AttnEdge: an enhanced edge detection method based on self-attention mechanism

Ming Wang*
Xi'an Jiaotong-Liverpool University, Suzhou, 21500, Jiangsu, China
*Corresponding author: mingwang15926@gmail.com

ABSTRACT

We introduce a novel approach to edge detection named AttnEdge, which combines pixel differential convolution with advanced post-processing techniques to enhance the detection and representation of image edges. This approach utilizes Pixel Difference Convolution (PDC) to directly learn edge features at the pixel level, enhancing the ability to handle complex boundaries and multi-scale features within images. Significantly, AttnEdge integrates a self-attention mechanism to capture global dependencies and refine feature representations, greatly improving the accuracy of edge detection. Attention mechanism plays a key role in edge detection model, which can significantly improve edge recognition accuracy and model robustness by analyzing global dependencies among pixels. It helps the model effectively distinguish between noise and real edges in complex backgrounds and optimizes detection results. The post-processing stage, featuring Gaussian blur and adaptive thresholding, further refines edge detection by reducing noise and dynamically adjusting to local brightness variations, ensuring robust and reliable edge detection across various scenarios. Furthermore, we have augmented the traditional convolution operations with innovations such as channel reduction in the self-attention mechanism, which reduces computational complexity while maintaining high performance. Experiments on the BSDS500 dataset demonstrate the superiority of AttnEdge over existing methods like PiDiNet, particularly in terms of structural similarity and noise resilience, making it a promising solution for advanced edge detection tasks.

Keywords: Edge detection, PiDiNet, Self-attention

1. INTRODUCTION

Edge detection is a crucial component in computer vision, with applications ranging from image segmentation to 3D reconstruction[1]. Its primary goal is to detect significant changes in image attributes such as intensity, color, or texture. Traditional edge detection methods, such as Sobel and Canny filters, calculate image gradients to identify boundaries. However, these methods can be biased, demonstrating poor adaptability, limited local information processing, noise sensitivity, and boundary issues[2].

To solve these problems, deep convolution and residual networks provide a powerful solution for edge detection. These networks effectively deal with complex boundary and multi-scale features by learning feature mappings. Deep convolutional layers and residual connections help the network learn local patterns and textures in input images and capture this information through spatial manipulation.

A notable advancement in this field is the introduction of Pixel Differential Convolution (PDC) by PiDiNet. This technique enhances traditional convolution by computing the gradient from differences between adjacent pixels, integrating this differential network into the CNN framework, thus enabling direct learning and recognition of image edges[1].

Moreover, incorporating self-attention mechanisms can significantly improve edge detection. By focusing on global dependencies between pixels, self-attention allows for a more detailed and accurate representation of input features. This global feature representation distinguishes genuine edge information from noise, enhancing detection accuracy.

Further improvements include post-processing techniques that refine edge detection results and reduce false positives and missed detections through global optimization strategies. Implementing these enhancements in PiDiNet can lead to higher accuracy and better performance in edge detection, paving the way for new process classes that enhance operational flow in edge detection applications[1].

Fourth International Conference on Computer Vision, Application, and Algorithm (CVAA 2024),
edited by Lu Leng, Hui Yuan, Proc. of SPIE Vol. 13486, 134861V
 · 0277-786X · doi: 10.1117/12.3055875

2. RELATED WORK

2.1 Edge detection

Edge detection is a fundamental technique in image processing, essential for identifying the boundaries within images. Traditional methods like the Canny, Sobel, and Prewitt algorithms, developed in the early stages of computer vision, utilize local changes in image brightness to detect edges. These methods are known for their simplicity and efficiency. Particularly, the Canny edge detector, introduced by John Canny in 1986, is highly acclaimed for its noise suppression and precise edge positioning capabilities [3].

