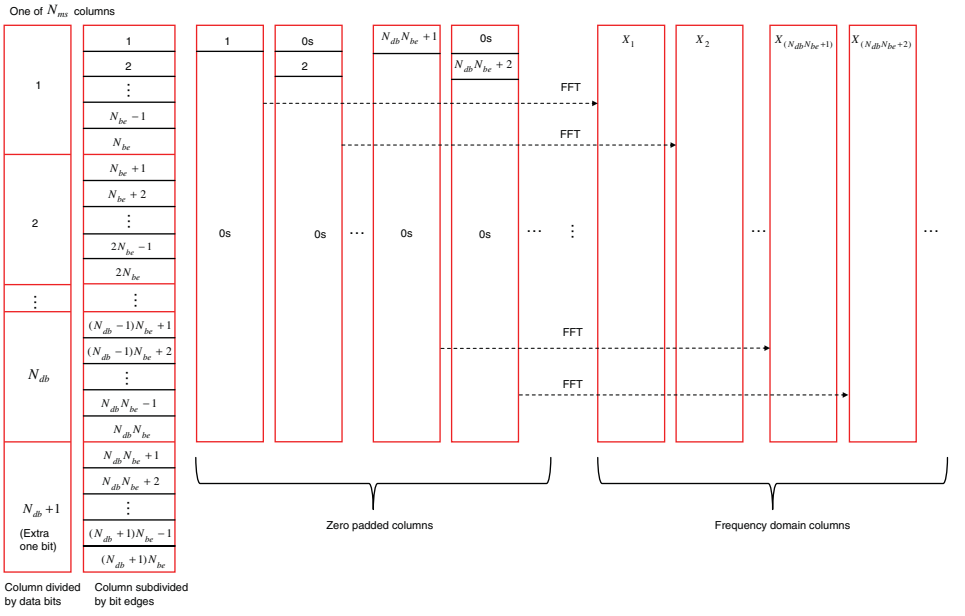


Figure 1. The division, zero padding and zoom FFT process of one column of the primary integration matrix M_{pi}.

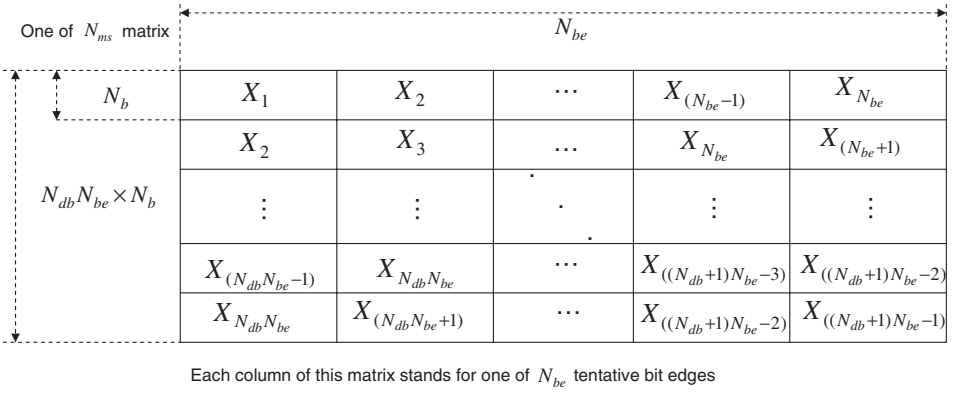


Figure 2. The matrix m_{ti} formed by taking into account the N_{be} possible navigation data bit edges.

similarly, the partial coherent integration matrix m_{ci} of size $2^{(N_{db}-1)}N_bN_{be}$ is obtained, which is shown in Figure 5. Each member is a vector of length N_b. Because all the $2^{(N_{db}-1)}$ possible navigation data combination paths have to be searched exhaustively, we call it the *All Path Search Method (APSM)*.

After all the N_{ms} columns of the primary integration matrix M_{pi} are processed using the same method as shown in Figures 1, 2, 3, 4, 5 and Equation Set (1), the full coherent integration matrix M_{ci} of size $2^{(N_{db}-1)}N_bN_{be}N_{ms}$ will be obtained, which is shown in Figure 6. Each member is a vector of length N_b.

