Grupa: Metroflex
Lukas Baltas, Lucija Kristić, Ivan Lipovac

December 2025.

Steganography
Matrix and tensor methods

Introduction

Nowadays, the problem of identifying and securing authentic content is gaining more light than ever. To protect the copyright of the multimedia authors, usually some kind of watermark is embedded into the author's image (host image), so that he/she can claim ownership over it when is later extracted from the host image.

A wide variety of image watermarking techniques has been proposed in the literature. They are usually grouped into two main categories: spatial-domain methods and transform-domain methods, depending on whether the watermark is embedded directly into pixel values or into transformed coefficients. Transform-domain approaches are generally more robust to common image-processing attacks, because the watermark is inserted into perceptually or statistically significant components of the image.

Among transform-domain techniques, Singular Value Decomposition (SVD) has become one of the most powerful tools due to its strong numerical properties and the meaningful interpretation of singular values. Depending on how SVD is used, many different watermarking schemes can be constructed. For example, SVD can be applied to the entire host image or only to selected sub-blocks; the watermark can be embedded by modifying the singular values (the diagonal matrix D) with an appropriate scaling factor; or SVD can be computed on both the host and the watermark image. However, schemes that require SVD of both images and a modified D matrix are typically non-blind, i.e. they require the original watermark during extraction. Later work proposed embedding the watermark only into the D matrix using quantization, but these methods often failed to achieve zero Bit Error Rate in extraction.

More recent approaches also exploit the U matrix of the SVD and introduce variants of dither or quantization-based embedding in the SVD domain. Unfortunately, many of these techniques either suffer from limited robustness to common attacks or remain non-blind in nature. This motivates the development of watermarking schemes that combine the advantages of SVD-based embedding with blind detection and improved robustness.

About our main paper

The proposed watermarking scheme in the paper is robust, has reasonably good capacity and is blind in nature. It uses SVD Domain and Dither quantization for embedding the watermark in both D and U. The largest singular values of the host image (D matrix coefficients) and coefficients of the U matrix are modified to embed the watermark data such as logo, so that if extraction of watermark image fails from the D matrix, there's a good chance that it can be done from the U matrix. The host image is partitioned into four sub images and the embedding is done in the upper left and bottom right sub image. The number of sub images can vary by partition of the host image, but by increasing the number of them information hiding capacity is reduced. This kind of algorithm is more secure and robust to various attacks such as JPEG2000 compression, rotation, scaling, cropping, salt and paper noise, filtering and gamma correction. Also, it was shown that it gives better results in terms of Bit Error Rate (BER), Normalized Cross correlation (NC) and Peak Signal to Noise Ration (PSNR) over recent proposed schemes.

Covered and used theory

Theorem 1 SVD
If $A \epsilon R^{mxn}$, then there exist orthogonal matrices U and V, also mxn, such that $U^T AV = diag(\sigma_1, \dots, \sigma_p)$, where p=min(m,n), and $\sigma_1 \geq \cdots \geq \sigma_p \geq 0$. σ_i are singular values or square roots of eigen values λ_i of $A^H A$ or AA^H.
Theorem 2 stability of SVD
The stability of singular value indicates that, when there is a little disturbance with A, the variation of its singular value is not greater than 2-norm of disturbance matrix. 2-norm is equal to the largest singular value of the matrix.
Theorem 3
If $\sigma_1, \dots, \sigma_k$ are singular values of matrix A and $\sigma_1^*, \dots, \sigma_k^*$ are singular values of $\alpha * A$, then $|\alpha|(\sigma_1, \dots, \sigma_k) = (\sigma_1^*, \dots, \sigma_k^*)$.
Theorem 4 rotation invariant property
If P is unitary and rotating matrix, the singular values od PA (rotated matrix) are the same as those od A
Theorem 5 translation invariance property
The original image A and its rows or columns of interchanged image, have the same singular values
Theorem 6 transposition invariance property
If $AA^H u = \lambda^2 u$, then $AA^H v = \lambda^2 v$, so that A and A^H have same singular values.
Dither quantization
Ideally, embedding of named two images should be done in a way that minimizes the distortion between the host image and watermarked image and maximizes the information embedding rate and robustness od embedding. These demands are usually very conflicting, so the wanted process has to be an efficient tradeoff of these requirements.
In the Dither quantization based watermarking schemes, the embedded image modulates a dither signal, and the host signal is quantized with an associated dithered quantizer. It has a lot od advantages over convention spread spectrum-based schemes, one of them being that it can effectively hide exact value of the host signal rather than combining the host image and watermark image in linear way.