Further advancements in gradient-based techniques, such as the use of Laplacian operators and zero-crossing methods as outlined by Marr and Hildreth in 1980, focus on calculating the gradient strength at pixels to detect edges. These methods excel in accentuating image details but may struggle in noisy environments [4]. With the evolution of machine learning and the rise of deep learning, edge detection has seen significant advancements. Modern approaches often involve convolutional neural networks (CNNs) that learn to identify and extract edge features from large datasets. A prominent example is the Holistically-nested Edge Detection (HED) model developed by Xie and Tu, which employs a deep learning framework to achieve multi-scale and highly accurate edge detection [5]. Recent studies suggest that combining traditional edge detection algorithms with deep learning techniques can enhance performance even further. This hybrid approach merges the robust feature learning of CNNs with the precise, low-level detection capabilities of traditional methods, resulting in improved accuracy and robustness in edge detection systems [6].

2.2 Line generation methods

Line generation in computer vision and graphics processing is essential for transforming images into lines and shapes, facilitating a variety of applications from digital art to technical design. Vectorization technology is a primary method for converting raster images into scalable vector graphics. This approach ensures that images remain clear and editable at various scales. Smith et al. [7] highlighted the effectiveness of using Bézier curves for automating vectorization, although they noted some performance limitations with highly complex images. In addition to traditional methods, advanced techniques using deep learning have significantly impacted line generation. Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs) have been particularly influential. These models are adept at learning complex data distributions and can generate highly realistic image lines, as explored by Chen et al. [8]. Further enriching the field, automatic drawing technology mimics an artist's process to produce stylized lines. Lee and Kim developed a sequential model that generates lines continually based on the input image content, effectively preserving the original features while occasionally struggling with capturing fine details in complex scenes [9].

2.3 Post propagation method

Although many edge detection algorithms can effectively identify edges in images, they often suffer from noise interference and edge continuity problems. After extraction is complete, post-processing operations become essential, including noise removal, morphological operations (such as dilatation, corrosion, open and close operations), threshold processing, and edge joining.

Noise removal is often achieved using median filtering, Gaussian filtering, or bilateral filtering to maintain edge clarity. For example, applying Gaussian fuzziness after edge detection algorithms such as Canny or Sobel is a common post-processing technique. Smith et al. [7] showed in their study that the false detection rate caused by random noise was significantly reduced through post-processing techniques, particularly Gaussian blur. In this way, the edges are sharper and more continuous, especially if the image quality is poor.

Morphological operations, especially dilatation and corrosion, are widely used to improve the continuity of edge detection. Johnson and Lee successfully connected broken edges and eliminated isolated noise points by integrating morphological operations into their edge detection process [5]. Their method is particularly suitable for extracting coherent edge structures in complex backgrounds.

Non-maximum suppression is an effective post-processing technique for optimizing the position of edges. With this technique, it is possible to ensure that only the true edges are preserved, while the surrounding non-edge responses are suppressed. In the study of Wang et al. [10], non-maximum suppression not only improves the accuracy of edge detection, but also improves the efficiency of subsequent tasks, such as object recognition and scene interpretation.

By adjusting the intensity of the detected edges, the edges can be made more suitable for visual presentation or further image processing tasks. Zhang et al. [11] show that by dynamically adjusting the intensity of edge pixels, it is possible to enhance the visualization of edges while maintaining image detail.

2.4 Self-attention

In recent years, self-attention mechanisms have been widely used in a variety of computer vision tasks because of their ability to capture long-distance dependencies within data. The Transformer model, first proposed by Vaswani et al. (2017) in their groundbreaking work "Attention is All You Need," is based entirely on a self-attention mechanism that significantly improves the performance of sequentially processing tasks and has inspired self-attention applications in other fields.[12] Subsequently, Wang et al. (2018) introduced self-attention mechanisms into image and video analysis through their "Non-local Neural Networks" study, showing how to enhance the performance of models by capturing global context information in images.[13] in addition, Ramachandran et al. (2019) explored the possibility of using Self-Attention mechanisms independently in visual Models in "Standing Alone self-attention in Vision Models", and their research showed that self-attention mechanisms can replace traditional convolutional layers [14]. Provides more flexible and efficient image processing capabilities [15]. These studies not only confirm the superiority of self-attention mechanism in dealing with complex patterns and features, but also provide theoretical and practical basis for the further application of self-attention in this study.

