*International Journal of Innovative Computing, Information and Control*
Volume **11**, Number **2**, April **2015**

ICIC International ©2015 ISSN 1349-4198
pp. 657–677

# AN OPTIMIZED SPLIT-RADIX FFT ALGORITHM IN DETECTING MECHANICAL RESONANCE FREQUENCY OF SERVO SYSTEM

Hongli Liu[1], Zheng Zhang[2], Qixin Zhu[1] and Guoping Zhang[3]

[1]School of Mechanical Engineering
Suzhou University of Science and Technology
No. 1701, Binhe Road, Suzhou 215009, P. R. China
{ liuhl_sz; bob21cn }@163.com

[2]School of Electrical and Electronic Engineering
East China Jiaotong University
No. 808, Shuanggang East Street, Nanchang 330013, P. R. China
451651248@qq.com

[3]Shenzhen Han's Motor Technology Company Limited
No. 9018, North Central Avenue, Shenzhen 518058, P. R. China
onlyIm2008@163.com

Received April 2014; revised August 2014

Abstract. *To suppress the vibration of the motor caused by the flexibility of mechanical transmission devices and load inertia, it is needed to use FFT algorithm to analyze the spectrum of speed or position of motion components in servo system. So a high precision and rapidity of the FFT algorithm is very necessary. Firstly, the radix-2, radix-4 and traditional split radix FFT algorithms are introduced in this paper. Secondly, an optimized split radix FFT algorithm is proposed in this paper, and the optimization of FFT algorithm is carried out mainly from three aspects. These three aspects are bit-reversed order, real sequence input instead of the traditional complex sequence input and rotation factor. Thirdly, the effectiveness of the radix-2, radix-4 and traditional split radix FFT algorithm is analyzed by simulations. Based on the XMC4500 floating-point ARM chip, 1024 points and 512 points of the optimized split-radix FFT algorithm's operation time and accuracy are tested. The experimental results show that the optimized split-radix FFT algorithm has certain quickness and accuracy, and it can be used to detect mechanical resonance frequency in the industrial servo system. Finally, split-radix FFT algorithm is adopted for spectral analysis of position signals when motor is positioned to obtain accurate buffeting resonant characteristics. The ultimate simulation experiments have shown the effectiveness of the optimized split-radix FFT algorithm which is utilized to suppress the vibration of the motor when positioning.*

**Keywords:** Resonance frequency, Optimized split-radix FFT algorithm, Bit-inversion sequence optimization, Real sequence input, Plural sequence input, Rotation factor, Motor vibration, Motor positioning

1. **Introduction.** The mechanical transmission part of industrial servo system generally chooses connection device, such as the coupling, transmission shaft, speed changer. The main functions of connection device are to transfer the torque of the servo motor to load and matching speed and torque [1]. However, the actual connection device is not ideal. There will be a certain elastic deformation when it is under torsion, which will cause mechanical resonance in the servo system [2-4]. Mechanical oscillation not only emits an acoustic noise, which will form noise pollution, but also can cause serious damage to mechanical connection device, thus affecting its life span. In addition, the mechanical oscillation seriously affects the stability of the whole servo system, and limits the bandwidth

of the system. Therefore, detection and suppression approaches of mechanical resonance frequency have become a very important topic in industrial servo drive system [5-7].

In recent years, suppression approaches of mechanical resonance frequency mainly include three types: (1) Use the sensor to measure load and motor's speed and position signals directly, according to the signals to estimate the states of the mechanical resonance, then to restrain [8,9]. However, it not only requires measuring the position of the load, but also makes the system more complex, raise the cost of the system. So it is difficult to realize in regular application. (2) Use the sensor to measure motor's speed and position signal, and use the observer to estimate the load's speed and other related variables, then to suppress resonance [10-12]. The control structure of this method is complicated and it is needed to design corresponding observers according to different systems. (3) Based on the control structure of industrial servo system, using only one sensor to measure the motor's speed and position signals, and then using notch filter to suppress mechanical resonance frequency in the control loop [13-19]. In his approach, it only needs to use simple compensation device or corrective action to suppress the resonance frequency, without changing the system control structure and the controller parameters. There are two major steps to implement this approach. Firstly, speed error signals of servo system control loop should be transformed using real-time FFT to obtain the mechanical resonance frequency of system. Then, the appropriate filter can be designed according to the resonance frequency. In recent years, to improve this method, many scholars have brought up a variety of ways to improve the filter. Since the low pass filter to suppress resonance is less reliable and the low pass filter will lead to a phase lag problem [2], the notch filter has been proposed to suppress resonance frequency [20]. In order to achieve better effect, on the basis of the traditional notch filter, a complex falling filter is proposed [21]. Furthermore, some suggestions are also proposed for improving the filtering performance in view of the deficiencies of the conventional filter [9,10]. In order to ensure both detection speed and precision of the resonant frequency, the traditional FFT transformation needs to be optimized. However, there are no relevant results reported in literature to investigate how to obtain resonance frequency quickly and accurately. So in this paper, operation time and detection precision of several different FFT algorithms are compared, emphasis on optimizing split-radix FFT algorithm so as to shorten the detection time and improve the precision of the resonant frequency.

When using FFT algorithms, specific to the algorithms of $N$ equals integral power of 2, there are three major algorithms; they are radix-2, radix-4 and split-radix FFT [22-24]. Firstly, the traditional split-radix FFT algorithm is optimized mainly from three aspects in this paper. These three aspects are bit-inversion sequence optimization, real sequence input instead of the traditional plural sequence input and rotation factor optimization respectively. Secondly, compared to the calculation of three kinds of FFT algorithms, the results showed that, with the increase of operational points, the split-radix FFT is more conducive to the real-time requirements, its computational time is much less than the radix-2 and radix-4 FFT algorithms. Then a comparative analysis between the traditional and improved split-radix FFT through MATLAB simulation needed to be done to verify the optimized split-radix FFT has certain quickness and accuracy in the resonant frequency detection. Thirdly, based on the XMC4500 floating-point ARM chip, the effectiveness of the traditional and improved split-radix FFT algorithms is verified in the DAVE (Digital Application Virtual Engineer) environment. Emphasis on testing optimized split-radix FFT algorithm's operation time and precision when detect oscillation frequency, and compared with theory operation time and actual operation time. Experimental results show that the optimized split-radix FFT algorithm can meet the requirements of speediness and accuracy of mechanical resonance frequency detection.

The actual operation time and accuracy do not appear to be much different from the theoretical analysis. The results also show that the optimized split-radix FFT algorithm has certain quickness and accuracy, and it can be used to detect mechanical resonance frequency in the industrial servo system.

At last, in this paper, the dynamic performance of PMSM servo system is investigated through the dynamic modeling, frequency property analysis, and computer simulation. The influence of the elastic deformation in transmission mechanism on the dynamic performance is studied. Then the flexible connection device between the motor and load will cause the motor vibration in PMSM when positioning. In order to suppress the vibration of motor, split-radix FFT algorithm is adapted for spectral analysis of position signals when motor positioning to obtain accurate buffeting resonant characteristics. According to the accurate buffeting resonant characteristics, filtering the resonant part in position signals could suppress the vibration of the motor end when positioning. The ultimate simulation experiments show the optimized split-radix FFT algorithm is effective to suppress the vibration of the motor when positioning.

## 2. Radix-2, Radix-4 and Traditional Split-Radix FFT Algorithms.

### 2.1. The radix-2 FFT algorithms.

The form of discrete Fourier transform (DFT) is as follows.

$$X\left(k\right)=\sum_{n=0}^{N-1}x\left(n\right)W_N^{nk}, \tag{1.1}$$

$$k=0,1,\ldots,N-1 \tag{1.2}$$

where $W_N^{nk}=e^{-j\frac{2\pi}{N}nk}$ is rotation factor, $X\left(k\right)$ is discrete Fourier transform of $N$ points sequence $x\left(n\right)$. The radix-2 FFT algorithm extracted by decimation-in-time (DIT) is to make $x\left(n\right)$ into two groups according to the odd and even. The expressions are described as follows.

$$X\left(k\right)=\sum_{r=0}^{N/2-1}x\left(2r\right)W_{N/2}^{rk}+W_N^k\sum_{r=0}^{N/2-1}x\left(2r+1\right)W_{N/2}^{rk} \tag{2.1}$$

$$X\left(N/2\right)=\sum_{r=0}^{N/2-1}x\left(2r\right)W_{N/2}^{rk}-W_N^k\sum_{r=0}^{N/2-1}x\left(2r+1\right)W_{N/2}^{rk} \tag{2.2}$$

$$k=0,1,\ldots,N/2-1 \tag{2.3}$$

### 2.2. The radix-4 FFT algorithms.

The radix-4 FFT algorithm extracted by DIT is to divide $x\left(n\right)$ into four groups, which are $x\left(4r\right)$, $x\left(4r+1\right)$, $x\left(4r+2\right)$ and $x\left(4r+3\right)$, and the expressions are described as follows.

$$X\left(k\right)=X_0\left(k\right)+W_N^kX_1\left(k\right)+W_N^{2k}X_2\left(k\right)+W_N^{3k}X_3\left(k\right) \tag{3.1}$$

$$X\left(k+N/4\right)=X_0\left(k\right)-jW_N^kX_1\left(k\right)-W_N^{2k}X_2\left(k\right)+jW_N^{3k}X_3\left(k\right) \tag{3.2}$$

$$X\left(k+2N/4\right)=X_0\left(k\right)-W_N^kX_1\left(k\right)+W_N^{2k}X_2\left(k\right)-W_N^{3k}X_3\left(k\right) \tag{3.3}$$

$$X\left(k+3N/4\right)=X_0\left(k\right)+jW_N^kX_1\left(k\right)-W_N^{2k}X_2\left(k\right)-jW_N^{3k}X_3\left(k\right) \tag{3.4}$$

$$X_0\left(k\right)=\sum_{r=0}^{N/4-1}x(4r)W_N^{4rk} \tag{3.5}$$

$$X_1\left(k\right)=\sum_{r=0}^{N/4-1}x(4r+1)W_N^{4rk} \tag{3.6}$$

$$X_2(k) = \sum_{r=0}^{N/4-1} x(4r+2)W_N^{4rk} \tag{3.7}$$

$$X_3(k) = \sum_{r=0}^{N/4-1} x(4r+3)W_N^{4rk} \tag{3.8}$$

$$k = 0, 1, \ldots, N/4-1 \tag{3.9}$$

2.3. **The traditional split-radix FFT algorithms.** The traditional split-radix FFT algorithm is extracted by decimation-in-frequency (DIF), and its first progression is decomposed into one $N/2$ points DFT and two $N/4$ points DFT, even sequence uses radix-2 algorithm, odd sequence uses radix-4 algorithm, and the expressions are described as follows.

