Design of Image Recognition Algorithm for Critical Flight Parameters in Civil Aircraft Display System

To cite this article: Maodi Zhang et al 2020 IOP Conf. Ser.: Mater. Sci. Eng. 751 012077

View the article online for updates and enhancements.
Design of Image Recognition Algorithm for Critical Flight Parameters in Civil Aircraft Display System

Maodi Zhang*, Yuanhui Zhou and Qingfan Gu
China National Aeronautical Radio Electronics Research Institute, Shanghai, 200241, China

*Corresponding e-mail: zhang_maodi@careri.com

Abstract. In the current civil aircraft display system design, complex COTS devices such as CPU and GPU have been widely used. While improving product performance, it also greatly increases the difficulty of airworthiness certification. Taking the display system as an example, the data integrity of the critical flight parameters which is displayed on the display unit must reach $1.0 \times 10^{-7}$ or higher, and the development assurance level of each software and hardware module must reach level B or higher. However, according to industry experience, the data integrity of CPU and GPU can only reach the level of $1.0 \times 10^{-6}$. Therefore, for the critical flight parameters in the display system, it must be monitored to cover the entire data path from data processing to image generation to prove the data validity. Aiming at the numeric critical flight parameters of civil aircraft display system, this paper proposes a numeric image recognition technology realized by FPGA image processing function, which can be used to monitor the integrity of numeric critical flight parameters, and this technology has been performed and verified in real display unit.

1. Introduction
According to the current display system HMI design concept, the airspeed, barometric altitude which is from the flight control, and the engine parameters from the FADEC are generally in the form of numbers or letters, and will have a scrolling display effect. According to the safety assessment of display system, the display function of airspeed, barometric altitude and engine parameters shall be developed to a minimum of FDAL B or A, and the corresponding failure probability shall be $1 \times 10^{-7}$ or $1 \times 10^{-9}$ [1] [2]. In order to meet the safety requirements of these critical flight parameters (including DAL level and failure probability), each type of aircraft (such as A380, B787, etc.) will design corresponding monitoring means, it may be a cross monitoring between display unit [3], or a monitor of video processing channels inside a single display unit. The core of all the above monitoring methods is to obtain critical flight data from the current display screen, and compare and judge the data with the original data to ensure the integrity of the data.

The most direct and effective way to get data from the screen is through image recognition technology [4] [5]. Currently, the image recognition technology has been widely used in daily life, such as license plate recognition [6], face recognition [7] [8], etc. Therefore, this paper proposes an image recognition method based on pixel matching, which is used to identify the numeric critical flight parameters in the civil aircraft display unit, which can be used for the integrity monitoring of these data and improve the safety of the display system.
2. Design and Implementation

2.1. Technical Scheme
The technical scheme adopted in this paper is based on FPGA image processing and pattern recognition technology. It is based on the display characteristics of the numeric display parameters of the airborne cockpit display system, using the pixel normalization matching technology to identify the numeric critical parameters in the current display screen. Firstly, according to the display characteristics of the numeric flight parameters, the corresponding matching template needs to be developed. For example, the airspeed and barometric altitude are scrolling display, and the engine parameters are in direct change form. Each parameter needs to be templated according to the possible display state. Then, binarize the received image data to reduce the matched noise, and the corresponding region is intercepted according to the display position of each parameter. Finally, FPGA-based pixel block matching is used to compare the pixel values of the captured image with the display template line by line and select the number corresponding to the position with the largest similarity measure as the matching result.

In order to verify the results of image recognition in a real display unit, we designed a data comparison function in the FPGA. While identifying the critical flight parameters in the display, the FPGA also accepts the original data sent from the CPU via the SPI and compares the two data. If the result of the data comparison is within an acceptable threshold range, then the identified data is considered correct. The whole structure is shown in figure 1.