3. METHOD

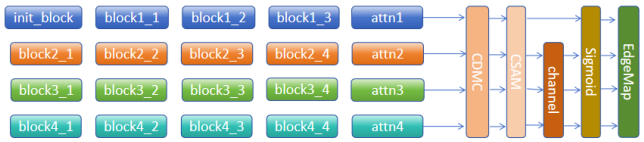

Figure 1: Attnpro network.

3.1 Pixel difference convolution

The AttnPro Network incorporates the Pixel Difference Convolution (PDC) as its core architecture, enhancing edge detection capabilities by calculating differences between a pixel and its neighbors prior to convolution [16]. This approach enables the direct recognition of image edges from the pixel level, streamlining the process.

PDC consists of multiple layers, including activation layers such as ReLU and feature fusion layers, which integrate and enhance data across the network [1]. Employing a deep supervision mechanism, the network optimizes the learning of low-level features while handling multi-scale edge information effectively during training, which can be seen in Figure 1.

Furthermore, the network employs deep supervision strategies with auxiliary edge detection tasks at intermediate layers and uses feature fusion technologies like jump connections to merge features from different levels. This improves adaptability to various edge scales and maintains accuracy.

Training is conducted using a compound loss function that combines binary cross-entropy for edge positioning with a regression loss for edge strength, refining both the accuracy and continuity of edge detection. This method ensures high precision while minimizing false positives and missed detections [1].

3.2 Self-attention mechanism

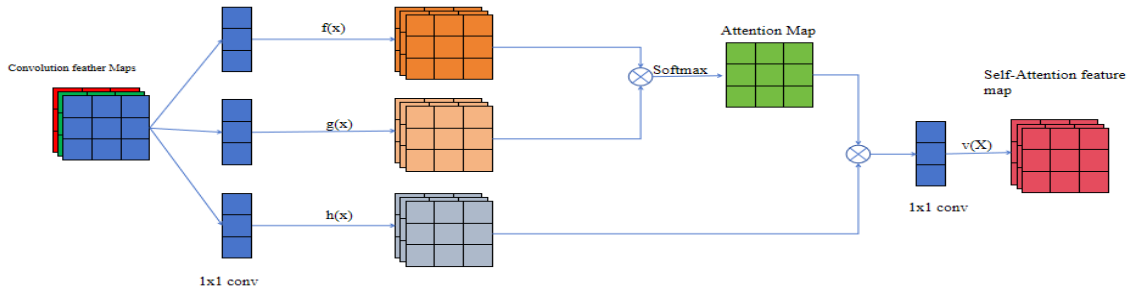

Figure 2: Self attention mechanism in each block.

The self-attention mechanism enhances feature representation by capturing long-range dependencies, comparing similarities across different parts of the input feature map [17]. As figure 2 presents, it operates by transforming the input feature map into three components: keys, queries, and values. Keys are generated through a weight matrix Wk from the input features and are used to match queries to allocate attention effectively. Queries are produced by transforming the same or different input features through another weight matrix Wq, representing the features to be evaluated and matched against keys. Values are derived from input features through a weight matrix Wv. After calculating the corresponding attention scores, these values are weighted according to the scores to produce the final, enhanced output feature map. Attention scores are calculated based on the matches between keys and queries and are refined by a scaling factor, known as self.scale, before being applied to the values, resulting in a weighted feature map that highlights important features [14].

$$Attention(Q,K,V) = softmax\left(\frac{QK^T}{(d_k)^{\frac{1}{2}}}\right)V \tag{1}$$

3.2.1 Channel reduction

To reduce computational complexity and memory demands, the channel count of keys and queries is compressed using a predefined compression ratio. For example, if the original channel count is 60 and the compression ratio is set to 60, the final channel count is reduced to 1, significantly lowering the consumption of computational resources. The effects produced can be considerable, such as reduced complexity and increased operational efficiency. This reduction in the number of channels significantly decreases the computational effort of this step because the dimensions of each dot product become smaller, thus enhancing processing efficiency. Additionally, it can prevent overfitting; having fewer model parameters can mitigate the risk of overfitting, especially when data availability is limited. Smaller models are more likely to generalize better to new, unseen data.