$$X(2r) = \sum_{n=0}^{N/2-1} [x(n) + x(n+N/2)]W_{N/2}^{nr}, \quad r = 0, 1, \ldots, N/2-1 \tag{4.1}$$

$$\begin{aligned} X(4r+1) = \sum_{n=0}^{N/4-1} &[(x(n) - x(n+N/2)) - j(x(n+N/4) - x(n+3N/4))] \\ &* W_N^n W_{N/4}^{nr}, \quad r = 0, 1, \ldots, N/4-1 \end{aligned} \tag{4.2}$$

$$\begin{aligned} X(4r+3) = \sum_{n=0}^{N/4-1} &[(x(n) - x(n+N/2)) + j(x(n+N/4) - x(n+3N/4))] \\ &* W_N^{3n} W_{N/4}^{nr}, \quad r = 0, 1, \ldots, N/4-1 \end{aligned} \tag{4.3}$$

2.4. **Comparison of calculation for radix-2, radix-4 and traditional split-radix FFT algorithms.** In the calculations of the radix-2, radix-4 and traditional split-radix FFT algorithms, the multiplication is more complicated than the addition operation in the computer implementation and it is the main factor to affect the computing speed. So here we mainly compared the multiplication operations of the three kinds of FFT algorithms when complete the same tasks. The theory complex multiplication of the radix-2, radix-4 and traditional split-radix FFT algorithms are $\frac{N}{2}\log_2^N$, $\frac{3N}{4}\log_4^N$ and $\frac{N}{3}\log_2^N$. It is obvious that the split-radix FFT algorithm needed the minimum calculation when accomplish the same tasks. In particular, by using the split-radix FFT algorithm will save much more computation when the sampling points $N$ is large. For the system needs to analyze the continuous time signals and has real-time requirements, and the comparison of calculation for radix-2, radix-4 and traditional split-radix FFT algorithms is showed in Figure 1.

From Figure 1, we can see the changes of the required computation of the three different FFT algorithms with the increase of sampling points. Through the MATLAB simulation, we can clearly see the computation time of split-radix FFT algorithm is less than the other two FFT algorithms.

## 3. Optimized Split-Radix FFT Algorithms.

3.1. **Bit-inversion sequence optimization.** For radix-2, radix-4 and split-radix FFT algorithms, no matter DIT or DIF, input or output need for bit-inversion sequence transformation. So a rapid bit-inversion sequence implementation method can improve the execution efficiency of FFT algorithms. In the traditional bit-inversion sequence algorithm, the data not only need for exchange judgment, but also shift judgment. The data requires multiplication and addition operations (multiplication can be done through a shift method) in the shifting judgment. The optimized bit-inversion sequence algorithm do not need for shift judgment.

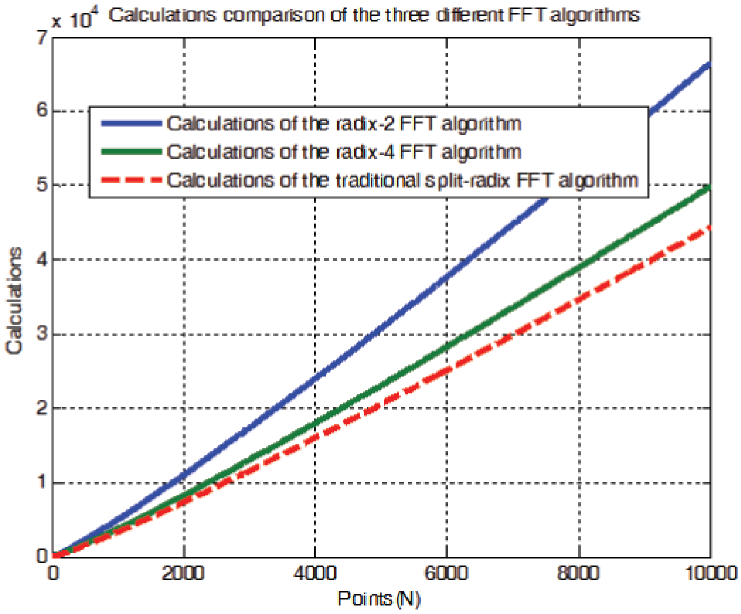


Figure 1. Comparison of calculations for three different FFT algorithms

Table 1. Implementation process table of optimized bit-inversion sequence

| Cloumn<br>Progression | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | | | | | | | | 1 | | | | | | | |
| 1 | 0 | | | | 2 | | | | 1 | | | | 3 | | | |
| 2 | 0 | | 4 | | 2 | | 6 | | 1 | | 5 | | 3 | | 7 | |
| 3 | 0 | 8 | 4 | 12 | 2 | 10 | 6 | 14 | 1 | 9 | 5 | 13 | 3 | 11 | 7 | 15 |

Take $N = 16$ as an example, Table 1 is the realization way of optimized bit-inversion sequence algorithm.

As one can see in Table 1, 0 progression adds one number only in the 8th column, that is to say, the 0 progression's increment is 1 and step length is 8. On the basis of the 0 progression, the first progression adds two numbers, respectively are number 2 (equals 2 + 0) in the fourth column and number 3 (equals 2 + 1) in the 12th column, that is to say, the first progression's increment is 2 and step length is 4. Similarly, on the basis of the first progression, the second progression increased by four numbers, the four numbers are 4 (equals 4 + 0), 6 (equals 4 + 2), 5 (equals 4 + 1) and 7 (equals 4 + 3) respectively, so the increment of the second progression is 4, the step length is 2. By that analogy, we can infer the third progression's new eight numbers respectively are 8 (equals 8 + 0), 12 (equals 8 + 4), 10 (equals 8 + 2), 14 (equals 8 + 6), 9 (equals 8 + 1), 13 (equals 8 + 5), 11 (equals 8 + 3), 15 (equals 8 + 7), thus the increment of the third progression is 8, the step length is 1.

Assuming the power of the data is $p$ ($2^p = N$). Through Table 1, bit-inversion sequence is $n_0n_1 \ldots n_{p-1}n_p$, original sequence is $n_pn_{p-1} \ldots n_1n_0$, the initial state of $n_0$ to $n_p$ is 0, thus original sequence is 0; $n_p$ in bit-inversion sequence at the 0 progression is 1 (the increment is $2^0$), $n_p$ in original sequence at the 0 progression is $2^p$ (that is on the basis of

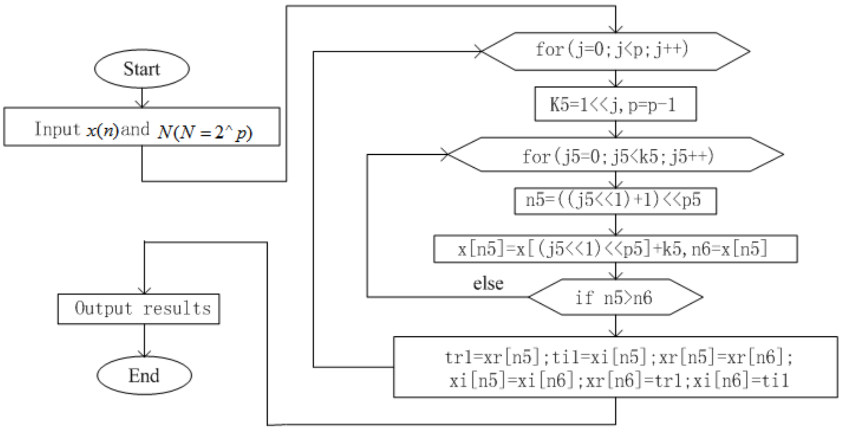

Start
Input $x(n)$and $N(N=2^\wedge p)$
for(j=0;j<p;j++)
K5=1<<j, p=p-1
for(j5=0;j5<k5;j5++)
n5=((j5<<1)+1)<<p5
x[n5]=x[(j5<<1)<<p5]+k5, n6=x[n5]
else
if n5>n6
tr1=xr[n5];ti1=xi[n5];xr[n5]=xr[n6];
xi[n5]=xi[n6];xr[n6]=tr1;xi[n6]=ti1
Output results
End

FIGURE 2. The programmed-thinking and flow chart of optimized bit-inversion sequence algorithm

the initial state add $2^p$). The first progression on the basis of the 0 progression adds two numbers, are number 2 (equals 2 + 0) and 3 (equals 2 + 1) respectively. Thus, original sequence are $2^{p-1} + 0 * 2^p$ and $2^{p-1} + 1 * 2^p$ respectively. By that analogy, suppose that $j$ is the current progression, thus the increment is $2^j$, step length is $2^{m-j-1}$, the number of new numbers is $2^j$.

Depending on the above rules, the programmed-thinking and flow chart of optimized bit-inversion sequence algorithm is showed in Figure 2.

As shown in Figure 2, with the increment of progression $j$, get the location of the new data $n5$ firstly, $n5$ is original sequence, and the bit-inversion sequence $n6$'s location is obtained by original sequence data plus increment. When execute exchange judgment, as long as the original sequence data is greater than the bit-inversion sequence data, then the two dates exchanged, keep the cycle going, the bit-inversion sequence of $N$ points date can be obtained.

From the above analysis, the optimized bit-inversion sequence algorithm has the following advantages:

(1) Bit-inversion sequence process of $N$ points dates can be completed only with $N - 1$ times operation.
(2) The optimized bit-inversion sequence is completed without the need to work in every bit of data. The process is easy and clear.
(3) The greater the $N$, the faster the computing speed.

