Two-Stage Method for Joint Time Delay and Doppler Shift Estimation

Ran Tao, Wei-Qiang Zhang, and En-Qing Chen

Abstract—The focus of this research is to provide a fast and precise method for joint time delay and Doppler shift estimation. The main procedure is divided into two stages. In the first stage, the pre-weighted Zoom FFT method is used for fast computing the ambiguity function and the quadratic surface fitting method is used for coarse estimation. In the second stage, the values near the coarse estimates are calculated and quadratic surface fitting method is used again for fine estimation. The two-stage method reduces the computational load without losing the precision. Simulation and experimental results are included to demonstrate the effectiveness of the proposed method.

Index Terms—ambiguity function, time delay, Doppler shift, quadratic surface fitting.

I. INTRODUCTION

In radar and sonar systems, time delay and Doppler shift are two important parameters. The time delay may occur due to different distances, while the Doppler shift may result from different velocity. Thus, the time delay and Doppler shift contain the range and relative velocity information of the moving targets, and they can be used for target localization and tracking [1].

In many cases, the time delay and Doppler shift have to be estimated from two measurements observed by two spatially separated sensors. For this estimation, a well-known tool is the cross ambiguity function (CAF), which can be viewed as a time delay and Doppler shift two-dimensional (2-D) correlation. However, the required accuracy may lead to a need for computation over long data sets that may become a challenge for real-time processing [2]. So there is a great need to produce fast algorithms.

Several algorithms have been designed for joint time delay and Doppler shift estimation. Some of them were concerned with fast computing the ambiguity function, such as the algorithms based on Fast Fourier Transform (FFT), fractional Fourier transformation (FRFT) and other methods [2]–[5]. These algorithms reduce the computational load, and can be used for near real-time processing. Other algorithms, such as that based on higher-order statistics (HOS), wavelets, and the adaptive methods have also been proposed [6]–[9]. In some special cases, these algorithms outperform the traditional methods.

In this paper, we propose an efficient method by fast computing the traditional CAF. The preliminaries are described in section II. The pre-weighted Zoom FFT method is developed in section III and the quadratic surface fitting method is presented in section IV. Section V describes the detailed steps of the two-stage method. In section VI, the computational complexities of the proposed algorithms are analyzed. Section VII offers the performance analysis based on Monte Carlo simulation and section VIII gives some experimental results. Finally, in section IX, the conclusion is given.

II. PRELIMINARIES

In many applications, such as the passive bistatic radar shown in Fig. 1, the signal of interest (SOI) can be received by two spatially separated receivers. Assuming operation at baseband, and the absence or prior removal of multi-path interference, the SOI \( s(t) \) is narrow-band, and the two measurements can be expressed as [2], [10] \(^1\)

\[
x(t) = s(t) + w_1(t)
\]

\[
y(t) = A_0 s(t - D) \exp\{j(2\pi f_d t + \varphi)\} + w_2(t)
\]

where \( A_0 \) is an attenuation constant, \( \varphi \) is a constant phase. The parameters \( D \) and \( f_d \) denote the time delay and Doppler shift between the two measurements, respectively. \( w_1(t) \) and \( w_2(t) \) are stationary, zero-mean, uncorrelated white Gaussian noises. The time delay and Doppler shift have to be estimated from finite length records of \( x(t) \) and \( y(t) \).

The cross-ambiguity function (CAF) is an extensively used tool for this estimation. The CAF of the narrow-band signals

\(^1\)In fact, the direct signal and its multi-path reflections received by the Receiver 2 is a major problem in this configuration. The separation can be implemented by some technologies such as beam-forming [11] and adaptive direct signal cancellation [12], [13].
\[ x(t) \text{ and } y(t) \text{ is defined as } \mathcal{A}(\tau, f) = \int_0^T x^*(t)y(t+\tau)\exp(-j2\pi ft)dt \] \hspace{1cm} (3)

where "*" denotes the complex conjugation, and \( \tau \) and \( f \) are the time delay and Doppler shift variables. If not considering the effect of noises, \( |\mathcal{A}(\tau, f)| \) will peak at the true values of the (time delay, Doppler shift) pairs, i.e.,

\[ (D, f_d) = \arg \max_{\tau, f} |\mathcal{A}(\tau, f)| \] \hspace{1cm} (4)

So according the peak location of \( |\mathcal{A}(\tau, f)| \), we can get the time delay and Doppler shift estimation.

