Research on Optimization Strategies of Infrared Image Enhancement Technology in Embedded Systems

Chen Xianbin

Aviation Industry Optoelectronics Institute, Luoyang, Henan, China

Abstract: Infrared images often face problems such as insufficient contrast and lack of texture information. Usually, image enhancement techniques are needed to improve the quality of infrared images. Embedded systems are the main hardware platforms for the implementation of infrared image enhancement algorithms. Their characteristics of small size, low power consumption and limited computing resources limit the possibility of implementing complex algorithms. This paper focuses on discussing the feasibility of efficiently implementing image enhancement algorithms in the context of embedded systems, and mainly introduces algorithm optimization strategies, hardware acceleration strategies and collaborative optimization strategies.

Keywords: Infrared Image; Embedded System; Resource Limitation; Image Enhancement; Optimization Strategy

1. Introduction

With the continuous advancement of embedded technology, the application scope of infrared image enhancement technology in multiple fields such as military, security, and medical care is expanding day by day. However, in view of the limitations of the hardware configuration of embedded systems, how to optimize and improve the image enhancement effect while ensuring processing speed and efficiency has become a challenging issue in this field. This paper conducts an in-depth analysis of the current application status of infrared image enhancement technology in the field of embedded systems, proposes improvement measures, and conducts an in-depth study on the problems encountered in the actual application process and their coping strategies.

2. Overview of Infrared Image Enhancement Technology

Infrared image enhancement technology is mainly applied to improve the clarity and quality of the images obtained by infrared image systems, optimize the visual effect and detail representation of the images, aiming to more efficiently meet various application requirements, such as security monitoring, military reconnaissance, medical imaging and other fields. Compared with visible light images, the image quality of infrared images is generally not high. There are often problems such as insufficient contrast, severe noise interference and low resolution. To effectively address such challenges, the emergence of infrared image enhancement technology is timely. This technology significantly improves the image quality and strengthens the detailed information of the image through a series of optimization measures.

Common infrared image processing techniques include histogram equalization, contrast expansion, noise cancellation and edge enhancement, etc. Histogram equalization technology involves the adjustment of pixel distribution and the equalization adjustment of image luminance distribution, and it is a widely applicable technical means to enhance image contrast. Contrast enhancement technology is achieved by broadening the gray level range of the image, which can effectively improve the texture resolution of the image. The elimination methods of image noise mainly rely on filtering techniques. Widely adopted filtering techniques include mean filtering, Gaussian filtering and median filtering, etc. The edge enhancement algorithm mainly improves the clarity of the image by strengthening the processing of the edge information of the image, and enhances the resolution and clarity level of the image.

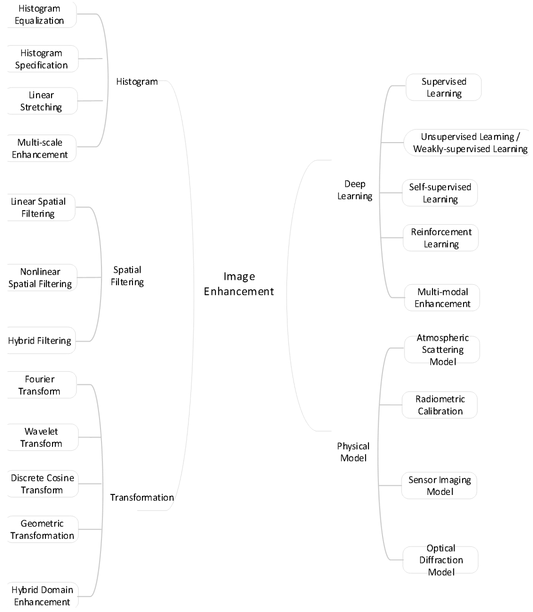

Figure 1. Classification of Image Enhancement Algorithms

3. Main Problems in the Implementation of Infrared Image Enhancement Technology in Embedded Hardware

Infrared image processing systems require extremely high real-time performance to meet the resolution and recognition requirements under different influence scenarios, and the realization of high real-time performance mainly relies on a large amount of computing resources. However, embedded devices usually have lower processor performance, less storage capacity and weaker computing power, which can lead to slow image processing speed and even the inability to process images in real time. These are a pair of prominent contradictions. How to achieve efficient image processing under limited resources is a key issue that must be considered when designing infrared image enhancement algorithms. Next, we will take a histogram equalization algorithm that integrates the detail layer as an example to conduct problem analysis and optimization description.