Proposed watermarking scheme is a binary watermark that is embedded in the gray scale host image, meaning that watermark image consist of ‘1’s or ‘0’s. Each quantization cell in the ensemble is constructed from basic quantizer. Basic quantizer is uniform scalar with a fixed step size T and is shifted to get the reconstruction point. Named quantizers are quantizer ensembles, consisting of two quantizers shifted by T/2 with respect to each other. The shift depends on the watermark bit. Largest component of D matrix is quantized using either quantizer 1 or 2 depending on watermark bit to be embedded. The center of the quantizer is the quantized value.

Used method explained

A. Host image partitioning

Let the host grayscale image be
$f(i,j), 1 \le i \le N, 1 \le j \le N$
1. The host image is divided into **four subimages** of size (N/2 × N/2):
Top-left: $f_{tl(p,q)} = f(p,q), 1 \le p \le \frac{N}{2}, 1 \le q \le \frac{N}{2}.$
Top-right: $f_{tr}(p,q) = f\left(p, q + \frac{N}{2}\right), 1 \le p \le \frac{N}{2}, 1 \le q \le \frac{N}{2}.$
Bottom-left: $f_{bl}(p,q) = f\left(p + \frac{N}{2}, q\right), 1 \le p \le \frac{N}{2}, 1 \le q \le \frac{N}{2}.$
Bottom-right: $f_{br}(p,q) = f\left(p + \frac{N}{2}, q + \frac{N}{2}\right), 1 \le p \le \frac{N}{2}, 1 \le q \le \frac{N}{2}.$
2. The watermark w(u,v) is resized to M × M and converted to a binary map:
$b(u,v) = w(u,v) / 255 \in \{0,1\}.$
3. The watermark bits are **permuted** using a pseudorandom permutation π_K derived from secret key K:
$b^p(u,v) = \text{permutation}(b(u,v)).$

B. Watermark embedding in D matrix

The f_{tl} subimage is **divided into M × M blocks** of size b × b, with N/2 = bM.
For each block B_{rs}:
1. Compute **SVD**:
$B_{rs} = U_{rs}\Sigma_{rs}V_{rs}^T,$
$\Sigma_{rs} = \text{diag}(\sigma_{rs},1 \ge \sigma_{rs},2 \ge \ldots \ge \sigma_{rs},b).$
2. Collect the largest singular values: $D_{large}(r,s) = \sigma_{rs},$
3. Let:
$d_{min} = \min D_{large},$
$d_{max} = \max D_{large}$
4. Divide the interval $[d_{min}, d_{max}]$ into **bins of width T.**
For coefficient $D_{large}(r,s)$, determine bin k:
$k = \text{floor}\left(\frac{(D_{large(r,s)} - d_{min})}{T}\right),$
and compute:
$d_{low} = d_{min} + kT,$
$d_{high} = d_{low} + T,$
$m = \frac{(d_{low} + d_{high})}{2}.$
5. Modify D_{large} according to the watermark bit $b^p(r,s)$:
If $b^p(r,s) = 1$ (Range 1: lower half):
$D_{large}'(r,s) = \frac{(d_{low} + m)}{2}.$
If $b^p(r,s) = 0$ (Range 2: upper half):
$D_{large}'(r,s) = \frac{(m + d_{high})}{2}.$
6. Reconstruct each block:
Replace $\sigma_{rs},1$ with $D_{large}'(r,s),$
$B_{rs}' = U_{rs}\Sigma_{rs}'V_{rs}^T.$
Assemble all B_{rs}' to obtain f_{tl}^w.