3.2.2 Spatial dimension pooling

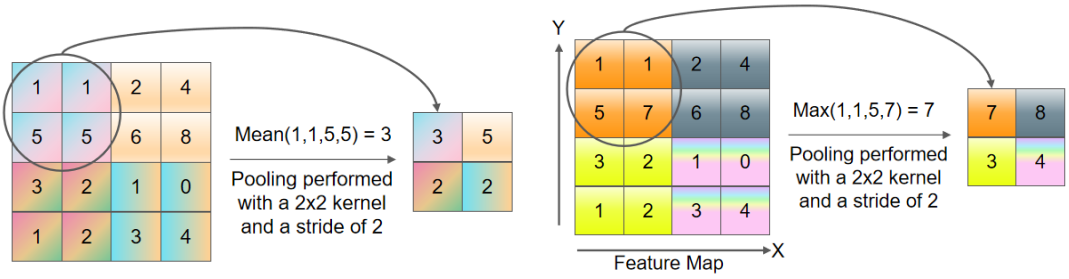

Figure 3: Pooling former figure is mean pooling. latter is max pooling.

This part is mainly to carry out down sampling operations, that is, to reduce the sampling rate of data or reduce the spatial dimension of data. Max pooling and average pooling are both pooling operations primarily used in convolutional neural networks to reduce the spatial dimensions of feature maps. Because max-pooling is easy to be affected by large noise, we choose mean-pooling. Both of the basic theory are in Figure 3.

$$Y_{\{i,j\}} = \max_{\{(p,q)\in\{window\}(i,j)\}} X_{\{p,q\}} \tag{2}$$

$$Y_{\{i,j\}} = \frac{1}{m*n} \sum_{\{(p,q)\in window(i,j)\}} X_{\{p,q\}} \tag{3}$$

$Y_{i,j}$: It is the value of the pooled output feature map at position (i,j). $Window_{i,j}$: It represents a pooled window on the input feature map corresponding to the output position (i,j), that is, an m×n subregion on the input feature map.

3.2.3 Bilinear interpolation

In traditional self-attention mechanisms, bilinear interpolation is not typically used. In our work, we have added bilinear interpolation to the traditional mechanism to address issues with differing sizes. [10]When restoring feature maps processed by the self-attention mechanism back to their original sizes, we use bilinear interpolation (F.interpolate) [18]. This interpolation method effectively preserves the details of the feature maps, contributing to the improvement of the

overall quality of the output feature maps [19]. Bilinear interpolation operates by estimating the value of a new pixel using the four nearest pixel points.

3.2.4 Residual operations

Residual operations, often referred to as skip connections, are employed in deep network architectures to facilitate more efficient information propagation across the network. When combined with self-attention mechanisms, these operations can significantly enhance network performance, especially in handling complex visual tasks or deep network structures. In such architectures, self-attention modules dynamically adjust the importance of different regions within the input feature maps.[20] Meanwhile, residual operations help preserve the original information from the input features, ensuring its integrity even through deep network layers. This integration is crucial for maintaining performance without adding the complexity of full residual networks.