An image enhancement algorithm integrating the extraction of the detail layer is as follows:

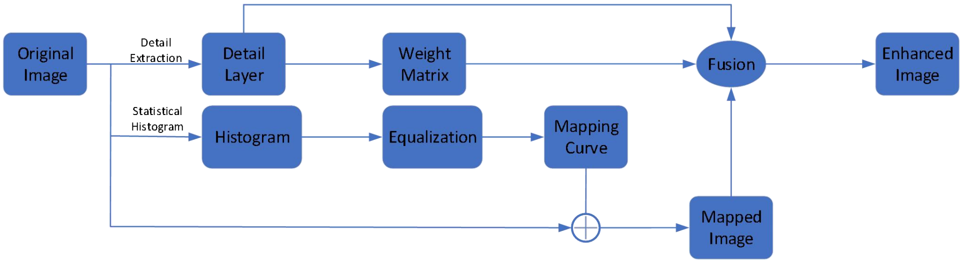

Figure 2. Shows the Process of a Histogram Equalization Enhancement Algorithm based on Weight Fusion

This algorithm controls the proportion of detailed features participating in the fusion through regional weight tables. It can not only highlight the high-frequency information, but also retain the low-frequency background information to the greatest extent. However, during the implementation process, a large number of shaping and floating-point operations need to be performed.

Suppose the original image is an infrared image with a pixel size of 2560*2048, a frame rate of 50Hz, and a quantization bit width of 16-bit. To meet the accuracy requirements, it is assumed that the bit width of the detail layer is 16 bits and that of the weight table is 16 bits. The main resource requirements are as follows:

★Data bandwidth = Pixel size * (original data bit width + detail layer bit width + weight table bit width) * frame rate = 1.3GB/s

★Storage space = Pixel size * (original data bit width + detail layer bit width + weight table bit width) *2= 60MB

★The computing resource requirement = approximately 2000M multiplication and addition operations. Under the RISC instruction set, it is equivalent to 1333MIPS.

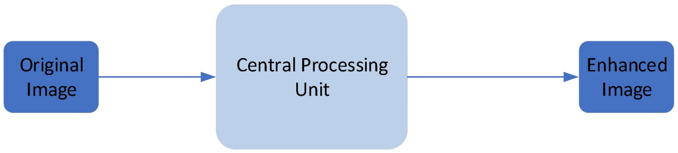

Figure 3. A common Hardware Processing Architecture

This processing architecture has its unique advantages when dealing with the image processing of small array detectors, featuring low power consumption, simple architecture and concise development process. However, the current mainstream infrared detector area array scale has reached the millions level, and some even reach the tens of millions level. Facing the explosive growth of pixel scale, the processing capacity of this architecture has been far from meeting the requirements.

Take the Cortex-M7 architecture as an example. It is a high-performance processor architecture and is suitable for application scenarios that require high computing power and real-time response. With its built-in FPU, DSP instruction set and rich peripheral interfaces, it can meet the diverse demands in fields such as industrial automation, audio processing and automotive electronics. The main resources of the hardware architecture built with the Cortex-M7 processor as the computing core are as follows:

★Maximum main frequency: 300Mhz, equivalent to 200MIPS

★Data bandwidth: 4000MB/s

★Maximum addressing space: 2GB

In reality, there are multiple devices competing for resources, and the resource utilization efficiency cannot reach 100%. The requirements of the algorithm and the resources provided by the platform are shown in the following table:

Table 1. Project Resource Requirements and Provision Status Table

Project	requirements	provision
Computing resources	1333MIPS	200MIPS
Bandwidth resources	1300MB/s	2000MB/s
Storage space	60MB	2GB

Through the comparison in the table, a conclusion can be drawn: The resources of a single Cortex-M7 architecture processor are completely unable to meet the requirements of algorithm implementation. Mainly, the computing resources and bandwidth resources limit the implementation of the algorithm.