### 3.2. *N* points real sequence input, instead of traditional plural sequence input.

In practical applications, the input data $x(n)$ are generally real sequence. However, when processing real sequence, the traditional FFT algorithms usually look the $x(n)$ as plural sequence which the imaginary part is zero. There is no doubt that this will increase operation time. In order to further reduce the computation time and storage, the improved real sequence FFT algorithm use $N/2$ points sequence's FFT to calculate $N$ points sequence's DFT. Set the real sequence's even sequence to be plural sequence's real part, set the real sequence's odd sequence to be plural sequence's imaginary part, and the improved FFT algorithm can reduce the half of the computation time.

For the input $x(n)$ ($x(n)$ is $N$ points real sequence), $n = 0, 1, \ldots, N - 1$, according to the following way, plural sequence $d(n)$ with $N/2$ points can be constituted, $n = 0, 1, \ldots, N/2 - 1$, $x(2n)$ is the real part of $d(n)$ and $x(2n + 1)$ is the imaginary part of

$d(n)$.

$$d(n) = x(2n) + jx(2n+1) \tag{5.1}$$

$$n = 0, 1, \ldots, N/2 - 1 \tag{5.2}$$

According to conjugate symmetry of DFT, the DFT of $x(2n)$ corresponds to the conjugate even symmetric component of $D(k)$ ($D(k)$ is the DFT of $d(n)$), the DFT of $x(2n+1)$ corresponds to the conjugate odd symmetric component of $D(k)$:

$$DFT\left[\mathrm{Re}\left\{d(n)\right\}\right] = DFT\left[x(2n)\right] = \frac{1}{2}\left[D(k) + D^*(N-k)\right] \tag{6.1}$$

$$DFT\left[\mathrm{Im}\left\{d(n)\right\}\right] = DFT\left[x(2n+1)\right] = \frac{1}{2j}\left[D(k) - D^*(N-k)\right] \tag{6.2}$$

Thus, the $X(k)$ ($X(k)$ is the DFT of $x(n)$) can be obtained

$$X(k) = \sum_{n=0}^{N-1} x(n)W_N^{nk} = \sum_{n=0}^{N/2-1} x(2n)W_{N/2}^{nk} + W_N^k \sum_{n=0}^{N/2-1} x(2n+1)W_{N/2}^{nk} \tag{7}$$

Submitting Equations (6.1) and (6.2) into Equation (7), then Equation (8) can be got

$$X(k) = \frac{1}{2}\left[D(k) + D^*(N-k)\right] + \frac{1}{2j}W_N^k\left[D(k) - D^*(N-k)\right] \tag{8}$$

Thus, the Fourier transform of $N$ points sequence can be obtained by FFT of $N/2$ points plural sequence.

$D(k)$ can be composed of real part and imaginary part

$$D(k) = R(k) + jI(k) \tag{9}$$

Submitting Equation (9) into Equation (8), $X(k)$ can be expressed as

$$\begin{aligned} X(k) = {} & XR(k) + jXI(k) = \frac{R(k) + jI(k) + R(N-k) - jI(N-k)}{2} \\ & + \left(\cos\frac{2k\pi}{N} - j\sin\frac{2k\pi}{N}\right)\frac{R(k) + jI(k) - R(N-k) + jI(N-k)}{2j} \end{aligned} \tag{10}$$

and

$$XR(k) = RP(k) + IP(k)\cos\left(\frac{2k\pi}{N}\right) - RM(k)\sin\left(\frac{2k\pi}{N}\right) \tag{11.1}$$

$$XI(k) = IM(k) - RM(k)\cos\left(\frac{2k\pi}{N}\right) - IP(k)\sin\left(\frac{2k\pi}{N}\right) \tag{11.2}$$

$$RP(k) = \frac{R(k) + R(N-k)}{2} \tag{11.3}$$

$$RM(k) = \frac{R(k) - R(N-k)}{2} \tag{11.4}$$

$$IP(k) = \frac{I(k) + I(N-k)}{2} \tag{11.5}$$

$$IM(k) = \frac{I(k) - I(N-k)}{2} \tag{11.6}$$

where $R(k)$ and $I(k)$ are the real part and imaginary part of $D(k)$ respectively, $XR(k)$ and $XI(k)$ are the real part and imaginary part of $X(k)$ respectively.

In programming, constitutes $d(n)$ with input $x(n)$ firstly and get real part and imaginary part of $D(k)$. That is to say, $R(k)$ and $I(k)$ can be obtained, thus $RP(k)$, $RM(k)$, $IP(k)$ and $IM(k)$ can be solved. Real part and imaginary part of $X(k)$ can also be obtained at last, that is to say, $XR(k)$ and $XI(k)$ can be obtained.

3.3. **The optimization of rotation factor.** After transforming Equation (4.2) and Equation (4.3), the following equations are available.

$$X\left(4r+1\right)=\sum_{n=0}^{N/4-1}f_1\left(n\right)W_{N/4}^{nr}\qquad r=0,1,\ldots,N/4\ -1 \tag{12.1}$$

$$X\left(4r+3\right)=\sum_{n=0}^{N/4-1}f_3\left(n\right)W_{N/4}^{n}W_{N/4}^{nr}\qquad r=0,1,\ldots,N/4\ -1 \tag{12.2}$$

$$\left[\begin{array}{c}f_1\left(n\right)\\ f_3\left(n\right)\end{array}\right]=W*R*\left[\begin{array}{l}x\left(n\right)\\ x\left(n+N/4\right)\\ x\left(n+N/4\right)\\ x\left(n+N/4\right)\end{array}\right]\qquad n=0,1,\ldots,N/4-1 \tag{12.3}$$

$$W=\left[\begin{array}{cc}W_N^n & 0\\ 0 & W_N^{-n}\end{array}\right],\qquad R=\left[\begin{array}{cccc}1 & -j & -1 & j\\ 1 & j & -1 & -j\end{array}\right] \tag{12.4}$$

Compared to the traditional split-radix algorithm, rotation factor is changed in the optimized split-radix algorithm. Equation (12.1) and Equation (12.2) can make the operation easier.

The rotation factor of Equation (12.4) is as follows.

$$W_N^n=\cos\frac{2\pi n}{N}-j\sin\frac{2\pi n}{N},\qquad W_N^{-n}=\cos\tfrac{2\pi n}{N}+j\sin\tfrac{2\pi n}{N} \tag{13}$$

The rotation factor of Equation (4.3) is as follows.

$$W_N^n=\cos\frac{2\pi n}{N}-j\sin\frac{2\pi n}{N},\qquad W_N^{3n}=\cos\tfrac{6\pi n}{N}-j\sin\tfrac{6\pi n}{N} \tag{14}$$

From Equation (13) and Equation (14), one knows the traditional split-radix algorithm needs four times calculations of sine and cosine and the optimized algorithm only needs two times calculations of sine and cosine. As a result, the optimized algorithm saves a lot of time.

4. **MATLAB Simulations of Traditional and Improved Split-Radix FFT Algorithms.** In order to verify the accuracy and rapidity of traditional and improved split-radix FFT algorithms in harmonic analysis, a simulation analysis in MATLAB is given for these two algorithms firstly.

Currently, the vast majority of PMSM servo systems are digital computer-controlled. It can be convenient to realize the resonance analysis of the signals which we concerned about. The purpose of the spectrum analysis of the servo system is to obtain the frequency and amplitude information of the resonant frequency signals in real time, which provide a basis for the design of the notch filter. Here, therefore, we are merely concerned with the amplitude information of each resonant frequency point and not to consider the phase information.

Due to the mechanical resonance frequency is commonly 0~1000 Hz, so the sampling frequency of FFT computing is set to fs = 2000 Hz. Considering the spectral resolution, so FFT transform points is set to 1024, thus the theoretical spectral resolution is 1.953 Hz. Assume that the input time-domain signal is $x(t)=200\sin(400\pi t)+400\sin(800\pi t)+600\sin(1200\pi t)+800\sin(1600\pi t)$.

Figure 3(a) shows the input time-domain signal $x(t)$, and Figures 3(b) and 3(c) show the spectrum results of the time-domain signal $x(t)$ which are respectively analyzed by the traditional and improved split-radix FFT algorithms.

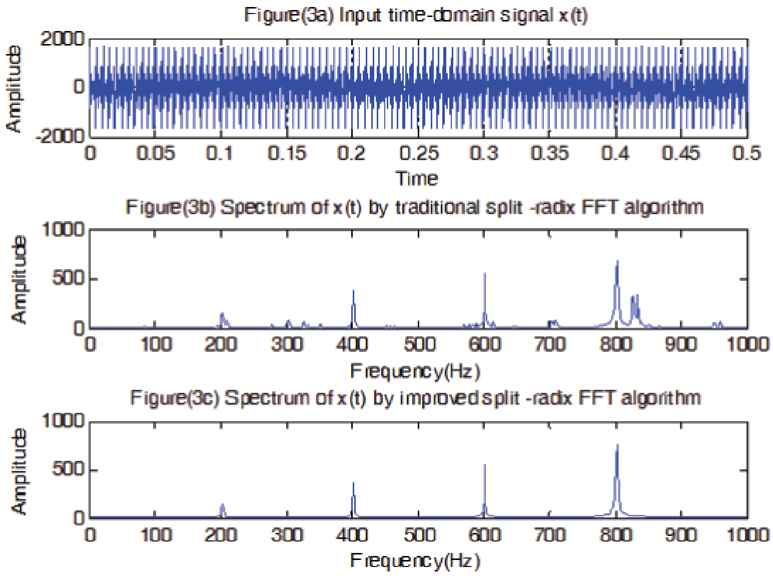


Figure 3. MATLAB simulations of traditional and improved split-radix FFT algorithms

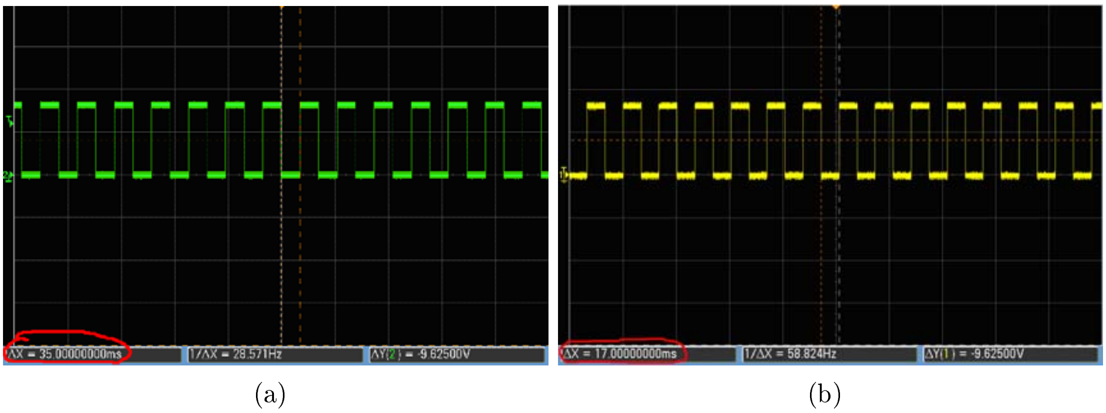
MATLAB programs can find out the frequency corresponding to the signal's amplitude, and the theory frequency is 800 Hz, through MATLAB program can find the actual frequency frec = 801.5625 Hz, and frec is within the range of spectral resolution.

