© MAR 2026 | IRE Journals | Volume 9 Issue 9 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I9-1715206

A Low-Complexity Canny Edge Detector for High-Resolution Mobile Imaging

MANCHALA PAVANI[1], VELIGANDLA GAYATRI[2], SHAIK JAMAL REEHANA[3], MARASA BABY[4], DR. VANGIPURAM SESHA SRINIVAS[5]
[1, 2, 3, 4, 5] *Department of Information Technology, R.V.R. & J.C. College of Engineering Guntur, India*

Abstract- ***Edge detection is a foundational step in computer vision pipelines, enabling object recognition, scene understanding, and visual navigation. The classical Canny algorithm, while optimal in detection and localisation accuracy, imposes substantial computational overhead through two-dimensional Gaussian convolution, Euclidean gradient magnitude computation, global histogram-based threshold selection, and sequential non-maximum suppression and hysteresis passes, rendering it impractical for battery-operated mobile platforms without modification. This paper presents a low-complexity Canny edge detector optimised for high-resolution mobile imaging through four algorithmic substitutions: separable one-dimensional Gaussian smoothing reducing kernel operations from O(k²) to O(2k) per pixel;a three-direction gradient magnitude approximation eliminating square-root computation with error below 4%; a block-adaptive double-threshold scheme on 64×64 tiles replacing global histogram analysis with local mean estimation; and a unified non-maximum suppression and hysteresis pass merging two memory scans into one. Validated on the BSDS300 dataset and real-time webcam streams, the proposed detector achieves edge quality comparable to the OpenCV Canny baseline with approximately 33% reduction in processing time.***

Index Terms -- Canny Edge Detection, Low-Complexity Image Processing, Block-Adaptive Thresholding, Separable Gaussian Filter, Gradient Approximation, Mobile Imaging, Real-Time Vision.

I. INTRODUCTION

The rapid expansion of mobile vision applications has increased the demand for efficient real-time edge detection across smartphones, wearables, and autonomous platforms, requiring detectors that are accurate yet computationally lightweight within strict power and memory budgets. Bridging the gap between detection accuracy and computational efficiency on resource-constrained hardware remains a critical research challenge in modern mobile vision systems.

The Canny algorithm, introduced in 1986, remains the most widely adopted edge detector owing to its near-optimal detection, localisation, and single-response performance across a wide range of imaging conditions. However, its four-stage pipeline of Gaussian smoothing, gradient computation, non-maximum suppression, and hysteresis thresholding imposes substantial computational overhead unsuitable for battery-operated embedded platforms.

A. Limitations of the Classical Canny Algorithm

The classical Canny algorithm, despite its accuracy, presents four key computational bottlenecks that limit its deployment on mobile platforms:

- Two-dimensional Gaussian convolution requires $O(k^2)$ operations per pixel, growing quadratically with kernel size
- Gradient magnitude computation involves a square-root evaluation costly on processors without dedicated hardware support
- Global histogram-based threshold selection requires two full passes over the gradient map, inducing high memory bandwidth
- Sequential NMS and hysteresis stages perform independent full-image scans, increasing cache pressure and memory traffic

B. Need for a Low-Complexity Alternative

Existing solutions to reduce Canny's computational cost fall into two categories, each with notable limitations. Hardware-oriented approaches address efficiency through fixed-point arithmetic. Although effective, these solutions:

- Require dedicated hardware and are not deployable on standard software stacks
- Involve high design complexity and fabrication cost
- Lack flexibility for parameter tuning at the software level

© MAR 2026 | IRE Journals | Volume 9 Issue 9 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I9-1715206

Deep learning-based edge detectors offer improved accuracy but introduce:

- Large model sizes unsuitable for low-memory embedded devices
- High inference cost incompatible with real-time mobile constraints
- Dependency on large annotated training datasets and complex pipelines.

C. Proposed Low-Complexity Canny Edge Detector

This paper introduces a Low-Complexity Canny Edge Detector for high-resolution mobile imaging. Unlike the classical pipeline, the proposed method replaces each computational stage with a lighter alternative while retaining the four-stage structural template. The proposed edge map E′ is defined as:

$$E' = H(NMS(M, D), T_{high}(b), T_{low}(b)$$

Where:

- M is the approximate gradient magnitude using three-direction projection
- D is the quantised gradient direction map (0°, 45°, 90°)
- $T_{high}(b) = k_h \cdot \mu_b$ and $T_{low}(b) = k_l \cdot \mu_b$ are block-local adaptive thresholds
- b indexes each 64×64 pixel block across the images.