4. Optimization Strategies of Infrared Image Enhancement Technology in Embedded Hardware

To solve this problem, the algorithm implementation architecture can be improved to reduce the computational redundancy. Meanwhile, the approach of parallel computing and multi-threading technology can be adopted to enhance the efficiency of image processing.

4.1 Achieve Strategy Optimization

The improvement of the algorithm implementation strategy is the core way to break the constraint of computing resources. The common methods are as follows:

①Downsampling technique. The number of pixels is reduced by merging adjacent pixels, thereby effectively reducing the computational load. The image with a 2560x2048 array size is merged pairwise to obtain a 1280x1024 image, which can reduce the computational load by 75%

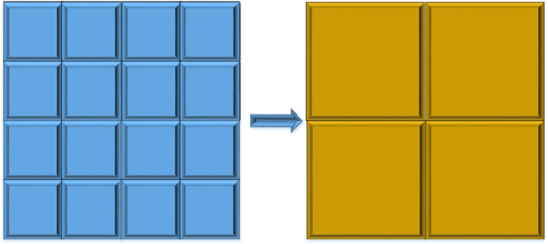

Figure 4. Schematic Diagram of Downsampling Technique

②Bit-width compression technology. Under the condition that the loss of accuracy is acceptable, compressing the data that is insensitive to the bit width can reduce the total amount of data processing. For instance, the detail layer is mostly high-frequency information, and the low bit width can meet the precision requirements of most scenarios. When extracting the detail layer, mapping the 16-bit image to 8-bit can reduce the computing requirements by approximately 40%.

③Partition processing technology. The method of merging later can achieve pipeline processing of images and effectively utilize idle computing resources.

④Adopting the stepwise accumulation computing strategy, that is, only implementing enhancement processing on specific areas in the image and reducing the unnecessary overall computational load, can effectively reduce the time resources required for processing.

⑤Adopt the image enhancement method based on deep learning. Deep learning algorithms are effectively trained through massive datasets to automatically extract and analyze features from image content, significantly enhancing the clarity and visual effect of the images. Although deep learning algorithms generally have a large demand for computing resources, lightweight neural network architectures (such as MobileNet, SqueezeNet, etc.) combine model compression techniques (such as pruning, knowledge distillation) and quantization methods, which can reduce the model size without significantly reducing performance. This enables it to better adapt to embedded devices with limited storage space, providing an efficient and flexible solution for the field of image processing.

4.2 Hardware Acceleration Strategy

Hardware acceleration is an important means to improve the processing capacity of embedded systems, mainly including two solutions: parallel computing and multi-threading technology. The implementation of parallel computing mainly relies on graphics processing units (Gpus), Field Programmable Gate Arrays (FPgas), or Application-specific Integrated circuits (ASics). The key to multithreading technology is the multi-core processor architecture.

①Parallel computing

GPU (Graphics Processing Unit) has outstanding parallel processing and computing performance and the ability to process massive image data in parallel. It is highly suitable for application requirements in the field of image processing. For example, the typical power consumption of NVIDIA's Jetson series products is only about 5 watts. This device integrates a high-performance GPU core, capable of handling thousands of threads simultaneously. It also provides rich apis and libraries, simplifying the development process of parallel computing tasks. Developers can utilize CUDA to write highly optimized parallel algorithms, which can significantly improve the efficiency of image processing, especially when performing large-scale image data processing tasks. Significantly reduce the processing time required.

FPGA is a kind of hardware device with programming capabilities, and this technology has also been widely applied and promoted in the field of image processing. In contrast to Gpus, FPgas demonstrate outstanding flexibility and customization advantages when performing specific tasks. For several specific infrared image processing technologies, the hardware acceleration technology based on FPGA can significantly improve the computing efficiency. Field Programmable Gate Array (FPGA) has demonstrated significant advantages in real-time image processing and low-latency applications due to its outstanding parallel computing performance and wideband characteristics.

