A Data-Collection and Fault Diagnosis System of Train Bearing Based on the C# and Multi-Wireless-Terminal

Haibin Zhang[1]*, Siliang Lu[1], Shangbin Zhang[1], and Fanrang Kong[1]

[1] Department of Precision Machinery and Precision Instrumentation, University of Science and Technology of China, Hefei, Anhui 230026, PR China.

* Corresponding author. Tel.: +8615805519140; email: zhbzhbyr@gmail.com
Manuscript submitted January 10, 2014; accepted March 28, 2014.

Abstract: With the development of modern railway transportation, the diagnosis of train bearing defects plays a significant role to maintain the safety of it and the higher speed, the more reliable detection we need. Among various defect detection techniques, acoustic diagnosis is widely used for detecting incipient defects of a train bearing as well as being suitable for wayside monitoring. But as we know, the acoustic signal we acquire from wayside is corrupted by the Doppler Effect and surrounding heavy noise including the noise of train body vibration, the noise of wheels, aerodynamic noise, etc. Besides, as the signal acts as a mixture of many bearings of the train, if we want to get the feature of one of them, we also need to have method to separate the signal. These are all quite difficult works. This paper will show us a new way to identify the conditions of bearings from a running machine. In this paper, we mainly describe the design of a software framework based on C# and the whole data transportation system. Ubiquitous computing technologies offer excitingly new possibilities for monitoring and analyzing of the train bearing conditions in real time. In our system, we put forward a whole serial port communication protocol which insures the data's accuracy during the wireless transmission process. We also get a thorough logic between the PC and wireless-terminals. In the software, we make it possible for engineers to monitor the bearings' condition in real time, check the parameters calculated from original data to identify the fault and set the communication parameters. Above all, the signals we acquire here are the vibration signal collected from the train bearings directly. It will make up most of the shortcomings compared to the acoustic diagnosis.

Key words: Software architecture, vibration data collection, wireless-terminal, wireless communication.

1. Introduction

The fault diagnosis of rotating machinery has attracted considerable research attention in recent years because such components as bearings and gears frequently suffer failure, resulting in unexpected machine break downs. And also rolling element bearings are considered as one of the critical mechanical components in industrial applications and their defects usually cause malfunction to some extent and even lead to failure of the machinery [1]. Besides, train speed increase is one of the main trends of railway development currently. For the speed increase, it is an important mission to guarantee the safety, the stability and uninterrupted operation of trains for passenger and freight transportation. There are hundreds of rolling bearings in a train with a significant relation for the train running [2]. As reported, bearing failure is the most common type of train faults [3], [4]. Therefore, the classification of rolling element bearing fault is an effective and necessary means to judge the bearing state and predict the lifetime so that some measures can be taken to avoid the occurrence of the catastrophic results.

In bearing defect diagnosis, increasing attentions have been paid to diagnostic techniques without disassembling the train bearings, such as oil monitoring, hot-box detection, acoustic signal and vibration signal analysis methods. These researches afford us the possibility to monitor the bearings' condition timely. Among the four methods, the oil monitoring means to obtain the lubrication and wear conditions related to the defects through analyzing the oil sample of lubricant used in train bearings. The technique is easy to be operated and effective to detect incipient fatigue damage of the bearing. However, as we know it is only suitable for oil lubrication bearings [5]. The hot-box detection method is to judge the degree of bearing wear through detecting the temperature condition of bearing box. There are two kinds of hot-box detection systems: one is on-vehicle system with a temperature sensor being set on the bearing housing, the other is wayside infrared detection system. Only in severe faulty conditions will the temperature of the train bearing raise. Thus, the temperature monitoring method does not have the capability of early failure detection, which is dangerous for the high speed trains [3].

The rest two methods, acoustic signal and vibration signal analysis, are the most two frequently-used ways. A lot of research works have been published, mostly in the last two decades, on the detection and diagnosis of bearing defects by vibration and acoustic methods. More and more researchers also pay much attention to these two ways as a result of their operability and reliability [1], [6]. But if we monitor the train bearing with the acoustic signal acquired from wayside, it is obvious that the measurement suffers from the drawback of signal attenuation and difficulty of signal processing, interpreting and classifying, as well as the difficulty of detecting inner race defects of a bearing [7]. Besides, as we discussed in the abstract, the acoustic signal we acquire from wayside is corrupted by the Doppler Effect and surrounding heavy noise [2]. So we need a lot of ways or algorithm to solve these challenges. But when it comes to the vibration signal analysis, all these problems can be avoided such as the Doppler Effect and so on. The analysis method for the vibration signal is studied by many researchers and it is more mature and easier relatively [1], [6], [8]. On the other hand, it will also bring us some inconvenience and difficulties. The accelerometers should be attached on the bearing housing for vibration signal acquisition, which would make the measurement system enormous and expensive. If we use wireless transmission with an independent power source, the system seems to be much simple and independent to the whole train system. But the accuracy of our data may be another challenge.