C. Watermark embedding in U matrix

For each block C_{rs} of f_{br}:

1. Compute SVD:
$C_{rs} = U_{rs}\Sigma_{rs}V_{rs}^{T}$.

Let:
$u11 = U_{rs(1,1)}$,
$u21 = U_{rs}(2,1)$,
$a = |u11|$,
$b = |u21|$,
$m = \frac{a+b}{2}$.

2. Embed bit $b^{p}(r,s)$ using margin α:

If $b^{p} = 1$ enforce $a \geq b + \alpha$:
    If $(a - b < \alpha)$:
        $a' = m + \frac{\alpha}{2}$,
        $b' = m - \frac{\alpha}{2}$,
    else
            keep a, b unchanged.

If $b^{p} = 0$ enforce $b \geq a + \alpha$:
    If $(b - a < \alpha)$:
        $a' = m - \frac{\alpha}{2}$,
      $b' = m + \frac{\alpha}{2}$,
    else
            keep a, b unchanged.

Restore original signs of u11 and u21 and rebuild block:
    $C_{rs}{}' = U_{rs}\Sigma_{rs}V_{rs}^{T}$.

Reassemble to obtain f_{br}^{w}.

D. Construction of Final Watermarked Image

The final watermarked image is:
    $f^{w} = [\, f_{tl}^{w} \;\; f_{tr}\, f_{bl} \;\; f_{br}^{w}\,]$.

Only f_{tl} and f_{br} contain watermark data; f_{tr} and f_{bl} remain unchanged.

E. Watermark extraction from D matrix

1. Partition watermarked image into subimages and process f_{tl}^{w}.
2. Block SVD yields: $\sigma_{rs}, 1^{ex} \rightarrow D_{large}^{ex(r,s)}$.
3. Using stored (d_{min}, d_{max}, T), compute bin (d_{low}, d_{high}, m). (
4. Bit decision:
    If $D_{large}^{ex(r,s)} < m \rightarrow$ extracted bit $= 1$
    else
            extracted bit = 0.
5. Apply inverse permutation $\pi_{K}^{\{-1\}}$ to recover original watermark bits.

F. Watermark extraction from U matrix

1. For each block of f_{br}^{w}, compute SVD.

Let:
    $u11 = U_{rs\,(1,1)}$,
    $u21 = U_{rs(2,1)}$.

2. Bit decision rule:
    If $|u11| > |u21| \rightarrow$ extracted bit = 1,
    else

extracted bit = 0.

3. Apply inverse permutation to obtain original watermark.

G. Performance measures

Mean Squared Error (MSE): $\mathrm{MSE} = \left(\frac{1}{N^2}\right)\Sigma\,(f - f^w)^2$

Peak Signal-to-Noise Ratio (PSNR): $\mathrm{PSNR} = 10\log10\left(\frac{255^2}{MSE}\right)$.

Bit Error Rate (BER): $\mathrm{BER} = \frac{(\#\text{ of mismatched bits})}{total\ bits}$

Normalized Correlation (NC): $\mathrm{NC} = \frac{\Sigma(w * w_{ex})}{\mathrm{sqrt}(\Sigma w^2 \Sigma w_{ex}^2)}$

Commentary on the used method

What's different from the paper?

Our implementation is basically the same as the one in the referenced main paper, except from the method used in embedding the watermark into the U matrix of the bottom right sub image. Their main idea of implementation goes like this:

For each M × M block of U matrix, u_{11} (first row and column) and u_{21} (second row first column) are modified as follows:

$u_{diff} = |u_{11}| - |u_{21}|$

$If\ w(i,j) = 1\ \&\ u_{diff} > \alpha\ or\quad w(i,j) = 0\ \&\ u_{diff} < \alpha$

$u_{21} = -||u_{21}| - (u_{diff} - \alpha)/2\,|$

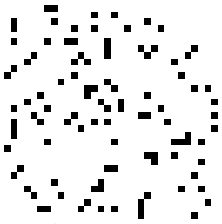
$u_{11} = -||u_{11}| + (u_{diff} - \alpha)/2\,|$

