PC-BASED FRAME OPTIMIZER USING MULTIPLE PCM FILES

Kwang-Ryul Koh, Sang-Bum Lee
Agency for Defense Development, Daejeon, Korea
kwang@add.re.kr

Taek-Joon Yi
Danam Systems, Seoul, Korea

Whan-Woo Kim
Department of Electronics Engineering,
Chungnam National University, Daejeon, Korea

ABSTRACT

Many engineers have tried to detect and correct erroneous data in telemetry communications. The best source selector can be used to combine data from two or more bit synchronizers to reduce frame error rates. An error-correcting code can be used as well. These techniques are absolutely helpful to obtain reliable telemetry data. However, some errors still remain and must be removed. This paper introduces the way to effectively merge multiple PCM files that are saved in different receiving sites, and shows nearly errorless data resulting from merging flight test data using a PC-based frame optimizer, which is a developed program.

Keywords : frame optimizer, best PCM frame, post data processing system.

INTRODUCTION

For many years, error detection and correction techniques have been employed in various communication systems, including telemetry units. Efforts to correct errors have been increasing as basic service for customers. Nowadays, customers take it for granted that the telecommunication systems include error reduction components. Most telecommunication environments are subject to channel noise: errors may occur during transmission from a transmitter to a receiver. Therefore, a lot of error-control methods for data transmission exist.

Among these techniques, forward error correction (FEC) codes are the most used techniques and have been used to improve link reliability in the telemetry field [1][2].

Automatic repeat request (ARQ) techniques are also used, and have played a key role in the development of error-control in many areas. However, in telemetry applications, FEC codes are used for error correction because of their one-way communication environments.

Along with these error-control techniques, using spatial diversity is another error reduction technique. For example, the best source selector can obtain a reliable bit stream because it produces a high fidelity output made up of the best combination of bits from multiple sources [3][4][5][6].

Many telemetry engineers use an error-correction code during encoding and select the best source selector, hoping to reduce errors. However, these methods do not satisfy their expectations every time. Thus, another method complementary to the abovementioned techniques must be found.

Most telemetry ground stations store a large amount of bits coming from multiple receivers located in different places. These bits can be used as very good resources for post error correction processing in a multiple-receiver configuration. In order to make a file by using bits, general information related to the bits is needed. This general information includes bit rate, word size, and file name, and the bits make a PCM file. The PCM file contains a lot of frames and a frame will be a unit of error correction processing.

In this paper, we propose a frame optimization method using multiple PCM files. This method has been designed and implemented using a sort-merge algorithm. To prove the effectiveness of this method, a PC-based frame optimizer, which is an application program, has been developed, and therefore can be commonly used on a desktop PC.

PC-BASED FRAME OPTIMIZER

In this section, we introduce a simple and plain proposal to solve the problem noted in the previous section. A PC-based frame optimizer is a frame-based error correction solution. The ultimate goal of this proposal is to choose an optimal frame among all candidates that are measured at the same time in different places, and to reconstruct the best PCM frame array using a sort-merge algorithm without any complex mathematical calculations.

For an optimal frame to be chosen, all frames in a PCM file must be extracted from the bit stream using sync codes in advance. Also, the frames have to be stored in a frame array for comparison with another array. All frames consist of a number of words. A word indicates a value, usually expressed in numbers, and obtained using a measurement sensor.

The basic problem of applying a sort-merge algorithm to a PC-based frame optimizer is to find a unique frame identifier that can consist of more than one word. Some words in a frame are usually used for time data. Each time data can be derived from a global positioning system (GPS) or can be increased by a minor frame counter. The frame counter will help to identify the frames in a PCM file. For this reason, the frame counter plays an important role in making an optimal frame array. Let C_{frame} be an n-bit counter[5]. The range of C_{frame} is expressed as follows.

$$C_{frame} \in [0, 2^{n} - 1] \quad (1)$$

If three fixed-length (n-bit) words in a frame are used for the frame counter, the number will increase to $2^{n\times 3} - 1$. Thus, the maximum measurement time can be calculated as follows.

$$T_{max} = \frac{2^{n\times 3}}{R_{frame}} \text{ (sec)} \quad (2)$$

where T_{max} is the maximum time of the frame counter and R_{frame} is the frame rate (frame per second). Thus the counter repeats every T_{max} seconds. To prevent the counter from wrapping to zero when it overflows, sufficient words have to be assigned into a frame for it.