There are already some famous acoustic signal acquisition system of train bearing which are RailBAM by Australia and TADS by America [9]. While in this paper, we will focus on such a vibration signal acquisition system with a software architecture based on C#. In order to set it an independent system, our system has its own power source and all the data of vibration signal is transported wirelessly. The following Section 2 will show the structure of our system and the key point may be the upper computer software [10], [11]. As the most difficult part of our work is to guarantee the reliability of wireless transmission, we put forward a total communication protocol as discussed in Section 3 and the software in detail shown in Section 4. This part mainly talks about the logic of its communication with lower computer system and the peripheral hardware used in the data acquisition system. Section 5 refers to the rest function of the software and the system with the join of database also the run results with function realization. Finally, we have some concluding remarks and outlook in Section 6.

2. System Structure and Software Architecture

In this section we describe the overall structure of our system and software. It will offer us the overview of every part and how does our system work.

2.1. System Structure

Generally speaking, the system can be separated into two parts: the upper computer and lower machine. The upper computer takes the tasks of giving orders, system monitoring and controlling, data acquisition and verification, data processing, data storage and so on. The lower machine contains the acceleration sensor, MCU system with STM32 and wireless module of SM55.

Fig. 1 shows the whole structure of our system. We can see that it is a distributed system and has its own advantages [12], [13].

The upper computer consists of four wireless modules used to communicate with the lower MCU and a PC. Here the PC has whole software architecture to complete the missions we talked before. Why we have four wireless modules as shown in Fig.1 here? We know it is a challenge to improve the wireless communication rate. And it is a very important aspect here as we want to realize real time monitoring of the train bearings when running. So we use four different communication channels with four wireless modules. The train has 16 carriages at most. In this case, each wireless module takes charge of 4 carriages with 32 bearings. With the highest sampling frequency-20kHz and sampling time -3s, we have tested that each bearing will take almost 4 minutes to transmit the data. So it needs about 2 hours to acquire all the train bearing vibration signals in four channels which will save three fourths by only one channel.

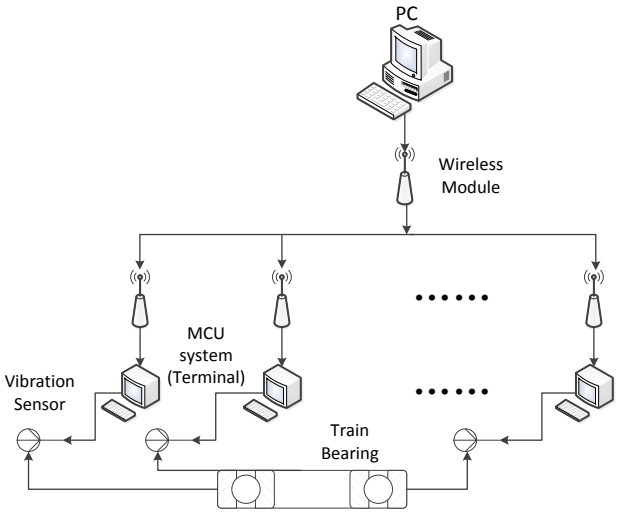


Fig. 1. System structure.

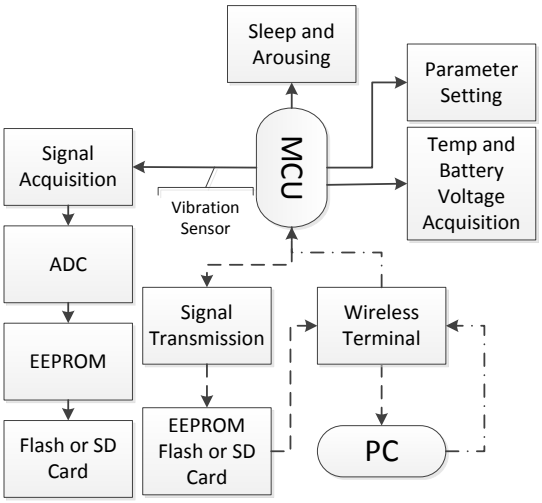


Fig. 2. Architecture of the MCU system.

The lower machine we call the terminal has a single chip microcomputer of STM32 which will execute functions of data acquisition, A/D converting, data transmission, sleeping and arousing, parameter setting, temperature and power checking, etc. Besides, each terminal is equipped with a wireless module to fulfill the data transmission. The architecture of the MCU system including hardware and software is displayed in Fig. 2.

2.2. Software Architecture

In the last part, we described the system structure. We refer to a PC which plays an important role in our system, and it works just as software we will show then. Fig. 3 gives the whole framework.

The part in the dashed box is the core of our system. The next two sections will focus on the communication protocol and logic of data acquisition program. As we use the P2P transfer method to save cost and it also facilitates the modular of our system, the logic of the program seems quite important to the robustness and reliability of the system.