![Figure 1. The verification architecture for the image recognition algorithm on display unit.](image)

2.2. Implementation Details
When the airspeed, barometric altitude and engine critical parameters are displayed normally, the display number changes continuously from 0 to 9 according to the input. During the display of
airspeed and barometric altitude, when the highest bit is 0, it is displayed as a grid line, and the barometric height may be negative. The display format and template is shown as figure 2.

Taking airspeed data as an example, after receiving the data input, the airspeed identification module intercepts the display areas of the digits, tens, and hundreds of digits, and then loads the respective display templates from the ROM. In the process of identifying each digit, it is necessary to instantiate one input data buffer RAM, one font template storage ROM, one matching module and one comparison positioning module. Then, the block image matching algorithm is used to match the input image data with the font template. The matching module adopts a parallel structure to improve the matching speed, and can complete the matching operation between the font template and the graphic data of 21 rows and 14 columns and each of the pattern shift registers inside the matching module has a tap number of 14 and a width of 8. During the matching operation, each unit registers one pixel of the graph, and shifting one unit to the right every other clock, to realize a point-by-point movement of a font template in an image, and then complete a real-time comparison between font template and image data. The arithmetic unit calculates the square of the error value between the image and the font template using (1), in which \( p_{ij}, q_{ij} \) represents the values of the corresponding pixels in the image block and the template. The main processing is shown as figure 3.

\[
BM_{pat} = \sum_{i,j} |p_{ij} - q_{ij}|^2
\]  

(1)

Figure 2. Display format and template for numeric critical parameters.

Figure 3. Airspeed identification process.
Since the digital display of airspeed and barometric altitude is scrolling, and it is possible that two consecutive numbers are incomplete, we have designed a digital positioning module. The digital positioning module is used to compare the calculation results of the matching metrics to determine the position of the best matching point. It has an internal input latching module which is used to latch the matching metrics to ensure data stability during operation. The comparison judging module compares the input data with the previous latched data and latches the smaller data for the next comparison. The address counting module is responsible for recording the number of matching metric values in the input latches participating in the comparison, so that the location of the font template search area corresponding to the metric value can be determined. When all the matching metrics are compared, what remains in the address latch is the position of the minimum matching metric in the entire sequence of matching metrics, thereby obtaining the position of the best matching point and converting it to the number of the corresponding data bit.

3. Experiments and Analysis
In order to verify the effectiveness of the technical method, we use simulation and real environment to implement verification activities.

3.1. Simulation Experiments
The simulation verification environment includes MATLAB and ModelSim. MATLAB simulation is mainly used for algorithm function verification in the early design stage. ModelSim simulation is mainly for FPGA code to verify that the FPGA code can meets the function and performance requirements.

The MATLAB simulation result is shown as figure 4:

![Figure 4. MATLAB simulation results.](image)

On the left is part of the input video frame, we can see that the currently displayed airspeed number is between 199 and 200. The MATLAB simulation program gives the recognition result of each digit. In the recognition result of each digit, the left side represents the picture taken in the video frame, the right side represents the display font template, and the middle represents the position of the picture taken on the left side in the font template.

The ModelSim simulation result is shown as figure 5:

![Figure 5. ModelSim simulation results.](image)
Through the picture, we can see that the airspeed readout in the picture is 31, and the Modelsim simulation result of the image recognition algorithm is 41F80000, which is converted to a decimal number of 31, and this result is consistent with the airspeed displayed in the picture. Meanwhile, we can see that the calculation time of the airspeed recognition algorithm is about 6.18ms, considering the refresh frequency of the display frame is 60Hz, which is about 16.7ms. Therefore, the algorithm can complete data identification in the display time of one frame, and to realize the recognition of every display frame.

3.2. Experiments on Display Unit

The verification activities in the real display unit cannot be observed by us directly, so we designed a SPI protocol between FPGA and CPU application, which is used for the CPU application sent original data to FPGA, and the FPGA return the compared result to CPU. We use the CPU as the master of the SPI bus to control the SPI clock and data transfer. As the Slave end, the FPGA accepts the original data and returns the comparison result according to the SPI clock given by the CPU.