$If\ w(i,j) = 1\ \&\ u_{diff} < \alpha\ or\quad w(i,j) = 0\ \&\ u_{diff} > \alpha$

$u_{21} = -||u_{21}| - (u_{diff} + \alpha)/2\,|$

$u_{11} = -||u_{11}| + (u_{diff} + \alpha)/2\,|$

While extracting the watermark from U matrix we got something completely off track and it didn't change when choosing different parameters, so we decided to change it up.



extracted watermark from U on "Shelby", NC=0.866106

The watermark was just unrecognizable and the NC value was, appropriately, unsatisfying. Our version of that embedding goes like this:

$If\ w(i,j) = 1$

$If\ u_{diff} < \alpha$

$mean = (|u_{11}| + |u_{21}|)/2$

$|u_{11}| = mean + \frac{\alpha}{2}$

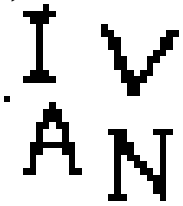
$|u_{21}| = mean - \frac{\alpha}{2}$

If not, they are staying the same

If $w(i,j) = 0$
If $u_{diff} > -\alpha$
$mean = (|u_{11}| + |u_{21}|)/2$
$|u_{11}| = mean + \frac{\alpha}{2}$
$|u_{21}| = mean - \frac{\alpha}{2}$
If not, they are staying the same

Return signs of u_{11}, u_{21} to original ones

To compare, for the same picture, we got this when we extracted the watermark from U. It was, compared to the previously shown image, a success.



extracted watermark from U on "Shelby", NC= 0.998875

Choosing parameters for model

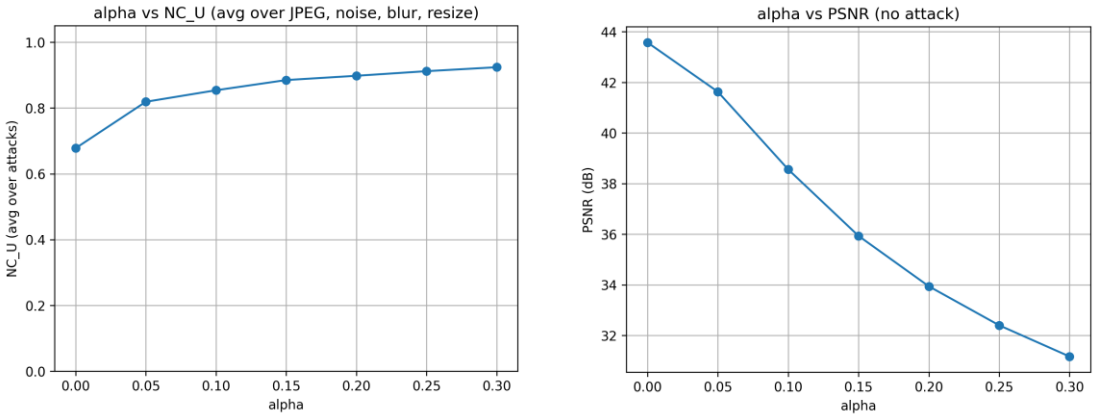
The crucial difference in our work of the one in paper, is how we choose the parameters for the method. In the paper they just choose "optimal values", as they said, without saying in which way it was optimal. We decided that we have to make up some measure to be certain to some extent it is a good choice.
To make the "right" decision while choosing the values for parameters, we did a certain iteration process. Firstly, we picked out a few attacks: JPEG_Q70, Gaussian noise 10, Blur and Resize by 0.5. Then, we applied those attack a few times, embedding the watermark as usual and extracting it. After that, we measured them with NC and BER and took the average values.
For possible values, for T we picked numbers in range of 10 to 100 by 10 step and for alpha in range of 0 to 0.30 by 0.05 step.
We ruled out immediately those who gave us PSNR lower then 40db (that was our measure of a "good enough" picture), so we were left with values 0, 0.05 and 0.10 for alpha.
To put it in prospective we will show you how the picture looked for alpha 0, 0.05, 0.10, 0.15 and 0.20 (for T=60), so that we show the visual difference in quality.