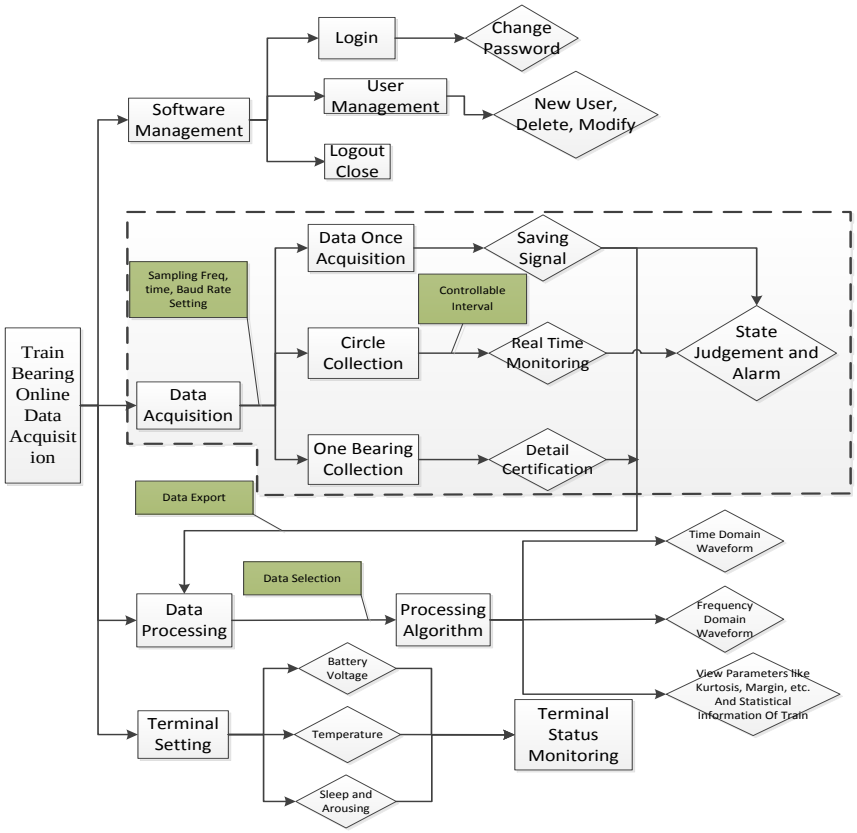


Fig. 3. Software architecture.

From Fig. 3, we know the software mainly includes four parts which are software management, data acquisition, data processing and terminal setting. The function of software management allows the user to modify password, build new users and delete users with using the database of Microsoft Access [14]-[16]. Because the signal we acquire here is the vibration signal from bearing directly, it gets rid of the noise, distortion and multi-source problem effectively. Then the data processing seems much easier here. We offer some indexes in the system like kurtosis, margin and some other statistical variables. It is also possible for the users to set the parameters like sampling frequency, sampling time, baud rate, etc. of each terminal from up computer. In the data acquisition part, there are three collecting modes. Data once acquisition means all the train bearings signal will be collected one by one until they are traversed. And all the data will be saved. Circle collection just monitors the bearings using the signal returned by terminals which just indicates the normal or not. In this mode, it will not offer the data in detail. The last mode is one bearing collection which allows the user to collect the data of a specified bearing. In this condition, if we suspect a bearing's result, we can collect only its signal again.

3. Data Communication Protocol

Communication protocol is formulated to make both of the facilities can understand each other. Due to the characteristics of wireless module, data may be transmitted with surrounding interference between the receiver and sender and the receiver will get the wrong data. So it needs communication protocols to ensure that the

receiver can get the data from sender correctly, and distinguish whether the data received conforms to the validation rules we make [17].

The wireless module we use here is industrial-grade narrowband called CC1101-SM55 working at a frequency band of 430/443MHz with power of 1000mW. Each terminal needs one wireless module and the PC is equipped with four modules working at different frequency. All the terminals may work at the same time. In order to guarantee the reliability of the data transmission, we need a robust and integrated communication protocol [18].

The data we need to transmit here includes instruction and vibration signal. The instructions can be separated into two kinds. One is up instruction and the other is called down instruction. And the original signal, considering the Bit Error Rate (BER) of the wireless module, is separated into each data package for transmitting. Every package carries data of a certain length. The next two parts will introduce them in detail.

3.1. Protocol for Instructions

According to the control commands and feedback commands we need to use in our system, a data format for instruction of three bytes is quite enough. As shown in Table 1, the first byte shows the order is up or down and the instruction type. Here the 0x (HEX) means the down instruction and the 4x (HEX) means the up instruction. The middle byte gives the information of the terminal's address. From a general perspective, there are at least 8 carriages in one train. Each carriage is equipped with eight bearings. So there are at least 64 terminals. This byte will be defined from 00 (HEX) to 3F (HEX). The last byte is made for error checking. Its value is the first byte OR the second one. With this information, we can recognize whether the instruction we receive is correct. For all the instructions, there are starting collecting, data transmitting, sleeping, arousing, sampling frequency setting, checking passed, etc. The up instructions are executing confirmation correspondingly to tell PC that I have executed it successfully.

Table 1. The Rule of Protocol for Instructions

Transmitting Direction	Instruction Contents	Terminal Address	Data Checking
Up	00-0F	00-FF	00-FF
Down	40-4F	00-FF	00-FF

Here we give an example for the protocol of instructions. Data of "020103" in HEX here is PC telling terminal to sleep. The first byte "02" shows the order is from PC to down machine and "2" here contains the information of sleeping. The second byte "01" means that the instruction is for the first terminal. "03" is the result of "02 | 01" for data checking.