To add this coherent integration matrix M_{ci} (size $2^{(N_{db}-1)}N_bN_{be}N_{ms}$) incoherently to the former noncoherent integration matrix (size $N_bN_{be}N_{ms}$), it has first to be circularly

https://doi.org/10.1017/S0373463311000737 Published online by Cambridge University Press

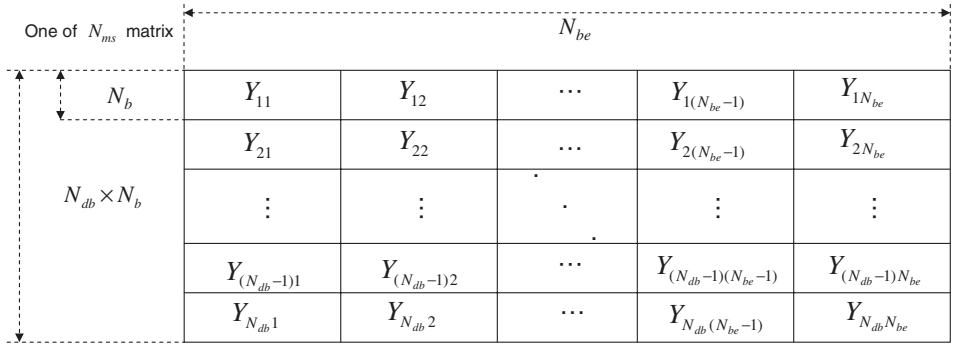


Figure 3. The matrix m_{bei} formed by taking into account that every N_{be} data lie in the same data bit.

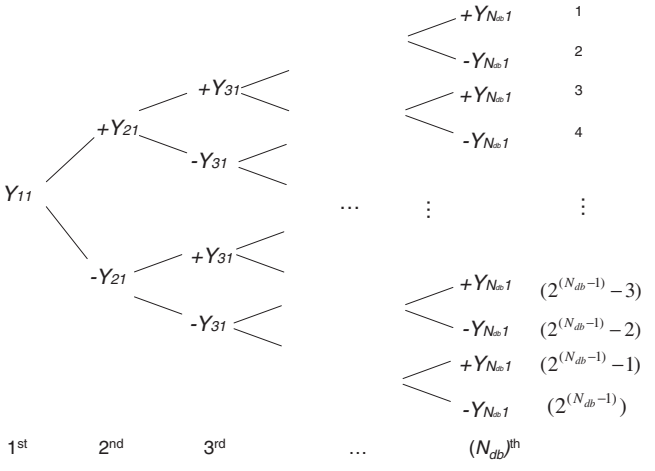


Figure 4. The APSM: coherent integration operation by taking into account all the $2^{(N_{db}-1)}$ possible navigation data combination paths.

rotated at each Doppler frequency bins (Ziedan and Garrison, 2004 [Equation 28]) to compensate the change in code length. Then among all the $2^{(N_{db}-1)}$ noncoherent integration matrices (size $N_b N_{be} N_{ms}$), preserve the one corresponding to the most likely data bit combination path and discard the others (Ziedan, 2006). The resulting new noncoherent integration matrix is still of size $N_b N_{be} N_{ms}$.

3. IMPROVED FMDBZP (IFMDBZP).

The APSM shown in Figure 4 has to search all the $2^{(N_{db}-1)}$ possible navigation data combination paths. In this Section, the OPSM is proposed to instead of the APSM. It consists of unlikely data bit combination path elimination during each coherent integration step to improve the FMDBZP.

The OPSM is based on the Viterbi algorithm theory (Ziedan, 2006). The theory is a technique that has been used in many applications. Its objective is to find the optimal

https://doi.org/10.1017/S0373463311000737 Published online by Cambridge University Press

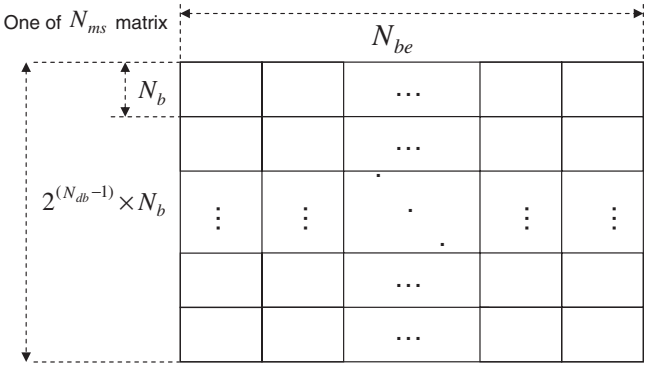


Figure 5. The partial coherent integration matrix m_{ci} formed by taking into account both the N_{be} possible navigation data bit edges and the $2^{(N_{db}-1)}$ possible navigation data combination paths.