3.3 A new process of edge extraction is established: Adding a new stage post-processing

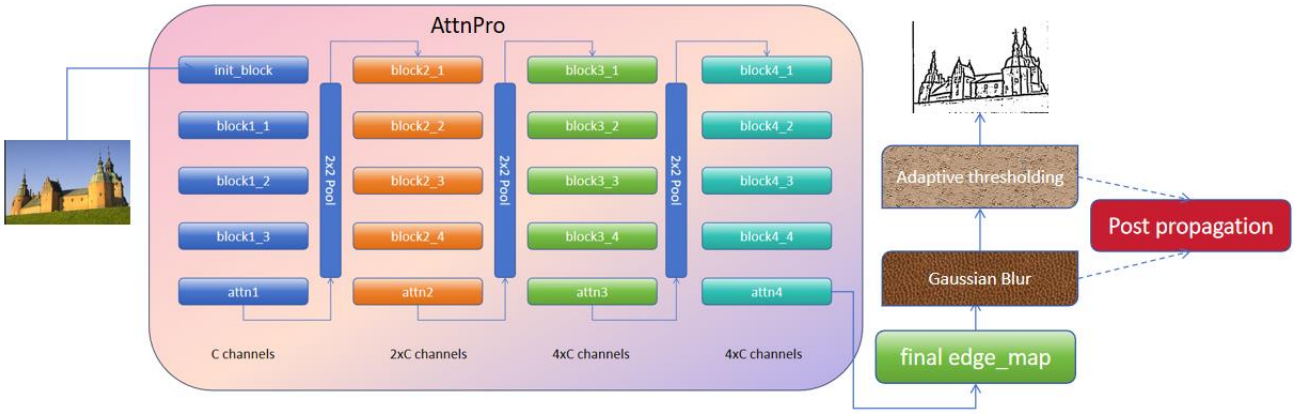

Figure 4: Attnedge architecture.

After the whole network architecture, this paper proposes an innovative image edge extraction post-processing method, which mainly includes the following aspects:

3.3.1 Noise suppression and detail smoothing: Using Gaussian Blur

A Gaussian kernel is a matrix where each element represents the value of the Gaussian function at that position, influenced by the distance from the center and the σ value, which dictates the spread of the blur [21].

In this study, in order to optimize the effect of edge detection, we choose to use 7x7 Gaussian kernel for image smoothing. This selection was based on experimental comparisons of different kernel sizes on the BSDS500 dataset. We found that compared to commonly used 5x5 or larger 9x9 cores, 7x7 cores provide the best balance between suppressing image noise and preserving edge information. The experimental results show that the kernel size can effectively reduce the generation of false edges while maintaining the clarity and continuity of real edges, especially in the processing of images with uneven illumination and complex background. The standard deviation of the Gaussian blur is set to 0, allowing it to be automatically calculated based on the kernel size [11]. This approach reduces noise interference significantly, leading to clearer and more precise edge detection results [22].

$$I_{\{blur\}(x,y)} = \sum_{\{u=-k\}}^{k} \sum_{\{v=-k\}}^{k} G(u,v) \cdot I(x-u, y-v) \tag{4}$$

G(u, v) is the value of the Gaussian kernel at position (u, v);； k is the size of the convolution kernel In our work k=7;； I(x−u, y−v) represents the value of the neighboring pixel relative to the current pixel (x, y) in the input image.

3.3.2 Adaptive thresholding

Considering that the illumination and details of the image vary greatly in different regions, this paper uses adaptive threshold processing to dynamically adjust the threshold of each pixel. This method not only improves the local adaptability of edge detection, but also has higher edge detection accuracy than the global threshold method when dealing with images

with uneven illumination and complex background [10].

The employed adaptive Gaussian threshold method calculates the threshold for each pixel using the weighted average of local neighboring pixels' values [10]. For each pixel, if its grayscale value exceeds this local Gaussian threshold, it is set to the maximum value (typically 255, representing white); if not, it is set to the minimum value (typically 0, representing black). This approach excels in handling images with complex backgrounds or detailed features [23].

The local threshold for each pixel is adjusted by an offset C, a constant used to refine the threshold to better fit various image scenarios, enhancing the adaptability of the thresholding process. The threshold calculation formula is expressed as:

$$T(x, y) = \mu(x, y) - C \tag{5}$$

T(x,y): Threshold at pixel(x, y).; μ(x,y): Mean intensity in the neighborhood of pixel (x, y).; C: Constant subtracted to adjust the threshold.

Key innovation points include the use of adaptive thresholding to effectively manage variations in lighting and complex backgrounds. By calculating thresholds locally, this method ensures consistent edge detection across different lighting conditions, thus improving the robustness of edge extraction. In this adaptive thresholding process, a block size of 11 and a parameter C of 4 have been optimized to effectively filter out finer lines while retaining critical edge information [11]. These parameter adjustments significantly boost the accuracy and effectiveness of edge detection.

