



Performance Comparison Study of Edge Detection Algorithms in Video Communication

Dinghao Zhang

School of Telecommunications Engineering, Xidian University, Xi'an, 710126, China
22009101693@stu.xidian.edu.cn

Abstract. With the rapid advancement of video communication technologies and the growing demand for real-time processing in applications such as autonomous driving and industrial automation, efficient edge detection has become increasingly crucial for optimizing video transmission and analysis. Traditional edge detection methods face significant challenges in balancing accuracy, speed, and power consumption, especially when processing high-resolution video streams in resource-constrained environments. This study evaluates edge detection algorithms for video communication optimization, comparing traditional methods, FPGA implementations, and deep learning approaches. Traditional Canny algorithms with FPGA acceleration achieve remarkable 4.628ns latency and 0.257W power consumption, outperforming CPUs by 450,000× in speed. Deep learning-based HED demonstrates superior accuracy (F-score: 0.78) but faces real-time processing challenges. Hybrid methods combining traditional and deep learning techniques show promising balance between accuracy and efficiency. In HEVC encoding applications, edge-guided CTU partitioning reduces encoding time by 41-58% while maintaining quality. The findings provide practical guidance for algorithm selection in autonomous systems and industrial applications, highlighting the need for adaptive solutions in next-generation video processing.

Keywords: Edge Detection Algorithms, Video Communication, HEVC Encoding, Deep Learning.

1 Introduction

In recent years, the importance of video communication in intelligent scenarios has become increasingly prominent. However, although HEVC encoding can save 50% of the bit rate compared with H.264, its encoding time increases by 9% to 502%, which is difficult to meet the real-time requirements. Edge detection technology can effectively reduce the computational burden of encoding by extracting the key edge information in the video. At present, edge detection algorithms are mainly divided into two categories: traditional algorithms and their improvements, and deep learning algorithms.

Traditional algorithms such as Sobel and Canny have high computational efficiency and are suitable for real-time processing, but their accuracy is insufficient in complex scenarios. FPGA acceleration solutions achieve high throughput through hardware parallel computing, but the development cost is relatively high. Deep learning algorithms

like HED significantly outperform traditional methods in edge detection accuracy, but their computational complexity is high, making them difficult to directly apply to real-time video communication. This study aims to provide the basis for the optimal algorithm selection for scenarios such as Industry 4.0 and autonomous driving through a comparative analysis of the performance of different algorithms, and to explore the collaborative optimization of edge detection and HEVC encoding.

2 Technical comparison and analysis

2.1 Traditional edge detection algorithms

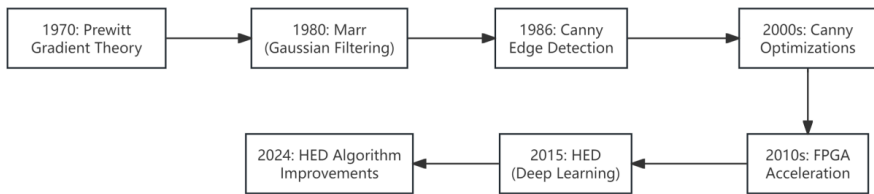


Fig. 1. Development process of edge detection algorithms.

As shown in Figure 1, the development of edge detection algorithms has undergone an evolution from traditional methods to deep learning techniques. Early research began in the 1970s, with Prewitt's gradient theory laying the foundation for edge detection. In the 1980s, Marr introduced Gaussian filtering into the field of edge detection, significantly enhancing the algorithm's noise resistance. The 1986 introduction of the Canny edge detection algorithm marked the maturity of traditional methods, achieving high-precision detection through steps such as Gaussian smoothing, gradient calculation, and non-maximum suppression. In the 21st century, researchers optimized the Canny algorithm in various ways to adapt it to more complex scenarios. In the 2010s, the application of FPGA hardware acceleration technology greatly improved the real-time performance of edge detection. In 2015, the advent of the deep learning algorithm HED initiated a new paradigm of end-to-end edge detection, and further improvements in 2024 integrated the advantages of traditional operators, achieving new breakthroughs in both accuracy and efficiency. Even today, the improvements of traditional edge detection and deep learning-based algorithms still have significant applications in video communication.