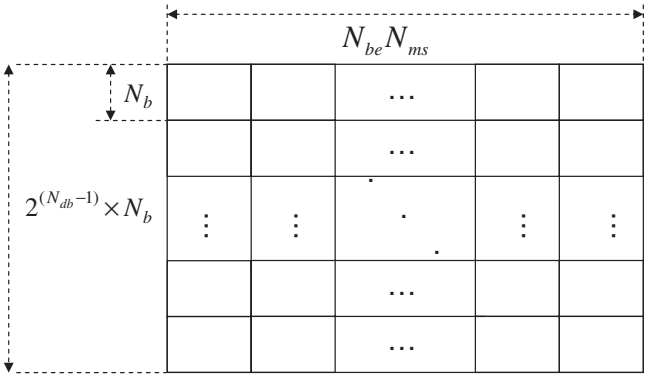


Figure 6. The full coherent integration matrix M_{ci} after all the N_{ms} columns of the primary integration matrix M_{pi} are processed.

solution to a problem, with optimal processing. It estimates the most likely sequence of data, given a set of observations. This is done by dynamic programming, which is an optimal recursive search process (Cormen et al., 1990). In general, dynamic programming can be used to solve a problem if it has the following properties:

- An optimal structure (i.e., the optimal solution to the problem consists of optimal solutions to its subproblems).
- The optimal solution can be expressed in a recursive form.

The idea behind the Viterbi algorithm is that if the most likely sequence of data between points P_0 and P_N passes through point P_i, then the subsequence from P_0 to P_i is also the most likely subsequence. The Viterbi algorithm can be used in the coherent integration operation in Figure 4 by taking into account all the $2^{(N_{db}-1)}$ possible navigation data combination paths. Thus the OPSM is formed to instead of the APSM.

Typically, a problem is formulated for the Viterbi algorithm by representing the system in question with a state diagram, then deriving a trellis graph from it. In our problem, there are 2 states, N_{db} observations and $2^{N_{db}-1}$ sequences representing all the

https://doi.org/10.1017/S0373463311000737 Published online by Cambridge University Press

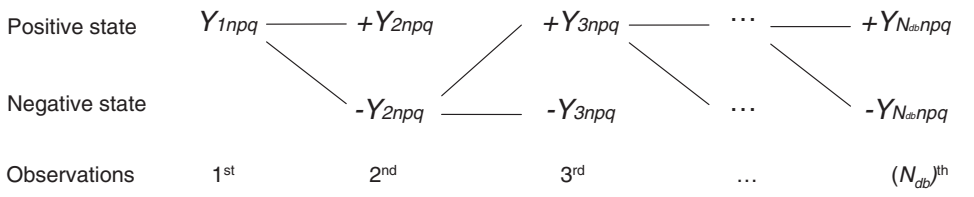


Figure 7. The trellis graph of the OPSM.

possible search paths. The optimal sequence can be found by finding the path with the maximum total weight. Although the optimal sequence can be found by searching all of the paths, such a process is impractical. Instead, the Viterbi algorithm operates recursively and keeps track only of the most likely subsequence after processing each observation. There are two paths to the same state at step i, then only the survivor path, the path with the maximum total weight up to this step, is retained. After processing each observation, there will be only one survivor path. After processing all of the observations, the survivor path with the maximum total weight will contain the most likely sequence.