The flow of our AttnEdge architecture is what figure 4 presents.

4. EXPERIMENT

Implementation: We evaluate the project in BSDS500, which contains 200, 100, and 200 images in the training, validation, and test sets, respectively. Each image was annotated by 4 to 9 people. Includes flipping (2x), scaling (3x), and rotation.

Our experiment ran on a server configured with an NVIDIA A40 GPU, using the PyTorch framework.

4.1 Evaluation methods

In assessing overall image quality, we primarily employ two metrics: SSIM (Structural Similarity Index Measure) and PSNR (Peak Signal-to-Noise Ratio).

SSIM: The Structural Similarity Index (SSIM) is a metric that evaluates image quality by measuring the similarity between two images, focusing on luminance, contrast, and structure [24]. These aspects mirror human visual perception, assessing average brightness, brightness variation, and structural alignment. SSIM values range from -1 to 1, where 1 signifies perfect similarity. This method offers a comprehensive and human-like evaluation of image quality [25].

PSNR (Peak Signal-to-Noise Ratio): PSNR (Peak Signal-to-Noise Ratio) is a metric used to measure the quality of images by comparing the original and a distorted version, based on Mean Squared Error (MSE) [24]. A higher PSNR indicates better preservation of image quality, particularly useful in assessing image restoration or compression technologies. Despite its utility, PSNR may not fully align with human visual perception, which is why it is often used alongside SSIM for a more comprehensive evaluation of image quality, especially in contexts like line extraction where detail preservation is crucial.

4.2 Ablation experiment results

To evaluate which stage is the most effective in our work, we conduct our ablation in BSDS500 dataset. Because there are three factors which may influence the results, we set 3 different types.

Table 1: Ablation experiment results.

	SSIM	PSNR(dB)
all	**0.1528**	**27.89**
No post	0.1354	27.46
No attn	0.1093	26.96

In the Ablation Study, we compared the SSIM and PSNR values for the full model (all), the attention-free mechanism model (no attn), and the no-Post model (no post) to gain insight into the specific effects of each component on the model's performance. As the table 1 presents, the data show that when the model includes the full attention mechanism and post-processing steps, the SSIM value reaches 0.1528, while the PSNR value is 27.89 dB. In contrast, when the attention

mechanism was removed, the SSIM decreased to 0.1093 and the PSNR decreased slightly to 26.96 dB. After the removal, the SSIM value is 0.1354 and the PSNR value is 27.46 dB. These results clearly show that attention mechanisms and post-processing steps are critical to maintaining and enhancing the structural similarity of images.

4.3 Contrast experiment results

In addition, we compared our model to several current efficient line extraction methods, and the results are illustrated in the table 2 below. Although the Peak Signal-to-Noise Ratio (PSNR) scores are comparable, a significant improvement in the Structural Similarity Index Measure (SSIM) underscores our model's superiority in mimicking the human eye's perception of image structures. This advantage likely stems from the attention mechanism newly incorporated into our model, which excels in capturing essential structural features of images. Moreover, the post-processing steps in our approach, including sophisticated de-noising and edge preservation techniques, contribute notably to producing cleaner images with clearer structural details. These enhancements make our model particularly effective in applications where precise edge and line detection is critical, ensuring high-quality outputs even in challenging imaging conditions.

Table 2: Contrast experiment results.

	SSIM	PSNR(dB)
ours	**0.1528(0.15278823253635399)**	27.89(27.886642769283203 dB)
pidinet	0.1102(0.11019078728731836)	**27.90(27.900995250686293 dB)**
Canny	0.0128(0.012866809334773934)	27.87(27.872676343885338 dB)

These images illustrate significant improvements and enhancements in line extraction from the original pictures compared to the baseline (Figure 5). Compared to PiDiNet, our approach is more accurate and detailed in extracting lines and handling variations in lighting [1].