Alpha= 0

alpha=0.05, alpha=0.10- form left to right

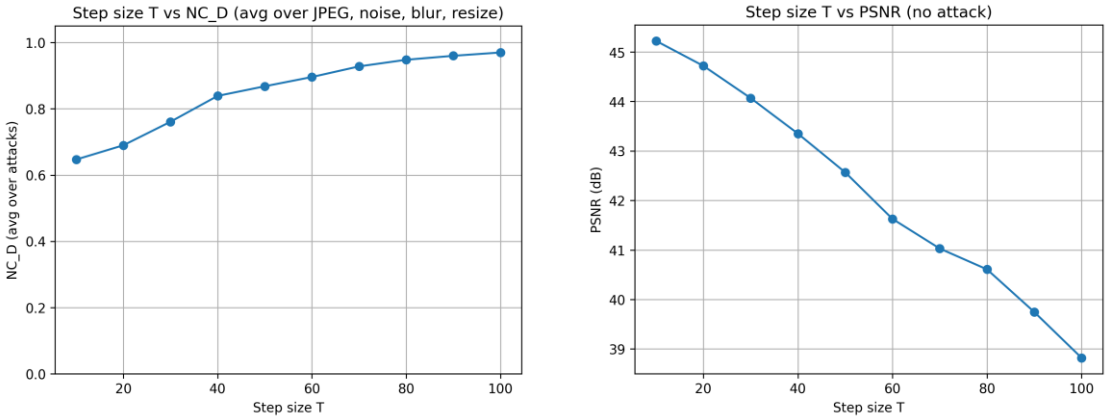
alpha=0.15, alpha=0.20- form left to right

As shown, the “lines” are very noticeable form alpha bigger then 0.10.
Below are also graphs that made us surer of our decision. The value 0 wasn’t nearly robust enough.



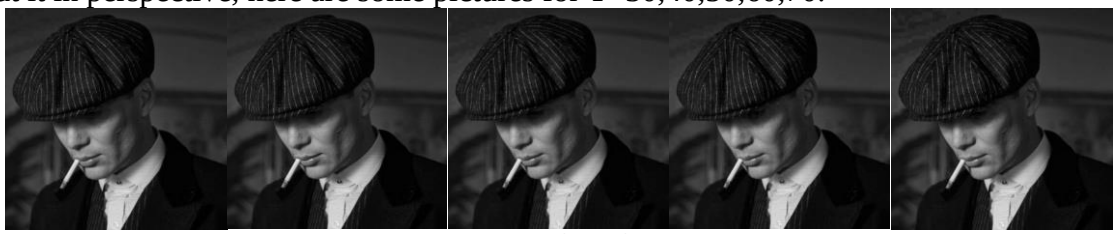
Graphs for "Shelby" picture – alpha with NC and PSNR

We still had to pick a value for T, after removing all the pairs that gave a PSNR under 40dB. After that we used the graphs that showed the connection between T, correlation and PSNR for fixed alpha 0.05. Paired with 0.05 were (in most cases, because it was a bit different every time) T=40 and T=60, values that pretty much agree with our graphs.



Graphs for Shelby picture – T with NC and PSNR

To put it in perspective, here are some pictures for T=30,40,50,60,70.