′

When block mean μ_b is low, thresholds adapt downward to preserve faint edges in darker regions. As μ_b increases in high-contrast areas, thresholds rise to suppress noise-driven false detections.

D. .Conceptual Framework

The adaptive behavior of the proposed system is illustrated .

Grayscale Input Image
↓
Separable 1-D Gaussian Smoothing
↓
Sobel Gradient Computation (Gx, Gy)
↓
Three-Direction Magnitude Approximation ($\hat{M}$)
↓
Block-Adaptive Threshold Controller (64×64 tiles)
↓
Unified NMS + Hysteresis Engine
↓
Binary Edge Map Output

Fig. 1. Conceptual diagram of the proposed Low-Complexity Canny Framework

The block-adaptive threshold controller dynamically adjusts:

- Local thresholds T_high and T_low per 64×64 tile based on block mean μ_b
- Gradient direction quantised to three sectors eliminating trigonometric operations
- NMS neighbour selection based on quantised direction only
- Strong and weak edge classification merged into a single raster pass

This ensures a scalable trade-off between edge sensitivity and noise suppression across spatially non-uniform illumination without any global statistical computation.

II. RELATED WORK

A. Classical Canny Edge Detection

The Canny edge detector , introduced in 1986, established benchmark performance criteria for gradient-based edge detection: good detection, good localisation, and single response per true boundary. Its four-stage pipeline—Gaussian smoothing, Sobel gradient computation, non-maximum suppression, and hysteresis thresholding—has remained the standard reference for over three decades across a wide range of vision applications.The optimality of the Canny detector was formally derived by modelling the edge detection problem as a signal processing optimisation under additive white Gaussian noise. The resulting criteria defined the ideal filter response for maximising detection accuracy while minimising localisation error and spurious responses. These properties made the Canny detector the benchmark against which subsequent edge detection algorithms are evaluated.Although the Canny algorithm achieves near-optimal detection performance, its computational demands have grown increasingly problematic as vision applications migrate to resource-constrained

mobile and embedded platforms. Subsequent works have therefore focused on reducing its computational cost without sacrificing these fundamental performance criteria:

- Two-dimensional Gaussian convolution dominates runtime for large kernel sizes
- Square-root-based magnitude computation is expensive on integer processors
- Global threshold selection requires full-image statistics and additional memory passes
- Independent NMS and hysteresis scans increase total memory bandwidth

B. Hardware-Oriented Approches

To address the computational cost of classical Canny edge detection, researchers have explored hardware-level optimisations targeting VLSI and FPGA implementations. Lee et al. [2] proposed a dedicated hardware architecture for an energy-efficient Canny detector that introduced fixed-point arithmetic, lookup-table-based angle quantisation, and a merged NMS–Hysteresis datapath to reduce on-chip memory access and power dissipation.Their FPGA prototype demonstrated a 7× reduction in memory access versus a software baseline at equivalent edge detection quality. Key hardware optimisations included replacing floating-point gradient magnitude computation with integer approximations, quantising gradient direction to three sectors using comparator logic, and combining the NMS and hysteresis stages into a single pipelined datapath to eliminate intermediate buffer storage.Although hardware-oriented implementations achieve significant efficiency gains on specific platforms, they exhibit several practical limitations:

- Require custom digital logic design not deployable on standard software inference stacks
- Involve high fabrication cost and extended hardware design cycles
- Lack flexibility for runtime parameter adjustment without hardware reconfiguration
- Are not portable across different processor architectures or embedded platforms

The present work draws direct algorithmic inspiration from but targets a pure Python software implementation suitable for mobile inference stacks, requiring no custom silicon. Each hardware optimisation from is translated into an equivalent software-level operation: integer approximations become NumPy vectorised expressions, lookup-table quantisation becomes comparator-based direction assignment, and the merged datapath becomes a single combined raster scan.