After making the frame array from a PCM file, a PC-based frame optimizer checks if the array is sorted by the frame counter properly. Because erroneous frames can increase the problem of frame duplication, all erroneous frames have to be removed and replaced by Null frames before the PC-based frame optimizer finds the optimal frame in multiple frame arrays. A null frame means that all words in the frame are filled with a specific value, except for the synchronization code and frame counter. The frame counter in the null frames must add one to the previous frame counter. Once a normal frame array is completed, a PC-based frame optimizer makes a frame array from another PCM file, and starts to create the best PCM file using these two frame arrays.

The task to sort the target frames is the most expensive part of performing this frame optimization. Fortunately, a PC-based frame optimizer does not require an additional sort operation because the frames containing a PCM file are pre-ordered. The frame optimizer has only to eliminate erroneous frames. If the sorted frame arrays are created, the next step is to merge them by choosing the optimal frame. The merge function has two input frame arrays. First of all, the function gets the frame counters from the input parameters and compares them. If there is one frame with the same counter, the frame will be chosen without any comparison task. If there are two frames with the same counter, the function will compare them and choose the optimal one. The best PCM file is created by the chosen frames. The sort-merge algorithm is shown in Figure 1.

```
Function SortMerge(Byval leftArray as FrameArray, Byval rightArray as FrameArray)
  ....
  While leftArray .count > 0 AndAlso rightArray.count > 0
    Dim lcounter as Int32 =  GetFrameCounter(leftArray (i))
    Dim rcounter as Int32 =  GetFrameCounter(rightArray (j))
    If lcounter < rcounter Then
      WriteFrame(leftArray(i))
      i += 1
    ElseIf lcounter = rcounter Then
      Dim f as Frame = GetBetterFrame(leftArray(i), rightArray(j))
      WriteFrame(f)
      i += 1
      j += 1
    Else
      WriteFrame(rightArray(j))
      j += 1
    End If
  End While
  .....
End Function
```

Figure 1. The sort-merge algorithm.

Let us say that we have 3 PCM files and the first frame array has 7 items, the second frame array has 7 items, and the third frame array has 4 items, and we create a best PCM file with these items. A PC-based frame optimizer tries to choose the optimal frame among the candidates with the same frame counter in 3 different frame arrays, as show in Figure 2.

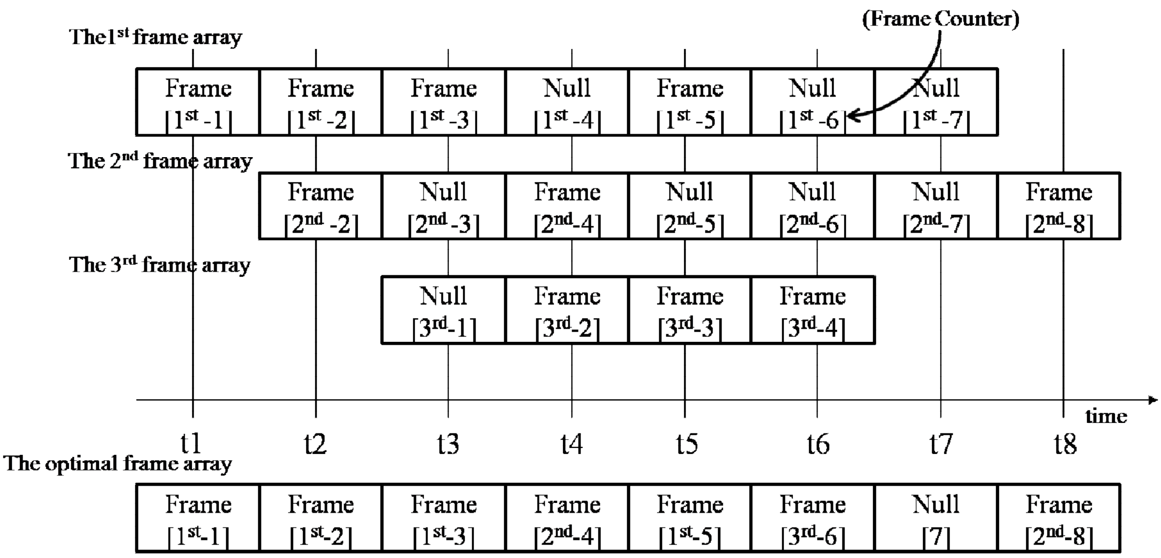

Figure 2. The method of optimal frame selection.