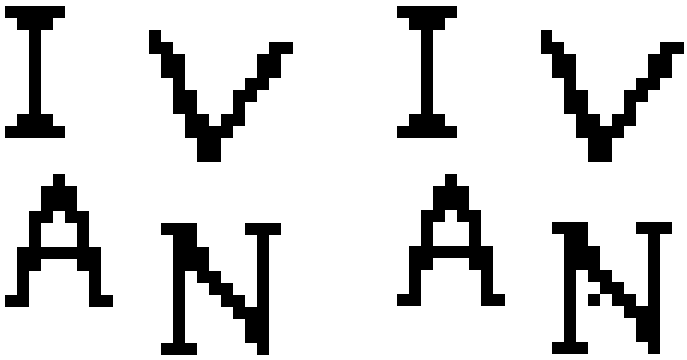
For T=30 to 70 from left to right

Choosing the right parameters is a balancing game. If you increase a parameter too much you get a better extraction, but lose on quality. In the end, among the left over candidates we choose the ones which produced pictures that "looked good" on the eye.
We did a similar iteration process for "Lena" image, so that we can compare images and results better to the paper and ended up choosing T=60 for "Shelby" and T=40 for "Lena".

Under every row of pictures from left to right: Attacked watermarked image, Extracted watermarks from D matrix, Extracted watermarks from U matrix.

ATTACKS (SHELBY)

No attack

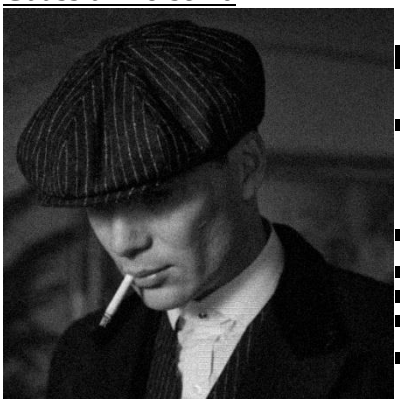


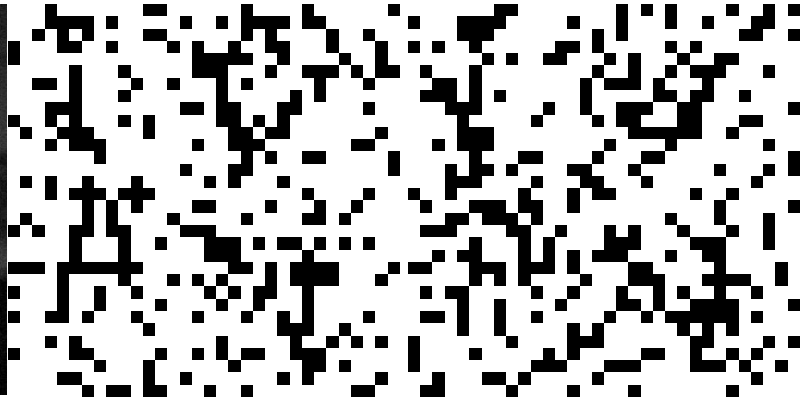
PSNR=41.645 dB (D: NC=1.000, BER=0.000) (U: NC=0.999, BER=0.001)

JPEG Q70

PSNR=38.594 dB (D: NC=1.000, BER=0.000) (U: NC=0.783, BER=0.336)