In the practical systems, we always get the discrete samples of \( x(t) \) and \( y(t) \), which assumed as \( \{x(nT_s), n = 0, 1, \ldots, N-1\} \) and \( \{y(nT_s), 0, 1, \ldots, N-1\} \), where \( T_s \) is the sampling period. In this case, the definition (3) will be become

\[ A(d\Delta\tau, f) = \frac{1}{N} \sum_{n=0}^{N-1} x^*(nT_s)y(nT_s+d\Delta\tau)\exp(-j2\pi fnT_s) \] \hspace{1cm} (5)

where \( d \) is the discrete time delay index and \( \tau = d\Delta\tau \), and \( \Delta\tau \) is the time delay resolution which is often selected as multiple of \( T_s \) for convenience.

III. Fast Algorithms for Computing the Discrete Ambiguity Function

The computation of the ambiguity function is very time-consuming. In this section, start from the FFT method, we will develop the pre-weighted Zoom FFT (PWZFFT) method for fast computing the ambiguity function step by step.

A. FFT method

We can see from (5) that for each \( d \), the CAF calculation can be regarded as operating on a product series \( r_N(n;d) \):

\[ r_N(n; d) = x^*(nT_s)y(nT_s+d\Delta\tau) \] \hspace{1cm} (6)

The rest operating on \( r_N(n;d) \) can be taken as a discrete Fourier transform (with \( n \) as the variable) and it can be carried on by FFT algorithm [2].

\[ A(d\Delta\tau, k\Delta f) = \frac{1}{N} \text{FFT}\{r_N(n;d)\} \] \hspace{1cm} (7)

where \( k \) is the discrete Doppler shift index and \( f = k\Delta f \), and \( \Delta f = f_s/N \) is the frequency resolution. In this way, the computational complexity can be lowered.

B. Zoom FFT (ZFFT) method

In FFT method, we need perform \( N \)-point FFT. But as said in section I, the real systems may require very long data sets, so \( N = 10^4 \) to \( 10^6 \) data samples are often involved. In this case, \( N \)-point FFT is usually impractical. On the other hand, FFT calculates the bandwidth of \( -f_s/2 \) to \( f_s/2 \) which is often much greater than the domain of uncertainty of Doppler shift. So there is a lot of waste in this method.

In fact, Zoom FFT can calculate a narrow band of the total spectrum with the same or higher frequency resolution [14], so it can be used to solve the problem mentioned above. Here we assume the frequency resolution needed is the same as that of FFT method, i.e., \( \Delta f = f_s/N \), and the frequency domain of interest is \( -f_s/(2D) \) to \( f_s/(2D) \), where \( D = N/M \). We can perform the main procedures of ZFFT method as follows. Firstly, let the \( N \)-point product series \( r_N(n;d) \) pass a lowpass filter (LPF) and we will get \( r'_N(n;d) \). Secondly, downsample \( r'_N(n;d) \) by a factor of \( D \), we will get a \( M \)-point series \( r_M(m;d) \). Finally perform \( M \)-point FFT on \( r_M(m;d) \), we can obtain the CAF.

Assume the LPF order is \( N_h - 1 \) and the coefficients are \( \{h(l), l = 0, 1, \ldots, N_h - 1\} \). Then the output of the LPF can be written as

\[ r'_N(n;d) = \sum_{l=0}^{N_h-1} r_N(n-l;d)h(l) \]

\[ = \sum_{l=0}^{N_h-1} r_N(n-l;d)h'(N_h - 1 - l) \]

\[ = \sum_{l=0}^{N_h-1} r_N(n-N_h + 1 + l;d)h'(l) \] \hspace{1cm} (8)

where \( h'(l) \) is the reverse order coefficients of \( h(l) \), i.e., \( h'(N_h - 1 - l) = h(l) \), for \( l = 0, 1, \ldots, N_h - 1 \). Let \( r'_N(n;d) \) be downsampled by a factor of \( D \), we obtain

\[ r_M(m;d) = r'_N(md; d) \]

\[ = \sum_{l=0}^{N_h-1} r_N(mD + l;d)h'(l) \] \hspace{1cm} (9)

Perform \( M \)-point FFT on \( r_M(m;d) \) for every \( d \), we can get the CAF. If we select suitable LPF, the spectral aliasing can be confined on a negligible level and a good result will be obtained. In this method, \( N \)-point FFT is substituted by \( M \)-point FFT, so the computational complexity can be lowered evidently.