The first frame array contains 4 normal frames and 3 null frames between t1 and t7. The second frame array contains 3 normal frames and 4 null frames between t2 and t3. Finally, the third frame array contains 2 normal frames and 2 null frames between t3 and t6. A normal frame means one of the candidates for the optimal frame at a particular moment. The start and end times of all frame arrays are slightly different. However, the best PCM file has all the frames between the earliest start time (t1) and latest finish time (t8) across all frame arrays. All items of the sorted frame array are compared to each other via a frame counter.

The frame comparison task does not use all the frame arrays at a time. If 3 frame arrays are extracted from 3 PCM files, a better PCM frame array will be created from two frame arrays. The better PCM frame array will be then compared to the third. As a result, two comparison and merging tasks are needed to make the best PCM frame array. This array will be the source to make an output PCM file that we wish to be error free. The frame comparison and merging process are shown in Figure 3.

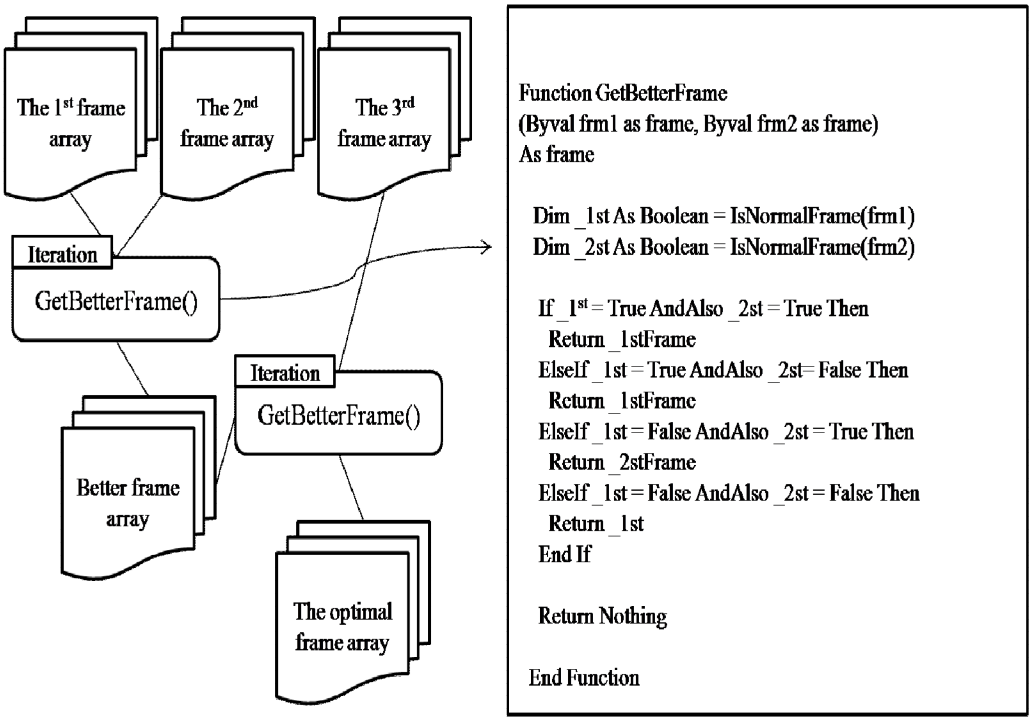

Figure 3. The frame comparison and merging process.