3.2. Protocol for Vibration Signal

The signal transmission is the core of our system. We must make the data we acquire is correct and complete. As a limit of the wireless module's property, we can't transmit data too long at one time. If there exists one or more error bit, all the data has to be abandon and the time will be wasted. If we make the data into pieces, there will be slim chance for the small pieces of data to have error. Even if the error occurs, it will take little time to transmit this piece of data again. This is our core idea that we separate the data into small package to transmit. Besides, every package has a robust checking method to distinguish every bit error. Here we use the Cyclic Redundancy Check (CRC) [19].

Table 2. The Rule of Protocol for Signal Transmitting

Package Info	Start Code	Signal Piece	CRC Checking	Data Address
Length	3 Bytes	25 Bytes	1 Byte	3 Bytes
Detail(Example)	C001C1(Hex)	...	...	000101

Every package of signal has 32 bytes data as Table 2. The first 3 bytes are the start code with the middle byte telling the terminal address. The last byte is still for start code checking. The next 25 bytes are the signal piece. As every data point's value needs two bytes to describe, two packages will carry 25 data points. That means if the whole signal contains 20,000 data points, it will need 40,000 packages. The 29th byte is the CRC checking data for the signal piece. The last 3 bytes are for the data address, which offers the location of the data piece in the whole signal. The first two bytes gives the address information and the last for address checking. With this information, we can avoid the packet loss and this phenomenon is quite serious in our system. For example, if the wireless module loses the 99th package, the next package of 100th will act as it. But the 99th signal piece is high 8-bit of the 50th data point and the 100th low one. Now the 100th signal piece becomes the high 8-bit and all the rest data get wrong. In section 4, we will describe what our transmission logic is and how the data address makes the system to avoid this risk.

4. The Optimization Design of Data Transmission

The last section mentions that the data transmission is the core of our system and the reliability of signal we acquire seems extremely important. We need to try our best to reduce the impact of data error. For bit error, we have used the Cyclic Redundancy Check (CRC) and this method can check every one bit error. But what about the package losing? Fig.4 gives the answer.

Fig.4 shows the transmission procedure in detail. Here we take sampling frequency of 10kHz and sampling time of 3s of the first terminal for example. The length of signal will be 30k which occupies 60k data bytes. With each package carrying signal of 25 bytes, we can see a 2400 times circulation.

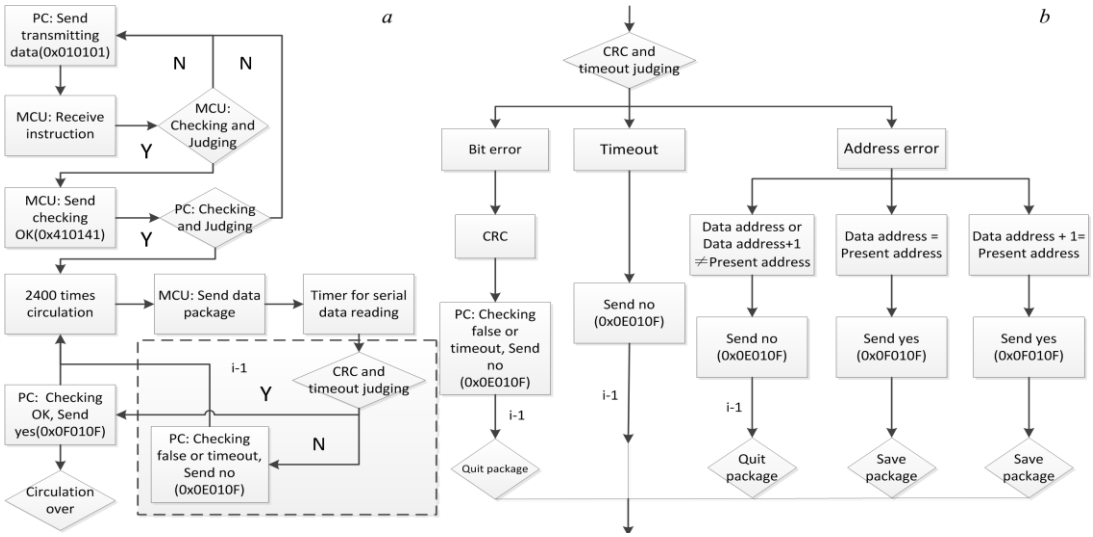


Fig. 4. Transmission logic and procedure. a). Original procedure. b). Optimized data checking procedure.

We try the transmission as procedure of Fig. 4. a) formerly. Firstly, PC sends an instruction of sending data to the MCU via the wireless module. If the MCU receives the instruction and checks OK, it will send a checking OK instruction to the PC and start to transmit the first package among 2400(60k bytes). Meanwhile, the PC get the checking OK and start a 2400 times' circulation. In every circulation, the PC waits for a 32 bytes' package in 1 second. If it receives the package successfully and checks OK, it will send an instruction of checking OK (0F010F) to the MCU, implying to transmit the next package. But if there is bit losing or bit error, the first one will lead to timeout and the other will lead to checking Failed. In both of the two cases, PC will send an instruction of checking Failed (0E010F) to the MCU, implying to transmit the last package.