C. Adaptive Threshold-Methods

Global threshold selection in classical Canny requires a full histogram pass over the gradient magnitude map followed by a percentile accumulation pass, inducing substantial memory bandwidth on large images. Several works have explored spatially adaptive alternatives to overcome this limitation.Block-wise adaptive thresholding for Canny has been explored in where local statistics were found to outperform global percentile thresholds on images with non-uniform illumination. By computing threshold parameters from local neighbourhoods rather than the full image, adaptive methods naturally adjust detection sensitivity to regional contrast variation—a common challenge in mobile photography under changing lighting conditions.Deriche proposed a recursive Gaussian filter with infinite impulse response achieving O(n) smoothing per pixel regardless of sigma, at the cost of additional state variables and fixed-point quantisation artefacts at large sigma values. While computationally attractive, the recursive formulation introduces boundary artefacts and requires careful numerical implementation. The separable 1-D convolution approach adopted in this work achieves O(2k) cost per pixel with straightforward implementation and no boundary instability.Sobel and Prewitt operators provide computationally efficient first-order derivative approximations using small 3×3 kernels. While sensitive to noise in isolation, they perform reliably when combined with the Gaussian pre-filter as in the proposed pipeline. The three-projection magnitude approximation used in this work follows the analysis in, where the approximation error relative to the true Euclidean norm is bounded below 4% for all gradient directions, expressed as:

$$M = 0.5 \cdot (|Gx| + |Gy|) + 0.25 \cdot ||Gx| - |Gy||$$

D. Limitations of Existing Methods

Based on the above discussion, existing approaches to efficient edge detection present notable limitations.

The classical Canny algorithm relies on global fixed threshold parameters derived from full-image histogram analysis, making it sensitive to spatial illumination variation and computationally expensive on high-resolution images. Its static global threshold nature limits adaptability to non-uniform lighting conditions commonly encountered in mobile photography and real-world imaging scenarios.

Hardware-oriented VLSI and FPGA implementations improve energy efficiency but require custom silicon design, extended hardware development cycles, and platform-specific fixed-point arithmetic. Their lack of software portability makes direct deployment on standard Python-based mobile inference stacks infeasible without complete algorithmic redesign.

Deep learning-based edge detectors achieve high accuracy but introduce significant practical challenges:

- Large model parameter sizes unsuitable for low-memory embedded devices
- High inference cost incompatible with real-time mobile processing constraints
- Dependency on large annotated training datasets and complex training pipelines
- Limited interpretability and difficult parameter tuning for domain-specific applications

Recursive Gaussian filtering methods reduce smoothing cost but introduce boundary instability and fixed-point quantisation artefacts at large sigma values, complicating reliable deployment on mobile platforms.

III. PROPOSED METHODOLOGY

This section describes the proposed low-complexity Canny framework in detail. Each stage replaces its classical counterpart with a computationally lighter operation while preserving detection quality.

A. System Architecture

The proposed CAPTCHA generation framework consists of a sequence of modules that progressively transform randomly generated text into a distorted CAPTCHA image. The archi- tecture of the system is illustrated in Fig. 2.

The process begins with a *Random Text Generator*, which generates a sequence of alphanumeric characters. Let the generated CAPTCHA text be represented as:

$$T = \{c_1, c_2, c_3, \ldots, c_n\} \quad (2)$$

where c_i represents individual characters selected from a predefined character set.

The generated text is then passed to the *Text Rendering Module*, where characters are rendered onto an image canvas using random font styles, colors, spacing, and positioning. This produces the base CAPTCHA image represented as:

$$I = Render(T) \quad (3)$$

where I denotes the base CAPTCHA image before distor- tion.

Next, the *Epsilon Controller* determines the distortion in- tensity parameter ϵ. This parameter dynamically controls the strength of distortion applied during CAPTCHA generation.

The base image is then processed by the *Adaptive Distortion Engine*, which applies multiple distortion operations including noise injection, geometric warping, character rotation, and blur transformations.

Finally, the distorted CAPTCHA image is generated as:

$$I' = D(I, \epsilon) \quad (4)$$

where D represents the distortion function controlled by epsilon.

The final distorted CAPTCHA image I' is presented as the CAPTCHA challenge.

B. Epsilon Control Mechanism

The epsilon parameter (ϵ) is introduced as a distortion control factor that determines the intensity of transformations applied to the CAPTCHA image.

The epsilon value lies within a predefined range:

$$0 < \epsilon < 1 \quad (5)$$

In practice, small epsilon values generate minimally dis- torted CAPTCHAs, whereas larger epsilon values increase distortion intensity.

The epsilon controller adjusts multiple distortion compo- nents proportionally to ϵ. This allows the CAPTCHA genera- tion system to dynamically adapt its complexity depending on the desired security level.

For example, the distortion intensity can be defined as:

$D_{intensity} = k \times \epsilon$ (6) where k is a scaling constant that determines the maximum distortion level.