The matrix m_{bei} in Figure 3 is only one of the N_{ms} matrices formed by taking into account that every N_{be} data lie in the same data bit. Let us mark the full matrix as M_{bei} and rearrange it into a four-dimension matrix. Using Y_{mnpq} represents the element, where $m = 1,2,\ldots, N_{db}$ means the mth navigation data bit, $n = 1,2,\ldots, N_b$ the nth Doppler frequency search bin, $p = 1,2,\ldots, N_{be}$ the pth navigation data bit edge, $q = 1,2,\ldots, N_{ms}$ the qth code phase search point. The OPSM should be processed for every navigation data bit at every Doppler frequency search bin, every navigation data bit edge and every code phase search point. For $m = 1$, the integration result is $I_{1npq} = Y_{1npq}$. For $m = 2,3,\ldots, N_{db}$, the integration result is determined by:

$$I_{mnpq} = \max\left(I_{(m-1)npq} \pm Y_{mnpq}\right) \tag{2}$$

For $m = 2$, if $I_{1npq} - Y_{2npq} > I_{1npq} + Y_{2npq}$, then $I_{2npq} = Y_{1npq} - Y_{2npq}$;
For $m = 3$, if $I_{2npq} + Y_{3npq} > I_{2npq} - Y_{3npq}$, then $I_{3npq} = I_{2npq} + Y_{3npq}$;
…;
For $m = N_{db}$, if $I_{(N_{db}-1)npq} - Y_{N_{db}npq} > I_{(N_{db}-1)npq} + Y_{N_{db}npq}$, then
$I_{N_{db}npq} = I_{(N_{db}-1)npq} - Y_{N_{db}npq}$.

For this example, the trellis graph of the OPSM is shown in Figure 7. $I_{N_{db}npq}$ is one of the elements of the final integration result. After all the elements are processed using Equation (2), the final coherent integration M_{oci} will be obtained. M_{oci} has size of $N_b N_{be} N_{ms}$.

To add this coherent integration matrix M_{oci} (size $N_b N_{be} N_{ms}$) incoherently to the former noncoherent integration matrix (size $N_b N_{be} N_{ms}$), first it has to be circularly rotated at each Doppler frequency bins (Ziedan and Garrison, 2004 [Equation 28]) to compensate the change in code length. The resulted new noncoherent integration matrix is still of size $N_b N_{be} N_{ms}$.

Because the OPSM needs comparison between the positive and negative states, it is a little more complicated than the APSM. But it does not have to search all the paths

https://doi.org/10.1017/S0373463311000737 Published online by Cambridge University Press

and overall it can reduce the calculation operations and save memory space. Furthermore, then the number of data bits N_{db} can be increased appropriately, that is to increase the coherent integration time, to increase the post correlation signal to noise ratio without increasing the overall processing time.

4. COMPLEXITY ANALYSIS AND COMPARISON.

The comparison between the APSM and the OPSM will be processed from the aspects of calculation operations and memory space. Because the operations before considering the possible data bit combination paths are the same, the comparison between the two methods starts from where they are different.

It should be pointed out that the primary integration matrix M_{pi} is a complex matrix. All the elements are complex numbers. For two complex numbers, a complex number multiplication equals to four real number multiplications and two real number additions; a complex number addition equals to two real number additions; a comparison operation limits between two real numbers. The addition and subtraction will not be distinguished because they need the same time to process. In this paper, the real number operations are applied to count the operations for both the APSM and the OPSM.

4.1. *The All Paths Search Method (APSM).*

- At the coherent integration step, when taking into account the $2^{(N_{db}-1)}$ navigation data bit combinations as shown in Figure 4, the number of real number additions is:

$$C^a_{CI_add} = 2\left(2+4+\cdots+2^{(N_{db}-1)}\right)N_b N_{be} N_{ms} = 4\cdot\left(2^{N_{db}-1}-1\right)N_b N_{be} N_{ms} \quad (3)$$

  Taking into account the storage of complex numbers, the needed maximum real number memory space is:

$$S^a_{CI} = 2\cdot 2^{(N_{db}-1)} N_b N_{be} N_{ms} = 2^{N_{db}} N_b N_{be} N_{ms} \quad (4)$$

- For each element of the full coherent integration matrix M_{ci}, square the real and image parts and then add them. The numbers of real number multiplications and additions are, respectively:

$$C^a_{IQ_multi} = S^a_{CI} = 2^{N_{db}} N_b N_{be} N_{ms} \quad (5)$$

$$C^a_{IQ_add} = S^a_{CI}/2 = 2^{(N_{db}-1)} N_b N_{be} N_{ms} \quad (6)$$

- At the noncoherent integration step, add the current coherent integration matrix (real numbers) to the former noncoherent integration matrix (real numbers). The number of real number additions is:

$$C^a_{NI_add} = S^a_{CI}/2 = 2^{(N_{db}-1)} N_b N_{be} N_{ms} \quad (7)$$

- Keep the result (real numbers) corresponding to the most likely data bit combination path. The number of comparisons is:

$$C^a_{NI_comp} = S^a_{CI}/2 - N_b N_{be} N_{ms} = \left(2^{(N_{db}-1)}-1\right)N_b N_{be} N_{ms} \quad (8)$$