Beyond the Graphics Processing Unit (GPU) and Field Programmable Gate Array (FPGA), in the field of embedded systems, Application-specific integrated circuit (ASIC) technology is also widely used in the implementation of hardware acceleration. Application-specific integrated circuits (ASics) have been specially optimized for specific image processing tasks, featuring outstanding processing performance and extremely low energy consumption. Although the research and development and production costs of Application-specific integrated circuits (ASics) are relatively high, they have demonstrated significant superiority in continuous operation and high-frequency computing tasks.

Table 2. Comparison of Several Parallel Computing Hardware

	Advantage	Disadvantage	Applicable scenarios
GPU	- Strong parallel computing capabilities - Mature application ecosystem	- Low flexibility - High power consumption - High latency	Deep learning, graphics rendering
FPGA	- High flexibility - High energy efficiency - Low latency	- Limited resources - High cost per chip	Scenarios with a strong demand for low power consumption and high flexibility, such as embedded devices, chip prototype development and

			verification
ASIC	- Optimal energy efficiency - High reliability - Low cost	- Extremely low flexibility - Long development cycle and high risk	Scenarios that require the best performance and energy efficiency for ultra-large-scale deployment, such as data centers, consumer electronics, and artificial intelligence model inference

②Multithreading technology

Multi-core processors are mainly responsible for performing tasks that require dynamic adjustment or high flexibility. By integrating multiple processing units, they can efficiently share the load of complex multi-tasks. Its main advantage lies in supporting the simultaneous execution of multiple threads or subtasks, thereby improving the overall computing efficiency and throughput. In addition, multi-core architectures are typically equipped with caches and optimized memory access mechanisms, further reducing data transmission latency. In the field of hardware acceleration, multi-core processors can flexibly allocate resources and fully leverage the potential of each core, making them suitable for scenarios that require high-performance computing.

4.3 System Integration and Collaborative Optimization

System integration and collaborative optimization are another core strategy for improving the overall performance of embedded image processing systems. Relying solely on implementing strategy optimization or hardware acceleration often fails to bring out the maximum capabilities of both software and hardware. The overall collaborative optimization of the system is the key to achieving efficient image enhancement.

During the implementation stage of system integration, the improved algorithm is deeply integrated with hardware acceleration technology to construct an efficient and comprehensive system. The characteristics of the hardware platform can be considered, the appropriate algorithm model can be selected, and it can be deeply optimized. For example, algorithms that perform inter-frame filtering based on the time-domain characteristics of images often have very high requirements for the data access bandwidth. When designing the algorithm architecture, attempts are made to adopt methods such as pixel merging and distributed storage to reduce the difficulty of algorithm implementation; During the hardware design process, the focus should be on optimizing the data transmission bandwidth, such as optimizing the bandwidth allocation mechanism, selecting high-speed memory, and deeply adapted memory controllers to improve the working efficiency of the memory. This mode reflects the characteristics of collaborative operation between software and hardware, and can significantly improve the processing efficiency of the system and the quality of image output.

5. Optimization Plan and Simulation Results

5.1 Optimization Plan

Combined with the actual scene requirements, volume constraints, power consumption constraints and cost constraints, it is proposed to adopt a hardware architecture of FPGA+CPU for the algorithm implementation. FPGA allows users to customize hardware logic and is capable of achieving highly parallel task processing. ARM processors excel in serial tasks and complex algorithms. By combining with FPgas, complex computing tasks can be allocated to appropriate parts to achieve efficient parallel processing.

After analyzing the implementation process of the algorithm, the algorithm optimization strategies of downsampling, bit width compression and partition processing are adopted to improve the feasibility. The total number of pixels in the original image is reduced to one quarter of the original by merging two adjacent pixels. Reduce the computational load of the detail layer by mapping 16-bit details to 8-bit. Parallel operations are achieved through block processing. The optimized cross-linking relationship diagram of software and hardware is shown as follows:

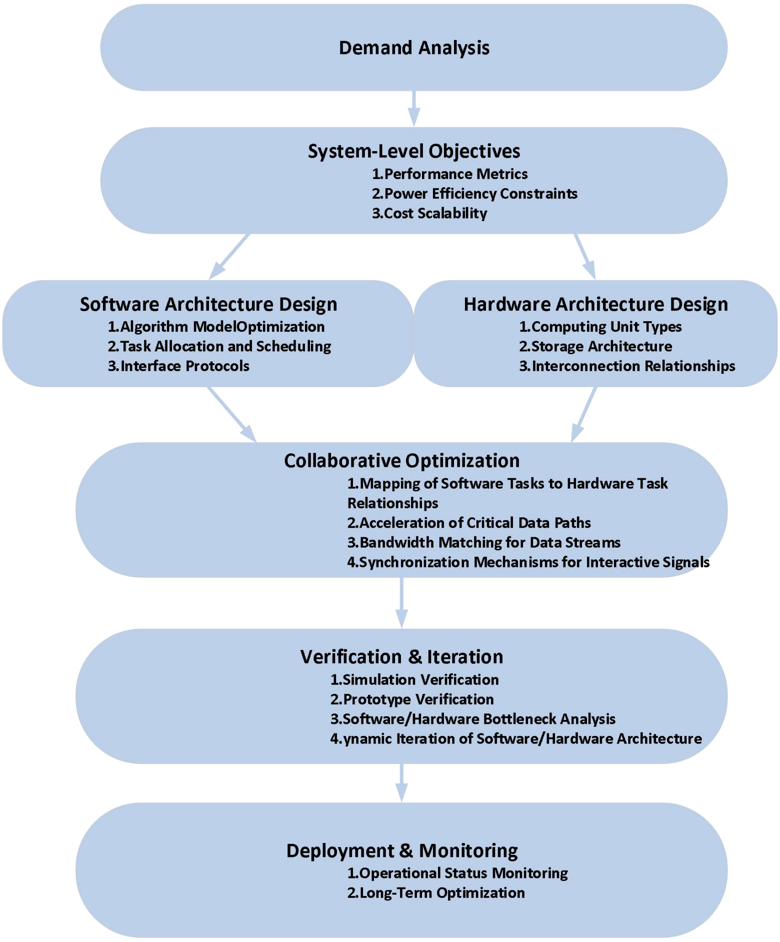
Demand Analysis
System-Level Objectives
1.Performance Metrics
2.Power Efficiency Constraints
3.Cost Scalability
Software Architecture Design
1.Algorithm ModelOptimization
2.Task Allocation and Scheduling
3.Interface Protocols
Hardware Architecture Design
1.Computing Unit Types
2.Storage Architecture
3.Interconnection Relationships
Collaborative Optimization
1.Mapping of Software Tasks to Hardware Task Relationships
2.Acceleration of Critical Data Paths
3.Bandwidth Matching for Data Streams
4.Synchronization Mechanisms for Interactive Signals
Verification & Iteration
1.Simulation Verification
2.Prototype Verification
3.Software/Hardware Bottleneck Analysis
4.ynamic Iteration of Software/Hardware Architecture
Deployment & Monitoring
1.Operational Status Monitoring
2.Long-Term Optimization

Figure 5. Shows the Collaborative Optimization Process of Software and Hardware

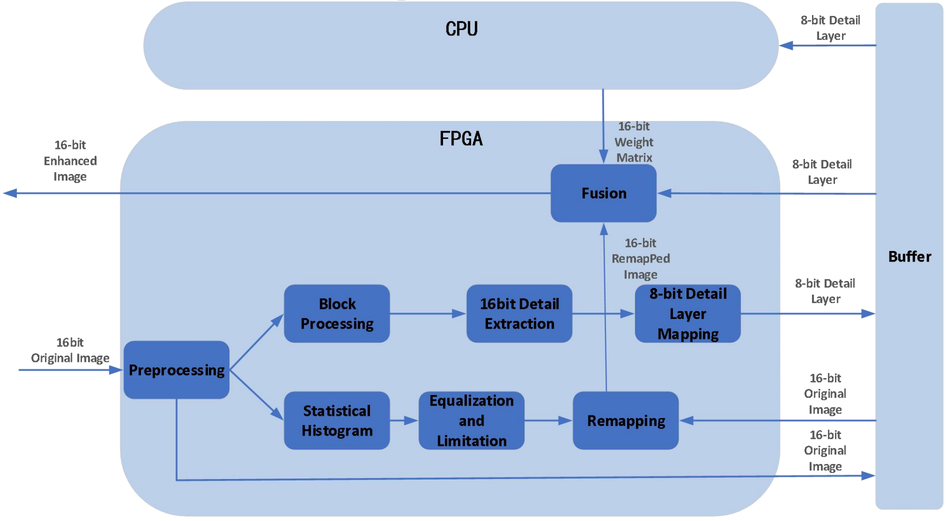

Figure 6. Optimized Architecture

Table 3. Optimized Resource Requirements Table

Project	Before optimization	After optimization
Computing resources	1333MIPS	65.5MIPS
Bandwidth resources	1300MB/s	437.5MB/s
Storage space	60MB	8.75MB