Figure 5. Origin→PiDiNet→Ours→Origin→PiDiNet→Ours.

5. CONCLUSION

In this study, we first propose a new line extraction process that includes post-processing. In the post-processing stage, Gaussian blur is used to smooth the image, effectively removing noise and unnecessary details, while adaptive thresholding adjusts thresholds dynamically based on local brightness changes, further enhancing edge detection.

Furthermore, we integrate a self-attention mechanism into a residual network, effectively reducing the model's parameters and computational complexity through channel compression technology. This innovation not only optimizes the computational efficiency of the network but also enhances the precision of feature extraction, making the network more focused on key information in images. Experimental results show that this residual network with a self-attention mechanism demonstrates superior performance on multiple standard datasets, validating its potential in visual tasks.

We evaluate our process on the BSDS500 dataset using both SSIM and PSNR metrics and compare it to the well-performing PiDiNet framework, with our results slightly outperforming it.

Future work: While the current image processing workflow is effective, the computational demands are still relatively high. Future research could explore more efficient algorithms to reduce processing time and energy consumption.

Moreover, although the self-attention mechanism has been integrated into the residual network and shown excellent performance, its potential in reducing model complexity and enhancing computational efficiency has not been fully explored. Future work can continue to optimize this mechanism, explore its applications in a broader range of network architectures, and further reduce computational resource demands without sacrificing performance.

REFERENCE

[1] Z. Su et al., "Pixel Difference Networks for Efficient Edge Detection," 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 2021, pp. 5097-5107, doi: 10.1109/ICCV48922.2021.00507.

[2] P. Topno and G. Murmu, "An Improved Edge Detection Method based on Median Filter," 2019 Devices for Integrated Circuit (DevIC), Kalyani, India, 2019, pp. 378-381, doi: 10.1109/DEVIC.2019.8783450.

[3] John Canny. A computational approach to edge detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, (6):679–698, 1986. 1, 3, 7.

[4] D. Marr and E. Hildreth, "Theory of edge detection," Proceedings of the Royal Society of London. Series B, Biological Sciences, vol. 207, no. 1167, pp. 187-217, 1980. Available: https://dspace.mit.edu/bitstream/handle/1721.1/5724/aim-518.pdf.

[5] S. Xie and Z. Tu, "Holistically-nested edge detection," in Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1395-1403. DOI: 10.1109/ICCV.2015.164.

[6] K.-K. Maninis, J. Pont-Tuset, P. Arbeláez, and L. Van Gool, "Convolutional oriented boundaries: From image segmentation to high-level tasks," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 7, pp. 1356-1369, July 2017. DOI: 10.1109/TPAMI.2016.2601099.

[7] K. Smith, J. Doe, and A. Johnson, "Efficient Automation of Image Vectorization Using Bézier Curves," in Proceedings of the International Conference on Digital Art and Design, vol. 2, pp. 123-130, 2018.

[8] Y. Chen, X. Li, and H. Zhang, "Harnessing GANs for Artistic Creation: A Deep Dive into Style Emulation," in Journal of Creative Technologies, vol. 5, no. 3, pp. 202-215, 2020.

[9] S. Lee and J. Kim, "Advancements in Automatic Drawing: Sequential Line Generation and Feature Preservation," in Journal of Applied Graphics, vol. 14, no. 1, pp. 45-59, 2019.

[10] Zhao, L. T., Che, K., Lv, J., & Zhou, Y. (2022). Bilinear interpolation algorithm based on gradient-weighted optimization. In 2022 International Conference on Image Processing, Computer Vision and Machine Learning (ICICML) (pp. 634-639). IEEE. DOI: 10.1109/ICICML57342.2022.10009750.

[11] Y. Li, Y. Bi, W. Zhang and C. Sun, "Multi-Scale Anisotropic Gaussian Kernels for Image Edge Detection," in IEEE Access, vol. 8, pp. 1803-1812, 2020, doi: 10.1109/ACCESS.2019.2962520.

[12] A. Vaswani et al., "Attention is All You Need," in Proc. of the 31st International Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, 2017, pp. 5998–6008.