C. Pre-Weighted Zoom FFT (PWZFFT) method

In ZFFT method, the product series \( r_N(n;d) \) is downsampled by a factor of \( D \) after filtering, so the passband cutoff frequency of LPF should be set at \( 1/D \) (normalized frequency). Generally speaking, we should use higher order LPF to get good filtering results. But in this situation, as shown in section II, \( x(t) \) and \( y(t) \) have identical frequency components which are derived from \( s(t) \), so after the operation in (6), these components will be eliminated. Thus \( r_N(n;d) \) will be a narrow-band signal with main frequency \( f_d \). This means that the performance of LPF is not very important in
some sense and we can select small order LPF. Based on this point, we develop another method to further reduce the computational load.

In ZFFT method, for each \( d \), the extra digital filtering process is needed. If we divide the filtering process into two steps, i.e., weighting and summing, (9) can be rewritten as

\[
r_M(m; d) = \sum_{l=0}^{Nh-1} x^*((mD+l)T_s)y((mD+l)T_s+d\Delta \tau)h'(l)
\]

where \( x^*((mD+l)T_s) = x^*((mD+l)T_s) = y^*((mD+l)T_s) \). Note that multiplying \( x^*((mD+l)T_s) \) by \( h'(l) \) is the weighting process. We do it first because it is independent of \( d \) and so that it can be done only once for the same \( m \) and all the different \( d \). By this method, we can save many complex multiplication operations. Because the weighting process is performed before the summing process, we call it pre-weighted Zoom FFT method.

The pre-weighted Zoom FFT method is summarized as follows.

- **Step 1**: Segment the data into \( M \) segments of \( N_h \) samples each with/without data overlapping, we get \( x^{(m)}(IT_s) \) and \( y^{(m)}(IT_s) \): for \( m = 0, 1, \ldots, M - 1 \) and \( l = 0, 1, \ldots, N_h - 1 \).

\[
\begin{align*}
x^{(m)}(IT_s) &= x((mD+l)T_s) \\
y^{(m)}(IT_s) &= y((mD+l)T_s)
\end{align*}
\]

- **Step 2**: Weight \( x^{(m)}(IT_s) \) by the coefficients \( h'(l) \): for each \( m \) and each \( l \).

\[
x^{(m)}(IT_s) = x^{(m)}(IT_s)h'(l)
\]

- **Step 3**: Calculate \( r_M(m; d) \) by using \( x^{(m)}(IT_s) \) and \( y^{(m)}(IT_s + d\Delta \tau) \): for each discrete \( d \) of interest.

\[
r_M(m; d) = \sum_{l=0}^{N_h-1} x^{(m)}(IT_s)y^{(m)}(IT_s+d\Delta \tau)
\]

- **Step 4**: Perform FFT on \( r_M(m; d) \) with \( d \) as a parameter and \( m \) as a variable.

\[
A(d, k) = \frac{1}{M} \text{FFT}\{r_M(m; d)\}
\]

\[
= \frac{1}{M} \sum_{m=0}^{M-1} r_M(m; d) \exp(-j2\pi km/M)
\]

where \( \tau = d\Delta \tau \), \( f = k\Delta f \), \( \Delta f = f_s/(MD) = f_s/N \).

**IV. Quadratic Surface Fitting Method**

In the previous section, we have discussed the fast algorithms for computing the ambiguity on discrete grids. After that, we can perform 2-D peak search and get the time delay and Doppler shift estimates. In fact, this will lead the estimates stepwise, while the true time delay and Doppler shift is continuous. Stein [2] have noticed this problem and presented a three-point polynomial fit method. However, this method is too simple to match all the complicated cases. We will develop it and present a quadratic surface fitting method in this section.

As shown in Fig. 2, assume the peak of interest is approximately located at \( (d\Delta \tau, k\Delta f) \) in the discrete ambiguity-plane. It is more convenient to translate the origin of the coordinates to \( (d\Delta \tau, k\Delta f) \). In the new coordinates, suppose the \( A(\tau, f) \) is a quadratic surface \(^3\), i.e.,

\[
|A(\tau, f)| = c_1\tau^2 + c_2\tau f + c_3f^2 + c_4\tau + c_5f + c_6
\]

(11)

In order to determine the coefficients, we can select nine points, which include the peak point and eight points around it. The values of the points are assumed as \( A_{--}, A_{-0}, A_{+0}, A_{0-}, A_{00}, A_{-+}, A_{+0}, A_{0+}, A_{++} \), and they are shown in Fig. 2. Substitute the nine points to the surface equation, we can get (12) at the top of the next page.