5.2 Effect Analysis

In this paper, two methods, namely Structural Similarity Index, SSIM, Peak Signal-to-Noise Ratio and PSNR, are used to evaluate the differences in image quality before and after optimization.

The structural similarity index well reflects the perception of image quality by the human visual system. It comprehensively considers luminance, contrast and structural information. The calculation formula is as follows:

$$\mathrm{SSIM(x,y)} = \frac{(2\mu_x + C_1)(2\sigma_{xy} + C_2)}{({\mu_x}^2 + {\mu_y}^2 + C_1)({\sigma_x}^2 + {\sigma_y}^2 + C_2)}$$

μ_x、 μ_y： The mean values of images x and y

σ_x、 σ_y： The standard deviations of images x and y

σ_{xy} ： The covariance of images x and y

C_1 、 C_2 ： Constant, used for harmonic calculation

The SSIM calculation result takes values within the range of [-1,1]. The closer it is to 1, the more similar the image is.

PSNR is a widely used indicator for measuring image quality. It assesses the superiority or inferiority of the reconstruction algorithm by comparing the differences between the original image and the reconstructed image. The calculation formula is as follows:

$$\mathrm{PSNR} = 10\mathrm{Log}(\frac{MAX_I^2}{MSE})$$

MAX_I ： The maximum pixel value in the image. The 16-bit image is 65535.

MSE： The mean square deviation between the original image and the reconstructed image.

More than 50 groups of images from different scenes were simulated and analyzed. The average value of the simulation results in various scenes was taken as shown in the following figure:

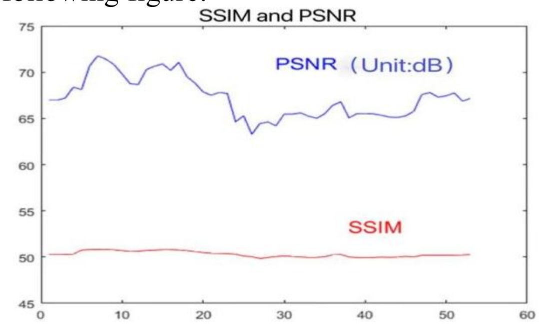

Figure 7. Simulation Results of the Two Evaluation Systems

In order to make an intuitive comparison and observation between SSIM and PSNR, the SSIM in the figure has been enlarged, with an magnification of 500. It can be seen through the SSIM index that the average similarity of the images before and after optimization is about 0.99, indicating that the error noise introduced by the optimization will not have a significant impact on the image quality. It can be seen that the PSNR indicators are all above 63dB, indicating that the image quality before and after optimization is very close. It can be concluded that the optimization strategy of the algorithm is effective, ensuring the image quality while reducing the reliance on hardware resources.

6. Conclusions

The application of infrared image enhancement in embedded systems still faces many challenges, especially the balance issue between processing speed and system resource limitations. Efficient image enhancement can be achieved in an embedded environment through algorithm optimization, hardware acceleration and overall system optimization. In the future, with the continuous advancement of technology, the embedded image processing capabilities will be further enhanced, and the application scope of infrared images will also continue to expand.

References

[1]Soundrapandiyan ,Rajkumar,Satapathy , et al.A comprehensive survey on image enhancement techniques with special emphasis on infrared images[J].Multimedia Tools and Applications,2021,81(7):1-33.

[2]El-Samie A E F ,Ashiba I H ,Shendy H , et al.Enhancement of Infrared Images Using Super Resolution Techniques Based on Big Data Processing[J].Multimedia Tools and Applications: An International Journal,2020,79(9-10):5671-5692.