Compared the frequency spectrum of Figures 3(b) and 3(c), we can find there is a large gap in accuracy between the traditional and improved split-radix FFT algorithms.

5. **Comparison of Operation Time and Accuracy between the Traditional and Improved Split-Radix FFT Algorithms.** The following results are based on Infineon XMC4500 chip, and the testing environment is DAVE environment which is developed by Infineon company itself. XMC4500 chip is a floating-point chip, and the CPU master clock frequency is 120 MHz. Due to the mechanical resonance frequency is commonly 0∼1000 Hz, so the sampling frequency of FFT computing is set to fs = 2000 Hz. Considering the spectrum resolution, set FFT transform points to 1024 and 512 which can be used to compare. In order to test the time of detecting the resonance frequency by FFT algorithms accurately, by controlling the voltage level of an I/O pin at the beginning and end of the whole code running, we can get the executing time of detecting programs with the help of the oscilloscope.

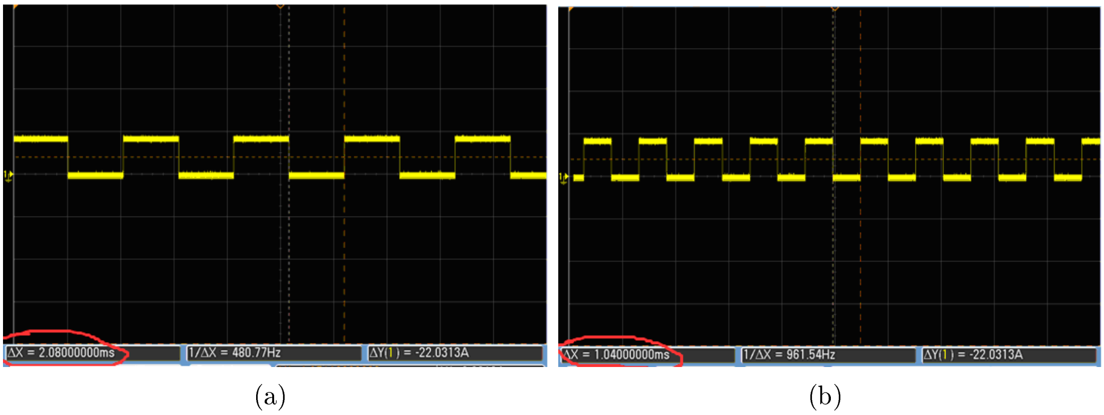
5.1. **Traditional split-radix FFT operation time.** Based on XMC4500 floating-point chip, through the oscilloscope measure operation time of the radix-2 FFT of 1024 points and 512 points, respectively, as shown in Figures 4(a) and 4(b).

5.2. **Bit-inversion sequence optimization.** In the traditional bit-inversion sequence algorithm, the data need for exchange judgment and shift judgment. However, in the optimized bit-inversion sequence algorithm the data do not need for shift judgment. Based on XMC4500 floating-point chip, by controlling the voltage level of an I/O pin at the beginning and end of the whole code running, we can get the executing time of the bit-inversion sequence (only is the time of bit-inversion sequence) with the help of the oscilloscope. In order to ensure detecting speed and precision of the resonant frequency simultaneously, FFT transformation points are set as 1024 and 512 respectively for comparison.



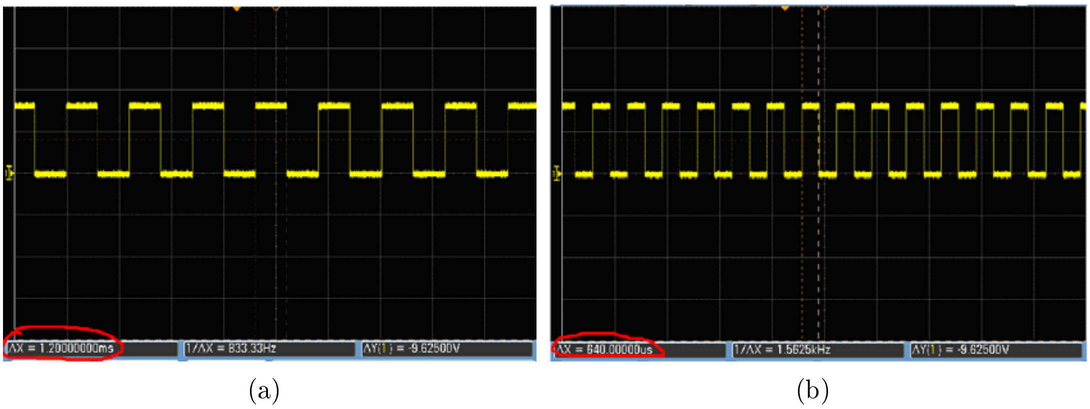
(a) (b)

FIGURE 4. (a) Time of traditional split-radix FFT of 1024 points; (b) time of traditional split-radix FFT of 512 points



(a) (b)

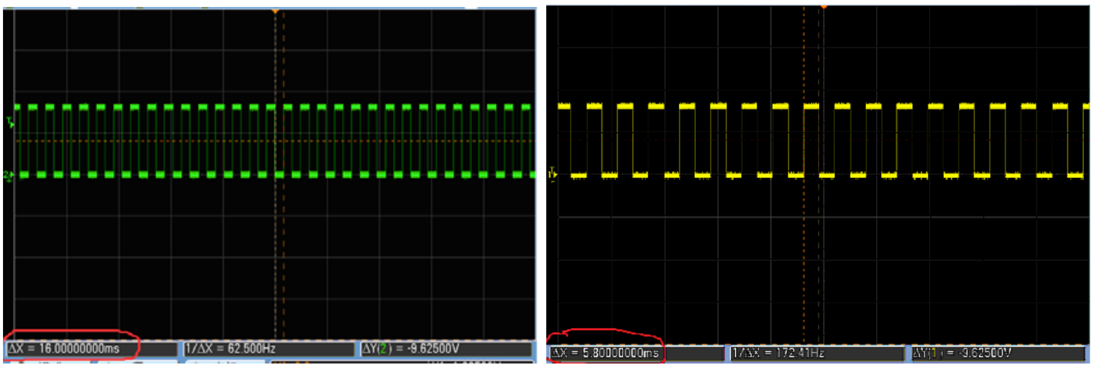
FIGURE 5. (a) Bit-inversion sequence time before optimization of 1024 points; (b) time of optimized bit-inversion sequence of 1024 points



(a) (b)

FIGURE 6. (a) Bit-inversion sequence time before optimization of 512 points; (b) time of optimized bit-inversion sequence of 512 points

The bit-inversion sequence operation time (only for the time of bit-inversion sequence) of before and after optimization of 1024 points is as shown in Figures 5(a) and 5(b). The bit-inversion sequence operation time (only for the time of bit-inversion sequence) before and after optimization of 512 points is as shown in Figures 6(a) and 6(b).



(a) (b)

Figure 7. (a) Optimized real sequence input, total time of 1024 points; (b) optimized real sequence input, total time of 512 points

5.3. **Real sequence input instead of traditional plural sequence input.** Instead of traditional plural sequence input, the improved real sequence FFT algorithm uses $N/2$ points sequence's FFT to calculate $N$ points sequence's DFT, and there is no doubt that the improved FFT algorithm can reduce a half of the computation time.

The improved split-radix FFT algorithm adopts real sequence input instead of traditional plural sequence input. Based on XMC4500 floating-point chip, by controlling the voltage level of an I/O pin at the beginning and end of the whole code running, we can get the executing time of the total operation time (including the time of optimized bit-inversion sequence, real sequence input and rotation factor before optimization) of 1024 points and 512 points with the help of the oscilloscope. The executing time of these two cases is shown in Figures 7(a) and 7(b).

From Figure 4(a), we can see the time of traditional split-radix FFT of 1024 points is 35 ms, from Figures 5(a) and 5(b), we know the traditional and optimized bit-inversion sequence operation time of 1024 points are 2.08 ms and 1.04 ms respectively, so the total time (including the time of optimized bit-inversion sequence, traditional plural sequence input and traditional rotation factor) of 1024 points is 33.96 ms (35 ms – 2.08 ms + 1.04 ms = 33.96 ms). From Figure 7(a), we know the total operation time (including the time of optimized bit-inversion sequence, optimized real sequence input and traditional rotation factor) of 1024 points is 16 ms. So the 1024 points optimized real sequence input spent 17.96 ms shorter than the traditional plural sequence input. Similarly, from Figures 4(b), 6(a), 6(b) and 7(b), the 512 points optimized real sequence input spent 8.64 ms shorter than the traditional plural sequence input.