Rewrite (12) in matrix form

\[
Mc = a
\]

(13)

where

\[
M = \begin{bmatrix}
1 & 1 & 1 & -1 & -1 & 1 \\
0 & 0 & 1 & 0 & -1 & 1 \\
1 & -1 & 1 & 1 & -1 & 1 \\
1 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & -1 & 0 & 1 \\
1 & -1 & 1 & -1 & 1 & 1 \\
0 & 0 & 1 & 0 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 \\
\end{bmatrix}
\]

\[
c = [c_1(\Delta \tau)^2, c_2(\Delta \tau)(\Delta f), c_3(\Delta f)^2, c_4(\Delta \tau), c_5(\Delta f), c_6]^T
\]

\[
a = [A_{--}, A_{0-}, A_{+0}, A_{00}, A_{-+}, A_{+0}, A_{0+}, A_{++}]^T
\]

\(^3\)In local, just as the curve can be seen as line, a general surface can be modeled as a quadratic surface. This will not introduce much error.
Equation (13) is overdetermined linear equations. Its least square (LS) solution is

\[ \mathbf{c} = \mathbf{M}^\dagger \mathbf{a} \]  

where \( \mathbf{M}^\dagger \) is the pseudoinverse of \( \mathbf{M} \). From (14), we can obtain (15) at the top of the next page.

On the other hand, from (11), we can easily find that the coordinates of the peak satisfy

\[
\begin{align*}
\frac{\partial A(\tau, f)}{\partial \tau} &= 2c_1 \tau + c_2 f + c_4 = 0 \\
\frac{\partial A(\tau, f)}{\partial f} &= c_2 \tau + 2c_3 f + c_5 = 0
\end{align*}
\]

Solve this equations, we obtain

\[
\begin{align*}
\tau &= \frac{2c_3 c_4 - c_2 c_5}{c_2^2 - 4c_1 c_3} \quad \text{(17a)} \\
f &= \frac{2c_1 c_5 - c_2 c_4}{c_2^2 - 4c_1 c_3} \quad \text{(17b)}
\end{align*}
\]

Return to the original coordinates, the time delay and Doppler shift estimates will be

\[
\begin{align*}
\hat{\tau} &= d \Delta \tau + \frac{2c_3 c_4 - c_2 c_5}{c_2^2 - 4c_1 c_3} \\
\hat{f} &= k \Delta f + \frac{2c_1 c_5 - c_2 c_4}{c_2^2 - 4c_1 c_3}
\end{align*}
\]

V. COMPUTATIONAL ALGORITHM

In this section, we will discuss the computational algorithm. In the PWZFFT method, in order to avoid aliasing, the lowpass filtering is needed. But considering the computational load, the filter order cannot be set very high, and this will introduce some errors. Although the errors are very low, they may affect the parameter estimation. In addition, in order to reduce the computational complexity, the step size may be relatively large when calculating the discrete ambiguity function. This may lead the surface of the ambiguity function not accord with the quadratic surface well. In this case, the time delay and Doppler shift estimation is not accurate, even though we use the quadratic surface fitting method.

To solve this problem, we can calculate nine points near the approximate estimates with little step and get the accurate estimates using surface fitting method again. Because the nine points are calculated by using (5) directly and the steps between them are little, this will introduce little error.

We can summarize the two-stage method as follows.

- **Stage I. Coarse estimation**
  - Step 1: Use PWZFFT method to calculate the ambiguity function in the discrete grid points with relatively large steps.
  - Step 2: Search the peak and get eight points around it in the time delay and Doppler shift plane.
  - Step 3: Use the quadratic surface fitting method to get the coarse time delay and Doppler shift estimation.

- **Stage II. Fine estimation**
  - Step 4: Calculate nine points value near estimated point in Stage I by relatively small steps.
  - Step 5: Get the accurate time delay and Doppler shift estimation by surface fitting method using the nine points obtained in Step 4.