  Assume that a real number addition and a comparison need the same time to process. Then for the APSM, the total numbers of real number multiplications,

https://doi.org/10.1017/S0373463311000737 Published online by Cambridge University Press

additions and maximum memory space are:

$$C^a_{Total_multi} = 2^{N_{db}} N_b N_{be} N_{ms} \tag{9}$$

$$C^a_{Total_add} = \left(7{\cdot}2^{N_{db}-1} - 5\right) N_b N_{be} N_{ms} \tag{10}$$

$$S^a_{Total} = 2^{N_{db}} N_b N_{be} N_{ms} \tag{11}$$

4.2. *The Optimal Path Search Method (OPSM).* The analysis of this method is based on $N_{db} \geqslant 2$.

- At the coherent integration step, when only taking into account the optimal solution among all the possible navigation data bit combination paths as shown in Figure 7, the number of real number additions is:

$$C^o_{CI_add} = 2 \cdot 2(N_{db} - 1) \cdot N_b N_{be} N_{ms} = 4(N_{db} - 1) N_b N_{be} N_{ms} \tag{12}$$

Take into account the storage of complex number, the needed maximum real number memory space is:

$$S^o_{CI} = 2 \cdot 2 N_b N_{be} N_{ms} = 4 N_b N_{be} N_{ms} \tag{13}$$

For each element of the full coherent integration matrix M_{oci}, square the real and image parts and then add them. The numbers of real number multiplications and additions are, respectively:

$$C^o_{CI_multi} = 4(N_{db} - 1) N_b N_{be} N_{ms} \tag{14}$$

$$C^o_{CI_add} = 2(N_{db} - 1) N_b N_{be} N_{ms} \tag{15}$$

The number of comparisons is:

$$C^o_{CI_comp} = (N_{db} - 1) N_b N_{be} N_{ms} \tag{16}$$

- At the noncoherent integration step, add the current coherent integration matrix (real numbers) to the former noncoherent integration matrix (real numbers). The number of real number additions is:

$$C^o_{NI_add} = N_b N_{be} N_{ms} \tag{17}$$

Assume that a real number addition operation and a real number comparison operation need the same time to process. Then for the APSM the total numbers of real number multiplications, additions and maximum memory space are:

$$C^o_{Total_multi} = 4(N_{db} - 1) N_b N_{be} N_{ms} \tag{18}$$

$$C^o_{Total_add} = (7N_{db} - 6) N_b N_{be} N_{ms} \tag{19}$$

$$S^o_{Total} = 4 N_b N_{be} N_{ms} \tag{20}$$

4.3. *The Comparison between the OPSM and APSM.* The ratios of the OPSM and the APSM for the total numbers of real number multiplications, additions and maximum memory space are, respectively:

$$\gamma_{C_multi} = \frac{C^o_{Total_multi}}{C^a_{Total_multi}} = \frac{4(N_{db} - 1)}{2^{N_{db}}} \tag{21}$$

https://doi.org/10.1017/S0373463311000737 Published online by Cambridge University Press

Table 1. The comparison between (a) the APSM and (b) the OPSM.

	Operations ($\times N_b N_{be} N_{ms}$)						Memory space ($\times N_b N_{be} N_{ms}$)		
N_{db}	× (a)	× (b)	γ_{C_multi}	+ (a)	+ (b)	γ_{C_add}	(a)	(b)	γ_S
1	2	2	100·00%	2	2	100·00%	2	2	100·00%
2	4	4	100·00%	9	8	88·89%	4	4	100·00%
3	8	8	100·00%	23	15	65·22%	8	4	50·00%
4	16	12	75·00%	51	22	43·14%	16	4	25·00%
5	32	16	50·00%	107	29	27·10%	32	4	12·50%
6	64	20	31·25%	219	36	16·44%	64	4	6·25%
7	128	24	18·75%	443	43	9·71%	128	4	3·13%
8	256	28	10·94%	891	50	5·61%	256	4	1·56%
9	512	32	6·25%	1787	57	3·19%	512	4	0·78%
10	1024	36	3·52%	3579	64	1·79%	1024	4	0·39%

$$\gamma_{C_add} = \frac{C^o_{Total_add}}{C^a_{Total_add}} = \frac{7N_{db} - 6}{7 \cdot 2^{N_{db}-1} - 5} \tag{22}$$

$$\gamma_S = \frac{S^o_{Total}}{S^a_{Total}} = \frac{4}{2^{N_{db}}} = \frac{1}{2^{N_{db}-2}} \tag{23}$$