Canny algorithm. The Canny edge detection algorithm is a classic method for edge detection [1], whose core idea is to precisely extract the edge information in the image through multiple processing steps. The algorithm first applies Gaussian smoothing filtering to the image to effectively suppress noise interference. Then, it calculates the gradient magnitude and direction of the image, which is usually achieved using the Sobel operator [1]. To obtain more refined edges, the algorithm then performs non-maximum suppression processing, that is, only retaining the local maximum points in

the gradient direction. Finally, through the double-threshold detection and edge connection steps, the extracted edges are ensured to be both complete and accurate [1].

The theoretical basis of this algorithm is based on three important criteria. The first criterion is that the algorithm should have a high signal-to-noise ratio, meaning that it needs to minimize the false detection caused by noise while ensuring that the real edges can be accurately detected. The second criterion emphasizes the precise positioning of edge points, requiring that the detected edge points must be as close as possible to the real edge positions. The third criterion stipulates that the algorithm should only generate one response for a single edge, so as to avoid repeated detection of the same edge and ensure the continuity of the edge. Through strict mathematical derivation, Canny proved that the optimal edge detection operator can be approximated as the first derivative of the Gaussian function, and innovatively introduced non-maximum suppression and double-threshold strategies, thereby achieving a good balance between detection accuracy and anti-noise ability.

However, this algorithm also has some inherent limitations. One of the main problems is that the double-threshold parameters need to be manually set, which makes the algorithm difficult to automatically adapt to different lighting conditions or noise levels. Although Gaussian smoothing can effectively suppress noise, it may also cause the loss of some weak edge information, especially being sensitive to salt-and-pepper noise. In terms of computational efficiency, due to the involvement of multiple processing steps, the algorithm's computational complexity is high, which to some extent affects its performance in real-time applications. These limitations have driven the further development of subsequent research.

The Direction of Canny's Improvements. Li Xuan et al. addressed the issues of fixed threshold and noise sensitivity in the traditional Canny algorithm and proposed an improved method. This method replaces Gaussian filtering with median filtering to enhance the anti-noise ability, and calculates the dual thresholds (high threshold ThH and low threshold $ThL = 0.4ThH$) adaptively through the difference operation of the gradient amplitude histogram. The improved algorithm also introduces the concept of generalized chains, filtering out false edges by calculating the average gradient value of the chain, and finally optimizing the edge connection through linear fitting [2]. Experiments show that this method can more clearly distinguish the target from the background in low-contrast images, and has stronger robustness against salt-and-pepper noise [2].

Further advancing real-time processing capabilities, Yang Liu et al. developed an adaptive edge detector leveraging 2D entropy-based thresholding. By statistically analyzing edge proportion distributions in BSDS500 dataset, their method automatically classifies images into three entropy groups (low/moderate/high) with corresponding edge percentage references (4.3%/6.5%/8.7%). The detector employs curvature-predictive edge linking that considers segment direction continuity, generating clean 1-pixel wide chains while avoiding staircase artifacts common in traditional operators [3]. This approach demonstrates superior accuracy (0.614 F-measure) at 8.17ms processing speed, particularly effective for post-processing tasks like line and ellipse detection [3].

2.2 Canny edge detection accelerated by FPGA

To meet the real-time requirements of video communication, Li Quan Tan et al. [4] utilized the parallel computing capability of FPGA to achieve hardware acceleration of the Canny algorithm. This design optimized the median filtering (by replacing Gaussian filtering with parallel sorting), Sobel gradient calculation (by calling IP cores and reducing memory access through row buffering), and non-maximum suppression (by comparing 8-direction interpolation) through pipeline technology [4]. The double-threshold connection adopted a neighborhood scanning strategy to avoid edge breakage [4]. The test results showed that the processing delay of the FPGA solution in a $640 \times 480 @ 60\text{fps}$ video stream was only 4.628 nanoseconds, with a power consumption as low as 0.257W. Compared to general-purpose processors (GPP), the speed was increased by approximately 450,000 times, and the energy efficiency ratio was improved by 665 times, significantly enhancing the real-time performance [4].

2.3 Edge detection based on deep learning (HED)

Holistically-Nested Edge Detection (HED) is a deep learning-based edge detection algorithm [4], proposed by Saining Xie and Zhuowen Tu. Its core idea is to achieve end-to-end image-to-edge map prediction through a fully convolutional neural network and a deep supervision mechanism [5]. This algorithm introduces side outputs at different network levels to automatically learn multi-scale features from local details to global semantics, thereby effectively improving the accuracy and robustness of edge detection [5]. The optimal dataset scale F-score (ODS F-score) of HED on the BSD500 dataset reached 0.782, significantly outperforming traditional methods such as the Canny detector (0.6) and SE (0.746) [5]. Moreover, HED has high computational efficiency, requiring only 0.4 seconds to process an image on a GPU, making it potentially advantageous in real-time video communication [5].