5.4. **Rotation factor optimization.** The traditional rotation factor needs four times calculations of sine and cosine, compared with the traditional rotation factor; the optimized rotation factor only needs two times calculations of sine and cosine, as a result, the optimized algorithm saves a lot of time.

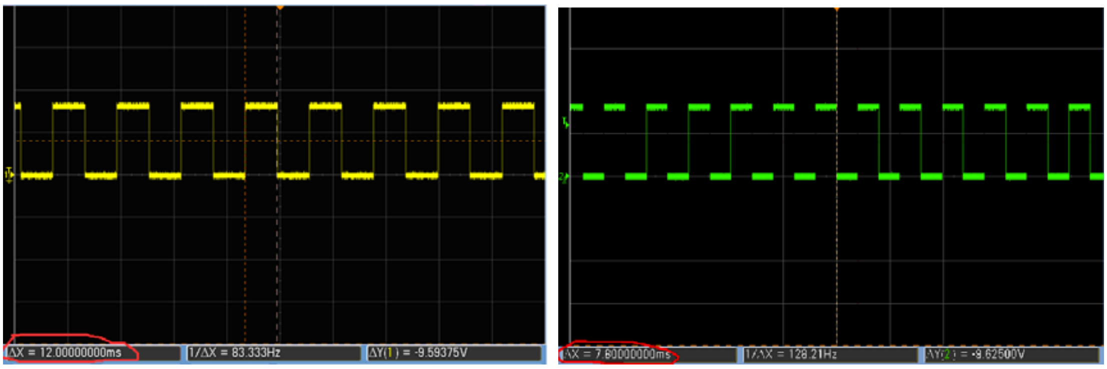
After the rotation factor optimized, through the oscilloscope respectively measure the total operation time (including the time of optimized bit-inversion sequence, optimized real sequence input and optimized rotation factor) of 1024 points and 512 points by controlling an I/O pin at the beginning and end of the whole code, as shown in Figures 8(a) and 8(b).

From Figure 7(a), we can see the total operation time (including the time of optimized bit-inversion sequence, optimized real sequence input and traditional rotation factor) of 1024 points is 16 ms. From Figure 8(a), we can see the total operation time (including the time of optimized bit-inversion sequence, optimized real sequence input and optimized rotation factor) of 1024 points is 12 ms. So the operation time of 1024 points optimized rotation factor is 4 ms shorter than the traditional rotation factor. Similarly, from Figures 7(b) and 8(b), we know the operation time of 512 points optimized rotation factor is 2 ms shorter than the traditional rotation factor.

Based on XMC4500 floating-point chip, through the oscilloscope measure split-radix FFT's operation time after three times optimization, and the split-radix FFT computation time before and after optimization of 1024 points and 512 points are summarized in Table 2.

Due to the single computation time of sequence input and rotation factor cannot be measured precisely, so in Table 2, assuming t1 and t2 are the time of sequence input before and after optimization respectively, and t3 and t4 are the time of rotation factor before and after optimization respectively. However, the time discrepancy between traditional plural sequence input and optimized real sequence input, the time discrepancy between traditional rotation factor and optimized rotation factor can be measured precisely. The total operation time of 1024 points and 512 points split-radix FFT can be measured precisely after three times of optimization.

5.5. **Comparison between actual and theory spectral resolution of optimized split-radix FFT.** The input signal is $x(t) = \max \sin(2\pi f_0 t) + y_1 \sin(2\pi f_1 t) + y_2 \sin(2\pi f_2 t)$, $f_0$ is the frequency corresponding to the biggest energy point of the input signal, $f_0 = 50$ Hz~950 Hz.



(a) (b)

FIGURE 8. (a) Optimized rotation factor, total time of 1024 points; (b) optimized rotation factor, total time of 512 points

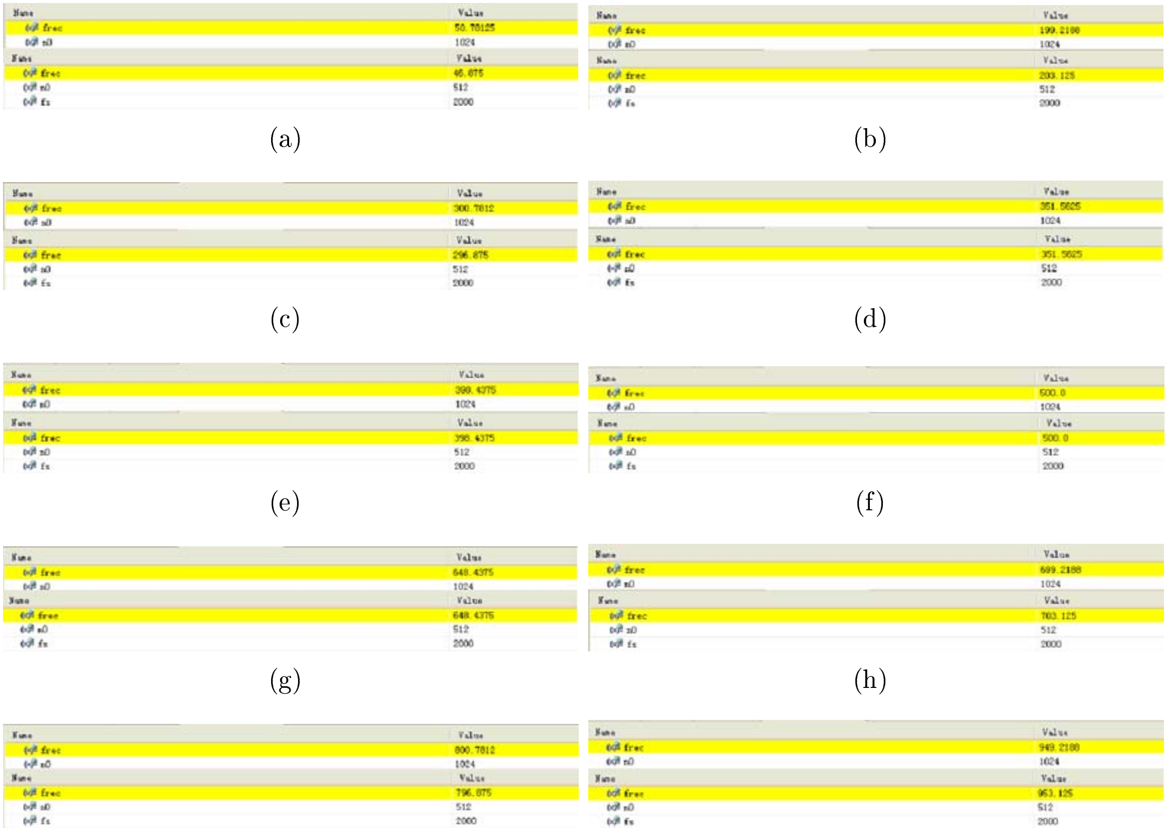
TABLE 2. Split-radix FFT computation time before and after optimization

| Time / Optimization | Before optimization (1024 points) | After optimization (1024 points) | Before optimization (512 points) | After optimization (512 points) |
|---|---|---|---|---|
| Bit-inversion sequence | 2.08ms | 1.04ms | 1.2ms | 0.64ms |
| Sequence input | t1ms | (t1-17.96)ms | t2ms | (t2-8.64)ms |
| Rotation factor | t3ms | (t3-4)ms | t4ms | (t4-2)ms |
| Total | 35ms | 12ms | 17ms | 5.8ms |

Based on XMC4500 floating-point chip, inputing a signal which contains different frequency, split-radix FFT program can identify the frequency which corresponding to the biggest energy point of the input signal, and the frequency can be read by DAVE environment window. frec can be identified by split-radix FFT programs and it is the actual frequency, f0 is the theory frequency, frec and f0 are corresponding to the biggest energy point of the input signal, n0 is split-radix FFT transformation points, and fs is sampling frequency.

Input theory frequency f0 are 50 Hz, 200 Hz, 300 Hz, 300 Hz, 400 Hz and 500 Hz, 650 Hz and 700 Hz and 800 Hz and 950 Hz respectively, the actual frequency can be identified by the optimized split-radix FFT programs, respectively as shown in Figure 9(a), Figure 9(b), Figure 9(c), Figure 9(d), Figure 9(e), Figure 9(f), Figure 9(g), Figure 9(h), Figure 9(i) and Figure 9(j).

Through the above DAVE environment window, actual frequency frec corresponding to the biggest energy point of the input signal can be got. Compared with theoretical frequency f0, spectral resolution corresponding to different point n0 and different frequency



```
Name        Value
  frec      50.78125
  n0        1024
Name        Value
  frec      46.875
  n0        512
  fs        2000
```

(a)

```
Name        Value
  frec      199.2188
  n0        1024
Name        Value
  frec      203.125
  n0        512
  fs        2000
```

(b)

```
Name        Value
  frec      300.7812
  n0        1024
Name        Value
  frec      296.875
  n0        512
  fs        2000
```

(c)

```
Name        Value
  frec      351.5625
  n0        1024
Name        Value
  frec      351.5625
  n0        512
  fs        2000
```

(d)

```
Name        Value
  frec      398.4375
  n0        1024
Name        Value
  frec      398.4375
  n0        512
  fs        2000
```

(e)

```
Name        Value
  frec      500.0
  n0        1024
Name        Value
  frec      500.0
  n0        512
  fs        2000
```

(f)

```
Name        Value
  frec      648.4375
  n0        1024
Name        Value
  frec      648.4375
  n0        512
  fs        2000
```

(g)

```
Name        Value
  frec      699.2188
  n0        1024
Name        Value
  frec      703.125
  n0        512
  fs        2000
```

(h)

```
Name        Value
  frec      800.7812
  n0        1024
Name        Value
  frec      796.875
  n0        512
  fs        2000
```