From Equations (21), (22) and (23), it can be concluded that the ratios are only related with N_{db} and they decrease as N_{db} increases. Numerical examples of $N_{db} = 1, 2, \ldots, 10$ are shown in Table 1. Compared to the APSM, the OPSM has no improvement for $N_{db} = 1$, slight improvement (88·89% addition operations) for $N_{db} = 2$, small improvement (65·22% addition operations and 50% memory space) for $N_{db} = 3$, obvious improvement (75% multiplication operations, 43·14% addition operations and 25% memory space) for $N_{db} = 4$, and larger and larger improvement as N_{db} increases. For $N_{db} \geqslant 2$, no matter how large N_{db} is, the maximum memory space needed is constant.

5. NUMERICAL RESULTS.

In this section, numerical simulations are run to demonstrate the performances of the FMDBZP algorithm which applies the APSM and the IFMDBZP algorithm which applies the proposed OPSM, respectively.

For signal generators, it generates the baseband GPS signal and sets the sampling frequency $f_s = 5{\cdot}7$ MHz, the Doppler frequency shift 200 Hz. The C/A code phase is 932·5 chips. The Carrier-to-Noise density ratio C/N_0 varies for different signal data. For signal acquisition using both FMDBZP and IFMDBZP algorithms, it sets the coherent integration time $T_I = 80$ ms (that is $N_{db} = 4$), the number of possible navigation bit edge $N_{be} = 10$, the number of noncoherent integration $N_{T_I} = 30$, and the Doppler frequency shift search range $[-500, 500]$ Hz. Then it can be calculated that the number of Doppler frequency search bins is 80, the number of code phase search points is 5700, and the size of the noncoherent integration matrix should be $80 \times 10 \times 5700 = 456\,000$.

The performances of both FMDBZP and IFMDBZP algorithms, which are shown in Figure 8, are obtained in terms of probability of detection using Monte Carlo

https://doi.org/10.1017/S0373463311000737 Published online by Cambridge University Press

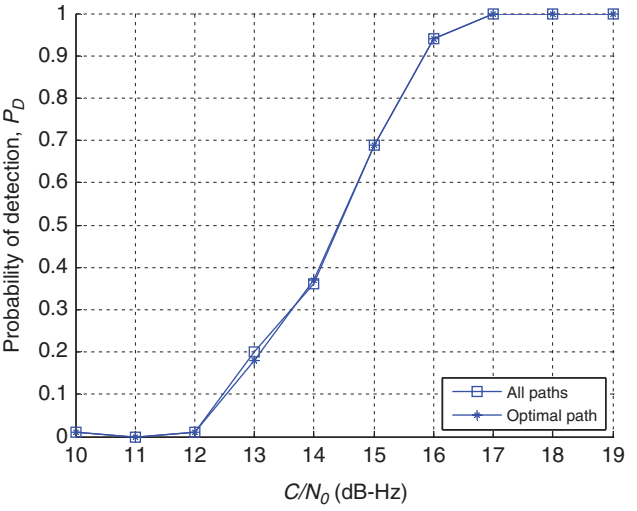


Figure 8. The final noncoherent integration result.