In terms of performance evaluation, HED optimizes the integration of multi-scale features through deep supervision and weighted fusion layers, achieving a good balance between precision and recall in edge detection results. Compared with traditional methods that rely solely on manually designed features, HED can automatically learn more discriminative features, reducing the need for manual intervention. However, its adaptability in handling dynamic video sequences still requires further research, such as how to deal with complex scenarios like illumination changes and motion blur.

2.4 Fusion of traditional operators and deep learning

Yeborui Yuan et al. [6] addressed the performance limitations of traditional Sobel operators in edge detection by proposing an ISAHED algorithm (Improved Sobel-based HED) that utilizes an improved Sobel operator as an auxiliary component of the HED network [6]. This algorithm first improved the traditional Sobel operator by adding 45-degree and 135-degree gradient calculations, expanding the convolution kernel template to capture multi-directional features of image edges more comprehensively [6]. On this basis, the researchers designed a simplified HED-like model based on the VGG-

16 network structure, which consists of five convolutional blocks and can extract multi-scale edge features and output five different-scale side outputs [6]. These side outputs are generated through bilinear interpolation and weighted fusion to produce the final edge map, achieving end-to-end training from the input image to the edge map [6].

Experimental results show that the ISAHED algorithm performs well in complex scenarios, being able to detect more complete edge contours and richer details [5]. Through quantitative analysis using the FOM (Figure of Merit) metric, the performance of this algorithm is 9 times that of the improved Sobel operator, significantly outperforming traditional Sobel, Prewitt operators, and their improved versions [6]. The innovation of this study lies in combining the interpretability of traditional operators with the feature learning ability of deep learning, not only improving the accuracy and robustness of edge detection but also providing an effective solution for practical applications [6].

Table 1. Comprehensive Performance Comparison of Edge Detection Algorithms.

Metric	Classic Canny	Adaptive Canny	FPGA-accelerated Canny	Deep Learning (HED)
F-score (BSD500)	0.60	0.72 (estimated)	Not mentioned	0.78
Real-time Performance	Medium (CPU 15fps)	Medium (CPU 15fps)	60fps	Low (2.5fps)
Adaptability	Low (fixed thresholds)	High (adaptive thresholds)	Low (fixed thresholds)	High (semantic understanding)

As shown in Table 1, by comparing and analyzing the performance of four edge detection algorithms, it can be found that different algorithms exhibit significant differences in terms of detection accuracy, real-time performance, and adaptability [1-5]. The classic Canny algorithm has an F-score of 0.60 on the BSD500 dataset and has moderate real-time performance (15 fps on CPU), but due to the use of a fixed threshold, its adaptability is relatively low [1]. The adaptive Canny algorithm improves the F-score to 0.72 (estimated value) by introducing an adaptive threshold mechanism, while maintaining the same real-time performance as the classic Canny algorithm (15 fps on CPU), significantly enhancing the adaptability of the algorithm [2]. The FPGA-accelerated Canny algorithm excels in real-time performance, achieving a detection speed of 60 fps, but still uses a fixed threshold, resulting in limited adaptability [4]. The deep learning-based HED algorithm performs the best in terms of detection accuracy, with an F-

score of 0.78 and the strongest semantic understanding ability, but has poor real-time performance (2.5 fps) and a significant disadvantage in terms of computational resource consumption [5].

Overall, the traditional Canny algorithm and its improved versions have advantages in real-time performance, while deep learning algorithms perform better in terms of detection accuracy and adaptability

3 Application research

3.1 Application of edge detection in HEVC encoding optimization

Researchers such as Md. Zahirul Islam proposed a fast CTU (Coding Tree Unit) partitioning method based on edge detection to address the high complexity issue in the CTU division process of HEVC/H.265 encoding [7]. This method identifies the key edge regions in the video frame using Canny or Sobel edge detection algorithms and only performs the maximum depth CTU partitioning on these regions, thereby skipping the traditional brute-force search rate-distortion optimization (RDO) process. Experimental results show that this method significantly reduces the encoding time while maintaining video quality: when using Canny edge detection, the average encoding time is reduced by 41.83% (up to 48.04%), and the BD-BR (Bit Rate Increment) is increased by an average of 5.24%; when using Sobel edge detection, the average encoding time is reduced by 58.41% (up to 72.98%), and the BD-BR is increased by an average of 3.93% [6]. This method provides an efficient solution for real-time video encoding, especially suitable for scenarios with limited computing resources.