3.2.1. Send frame data definition. The send frame data includes: frame header, airspeed data, airspeed invalid or loss flag, barometric altitude data, barometric altitude invalid or loss flag, wherein each part is a 32-bit data. The send frame format is as table 1 (Take airspeed as an example):

| Definition          | Type          | Length (32 bit) | Note                                                                 |
|---------------------|---------------|-----------------|----------------------------------------------------------------------|
| Frame Head          | UINT 32       | 1               | Frame header information, can be defined by a 32bit like 0xFFFFFFF0;  |
|                     |               |                 | Send by CPU                                                          |
| Airspeed_Num        | Float 32      | 1               | 32bit float data, ranging from 30 to 450. Send by CPU              |
| SPD_Loss_Tag        | UINT 32       | 1               | 32bit integer data, send by CPU: 1) If the airspeed is valid, set as 0x00000000; 2) If the airspeed is invalid, set as 0x00000001; 3) If the airspeed is loss, set as 0x00000002; |

3.2.2. Receive frame data definition. The received frame data includes: a frame header, an airspeed data comparison result, a barometric altitude data comparison result, an engine parameter comparison result, wherein each part is a 32-bit data. The received frame format is as table 2 (Take airspeed as an example):

| Definition          | Type          | Length (32 bit) | Note                                                                 |
|---------------------|---------------|-----------------|----------------------------------------------------------------------|
| Frame Head          | UINT 32       | 1               | Frame header information, can be defined by a 32bit like 0xFFFFFFF1;  |
|                     |               |                 | Send by CPU                                                          |
| Airspeed_Result     | Float 32      | 1               | 32bit integer data, send by FPGA. 1) If the comparison data is normal, set as 0x00000000; 2) If the comparison data is error, set as 0x00000001; 3) If the comparison data is fault flag, set as 0x00000002; |

The data comparison between original data and identified data need a reasonable threshold. We set the threshold for airspeed as 5, and the threshold for altitude as 20. During the FPGA operation, the SPI data and identified data can be captured through SignalTap. The result is shown as figure 6:
Figure 6. SignalTap result of experiment on display unit.

Table 3. Data compare result.

| Parameter | Recognition Result | SPI Input     |
|-----------|--------------------|---------------|
| Airspeed  | 78 (429C0000)     | 78.33 (429CA9D8) |
| Altitude  | 9520 (4614C000)   | 9514 (4614A800)   |

According to the recognition result obtained by the capture and the SPI input data shown in table 3, the data of the two are basically consistent in consideration of the delay of the data transmission and the recognition accuracy. This also proves the effectiveness of this monitoring function.

4. Conclusion
This paper proposes an image recognition method based on FPGA pixel matching, which can be used to identify digital critical flight parameters in civil aircraft display systems. After tens of thousands of simulation tests, and dozens of consecutive tests in the display unit, the airspeed maximum recognition error does not exceed 1, and the barometric altitude maximum recognition error does not exceed 20. This error is acceptable for safety requirement. So this image recognition method is effective for critical flight parameters in civil aircraft display system.

References
[1] AC 25-11A 2007 Electronic Flight Deck Displays (Washington D.C.: FAA)
[2] Dong H, Gu Q, Wang G, Zhai Z, Lu Y and Wang M 2019 Processes 7 p 117
[3] TC 15/62 2016 Single Event Effects Mitigation Techniques Report (Washington D.C.: FAA)
[4] McAndrew A 2004 Introduction to Digital Image Processing with MATLAB (Boston: Victoria University of Technology)
[5] Seo J and Yoo C 2006 IEEE Trans. Signal Process 54 pp. 1537-1549
[6] Wang F, Zuo W, Lin L, Zhang D and Zhang L 2016 Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (Las Vegas: IEEE) pp. 1288-1296
[7] Boles W and Boashah B 1998 IEEE Trans. on Signal Processing 46 pp. 1185-1188
[8] Jain A, Bolle R and Pankanti S 2006 Biometrics: personal identification in networked society (New Jersey: Springer Science & Business Media)