VI. ALGORITHM ANALYSIS

In this section, we analyze the computational complexity of proposed method. Because almost all the computational loads lie in the first stage, we only analyze this stage. First, we compare the computational complexity of direct calculation, FFT method, ZFFT method and PWZFFT method. We use the number of complex multiplications (NCM) as a figure of merit. In the analysis, we assume the number of discrete \( d \) of interest is \( N \tau \) and the number of discrete \( f \) of interest is \( M \). We also assume that the NCM for \( N \)-point FFT is \( N/2 \log_2 N \).

From (5), we can know easily that the NCM for direct calculation is \( 2N \tau N M \). Usually, \( N \) is so great that the NCM will be enormous and this method do not feasible for many real systems.

In FFT method, for each \( d \), the NCM for obtaining \( r_N(n; d) \) is \( N \) and for \( N \)-point FFT is \( (N/2) \log_2 N \). Thus the total NCM is \( N \tau [N + (N/2) \log_2 N] \).

In ZFFT method, for each \( d \), besides the NCM for obtaining, the NCM for lowpass filtering is \( M N_h \) and for \( M \)-point FFT is \( (M/2) \log_2 M \). So the total NCM is \( N \tau [N + M N_h + (M/2) \log_2 M] \). Note that \( M \ll N \), the NCM of ZFFT method will be less than that of FFT method.
\[ c_1 = \frac{1}{6(\Delta t)^2} (A_- - 2A_0 + A_+ + A_0 + 2A_0 - 2A_{00} + A_{00} + 2A_{0} + A_{0}) \]  
(15a)

\[ c_2 = \frac{1}{4(\Delta t)(\Delta f)} (A_- - A_+ - A_+ + A_+ + A_+) \]  
(15b)

\[ c_3 = \frac{1}{6(\Delta f)^2} (A_- + A_0 + A_0 - 2A_0 + 2A_0 - 2A_0 + A_+ + A_+ + A_+) \]  
(15c)

\[ c_4 = \frac{1}{6(\Delta t)} (-A_- + A_0 + A_0 - A_+ - A_+ + A_+) \]  
(15d)

\[ c_5 = \frac{1}{6(\Delta f)} (-A_- - A_0 - A_+ + A_+ + A_0 + A_+) \]  
(15e)

![Graph showing number of complex multiplications for different methods](image)

**TABLE I: Number of complex multiplications for each method**

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>NCM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct Calculation</td>
<td>(2N_N M)</td>
</tr>
<tr>
<td>FFT Method</td>
<td>(N_d [N + (N/2) \log_2 N])</td>
</tr>
<tr>
<td>ZFFT Method</td>
<td>(N_d [N + N_N + (N/2) \log_2 M])</td>
</tr>
<tr>
<td>PWZFFT Method</td>
<td>(N_d [M N_N + (N/2) \log_2 M] + MN_N)</td>
</tr>
</tbody>
</table>

**VII. PERFORMANCE ANALYSIS**

In this section, we will use placeMonte Carlo simulation to analysis the performance of proposed algorithm. For simplicity, we ignore the noise \(w_1(t)\) in the measurement \(x(t)\). In this case, the Cramer-Rao lower bounds (CRLB) for time joint time delay and Doppler shift estimation is given by [1], [15]

\[ \text{var}(\hat{D}) = \frac{1}{2SNR} \beta^2 - \alpha^2 \]  
(20a)

\[ \text{var}(\hat{\delta}) = \frac{1}{8\pi^2 SNR} \beta^2 \delta^2 - \alpha^2 \]  
(20b)

where

\[ \alpha = \bar{\omega}_t - \bar{\omega} \]  
(21a)

\[ \beta^2 = \bar{\omega}_t^2 - \bar{\omega}^2 \]  
(21b)

\[ \delta^2 = \bar{\tau}^2 - \bar{\tau}^2 \]  
(21c)

\[ \text{SNR} = \frac{E}{N_0} \]  
(21d)

is the signal-energy to noise-density ratio (SNR).