3.2 FPGA accelerated real-time edge detection architecture

Researchers such as Venkata Ganeswara Rao Maddipati designed a real-time video edge detection system based on VLSI architecture, using the Canny algorithm and implementing it on Artix 7 FPGA [8]. This architecture significantly improves the edge detection speed through parallel processing, optimized data flow, and efficient memory management. Experimental results show that the detection delay is only 4.628 nanoseconds, which is approximately 450,000 times faster than the 2.1 milliseconds achieved by general-purpose processors (GPP). Additionally, this design has a relatively low resource utilization on the FPGA (LUT occupancy rate of 5.17%, BRAM occupancy rate of 5%), with excellent power and area efficiency. The system captures video through a VGA camera and outputs edge detection results, verifying its practicality in real-time video processing and providing hardware support for applications requiring low-latency edge detection, such as autonomous driving and surveillance.

Table 2. Performance Comparison Between Proposed VLSI Implementation and General-Purpose Processor (GPP) Implementation.

Platform	Detection time (ns)	Estimated Power (W)		
		Static	Dy-namic	To-tal
Artix 7 FPGA	4.628	0.072	0.185	0.257
GPP (Intel Core i7)	2.1×10^6	-	171 (avg)	-

Table 2 compares the performance of two hardware platforms when performing edge detection tasks. On the Artix 7 FPGA platform, the detection time is only 4.628 nano-seconds, with its power consumption consisting of static power of 0.072W and dynamic power of 0.185W, totaling 0.257W. In contrast, the detection time of the general-purpose processor (Intel Core i7) reaches 2.1 microseconds, with a power consumption as high as 171W. The data indicates that the FPGA solution has significant advantages in real-time performance and energy efficiency. Its processing speed is approximately 450,000 times faster than that of the general-purpose processor, and the energy efficiency is improved by 665 times [8]. This is mainly attributed to the parallel computing architecture and hardware optimization design of FPGA.

4 Challenges and prospects

Current research in the field of edge detection and video coding integration faces multiple challenges. Traditional algorithms such as Canny and Sobel, although having high computational efficiency, significantly reduce detection accuracy in complex scenarios like dynamic lighting or motion blur, leading to misjudgments in the video coding process. While deep learning-based detection methods can improve accuracy, their high computational complexity makes them difficult to meet real-time requirements. In terms of hardware implementation, the FPGA acceleration scheme significantly reduces latency through parallel computing, but its high development cost and fixed hardware architecture limit the flexibility of algorithm iteration. Moreover, general-purpose processors (GPP) have an inferior energy efficiency ratio, making it difficult to support long-term operation in low-power scenarios. Additionally, existing technologies have insufficient adaptability to high dynamic scenes, especially in strong motion scenarios such as autonomous driving or industrial robots, where the stability of edge detection and the robustness of coding still need to be improved.

Future research should focus on breaking through in the directions of intelligence and systematization. By integrating the feature extraction capability of deep learning with the lightweight advantages of traditional algorithms, an adaptive model that can

dynamically adjust the detection strategy can be constructed, which is expected to achieve a better balance between accuracy and efficiency. The collaborative design of heterogeneous computing architectures will be crucial, such as combining the real-time processing capability of FPGA with the deep learning acceleration feature of GPU to build a reconfigurable hardware platform to support rapid algorithm iteration. At the same time, it is necessary to deepen the end-to-end collaboration of edge detection and video encoding, using edge information to optimize bitrate allocation and coding unit division, and reducing redundant computations. With the development of 6G networks and edge computing technologies, the detection-coding integrated architecture of distributed edge nodes and cloud collaboration will promote ultra-low latency video communication in scenarios such as Industry 4.0 and autonomous driving, providing more efficient and reliable technical support for real-time video processing systems.

To address these challenges, recent research breakthroughs demonstrate two complementary approaches that could inform future system designs. First, Zhao et al.'s Ubiquitous Target Awareness (UTA) network introduces a depth-aware architecture that dynamically corrects edge detection errors in challenging conditions through adaptive error-weighted learning [9]. This approach proves particularly effective in preserving boundary accuracy during motion blur or lighting variations, precisely the scenarios where traditional edge detectors typically fail. Second, Yang et al.'s task-switchable preprocessing framework achieves remarkable 30-74% bitrate reductions by selectively preserving semantic features critical for machine vision tasks [10], while maintaining compatibility with existing codecs. Together, these studies suggest three key implementation pathways:

Hybrid feature extraction can be achieved by combining the depth-aware edge enhancement capabilities of the UTA network with the computational efficiency of traditional algorithms, which may produce adaptive detectors capable of maintaining high accuracy under dynamic environmental conditions while avoiding prohibitive computational costs [9].