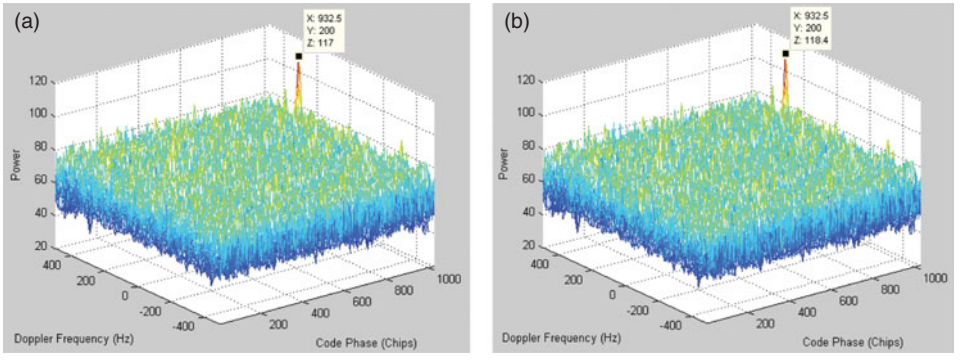


Figure 9. The final noncoherent integration result (a) the APSM (b) the OPSM.

simulation method. The probability of detection P_D is defined as the probability that the maximum of the final noncoherent integration matrix occurs in the correct cell when a signal is present. The results show that the IFMDBZP algorithm has no performance loss compared to the FMDBZP algorithm. So the proposed OPSM provides a design tool to reduce computation and save memory space without suffering any performance loss.

For $C/N_0 = 16$ dB-Hz, one successful acquisition using the APSM and the OPSM are shown in Figure 9, Figure 10, and Figure 11, respectively.

Both methods give the correct acquisition results. But according to the analysis in Table 1 for $N_{db} = 4$, the OPSM only needs 75·00% real number multiplications, 43·14% real number additions and 25·00% real number memory space comparing to the APSM during the acquisition process. The examples verified that the proposed OPSM can save processing time without suffering any loss compared with the APSM.

https://doi.org/10.1017/S0373463311000737 Published online by Cambridge University Press

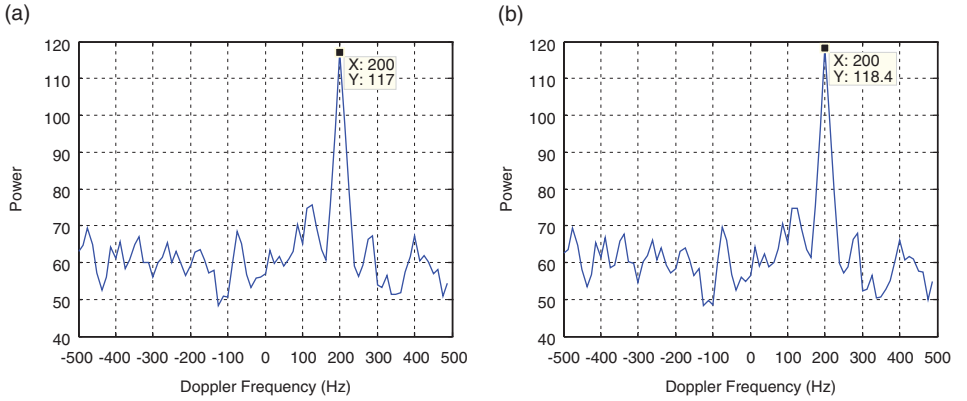


Figure 10. Power versus Doppler frequency shift at the correct code phase (a) the APSM (b) the OPSM.

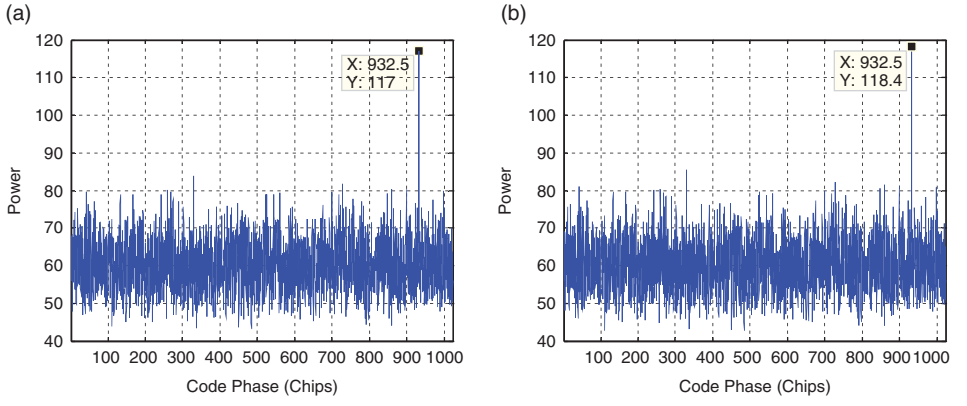


Figure 11. Power versus code phase at the correct Doppler frequency shift (a) the APSM (b) the OPSM.