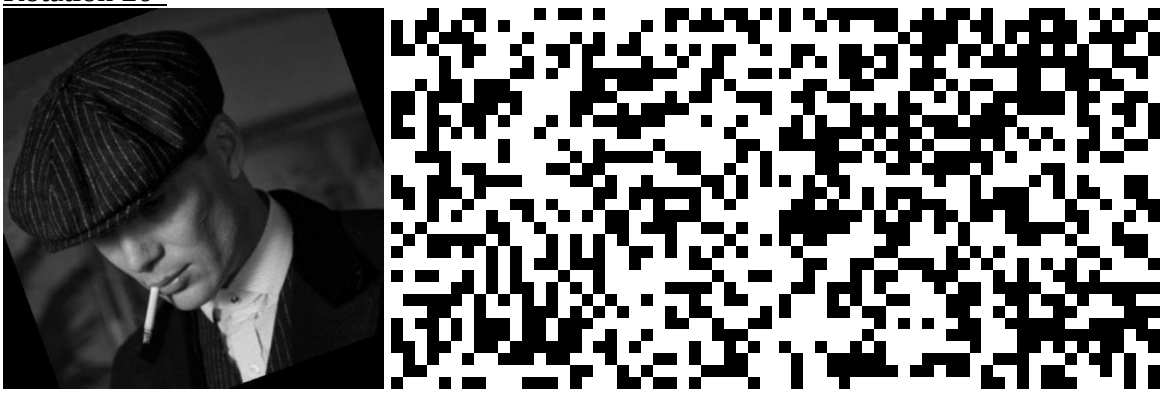
Gaussian noise 10





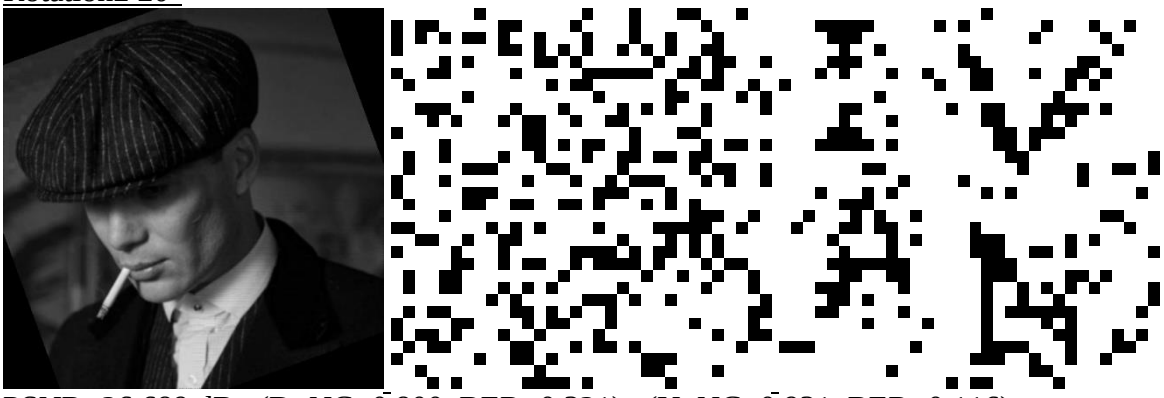
PSNR=28.286 dB (D: NC=0.876, BER=0.203) (U: NC=0.852, BER=0.240)

Rotation 20°



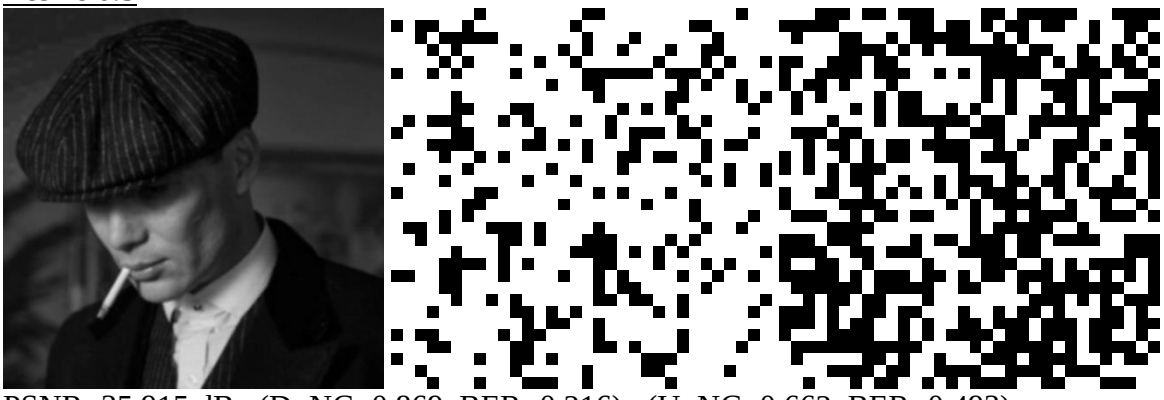
PSNR=14.094 dB (D: NC=0.706, BER=0.446) (U: NC=0.667, BER=0.490)

Rotation2 20°



PSNR=26.689 dB (D: NC=0.800, BER=0.321) (U: NC=0.931, BER=0.116)