\[ E = A_0^2 \int_{-\infty}^{\infty} |s(t)|^2 dt \]  
(22a)

\[ \bar{\omega}_t = \frac{A_0^2}{E} \int_{-\infty}^{\infty} t |s(t)|^2 dt \]  
(22b)

\[ \bar{\tau}^2 = \frac{A_0^2}{E} \int_{-\infty}^{\infty} t^2 |s(t)|^2 dt \]  
(22c)

\[ \bar{\omega} = \frac{A_0^2}{E} \int_{-\infty}^{\infty} s^*(t) \dot{s}(t) dt \]  
(22d)

\[ \bar{\omega}^2 = \frac{A_0^2}{E} \int_{-\infty}^{\infty} |\dot{s}(t)|^2 dt \]  
(22e)

\[ \bar{\omega}_t = \frac{A_0^2}{E} \int_{-\infty}^{\infty} t s^*(t) \dot{s}(t) dt \]  
(22f)

\[ \hat{s}(t) = \frac{ds(t)}{dt} \]  
(22g)

\^\footnote{Note that in this example, we suppose \( s^* \) is a fixed value. In this case, in order to get the same frequency domain of interest, \( M \) must be increased with \( N \).}
function, respectively. In our experimental system, the two measurements, \( x \) and \( y \), are compared with the CRLB. We use the Monte Carlo simulation to get the variance of the time delay and Doppler shift and compared them with the CRLB.

In our simulation, a Gaussian signal, \( s(t) = \sqrt{\pi} \exp\{- (t - 4.096)^2 / 2\}, 0 \leq t \leq 8.192 \) was used as the SOI. The sampling period is set as \( T_s = 0.001 \). We set the true parameters of time delay and Doppler shift at \( (D = 0, f_d = 0) \). We vary the noise variances to get different SNR’s. The results are shown in Fig. 4. From Fig. 4, we can see the estimations are very close to the CRLB. This shows the effectiveness of the proposed method.

VIII. EXPERIMENTAL RESULTS

We present some experimental results to demonstrate the performance of the proposed algorithms. The data we use is the television voice signals which are obtained through field trials. The block diagram of the experimental system is shown in Fig. 5 and the parameters are listed in Table II. In our experimental system, the two measurements, \( x(t) \) and \( y(t) \), are first sampled through analog-to-digital converter (A/D). Then they are down-sampled by digital down converter (DDC). After direct signal cancellation (DSC), the CAF is calculated. The CAF result is sent to CFAR module to detect and estimate the (time delay, Doppler shift) pairs. At last, combined with other information, such as direction-of-arrival (DOA), the trajectory is formed.

We mainly focus on the CAF module, which is implemented by a digital signal processor (DSP). In this module, the parameters are \( N = 131072, D = N_h = 256, M = 512 \), and \( N_r = 50 \). The DSP we used is TMS320C6701 [16] whose clock rate is 167 MHz. It has two floating-multipliers and its highest performance is up to 1 giga floating point operations per second (GFLOPS). In this specific platform, the processing time of each algorithm is listed in Table III. We can see that, for 0.6554 s integration time, the PWZFFT method only needs 0.0835 s processing time. This shows that PWZFFT method is feasible in real-time systems.

In order to verify the proposed algorithm, we save the measurement data into the storage device and process them offline. The CAF results obtained by direct calculation and PWZFFT method are shown in Fig. 6. We can see the results are almost the same as each other.

For further quantitative verification, we use two approaches to estimate the parameters. One is the direct calculation without DDS and surface fitting, in which the time delay and Doppler shift steps are set as \( \Delta \tau_1 = 0.1250 \) \( \mu s \) and \( \Delta f_1 = 0.03815 \) Hz, respectively. The other is the two-stage method, in which the calculation steps in the second stage are set as \( \Delta \tau_2 = 5.0000 \) \( \mu s \) and \( \Delta f_2 = 1.5259 \) Hz.

![Fig. 5: Block diagram of the experimental system.](image)

**TABLE II: Parameters of the experimental system**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transmitter carrier frequency</td>
<td>222.75 MHz</td>
</tr>
<tr>
<td>Bandwidth of the measured signal</td>
<td>50 kHz</td>
</tr>
<tr>
<td>A/D resolution</td>
<td>16 bit</td>
</tr>
<tr>
<td>Sampling frequency</td>
<td>8 MHz</td>
</tr>
<tr>
<td>DDC factor</td>
<td>40</td>
</tr>
<tr>
<td>Integration time</td>
<td>0.6554 s</td>
</tr>
</tbody>
</table>

**TABLE III: Processing time of each method (Integration time 0.6554 s)**

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Processing Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct Calculation</td>
<td>80.0702</td>
</tr>
<tr>
<td>FFT Method</td>
<td>1.5112</td>
</tr>
<tr>
<td>ZFFT Method</td>
<td>0.1604</td>
</tr>
<tr>
<td>PWZFFT Method</td>
<td>0.0835</td>
</tr>
</tbody>
</table>
a two-stage method, which includes the coarse search and fine search, for joint time delay and Doppler shift estimation. The algorithm analysis shows that the two-stage method has lower computational complexity, requires less memory and can meet the need of real-time processing. Simulation results show that the two-stage method induces negligible error and can achieve the CRLB for high SNR.

ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundations of China (No. 60232010) and the Teaching and Research Award Program for Outstanding Young Teachers in Higher Education Institutions of MOE, China.

REFERENCES


The RMSE is defined as $R = \sqrt{\frac{1}{N} \sum_{i=1}^{N} e_i^2}$, where $N$ is the number of estimation and $e_i$ is the $i$-th error of time delay or Doppler shift.

IX. CONCLUSION

In this paper, we present the pre-weighed Zoom FFT method for fast computation of the ambiguity function in the discrete grid and quadratic surface fitting method for solving the stepwise problem. Combining these two methods, we present respectively. We use the difference between the results of the two approaches to evaluate the proposed method. In fact, although the difference is not the true estimation error, it can depict the errors induced by the PWZFFT method and the quadratic surface fitting method. So we refer it as error herein. The results are given in Table IV. Note that the data of observation 1 are long enough to be divided into several segments for CAF integration. According to Table IV, we can obtain that the root mean square errors (RMSEs) $^5$ are (0.7119 µs, 0.3880 Hz). The RMSEs are much less than the calculation steps ($\Delta \tau_2 = 5.0000 \mu s, \Delta f_2 = 1.5259 \text{Hz}$), which shows that the error induced by the two-stage method is negligible.

\[ \sum_{i=1}^{N} e_i^2 \]

5The RMSE is defined as $R = \sqrt{\frac{1}{N} \sum_{i=1}^{N} e_i^2}$, where $N$ is the number of estimation and $e_i$ is the $i$-th error of time delay or Doppler shift.
TABLE IV: (Time delay, Doppler shift) pairs obtained by direct calculation and two-stage method, unit (µs, Hz)

<table>
<thead>
<tr>
<th>Observation</th>
<th>Direct Calculation</th>
<th>Two-Stage Method</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(75.2500, 200.7675)</td>
<td>(75.1436, 200.8756)</td>
<td>(−0.1064, 0.1081)</td>
</tr>
<tr>
<td></td>
<td>(74.5000, 204.5822)</td>
<td>(74.3791, 204.3691)</td>
<td>(−0.1209, −0.2131)</td>
</tr>
<tr>
<td></td>
<td>(73.3750, 204.9255)</td>
<td>(72.5575, 204.6851)</td>
<td>(−0.8175, −0.2405)</td>
</tr>
<tr>
<td>2</td>
<td>(3.7500, 146.7133)</td>
<td>(4.1043, 147.1986)</td>
<td>(0.3543, 0.4853)</td>
</tr>
<tr>
<td>3</td>
<td>(14.2500, −165.0620)</td>
<td>(13.7537, −165.0727)</td>
<td>(−0.4963, −0.0107)</td>
</tr>
<tr>
<td>4</td>
<td>(18.1250, −206.4133)</td>
<td>(19.0003, −206.0072)</td>
<td>(0.8753, 0.4060)</td>
</tr>
<tr>
<td>5</td>
<td>(14.5000, 247.5357)</td>
<td>(14.4279, 247.6019)</td>
<td>(−0.0721, 0.0662)</td>
</tr>
<tr>
<td>6</td>
<td>(18.0000, 224.4186)</td>
<td>(19.1796, 224.3306)</td>
<td>(1.1796, −0.0880)</td>
</tr>
<tr>
<td>7</td>
<td>(33.0000, 65.1550)</td>
<td>(34.2701, 66.1321)</td>
<td>(1.2701, 0.9771)</td>
</tr>
<tr>
<td>8</td>
<td>(2.7500, 250.7401)</td>
<td>(2.2743, 250.5874)</td>
<td>(−0.4757, −0.1526)</td>
</tr>
</tbody>
</table>

RMSE = (0.7119, 0.3880)