Table 3. The Comparison of 3 Methods to Respond the Event of Data's Coming in Serial Port

Methods / Interval	Do While{ }	Timer	Event Trigger
10ms	1'58"	1'56"	Cannot respond
50ms	2'50"	3'02"	4'42"
100ms	5'10"	5'12"	6'18"
Thread Blocking	Yes	No	No

For the data reading, we use a timer from C# itself. System checks data to read in serial port every time tick kept by the timer. If the data length reaches 32 bytes, the timer will stop and system will catch the data in serial port. The next step is data checking as described before. After checking and sending of OK or Failure, the timer will start again to wait for the next package. There are several methods to respond the event of data's coming in serial port, why we use method or timer here? From Table 3, we can find the reason. The result in table is measured with 2400 packages. If the interval of every two packages is too short, it cannot respond every package successfully. So we find with a multiple interval, the consuming time is not of the same multiple. The first method we call Do While means we use a circulation to check the serial port. We have explained Timer before. The last one is Event Trigger which is an event delegation of C#. In the table, we can see that this method cannot respond rapidly when compared with the other two. And for the first one, it will occupy the main thread. So we use a timer for data reading here.

Now we have introduced the whole procedure of data transmission. But for the original procedure, there will be some deficiencies. The part in dashed box of Fig.4.a shows the process of checking Failed. It contains two situations. One is bit error and the other is timeout. The first error can be checked by CRC perfectly. In the second case, there are several reasons leading to timeout. Firstly, if the instruction of OK (0F010F) from PC to MCU loses, it means the lower machine will not get the order of checking OK. It will bring out the timeout and the PC sends another instruction of failing (0E010F) and MCU transmits the last package. But this package is already in and there will be one more which will bring reversal of high and low byte described as 3.2. Secondly, if the data package from MCU to PC loses, it will also lead to timeout. In this case, the PC will send instruction of checking Failed (0E010F). There will be no mistake. Thirdly, the lower machine has received the instruction of OK successfully. But it do not transmit data package for some reason. This case will lead to timeout again and the reversal of high and low byte may occur.

According to the description, we find the procedure of Fig.4.a is not reliable. In order to solve this problem, we detailing the process of checking in dashed box with package address as Fig. 4. b). The package address means the location of current package among the 2400. With this parameter, we can combine every package in the right order. For the bit error and timeout, it still works as before. We add another checking with package address. The PC has a criterion and it increases progressively during the circulation. The address information carried by every package will be matched by the criterion. If it equals the criterion, PC will save this package's signal and send instruction of checking OK and MCU will transmit the next package. If it equals the criterion minus 1, PC will also send instruction of checking OK but quit this package. In other situation, PC will send instruction of checking Failed and quit present package. MCU will transmit the last package. With this checking method, we can avoid the increase or decrease of package quantity which will lead to wrong signal.

5. Hardware, Software Interface and Experimental Result

Fig.5 shows us the main hardware of the system. The up computer consists of a PC and four wireless modules in different channel. The keynote of hardware design is the down machine we can also call the terminal. Each terminal is made of a control panel, a wireless module and an acceleration sensor. All the wireless modules are separated into 4 kinds of different channel to match the up computer. The control panel is the most important part of a terminal. It is equipped with an EEPROM of 1M bits, NAND Flash of 1G bits, 5 channels of ADCs, 4 channels of ULN 2803 and a temp sensor. Among the 5 channels of ADCs, four of them are set for acceleration

data (vibration signal) and the last for battery voltage. A temp sensor we used here is for the monitoring of terminal's temperature. These are the whole components of our hardware equipment. And now, we are getting down designing a structure to fix all the hardware, including the battery. We want our system to be an independent system from the train, even in the aspect of power source.