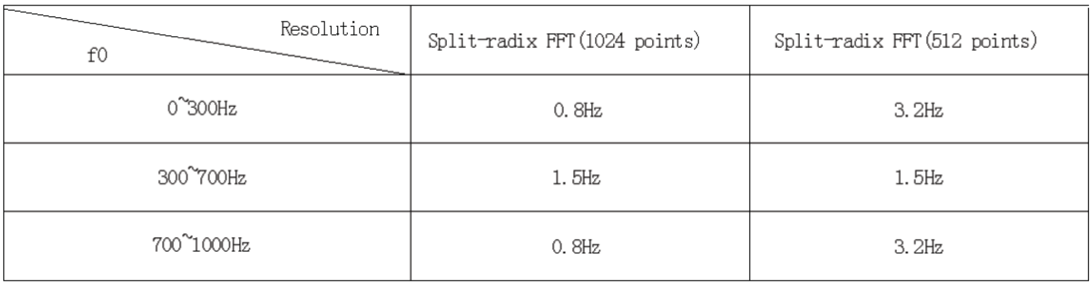
(i)

```
Name        Value
  frec      949.2188
  n0        1024
Name        Value
  frec      953.125
  n0        512
  fs        2000
```

(j)

Figure 9. (a) f0 = 50 Hz, actual frequency is identified by programs; (b) f0 = 200 Hz, actual frequency is identified by programs; (c) f0 = 300 Hz, actual frequency is identified by programs; (d) f0 = 350 Hz, actual frequency is identified by programs; (e) f0 = 400 Hz, actual frequency is identified by programs; (f) f0 = 500 Hz, actual frequency is identified by programs; (g) f0 = 650 Hz, actual frequency is identified by programs; (h) f0 = 700 Hz, actual frequency is identified by programs; (i) f0 = 800 Hz, actual frequency is identified by programs; (j) f0 = 950 Hz, actual frequency is identified by programs

Table 3. The spectral resolution under different points and different frequency f0



| Resolution<br>f0 | Split-radix FFT(1024 points) | Split-radix FFT(512 points) |
|---|---|---|
| 0~300Hz | 0.8Hz | 3.2Hz |
| 300~700Hz | 1.5Hz | 1.5Hz |
| 700~1000Hz | 0.8Hz | 3.2Hz |

f0 can be obtained. According to Figure 9, the spectral resolution under different points and different frequency f0 can be summarized in Table 3.

From Table 3, considering the FFT operation time and spectral resolution, $N$ takes 1024 at the best point.

5.6. **The comparison of actual time and theory time of optimized split-radix FFT algorithm.** Theoretically, split-radix FFT algorithm's butterfly computation have $N/3 * \log 2(N)$ times plural multiplication and $N * \log 2(N)$ times plural addition. Each plural multiplication needs four times real multiplication and two times real addition, and each plural addition needs two times real addition. For XMC4500 chip, completing one time of real addition needs one clock cycle and one time real multiplication needs two clock cycles. The master clock frequency of XMC4500 is 120 MHz. $N$ is split-radix FFT transformation points. Thus theoretically, butterfly computation of 1024 points takes about 0.5 ms; butterfly computation of 512 points takes about 0.21 ms.

The operation time of 1024 points split-radix FFT programs is theoretically about 3.74 ms which include bit-inversion sequence time (1.04 ms), butterfly computation (about 0.5 ms) and cycle to find the frequency corresponding to the biggest energy point of the input signal (2.2 ms). However, the actual operation time of 1024 points split-radix FFT programs is 12 ms by oscilloscope measured. Similarly, the operation time 1024 points split-radix FFT programs is theoretically about 1.95 ms, and the actual operation time is 5.8 ms by oscilloscope measured.

Compared the actual time and the theory time, we know actual time is about three times as much as theory time. The reasons are as follows. (1) Butterfly operation theoretically only considered the real multiplication and real addition computation times, but the sine and cosine computation should spend a lot of time when does butterfly operation. (2) Butterfly computation theoretically is complex operation, however, in the actual complex operation programming, real part and imaginary part need for separation operation which consumed much time. So the operation time of optimized split-radix FFT algorithm is acceptable.

## 6. Positioning Vibration for PMSM Servo System.

6.1. **Position loop model frame of PMSM servo system without flexibility distortion.** In order to simplify the model, the design and simulation of PMSM servo system generally do not consider flexibility distortion of the transmission mechanism.

Figure 10 is a simplified block diagram of PMSM servo system without the flexibility distortion. Due to the current response is very fast, so merged the current loop time constant and other small time constants of speed loop into a small time constant $\sigma$ inertia unit. The command of motor position is $\theta_i$, and position controller APR is a PI adjuster. The output of APR controller is speed command. The inside loop is speed loop and speed

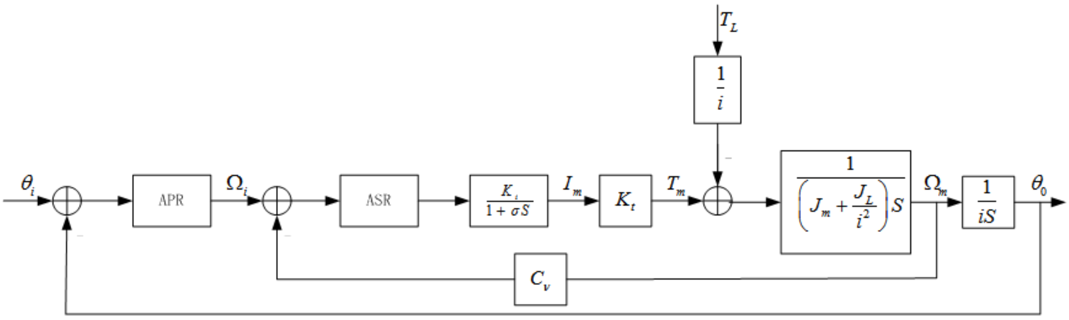


Figure 10. Block diagram of PMSM servo system without flexibility distortion

controller ASR is a PI adjuster. The output of ASR controller is current command. The inner loop is current loop and controller ACR is a PI adjuster. This paper focuses on the study of performance of the position loop and speed loop. In Figure 10, $C_v$ is speed feedback coefficient, $i$ is reducer ratio, $K_i$ is magnification of the current closed-loop, $K_t$ is motor torque coefficient, $J_m$ is motor rotor inertia, $J_L$ is load inertia, $T_m$ is output torque of elastic axis, and $\theta_0$ is motor position.

6.2. **Position loop model frame of PMSM servo system with flexibility distortion.** In the actual industrial servo system, the torque of the servo motor transferred to the load usually through mechanical transmission devices. However, the actual mechanical transmission devices are not the ideal rigid, and the devices will appear certain flexibility distortion subjected to torsion, which cause the mechanical resonance in the servo system.

Reference [3] gives the elastic model and its structure diagram. As showed in Figure 11 is position loop model frame of PMSM servo system with flexible distortion which can be obtained by combining the elastic model with the ideal system. In Figure 11, $\theta_L$ is load position, $K_L$ is transmission mechanism stiffness, and $F_L$ is speed friction coefficient of load axis.

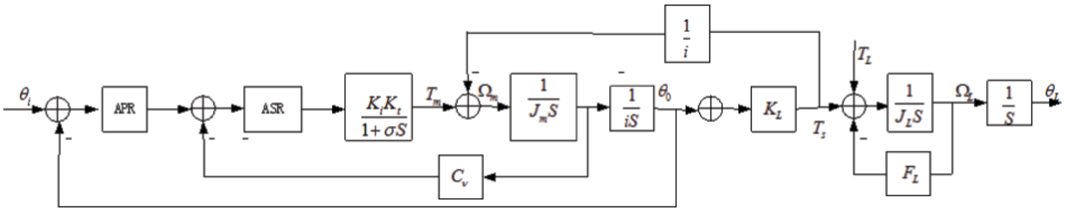


Figure 11. Block diagram of PMSM servo system with flexibility distortion

6.3. **The influence of mechanical resonance on system performance in MATLAB simulation.** The key parameters of PMSM servo system used in the simulation respectively are motor torque coefficient $K_t$ which is 1.05, transmission mechanism stiffness $K_L$ which is 500, motor rotor inertia $J_m$ which is $3.617 * 10^{(-4)} N \bullet m^2$, load inertia $J_L$ which is $10^{(-4)} N \bullet m^2$, speed friction coefficient of load axis $F_L$ which is 0.05. Adopted the same normal design method to configure parameters to ensure normal work and bandwidth for the system with and without flexibility distortion.

The transfer function of the system is very complex when flexibility distortion is considered. And using the Bode diagram is easier to obtain conclusions intuitively and clearly. So in order to analyze the positioning vibration problem visually, it is needed to draw the