To select the best frame, a very simple function is used. In figure 3, the right side code shows the function to select the optimal one from two input frames. First of all, each frame must be checked to determine whether the frame is normal. If the first input frame is normal, the function returns the first input frame without checking the second input frame. And if two input frames have an error, the function returns the first input frame as well. A checksum, CRC or constant matrix can be used for determining whether the frame is normal. This comparison task continues to the end of the latest frame across all the frame arrays.

PERFORMANCES

To show the performance of the PC-based frame optimizer, two PCM files stored at different ground stations, and one PCM file stored by the best source selector are used. The configuration of two ground stations and the best source selector is shown in Figure 4.

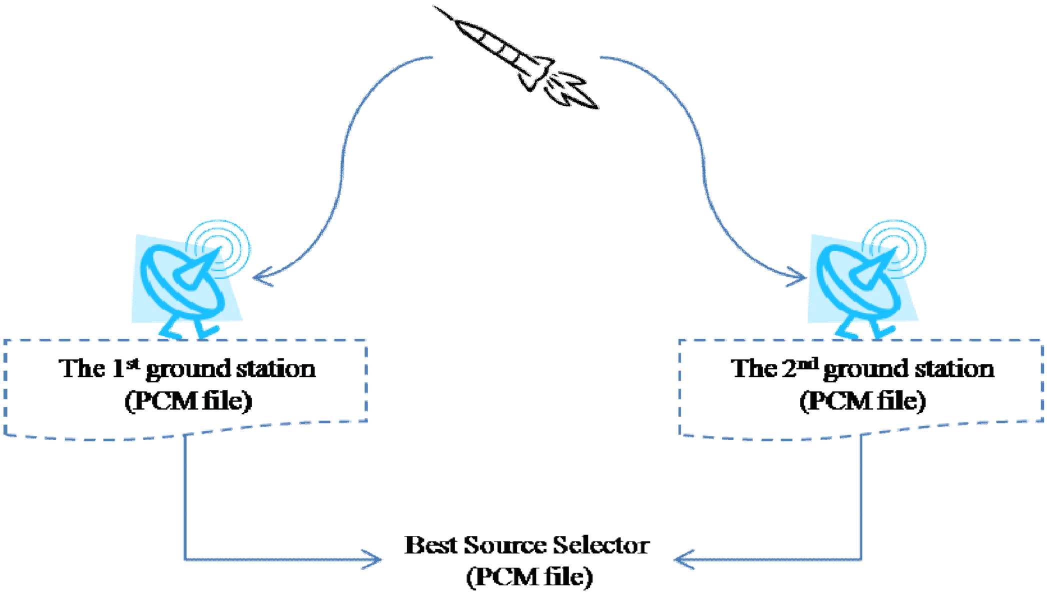

Figure 4. The two ground stations and the best source selector.

We usually operate multiple receiving sites for flight tests. Once a test is completed, multiple PCM files are created. Also the best source selector is used to obtain more reliable data. Errors caused by noise or other impairments may occur in different time slots during transmission from the transmitter to the receiver. For this reason, these files are complementary to each other. If a frame has an error, 1 is plotted on the y axis. If not, 0 is plotted on the same axis. The frame counter is plotted on the x axis. The frame errors are drawn in many different forms. The frame errors that occur in the 1st ground station are shown in Figure 5.

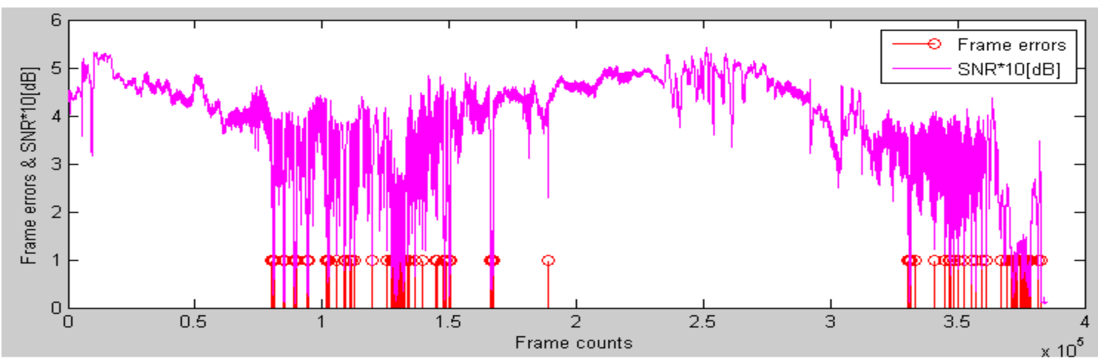

Figure 5. The frame errors that occur in the 1st ground station.