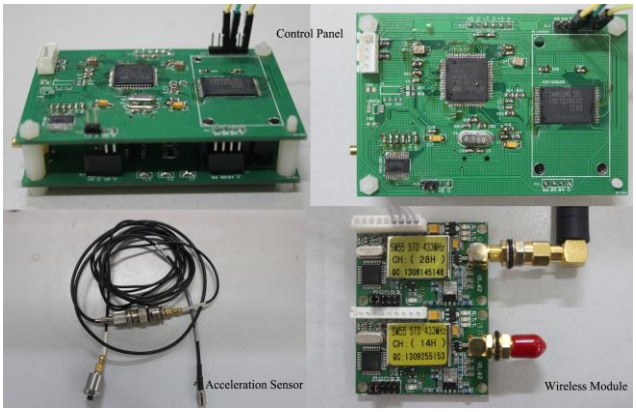
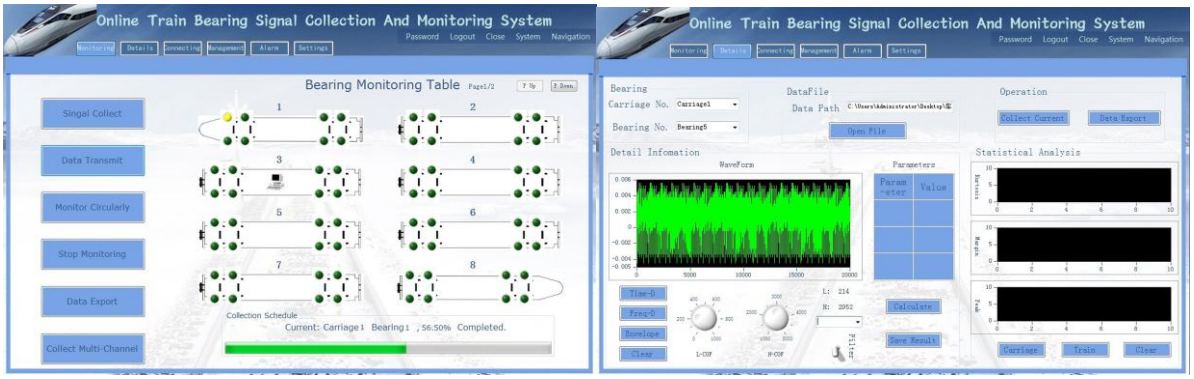
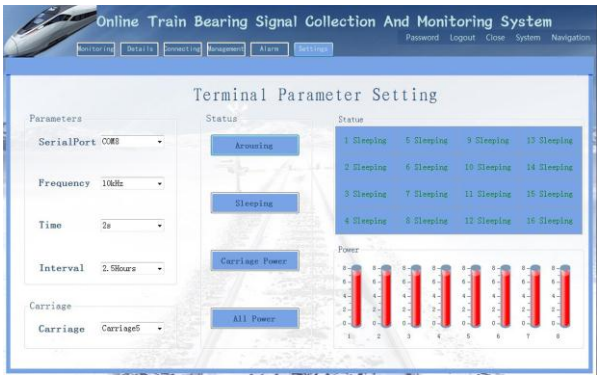


Fig. 5. Primary hardware of the system.

Next, we will show the design of software interface and some of the run-time interface. Fig.6.a is the all train monitoring interface. The software consists of six main interfaces. The first one allows us to monitor the whole train's status. And we used LED controls from Measurement Studio offered by NI to represent every bearing shown as Fig. 6. a). Yellow color shows the terminal is collecting data. Red color is an alarming color while green means normal. In this interface, we can control the terminals to collect or transmit data, start monitoring and so on. Fig. 6. b) is the detail interface of the software. If we click the "Details" button in menu bar, the interface will change to it. It can provide us with the signal waveform or spectrum which can be selected and has the function of processing it. In this panel, users can do filtering, see time domain and frequency domain waveform and calculate parameters and so on. The signal source can be either a new collecting one or the existing one. It allows users to open the existing data file and load them to the cache. The third to fifth interfaces are connecting checking, user management and alarm record which can be controlled by the menu bar still. We can switch the GUI just by clicking them. The last panel is parameter and terminal status setting shown as Fig. 6. c), with which we can select the serial port, set sampling frequency and time, arouse or make terminal sleep or watch their power conditions. The software also has the feature to decide whether the operation is for all terminals or just some of them.



a) b)



c)

Fig. 6. Primary interfaces of the software. a. Monitoring interface. b. Detail interface. c. Parameters setting interface.

Until now, our system design has been completed. It can do a perfect collecting and transmitting work already. We did some experiments using Matlab to simulate the bearing's vibration. The signal is imported into a loudspeaker box. When the sound plays, the cavity and diaphragm of the loudspeaker will start to vibrate. We have adhered the acceleration sensor to the diaphragm so that it can acquire the vibration signal (Fig. 7). In the other side, MCU will be controlled by the PC to collect and transmit the signal.

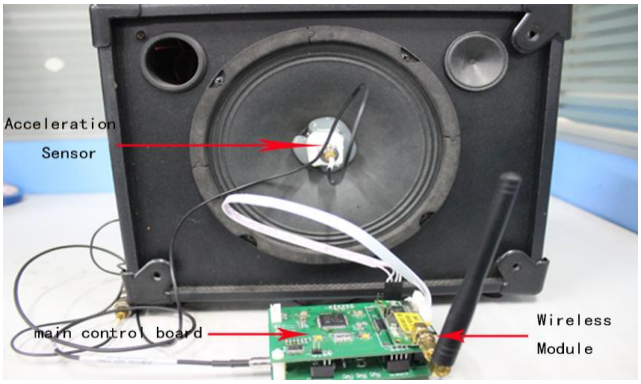


Fig. 7. Experimental device of loudspeaker box.

Here we use a simulation signal to examine the system:

$$x(k) = e^{-900\,\mathrm{mod}(k/8192,1/100)} \sin(17k/60) \tag{1}$$

In this equation, k= 0, 1, 2, …, 2047. Mod (a, b) here means the remainder. Characteristic frequency is 100Hz with sampling frequency of 8192Hz. We use 8192Hz here because the function of Sound (x, Fs) in Matlab needs a sampling frequency of integer power of 2.