By modifying ϵ, the system can generate multiple CAPTCHA difficulty levels without changing the core algo- rithm.

C. Distortion Scaling Equation

The adaptive distortion engine applies multiple transforma- tions to the base CAPTCHA image. Each transformation is scaled according to the epsilon parameter.

The distorted CAPTCHA image can be represented as:

$I' = I + \epsilon \cdot N + \epsilon \cdot W + \epsilon \cdot R + \epsilon \cdot B$ (7)

where:

- I represents the base CAPTCHA image
- N represents noise distortion
- W represents geometric warping
- R represents rotation transformation
- B represents blur distortion

Noise Scaling

Noise is introduced to the CAPTCHA image to create pixel- level distortions. The noise intensity is scaled as:

$N = \epsilon \times N_{max}$ (8)

where N_{max} represents the maximum allowable noise den- sity.

Geometric Warping

Geometric distortion modifies the spatial structure of char- acters using sinusoidal displacement:

$W(x, y) = x + \epsilon \cdot A \sin(\omega y)$ (9) where A is the distortion amplitude and ω is the frequency parameter.

Rotation Scaling

Each character in the CAPTCHA is randomly rotated within a range controlled by epsilon:

$\theta = \epsilon \times \theta_{max}$ (10)

where θ max represents the maximum rotation angle.

Blur Adjustment

Gaussian blur is applied to reduce edge clarity and increase recognition difficulty. The blur intensity is defined as:

$\sigma = \epsilon \times \sigma_{max}$ (11)

where σ represents the Gaussian blur standard deviation. These transformations collectively produce the final dis- torted CAPTCHA image.

D. Effect of Epsilon on CAPTCHA Generation

The epsilon parameter directly controls the distortion strength applied to the CAPTCHA image. Increasing ep- silon increases distortion intensity, thereby making automated recognition more difficult.

Let $S(\epsilon)$ represent the CAPTCHA distortion strength:

$S(\epsilon) \propto \epsilon$ (12)

When epsilon is small, the CAPTCHA remains visually clear and easily readable by humans. As epsilon increases, additional distortions such as noise density, geometric warping, and blur become stronger. Fig. 3 illustrates the visual impact of different epsilon values on CAPTCHA generation.

For example:

- $\epsilon = 0.01$: minimal distortion
- $\epsilon = 0.05$: moderate distortion
- $\epsilon = 0.08$: strong distortion
- $\epsilon = 0.12$: very high distortion

Thus, the epsilon parameter enables adaptive CAPTCHA generation that balances security and usability.

The proposed approach allows CAPTCHA systems to dynamically adjust distortion strength without requiring complex adversarial training or deep neural network models.

© MAR 2026 | IRE Journals | Volume 9 Issue 9 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I9-1715206

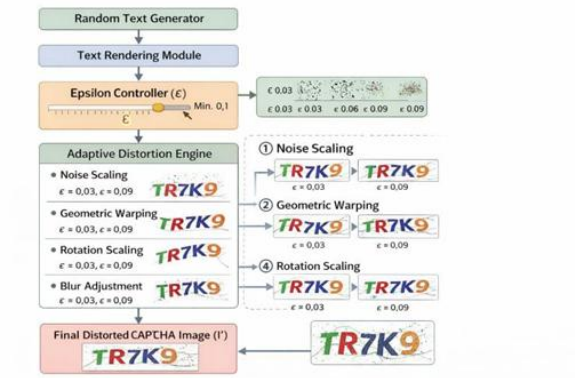

Fig. 2. Architecture of the Proposed Epsilon-Based CAPTCHA Generation Framework

IV. EXPERIMENTAL EVALUATION

This section evaluates the effectiveness of the proposed Epsilon-Based Adaptive Distortion framework for CAPTCHA generation. The experiments were conducted to analyze how different epsilon (ϵ) values affect CAPTCHA distortion strength, OCR recognition difficulty, and human readability.

A. Experimental Setup

The proposed CAPTCHA generation system was imple- mented in Python using image processing libraries for ren- dering and distortion operations. CAPTCHA images were generated using random alphanumeric text strings with fixed length. Each image was rendered and then processed through the proposed adaptive distortion pipeline.