The total frame length of the 1st ground station is 382,776, and the number of frame errors of the 1st ground station is 9,983. As a result, the frame error rate of the 1st ground station is about 0.026. The 1st ground station's frame errors mostly occur when the signal-to-noise ratio drops below 20 dB. A weak section of signal-to-noise ratio is when frame counter is between 0.8×10^5 and 1.5×10^5, or between 3.3×10^5 and 3.8×10^5.

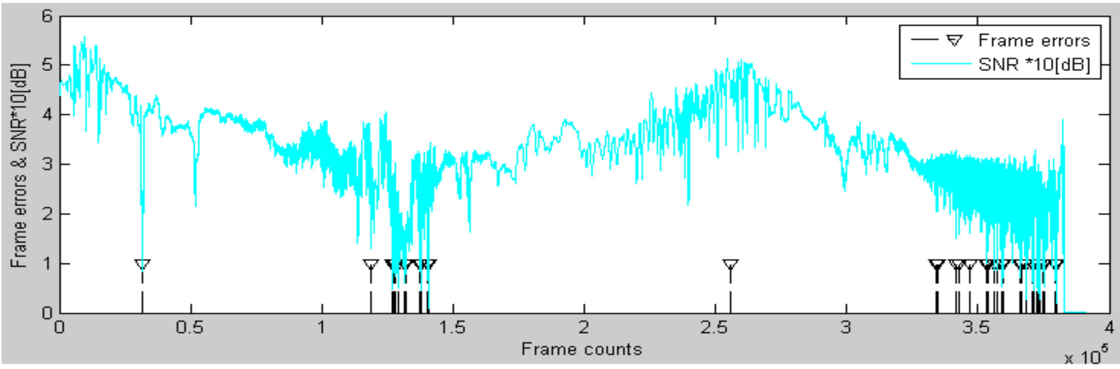

Figure 6. The frame errors that occur in the 2nd ground station.

Frame errors that occurred in the 2nd ground station are shown in Figure 6. The total frame length of the 2nd ground station is 382,776, and the number of frame errors of the 2nd ground station is 1,494. As a result, the frame error rate of the 2nd ground station is about 0.0039. This figure is lower than the frame error rate of the 1st ground station. The 2nd ground station's frame errors mostly occur when the signal-to-noise ratio drops below 20 dB as in the 1st ground station. A weak section of signal-to-noise ratio occurs when the frame counter is between 1.3×10^5 and 1.4×10^5, or between 3.5×10^5 and 3.7×10^5. Frame errors in the best source selector's frame array are shown in Figure 7.

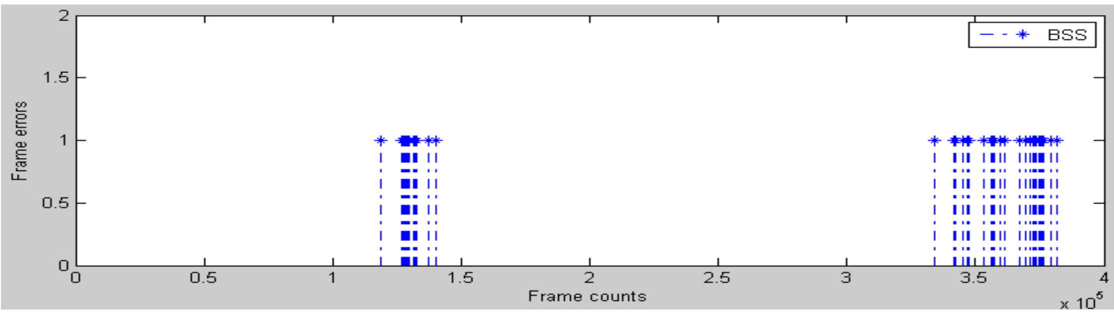

Figure 7. The errors in best source selector's frame array.