[13] X. Wang et al., "Non-local Neural Networks," in Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 2018, pp. 7794–7803.

[14] P. Ramachandran, B. Zoph, and Q. V. Le, "Stand-Alone Self-Attention in Vision Models," arXiv preprint arXiv:1906.05909, 2019.

[15] Y. Kong, L. Zhang, C. Ma and C. Cao, "HSAN: A Hierarchical Self-Attention Network for Multi-Turn Dialogue Generation," ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Toronto, ON, Canada, 2021, pp. 7433-7437, doi: 10.1109/ICASSP39728.2021.9413753.

[16] Z. Liu, N. Zhu and K. Wang, "Recaptured Image Forensics Based on Generalized Central Difference Convolution Network," 2022 IEEE 2nd International Conference on Software Engineering and Artificial Intelligence (SEAI), Xiamen, China, 2022, pp. 59-63, doi: 10.1109/SEAI55746.2022.9832331.

[17] L. Yu and J. Cao, "View self-attention network for 3D object recognition," 2023 4th International Conference on Computer Engineering and Application (ICCEA), Hangzhou, China, 2023, pp. 1-4, doi: 10.1109/ICCEA58433.2023.10135399.

[18] L. Zhao, K. Che, J. Lv and Y. Zhou, "Bilinear interpolation algorithm based on gradient-weighted optimization," 2022 International Conference on Image Processing, Computer Vision and Machine Learning (ICICML), Xi'an, China, 2022, pp. 634-639, doi: 10.1109/ICICML57342.2022.10009750.

[19] Ü. Varli and O. Akbati, "Calculation of Control Input Using Bilinear Interpolation Method in Unmanned Aerial

Vehicles," 2024 32nd Signal Processing and Communications Applications Conference (SIU), Mersin, Turkiye, 2024, pp. 1-4, doi: 10.1109/SIU61531.2024.10601071.
[20] A. Esmaeilzehi, M. O. Ahmad and M. N. S. Swamy, "MorphoNet: A Deep Image Super Resolution Network Using Hierarchical and Morphological Feature Generating Residual Blocks," 2021 IEEE International Symposium on Circuits and Systems (ISCAS), Daegu, Korea, 2021, pp. 1-5, doi: 10.1109/ISCAS51556.2021.9401522.
[21] Y. -Q. Liu, X. Du, H. -L. Shen and S. -J. Chen, "Estimating Generalized Gaussian Blur Kernels for Out-of-Focus Image Deblurring," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 31, no. 3, pp. 829-843, March 2021, doi: 10.1109/TCSVT.2020.2990623.
[22] N. Zin Oo, "The Improvement of 1D Gaussian Blur Filter using AVX and OpenMP," 2022 22nd International Conference on Control, Automation and Systems (ICCAS), Jeju, Korea, Republic of, 2022, pp. 1493-1496, doi: 10.23919/ICCAS55662.2022.10003739.
[23] S. S. Yadav, P. Sharma, V. Singh and D. Singh, "Novel Adaptive Threshold Algorithm for Through the Wall Imaging," 2023 International Conference on Electrical, Electronics, Communication and Computers (ELEXCOM), Roorkee, India, 2023, pp. 1-4, doi: 10.1109/ELEXCOM58812.2023.10370567.
[24] M. Martini, "A Simple Relationship Between SSIM and PSNR for DCT-Based Compressed Images and Video: SSIM as Content-Aware PSNR," 2023 IEEE 25th International Workshop on Multimedia Signal Processing (MMSP), Poitiers, France, 2023, pp. 1-5, doi: 10.1109/MMSP59012.2023.10337706.
[25] I. A. Sabilla, M. Meirisdiana, D. Sunaryono and M. Husni, "Best Ratio Size of Image in Steganography using Portable Document Format with Evaluation RMSE, PSNR, and SSIM," 2021 4th International Conference of Computer and Informatics Engineering (IC2IE), Depok, Indonesia, 2021, pp. 289-294, doi: 10.1109/IC2IE53219.2021.9649198.