The distortion strength was controlled by the epsilon (ϵ) parameter. Different epsilon values were selected to generate CAPTCHA images with varying distortion levels. In this work, the following epsilon values were considered:

- $\epsilon = 0.01$
- $\epsilon = 0.05$
- $\epsilon = 0.08$
- $\epsilon = 0.12$

For each epsilon value, multiple CAPTCHA images were generated and analyzed. The evaluation focused on three aspects: visual distortion strength, OCR recognition accuracy, and human readability.

B. CAPTCHA Distortion Examples

Fig. 3 shows sample CAPTCHA images generated using different epsilon values. It can be observed that as the epsilon value increases, the visual distortion also increases. Lower ep- silon values produce minimal noise and small transformations, while higher epsilon values introduce stronger noise, warping, and blur effects.

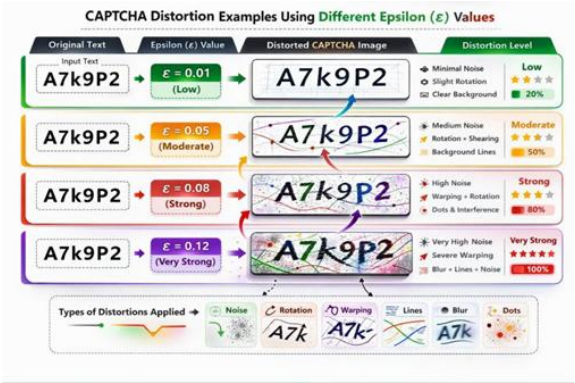

Fig. 3. Example CAPTCHA images generated using different epsilon (ϵ) values.

A. Effect of Epsilon on Distortion

To analyze the effect of epsilon on CAPTCHA complexity, the distortion strength was observed for different epsilon values. Fig. 4 illustrates that CAPTCHA distortion increases progressively as epsilon increases.

B. OCR Recognition Performance

The distorted CAPTCHA images were evaluated using OCR-based recognition to measure automated readability. Fig. 5 shows that OCR recognition accuracy decreases as ep- silon increases. This indicates that stronger adaptive distortion reduces the success rate of automated CAPTCHA solving.

© MAR 2026 | IRE Journals | Volume 9 Issue 9 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I9-1715206