The time domain waveform of original signal produced by equation (1) is shown as Fig. 8. a). It is a typical bearing fault signal which is a sinusoidal signal modulated by impulse signal. By the wave form, we can clearly distinguish an impact frequency of 100Hz. When we transform the signal to vibration, reacquire by the acceleration sensor with sampling frequency of 10 kHz and transmit to PC via our system, the signal we get is shown as Fig. 8. b). The initial phases of the signal are different which is easy to understand that the beginning of signal collecting doesn't match the starting of original signal. That means we produced the signal first and started the data collecting during the signal sequence. But we can see a consistent signal with the same impulse and impact frequency, with only attenuation of amplitude and quite little noise of high frequency. It gives the evidence of the reliability of our system.

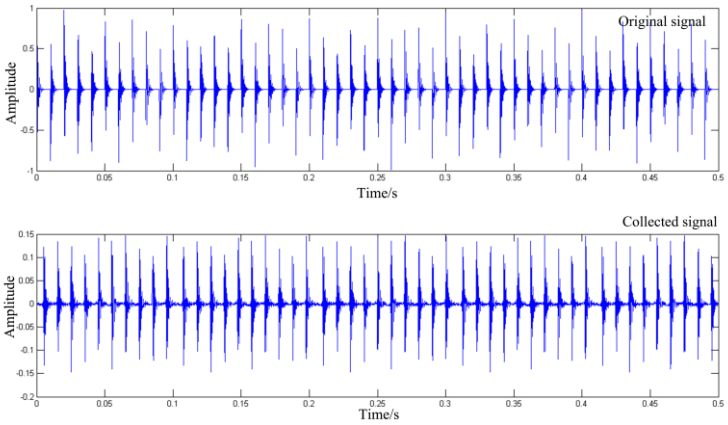


Fig. 8. Experimental result. a Original signal. b) Ignal acquired by the system.

6. Conclusions and Future Research

This paper proposes a new system for vibration signal collecting and transmitting wirelessly. Two serious problems affect the wayside acoustic signal processing, including Doppler shift and heavy noise interference. Many researchers try their best to find algorithm to reduce the noise or compensate the distortion. But when we use signal directly from the train bearing, the problem will be solved. So we try to design a system to acquire vibration signal of train bearing without impurity or distortion.

In this paper, we aim at the robust of the signal transmitting and user-friendliness as the reliability during the signal's transmission is the most important. We propose a whole communication protocol and optimized data package to increase the reliability during the process of data transmission. Experiment has approved our work's effectiveness.

While current study is only primary work. In next step, we will try to design the fixing device to fix our terminal on the train and try to collect real signal and check the reliability. We have already examined the shape below the train's bogie and designed a device. But it still need test and modify some parameters. Besides, we will improve the vibration signal processing method with the real signal.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (Grant No.51475441). The valuable comments and suggestions from anonymous reviewers are sincerely appreciated for their help to further improve the paper. Haibin Zhang thanks to Siliang Lu's work in this paper of some simulations and deduction. And also Fanrang Kong, Shangbin Zhang provide lots of suggestions and guidance. The authors also would like to thank the anonymous reviewers for their valuable comments and suggestions.

References

[1] Tandon, N., & Choudhury, A. (1999). A review of vibration and acoustic measurement methods for the detection of defects in rolling element bearings. *Tribology International, 32(8)*, 469-480s.

[2] He, Q., Wang, J., Hu, F., & Kong, F. (2013). Wayside acoustic diagnosis of defective train bearings based on signal resampling and information enhancement. *Journal of Sound and Vibration, 332,* 5635-5649

[3] Sneed, W. H., & Smith, R. L. (1998). On-board real-time bearing defect detection and monitoring. *Proceedings of the 1998 ASME/IEEE Joint Railroad Conference*, *Philadelphia* (pp. 149-153).

[4] Donelson III, J., & Dicus, R. L. (2002). Bearing defect detection using on-board accelerometer measurements, *Proceedings of the 2002 ASME/IEEE Joint Rail Conference*. Washington (pp. 95-102).

[5] Yan, X., Zhao, C. H., Lu, Z. Y., Zhou, X. C., & Xiao, H. L. (2005). A study of information technology used in oil monitoring, *Tribology International, 38*, 879-886.

[6] Shiroishi, J., Li, Y., Liang, S.,Kurfess, T., & Danyluk, S. (1997). Bearing condition diagnostics via vibration and acoustic emission measurement. *Mechanical Systems and Signal Processing, 11(5)*, 693-705.

[7] Al-Dossary, S., Hamzah, R. I., & Mba, D. (2009). Observations of changes in acoustic emission waveform for varying seeded defect sizes in a rolling element bearing. *Applied Acoustics, 70*, 58–81.

[8] Sheen, Y.-T. (2007). An analysis method for the vibration signal with amplitude modulation in a bearing system. *Journal of Sound and Vibration, 303*, 538-552.