The total frame length of the best source selector is 382,776, and the number of frame errors of best source selector is 2,558. As a result, the frame error rate of the best source selector is about 0.0066. This figure is higher than the frame error rate of the 2nd ground station. Unfortunately, the result does not meet our expectations. Finally, errors in the optimal frame array are shown in Figure 8.

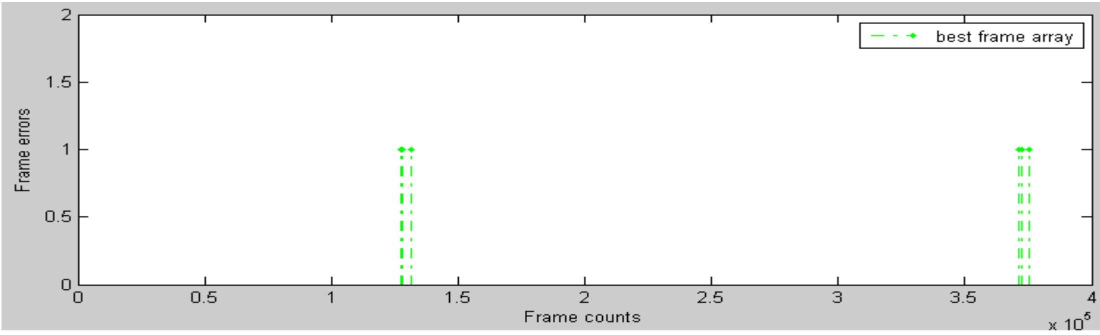

Figure 8. The errors in the optimal frame array.

The total frame length of the best source selector is 382,776, and the number of frame errors in the optimal frame array is 184. As a result, the frame error rate of the optimal frame array is about 0.0004.

Site	Total Frame	Error Frame	Error Rate
The 1st ground station	382776	9983	0.0260
The 2nd ground station		1494	0.0039
Best Source Selector		2558	0.0066
PC-based frame optimizer*		184*	0.0004*

Table 1. The performance of a PC-based frame optimizer

Table 1 shows the performance of the PC-based frame optimizer. As can be seen in Table 1, the error rate is drastically reduced. This is a remarkable result. In addition, it does not influence the complexity of the system, and it is not difficult to design and implement a PC-based frame optimizer at low cost.

CONCLUSIONS

In this paper, a PC-based frame optimization method using a sort-merge algorithm has been introduced and its validity has been verified. As a result, the method can provide more reliable telemetry data to test and analysis engineers. In a flight test, a simple bit error or frame error can be detected and corrected by a parity check or error-correcting code. However, if burst errors occur by accident, it is not easy to recover the original data. A PC-based frame optimizer can help to correct consecutive corrupted frames without requiring any additional equipment. The design of the application is simple and easy enough for anyone to understand. If burst errors occur in different time slots, the PC-based frame optimizer will provide more satisfying results.

REFERENCES

[1] Brett W. Werling, “Implementation of an LPDC Decoder Using Functional Programming Languages,” Proceedings of the 45th Annual International Telemetry Conference, USA, October 26-29, 2009.

[2] Bob Hayes, "Low Density Parity Check Codes For Telemetry Applications," Proceedings of the 43th Annual International Telemetry Conference, USA, October 23-25, 2007.

[3] Richard (Ray) Engler, “Telemetry Best Source Selection at White Sands Missile Range,” Proceedings of the 40th Annual International Telemetry Conference, USA, October 18-21, 2004.

[4] Diarmuid Corry MSc, “Measuring and Evaluating Best Source Selection,” Proceedings of the 45th Annual International Telemetry Conference, USA, October 26-29, 2009.

[5] Michael J. Wilson, “Merging Telemetry Data from Multiple Receivers,” Proceedings of the 40^{th} Annual International Telemetry Conference, USA, October 18-21, 2004

[6] Michael A. Forman, “Quantifying Coding Gain from Telemetry Data Combining,” Proceeding of the 44^{th} Annual International Telemetry Conference, USA, October 27-30, 2008.