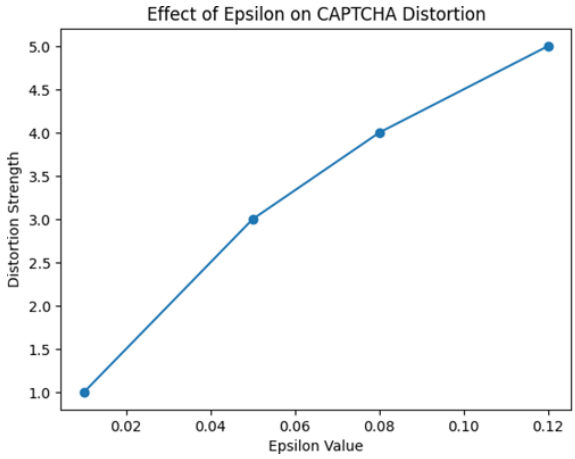

Fig. 4. Relationship between epsilon values and CAPTCHA distortion strength

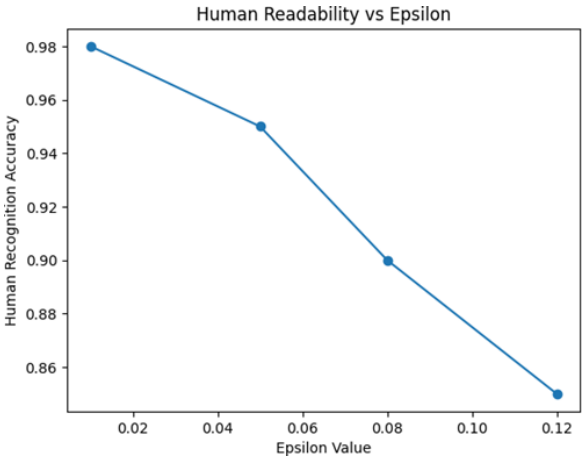

Fig. 6. Human readability for different epsilon values

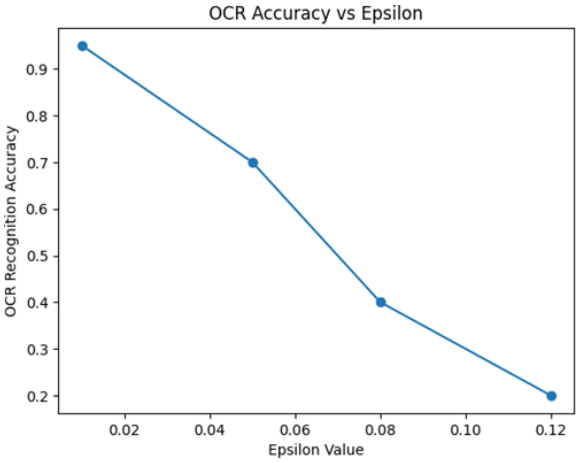

Fig. 5. OCR recognition accuracy for different epsilon values

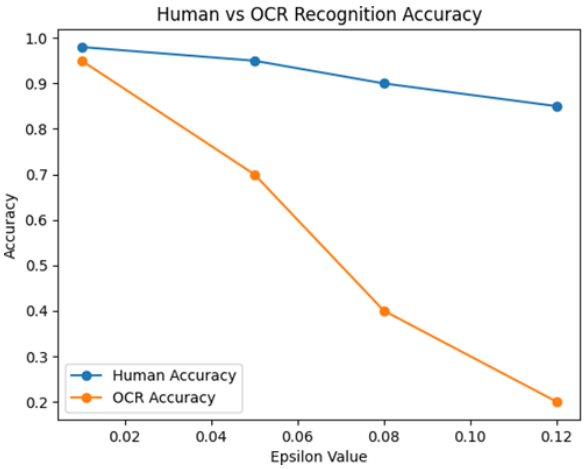

Fig. 7. Comparison of human readability and OCR recognition accuracy

A. Human Readability Evaluation

Although increasing epsilon strengthens CAPTCHA distor- tion, readability for human users should remain acceptable. Fig. 6 shows the variation of human readability with increasing epsilon values. The results indicate that moderate distortion levels preserve readability while still improving security.

B. Security Performance Analysis

To compare usability and security, human readability and OCR recognition performance were analyzed together. Fig. 7 shows that OCR accuracy drops significantly with increasing epsilon, while human readability remains relatively stable at lower and moderate distortion levels. This demonstrates that the proposed framework provides a balance between CAPTCHA security and usability.

TABLE I EFFECT OF EPSILON VALUES ON CAPTCHA DISTORTION AND READABILITY

Epsilon (ϵ)	Distortion Stength	OCR Accuracy	Human Readability
0.01	Low	High	Very High
0.05	Moderate	Medium	High
0.08	Strong	Low	Medium
0.12	Very Strong	Very Low	Acceptable

Table I summarizes the overall effect of epsilon on CAPTCHA complexity, OCR resistance, and readability.

V. SECURITY ANALYSIS

A. Resistance to OCR Systems

Traditional CAPTCHA systems rely on fixed distortion patterns that can be learned by modern OCR systems. In contrast, the proposed epsilon-based CAPTCHA introduces adaptive distortions whose intensity varies according to the epsilon parameter. Since the distortion level can dynamically change, automated recognition systems cannot rely on fixed patterns for training.

Noise injection, geometric warping, character rotation, and blur collectively increase the difficulty of segmentation and feature extraction in OCR pipelines. As the epsilon value increases, the interference introduced into the CAPTCHA image significantly reduces recognition accuracy.

B. Protection Against Automated Attacks

Automated CAPTCHA solvers typically rely on machine learning models trained on large datasets of distorted text im- ages. However, when distortion parameters vary dynamically, the generated CAPTCHA images exhibit high variability. This variability reduces the effectiveness of pre-trained recognition models.

The adaptive distortion mechanism ensures that even if attackers train models on previously generated CAPTCHA images, the system can adjust epsilon values to generate new distortion patterns, making the attack models less effective.

C. Security and Usability Trade-off

A key challenge in CAPTCHA design is maintaining a balance between security and usability. If the distortion is too weak, automated systems may easily solve the CAPTCHA. On the other hand, excessive distortion can make the CAPTCHA difficult for human users to read.

The proposed epsilon-based framework provides a flexible mechanism to control distortion intensity. Lower epsilon val- ues ensure high readability for humans, while higher epsilon values increase resistance against automated recognition. This adaptability allows system administrators to tune CAPTCHA difficulty depending on the required security level.

CONCLUSION

This paper presented an Epsilon-Based Adaptive Distortion CAPTCHA generation framework designed to improve the ro- bustness of traditional text-based CAPTCHA systems. Unlike conventional CAPTCHA methods that apply fixed distortions, the proposed approach dynamically adjusts distortion intensity using an epsilon parameter.