[9] He, P., Li, P., Sun, H., & Sun, N. (2011). Acoustic signal acquisition system of train bearing. *Automation and Instrumentation, 10*, 8-11.

[10] Jeske, D. R., & Zhang, X. (2005). Some successful approaches to software reliability modeling in industry. *Journal of Systems and Software, 74*, 85-099.

[11] Herman, H., Trew, T., & Bosch, J. (2012). The changing industry structure of software development for consumer electronics and its consequences for software architectures. *Journal of Systems and Software 82*, 178-192.

[12] Andrea, G., Giovanni, P., Gennaro, T., *et al.* (2013). A mobile data collection platform for mental health research. *Pers Ubiquit Comput, 17,* 241–251

[13] George, A. D., Fogarty, R. B., Markwell, J. S., & Miars, M. D. (1999). An integrated simulation environment for parallel and distributed system prototyping. *Simulation, 72(5)*, 283-294.

[14] Liu, X., & Li, D. (2011). Creation and accessing of access database based on C#.NET. *Database and Information Management (In Chinese)*, *6*, 52-61.

[15] Ozgui, U. (1997). An evaluation of network access protocols for distributed real-time database systems. *Journal of Systems and Software, 37,* 49-60.

[16] Wang, W., & Wang, R. (2011). The encapsulation and reuse of ADO database access interface in software engineering. *Proceedings of 2011 Fourth International Symposium on Knowledge Acquisition and Modeling (KAM)* (pp. 47-50).

[17] Zhang, D., & Huang, B. (2002). Application of radio communication protocol and radio communication chip nRF401 in information-appliance. *Journal Of Zheng Zhou Institute Of Light Industry (Natural Science) (In Chinese), 17(2)*, 67-70.

[18] Al-Saqabi, K., Saleh, K., & Ahmad, L. (1996). Recovery from concurrent failures in communication protocols. *J. Systems Software, 35*, 55-65.

[19] Patrick, G. (1999). Cyclic redundancy check computation: An implementation using the TMS320C54x. *Digital Signal Processing Solutions*.

[20] Shen, C., Zhu, Z., Kong, F., & Huang, W. (2012). An improved morphological filtering method and its application in bearing fault feature extraction. *Journal of Vibration and Engineering, 25(4),* 468-473

[21] Abowd, G. D., Allen, R., & Garlan, D. (1995). Formalizing style to understand descriptions of software architecture. *ACM Transactions on Software Engineering and Methodology, 4(4)*, 319-364.

[22] Bieman, J., & Zhao, J. (1995). Reuse through inheritance: A quantitative study of C++ software. *ACM Symposium on Software Reuse*.

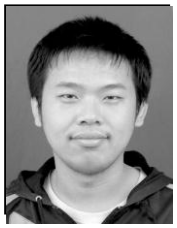
[23] Bucur, D., & Kwiatkowska, M. (2011). On software verification for sensor nodes. *Journal of Systems and Software*, *84*, 1693-1707.

[24] Kazman, R., Len, B., & Klein, M. (2006). The essential components of software architecture design and analysis. *Journal of Systems and Software, 79*, 1207–1216.

Haibin Zhang received the B.S. degree in mechanical engineering from the University of Science and Technology of China (USTC), Hefei, China, in 2010. He is currently a Ph.D. candidate with the Department of Precision Machinery and Precision Instrumentation, USTC. His major is digital signal processing, stochastic resonance, machine fault diagnosis and software system design.

He works in the PMPI of USTC as a Ph.D. candidate since 2011. During the period, he designed several software systems and had several publications like "Stochastic resonance with a Joint Woods-Saxon and Gaussian potential for bearing fault diagnosis", "A Tri-Stable stochastic resonance model and its applying in detection of weak signal". Mr. Zhang is now an IEEE student member.

Siliang Lu received the B.S. degree in mechanical engineering from the University of Science and Technology of China (USTC), Hefei, China, in 2010. He is currently a Ph.D. candidate with the Department of Precision Machinery and Precision Instrumentation, USTC. His research interests include condition-based monitoring, fault diagnosis and maintenance related to mechanical, electrical, and manufacturing systems by using the analog/digital signal processing approaches.



Shangbin Zhang received the B.S. degree in mechanical engineering from the University of Science and Technology of China (USTC), Hefei, China, in 2012. He is currently a Ph.D. candidate with the Department of Precision Machinery and Precision Instrumentation, USTC. His research interests include maintenance to mechanical, electrical, and manufacturing systems, application of singular value decomposition in fault diagnosis and data collecting system design.

Fanrang Kong received the B.S. degree in mechanical engineering from the Tsinghua University, Beijing, China, in 1978, the M.S. degree in mechanical engineering from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 1983, and the Ph.D. degree in mechanical engineering from Huazhong University of Science and Technology, Wuhan, China, in 1990. He is currently a professor with the Department of Precision Machinery and Precision Instrumentation, University of Science and Technology of China, Hefei, China. His research interests include innovative design, machine condition monitoring, automation and fault diagnosis.