6. CONCLUSIONS. Acquisition of weak GPS signals requires long coherent integration time. The Double-Block Zero Padding (DBZP) algorithm can use a long pre-detection integration time without any assisting information to acquire weak GPS signals. The Modified Double Block Zero Padding (MDBZP) algorithm accounts for the Doppler Effect on the code duration. It also develops the Doppler frequency bin elimination method during the noncoherent integration steps. The Fast Modified Double-Block Zero Padding (FMDBZP) algorithm based on the MDBZP algorithm eliminates the redundant Fast Fourier Transform (FFT) computations when considering possible data bit combination paths and bit edge positions. Although the computation burden of FMDBZP is lower than that of MDBZP, it may still be very heavy.

In this paper, the *Optimal Path Search Method (OPSM)* is proposed to improve the FMDBZP to reduce further the computation processing and save memory space during the coherent integration steps. The analyses show that the proposed OPSM can reduce the computation calculations and save memory space without suffering any loss compared to the *All Path Search Method (APSM*). And the longer the coherent

https://doi.org/10.1017/S0373463311000737 Published online by Cambridge University Press

integration time is, the more benefit one can get from the proposed method. The simulation examples also show that the Improved FMDBZP (IFMDBZP) algorithm using the proposed OPSM has the same acquisition performance as the one using the APSM.

It should be pointed out that, the proposed OPSM can only be used based on FMDBZP but not DBZP or MDBZP. This is because during the coherent integration steps, the FMDBZP eliminates the redundant FFT computations by doing FFT operations (Heckler and Garrison, 2009) before taking into account all the possible data bit combination paths.

REFERENCES

Cormen, T., Leisersen, C. (1990). *Introduction to Algorithms*. MIT Press, Boston.

He, S. and Torkelson, M. (1996). Computing Partial DFT for Comb Spectrum Evaluation. *IEEE Signal Processing Letters*, **3**(6), 173–175.

Heckler, G. W. and Garrison, J. L. (2009). Implementation and Testing of an Unaided Method for the Acquisition of Weak GPS C/A Code Signals. *Navigation*, **56**(4), 241–259.

Lin, D. M. and Tsui, L. B. Y. (2000). Comparison of Acquisition Methods for Software GPS Receiver. *Proceedings of ION GPS-2000*.

Lin, D. M., Tsui, L. B. Y. and Howell, D. (1999). Direct P(Y)-Code Acquisition Algorithm for Software GPS Receivers. *Proceedings of ION GPS-1999*.

Misra, P. and Enge, P. (2001). *Global Positioning System: Signals, Measurements, and Performance*. Ganga-Jamuna Press.

Parkinson, B. and Spilker, J. (1996). *Global Positioning System: Theory and Applications*. American Institute of Aeronautics and Astronautics.

Sorensen, H. V. and Burrus, C. S. (1993). Efficient Computation of the DFT with Only a Subset of Input or Output Points. *IEEE Transactions on Signal Processing*, **41**(3), 1184–1200.

Tsui, J. B. Y. (2004). *Fundamentals of Global Positioning System Receivers: A Software Approach (Second Edition)*. Wiley-Interscience.

Van Nee, D. J. R. and Coenen, A. J. R. M. (1991). New Fast GPS Code-Acquisition Technique Using FFT. *IEEE Electronics Letters*, **27**(2).

Wild, R. de, Nieuwkerk, L. R., and Sittruyen, J. S van. (1987). Method for Partial Spectrum Computation. *IEE Proceedings, Part F, Communications Radar and Signal Processing*, **134**(7), 659–666.

Yang, C. (2001). Zoom, Pruning, and Partial FFT for GPS Signal Tracking. *Proceedings of the Institute of Navigation's ION NTM 2001*.

Ziedan, N. I. (2006). *GNSS Receivers for Weak Signals*. Artech House, London.

Ziedan, N. I. and Garrison, J. L. (2004). Unaided Acquisition of Weak GPS Signals Using Circular Correlation or Double-Block Zero Padding. *Position Location and Navigation Symposium (PLANS)*. 461–470.

https://doi.org/10.1017/S0373463311000737 Published online by Cambridge University Press