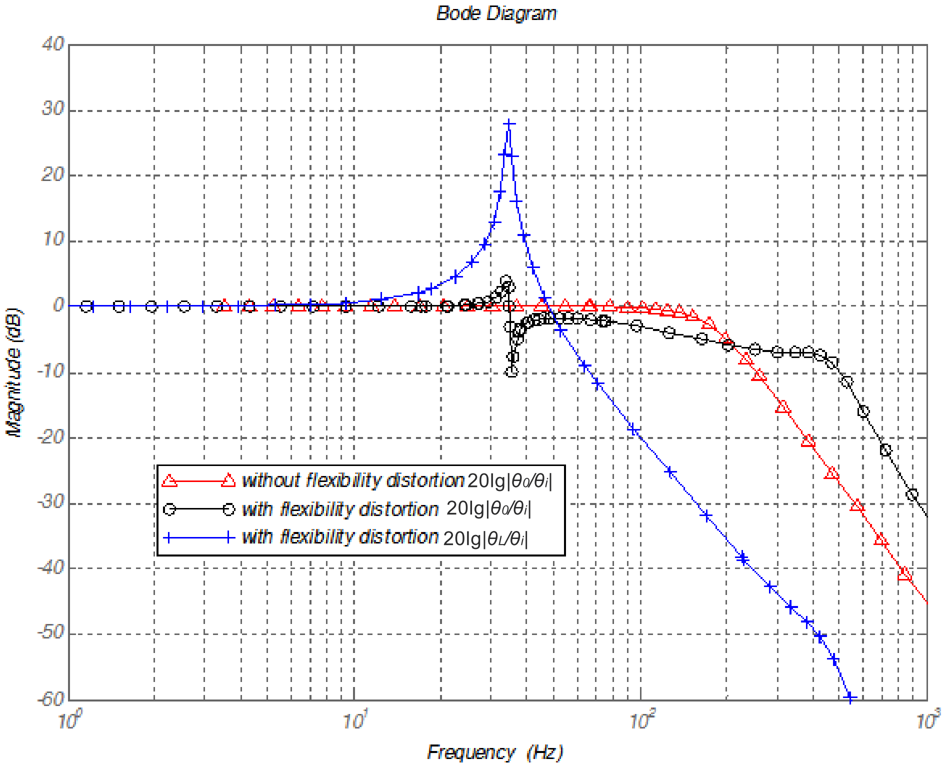


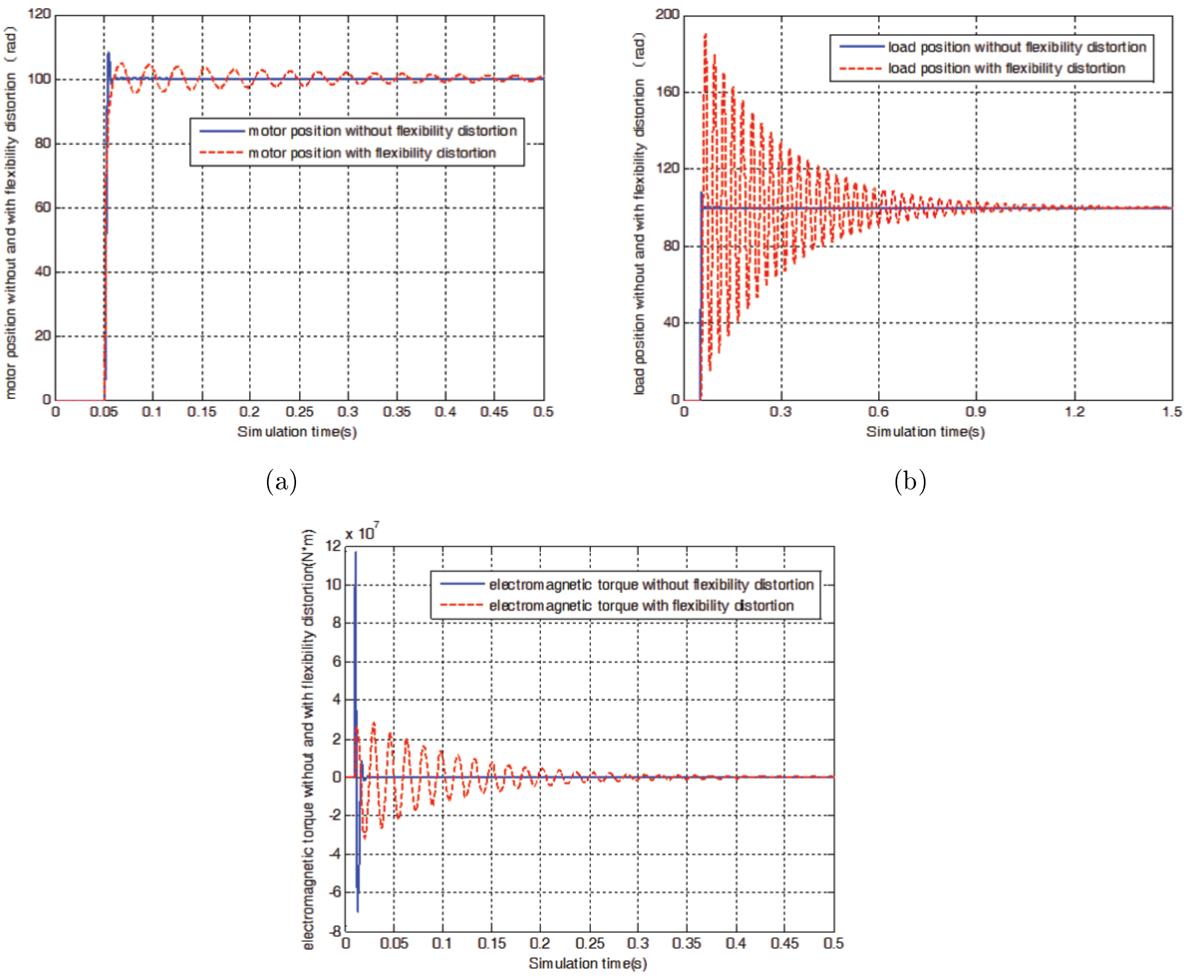
FIGURE 12. Bode diagram with and without flexibility distortion of the PMSM servo system

Bode diagram of the system in MATLAB simulation. Bode diagram with and without flexibility distortion of the PMSM servo system is obtained by the MATLAB simulation as showed in Figure 12.

Regardless of flexibility distortion, according to the normal design method to configure parameters could meet the requirement of the expected effect. However, harmonic oscillation exists in the system with flexible distortion. Figure 12 shows the frequency characteristics curves with flexibility distortion and without flexibility distortion, we can see that there is an overlap between these two curves in the low frequency band, but a significant difference in the high frequency band. The curve with flexibility distortion of $20\lg|\theta_0/\theta_i|$ has a +7.17 dB resonance peak and a mechanical anti-resonance peak; $20\lg|\theta_L/\theta_i|$ is the curve with flexibility distortion which has a +17.7 dB electromechanical combination resonance peak and the mechanical resonance offsets the anti-resonance, and the existence of a positive resonance peak indicates that there is a electromechanical combination resonance peak in the same frequency.

Figures 13(a), 13(b) and 13(c) show the simulated performance of the influence of mechanical resonance on system. Figure 13(a) shows the comparison of motor position step response between the system without and with flexibility distortion; Figure 13(b) shows the comparison of load position step response between the system without and with flexibility distortion; Figure 13(c) shows the comparison of electromagnetic torque step response between the system without and with flexibility distortion.

We can see from the simulation results, when compared to the ideal rigid transmission device, in the actual transmission device, the motor speed is much more stable depending on the closed-loop control, however, the load speed and electromagnetic torque are serious oscillation. Oscillation will damage the mechanical transmission device and shorten its working life.



(a)

(b)

(c)

FIGURE 13. (a) Comparison of motor position step responses between the system without and with flexibility; (b) comparison of load position step responses between the system without and with flexibility; (c) comparison of electromagnetic torque step responses between the system without and with flexibility distortion

## 7. The Application of Optimized Split-Radix FFT Algorithm in Suppression of PMSM Servo System Positioning Vibration.

### 7.1. The scheme for suppression of PMSM servo system positioning vibration.

In order to suppress the PMSM servo system positioning vibration, it is necessary to eliminate the position command signal's resonance frequency components. The scheme of suppressing position vibration of PMSM servo system is showed in Figure 14.

When positioning, speed vibration and position vibration are the same, so adopt a split-radix FFT algorithm to analyze position error to get vibration frequency, then filtering the position error through the notch filter.

Here the notch filter chooses improved double-$T$ net notch filter whose transfer function is as follows.

$$H(s) = \frac{as^2 + cs + 1}{as^2 + bs + 1} \qquad a = \frac{1}{\omega_0^2},\ b = \frac{k_1}{\omega_0},\ c = \frac{k_2}{\omega_o} \tag{15}$$

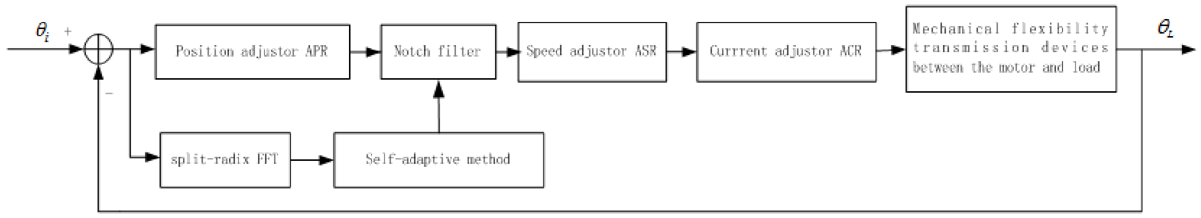


Figure 14. Scheme of suppressing position vibration

Among them, notch frequency $\omega_0$, notch bandwidth parameters $k_1$ and notch depth parameter $k_2$, these three variables can determine notch filter coefficients $a$, $b$, $c$. Configure the filter parameters $a$, $b$, $c$ needed to quickly and accurately obtain the resonance point, which requires the split-radix FFT algorithm to analyze resonance frequency.

7.2. **The split-radix FFT algorithm to detect and suppress vibration in MATLAB simulations.** As showed in Figure 15, it is the split-radix FFT spectrum of position error when positioning. Exclude the impact of the DC frequency components near 0 Hz, we can see the position vibration frequency is 17.58 Hz, which is the same resonance frequency as observed in Figure 13(a), and also verify the accuracy of the split-radix FFT spectrum analysis. According to the analysis results of resonance frequency, notch filter parameters can be configured and the notch filter can cut the resonance frequency part.

Focused on the PMSM servo system mechanical vibration, Figure 16 is the position waveform comparison of before and after suppression; Figure 17 is the speed waveform comparison of before and after suppression. According to the spectrum analysis results through the split-radix FFT algorithm transformation, then to configure the notch filter parameters, positioning vibration can be effectively suppressed.