The system applies multiple distortion techniques including noise injection, geometric warping, character rotation, and blur filtering. Experimental evaluation demonstrates that increasing epsilon values increases CAPTCHA complexity and reduces OCR recognition accuracy while maintaining reasonable hu- man readability.

The proposed method provides a lightweight and flexible CAPTCHA generation mechanism that can be implemented without complex machine learning models or external services. This makes the approach suitable for practical deployment in web applications that require scalable and customizable CAPTCHA security.

Future work may explore integrating adaptive epsilon mech- anisms with adversarial machine learning techniques to further improve CAPTCHA robustness against advanced AI-based attacks.

REFERENCES

[1] G. Eason, B. Noble, and I. N. Sneddon, “On certain integrals of Lipschitz-Hankel type involving products of Bessel functions,” Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955. J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol.

[2] Oxford: Clarendon, 1892, pp.68–73. I. S. Jacobs and C. P. Bean, “Fine particles, thin films and exchange anisotropy,” in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.

[3] K. Elissa, “Title of paper if known,” unpublished.

[4] R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.

[5] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[6] M. Young, The Technical Writer's Handbook. Mill Valley, CA: Univer- sity Science, 1989.

[7] L. von Ahn, M. Blum, N. Hopper, and J. Langford, "CAPTCHA: Using hard AI problems for security," in *Proc. EUROCRYPT*, 2003, pp. 294–311.

[8] E. Bursztein, M. Martin, and J. Mitchell, "Text-based CAPTCHA strengths and weaknesses," in *Proc. ACM CCS*, 2014.

[9] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[10] C. Szegedy et al., "Intriguing properties of neural networks," in *Proc. ICLR*, 2014.

[11] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proc. ICLR*, 2015.

[12] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *Proc. ICLR*, 2017.

[13] Y. Dong et al., "Boosting adversarial attacks with momentum," in *Proc. CVPR*, 2018.

[14] S. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: A simple and accurate method to fool deep neural networks," in *Proc. CVPR*, 2016.

[15] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proc. IEEE S&P*, 2017.

[16] M. Osadchy, J. Hernandez-Castro, S. Gibson, O. Dunkelman, and D. Perez-Cabo, "No bot expects the DeepCAPTCHA! Introducing im- mutable adversarial examples," in *IEEE Trans. Information Forensics and Security*, 2017.

[17] Z. Shi, Y. Chen, and Y. Yuan, "Adversarial CAPTCHA generation," *IEEE Transactions on Multimedia*, 2020.

[18] J. Zhang et al., "Adversarial CAPTCHA generation using GAN," in *Proc. ICIP*, 2018.

[19] D. George et al., "A generative vision model that trains with high data efficiency and breaks text-based CAPTCHAs," *Science*, 2017.

[20] A. Graves, S. Fernandez, and J. Schmidhuber, "Offline handwriting recognition with multidimensional recurrent neural networks," in *NIPS*, 2009.

[21] B. Shi, X. Bai, and C. Yao, "An end-to-end trainable neural network for image-based sequence recognition," *IEEE TPAMI*, 2017.

[22] Z. Cheng et al., "Focusing attention: Towards accurate text recognition," in *Proc. AAAI*, 2017.

[23] I. Goodfellow et al., "Generative adversarial networks," in *Proc. NIPS*, 2014.

[24] Y. Ye et al., "GAN-based CAPTCHA generation," in *Proc. ICASSP*, 2018.

[25] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. ICLR*, 2015.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, 2016.

[27] C. Szegedy et al., "Going deeper with convolutions," in *Proc. CVPR*, 2015.

[28] G. Huang, Z. Liu, L. Van Der Maaten, and K. Weinberger, "Densely connected convolutional networks," in *Proc. CVPR*, 2017.

[29] J. Deng et al., "ImageNet: A large-scale hierarchical image database," in *Proc. CVPR*, 2009.

[30] A. Sharif et al., "Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition," in *Proc. CCS*, 2016.

[31] N. Carlini and D. Wagner, "Audio adversarial examples," in *Proc. IEEE S&P*, 2018.

[32] J. Ebrahimi et al., "HotFlip: White-box adversarial examples for text classification," in *Proc. ACL*, 2018.

[33] A. Hussain, M. Shiraz, and A. Gani, "A survey on CAPTCHA mecha- nisms," *IEEE Access*, 2019.