The feature modulation technique developed by Yang et al. provides compelling evidence that edge information can effectively guide bit allocation decisions in video coding systems, offering significant potential for reducing redundant computations throughout the encoding pipeline [10].

The demonstrated real-time performance of these architectures, with the UTA network achieving 43 FPS and the pre-processor reaching 111 FPS, strongly suggests their suitability for implementation on FPGA-GPU heterogeneous platforms, potentially overcoming existing limitations in both energy efficiency and system flexibility.

These advances lay the groundwork for developing end-to-end systems where robust edge detection directly informs adaptive coding strategies - a crucial capability for emerging 6G-enabled applications requiring ultra-low latency processing.

5 Conclusion

This study systematically compared the performance of traditional edge detection algorithms, FPGA acceleration schemes, and deep learning techniques in video communication. The experiments showed that the traditional Canny algorithm and its improved versions have significant advantages in terms of real-time performance (the FPGA acceleration scheme can reach 60 fps), while the deep learning algorithm HED achieved a breakthrough in detection accuracy (with a F-score of 0.78), but due to its high computational complexity, it is difficult to meet real-time requirements. Through the collaborative optimization of edge detection and HEVC encoding, the encoding time can be reduced by 41% - 58%, verifying the key value of this technology in real-time video communication scenarios such as Industry 4.0 and autonomous driving.

The research results provide a theoretical basis for algorithm selection in intelligent scenarios: In scenarios with complex dynamic environments and limited computing power, the adaptive Canny algorithm based on FPGA demonstrates the best cost-performance ratio; while in offline analysis scenarios with higher detection accuracy requirements, deep learning algorithms have greater potential. The research reveals the necessity of collaborative innovation between algorithm lightweighting and hardware acceleration, providing a new idea for building an "analysis-coding" integrated system.

Looking to the future, edge detection technology will evolve towards intelligence and heterogeneity. On one hand, lightweight models based on neural network architecture search (NAS) are expected to break through the real-time performance bottleneck; on the other hand, the heterogeneous computing architecture of FPGA + GPU can meet both accuracy and efficiency requirements. With the development of 6G networks and edge computing, distributed video coding technology integrating edge detection will promote ultra-low latency communication in fields such as industrial robots and autonomous driving, and build a high-reliability and high-energy-efficient visual perception infrastructure for intelligent scenarios.

References

1. J. Canny, A computational approach to edge detection. Massachusetts Institute of Technology, Cambridge, MA 02139. (1986)
2. X. Li, H. Zhang, An improved Canny edge detection algorithm. Shenyang Aerospace University, Shenyang, China. (2018)
3. L. Yang et al., An adaptive and robust edge detection method based on edge proportion statistics. *IEEE Trans. Image Process.* **29**, 5206-5215 (2020)
4. Q. L. Tan, The research of implementation method of Canny edge detection of video on FPGA. Beijing University of Chemical Technology, 100029. (2017)
5. S. Xie, Z. Tu, Holistically-nested edge detection. University of California, San Diego. (2015)
6. Y. Yuan et al., A high-precision edge detection algorithm based on improved Sobel operator-assisted HED. Tianjin University of Technology, Tianjin, 300384, China. (2022)
7. M. Z. Islam et al., Fast CTU splitting using edge detection in video coding. *IEEE Trans. Circuits Syst. Video Technol.* **31**, 3681-3695 (2021)

8. V. G. R. Maddipati et al., High performance VLSI architecture for real-time video edge detection. *IEEE Trans. Circuits Syst. I.* **68**, 2986-2999 (2021)
9. Y. Zhao et al., RGB-D salient object detection with ubiquitous target awareness. *IEEE Trans. Image Process.* **30**, 7707-7717 (2021)
10. M. Yang et al., Task-switchable pre-processor for image compression for multiple machine vision tasks. *IEEE Trans. Circuits Syst. Video Technol.* **34**, 1234-1245 (2024)

Open Access This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (<http://creativecommons.org/licenses/by-nc/4.0/>), which permits any noncommercial use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