8. **Conclusions.** In detecting mechanical resonance frequency of servo system, how to improve the accuracy of detection resonance frequency and shorten the time of the identification is a very important challenge. An optimized split radix FFT algorithm is proposed to detect mechanical resonance frequency of servo system in this paper. Verified by many

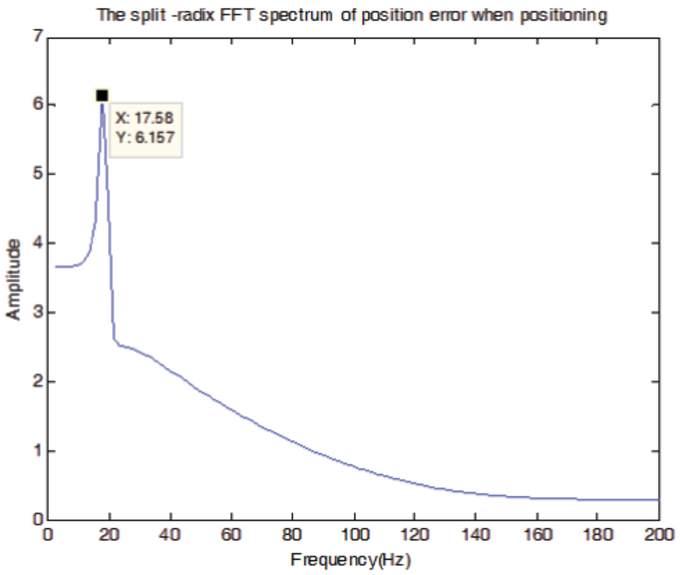


Figure 15. The split-radix FFT spectrum of position error when positioning

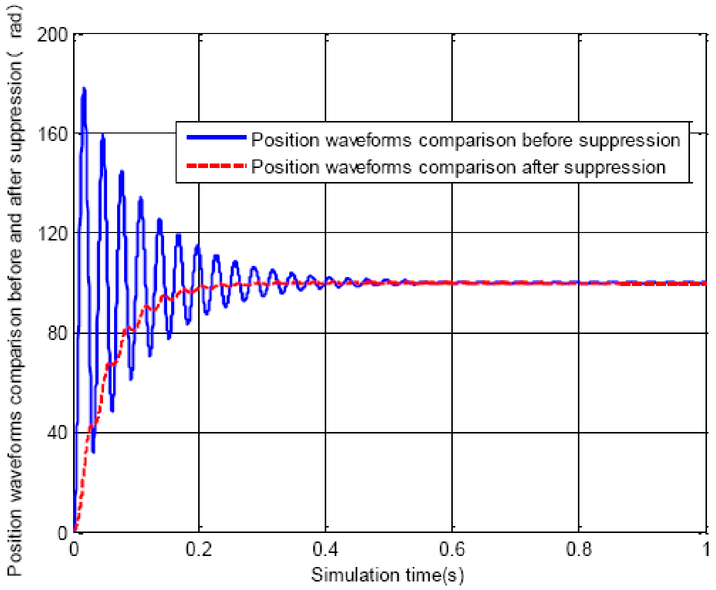


FIGURE 16. Position waveform comparison of before and after suppression

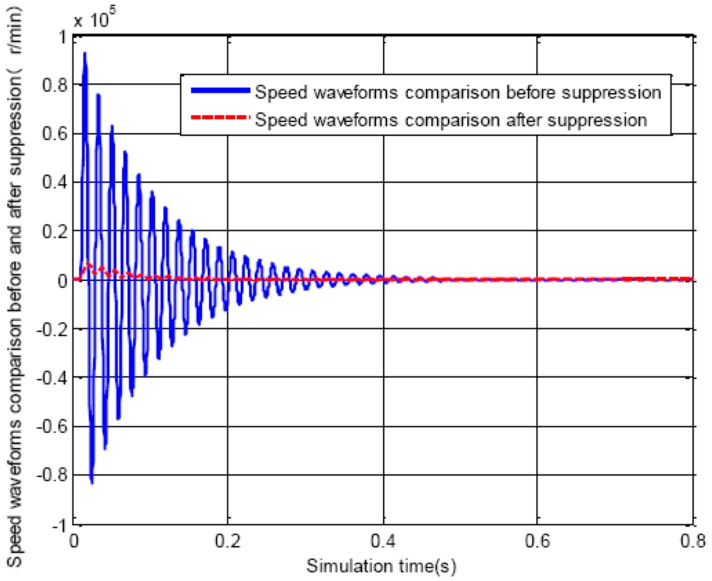


FIGURE 17. Speed waveform comparison of before and after suppression

experimental results based on Infineon XMC4500 floating-point chip, the optimized split-radix FFT algorithm can effectively detect the mechanical resonance frequency. In the case of guaranteeing the detection accuracy, the optimized split-radix FFT is much faster than radix-2, radix-4 and traditional split-radix FFT algorithms.

Finally, due to the elastic transmission device will cause PMSM servo system positioning vibration, the improved split-radix FFT algorithm is adopted to analyze position error signal, and according to the spectrum results obtain resonance frequency information to configure the notch filter parameters appropriately, and then filter resonance frequency parts position signal. So positioning vibration of PMSM servo system can be suppressed effectively.

**Acknowledgement.** This work was partly supported by National Nature Science Foundation of China (51375323, 61164014) and Qing Lan Project of Jiangsu Province, China. The authors also gratefully acknowledge the helpful comments and suggestions of the editor and reviewers, which have improved the presentation.

## REFERENCES

[1] T. M. O'Sullivan, C. M. Bingham and N. Schofield, High performance control of dual-inertia servo-drive systems using low-cost integrated SAW torque transducers, *IEEE Trans. Industrial Electronics*, vol.53, no.4, pp.1226-1237, 2006.

[2] G. Ellis and R. D. Lorenz, Resonant load control methods for industrial servo drives, *Proc. of IEEE Conf. on Industry Applications*, Rome, Italy, pp.1438-1445, 2000.

[3] K. Sugiura and Y. Hori, Vibration suppression in 2- and 3-mass system based on the feedback of imperfect derivative of the estimated torsional torque, *IEEE Trans. Industrial Electronics*, vol.43, no.1, pp.56-64, 1996.

[4] M. A. Valenzuela, J. M. Bentley and R. D. Lorenz, Evaluation of torsional oscillations in paper machine sections, *IEEE Trans. Industry Applications*, vol.41, no.2, pp.493-501, 2005.

[5] S. N. Vukosavic and M. R. Stojic, Suppression of torsional oscillations in a high-performance speed servo drive, *IEEE Trans. Industrial Electronics*, vol.45, no.1, pp.108-117, 1998.

[6] J. H. Wang, Y. X. Zhang, L. Xun et al., Torsional vibration suppression of rolling mill with constrained model predictive control, *Proc. of the 6th World Congress on Intelligent Control and Automation*, Dalian, China, pp.6401-6405, 2006.

[7] I. Masalliko, Torsional vibration suppression of a twin-drive geared system using model-based control, *Proc. of the 10th IEEE International Workshop on Advanced Motion Control*, Trento, Italy, pp.176-181, 2008.

[8] S. Colombi and T. Raimondi, Compliance compensation in mechatronic systems, *Proc. of IEEE the 20th International Conf. on Industrial Electronics, Control and Instrumentation*, Bologna, Italy, pp.946-951, 1994.

[9] J. Y. Hung, Control of industrial robots that have transmission elasticity, *IEEE Trans. Industrial Electronics*, vol.38, no.6, pp.421-427, 1991.

[10] J. Vittek, P. Bris, P. Makys et al., Control of flexible drive with PMSM employing forced dynamics, *Proc. of the 13th International Conf. on Power Electronics and Motion Control*, Pozan, Poland, pp.2219-2226, 2008.

[11] T. Ohmae, T. Matsuda, M. Kanno et al., A microprocessor based motor speed regulator using fast response state observer for reduction of torsional vibrations, *IEEE Trans. Industry Applications*, vol.23, no.5, pp.863-871,1987.

[12] K. Ohishi, Robust position servo system based on vibration suppression control for industrial robotics, *Proc. of the 2010 International Power Electronics Conference*, Sapporo, Japan, pp.2230-2237, 2010.

[13] H. Yang, Y. K. Fan and H. L. Shu, An improved digital filter for restraining mechanical resonance frequency, *Optoelectronic Engineering*, vol.34, no.1, pp.17-20, 2012 (in Chinese).

[14] Q. X. Zhu, H. L. Liu and K. Nikolay, A new design of notch filter in servo systems, *Microcomputer Information*, vol.24, no.7-1, pp.56-58, 2008 (in Chinese).

[15] L. Zhang, Y. G. Liu, Y. L. Fu et al., Study on simulation of AVI based on adaptive notch filter, *Journal of System Simulation*, vol.17, no.1, pp.234-237, 2005 (in Chinese).

[16] M. Yang, H. Hu and D. G. Xu, Cause and suppression of mechanical resonance in PMSM servo system, *Electric Machines and Control*, vol.16, no.1, pp.79-84, 2012 (in Chinese).

[17] S. Kumagai, K. Ohishi and T. Miyazaki, High-performance robot motion control based on zero-phase notch filter for industrial robot, *Proc. of the 11th IEEE International Workshop on Advanced Motion Control*, Nagaoka, pp.626-630, 2010.

[18] W. Y. Wang, J. B. Xu and A. W. Shen, Detection and reduction of high damping resonance for industrial servo, *Micromotors*, vol.45, no.11, pp.23-26, 2012 (in Chinese).

[19] J. A. Ferreira, P. Dorland and G. B. Freddie, An active inline notch filter for reducing acoustic noise in drives, *IEEE Trans. Industrial Electronics*, vol.43, no.3, pp.798-804, 2007.

[20] P. Schmidt and T. Rehm, Notch filter tuning for resonant frequency reduction in dual inertia systems, *Proc. of IEEE Conference on Industry Applications*, Phoenix, America, pp.1730-1734, 1999.

[21] L. H. Cai, H. B. Gao and S. M. Zhang, Research on method of eliminating accurately mechanical resonance frequency of the theodolite by complex falling filter, *Measurement & Control Technology*, vol.28, no.2, pp.58-61, 2009 (in Chinese).

[22] Y. Wang and H. Q. Huang, The application of split-radix FFT on harmonic detection in electric power system, *Instrumentation and Measurement*, vol.29, no.4, pp.73-76, 2010 (in Chinese).

[23] S. K. Sui, C. J. Sun and X. X. Zhou, The application of split-radix FFT on harmonic detection in electric power system, *Manufacturing Automation*, vol.33, no.6, pp.79-82, 2011 (in Chinese).

[24] H. Ai, Q. Q. Chang and D. W. Deng, Realization of FFT algorithm based on DSP, *Manufacturing Automation*, vol.34, no.1, pp.17-20, 2012 (in Chinese).