Resize 0.5

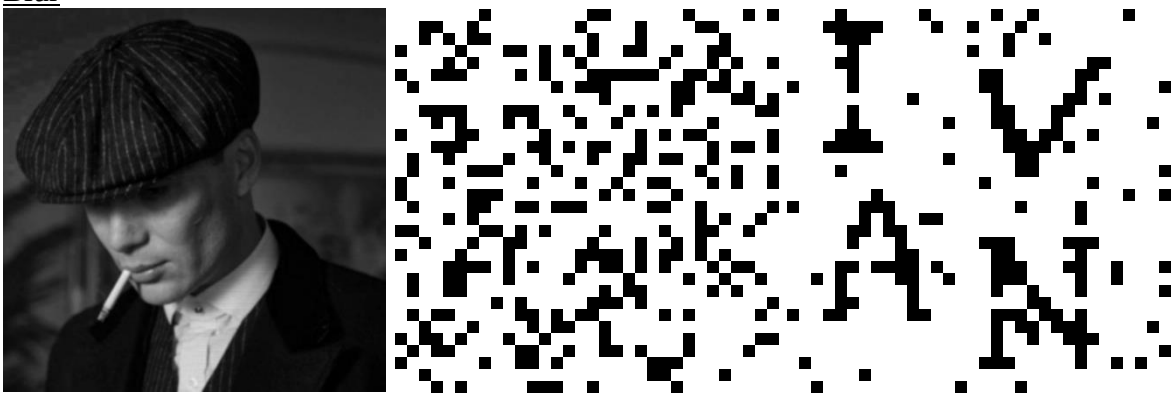


PSNR=35.915 dB (D: NC=0.869, BER=0.216) (U: NC=0.662, BER=0.493)

Median 3x3

PSNR=38.461 dB (D: NC=0.833, BER=0.269) (U: NC=0.641, BER=0.518)

Blur

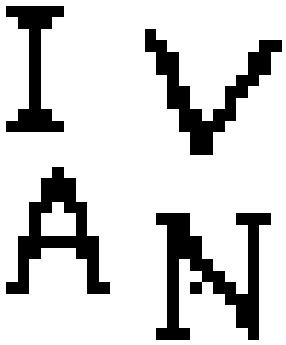


PSNR=37.345 dB (D: NC=0.833, BER=0.271) (U: NC=0.959, BER=0.069)

Salt and Pepper

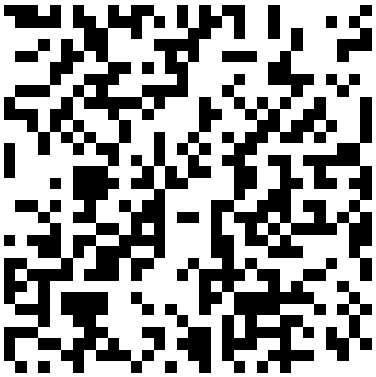
PSNR=24.008 dB (D: NC=0.922, BER=0.131) (U: NC=0.976, BER=0.04

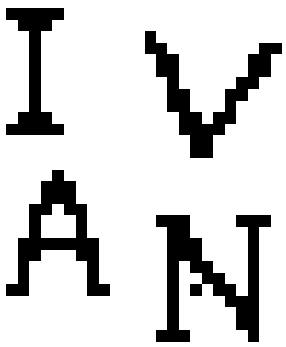
Cropping 25%



PSNR=22.604 dB (D: NC=0.932, BER=0.132) (U: NC=0.999, BER=0.001)

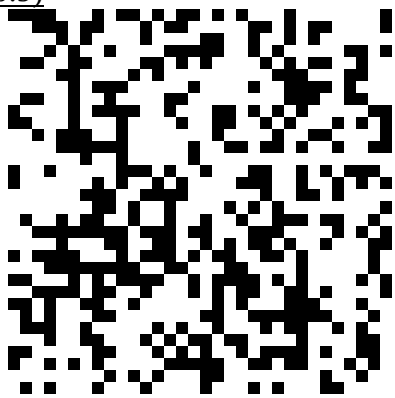
Brightness 20%

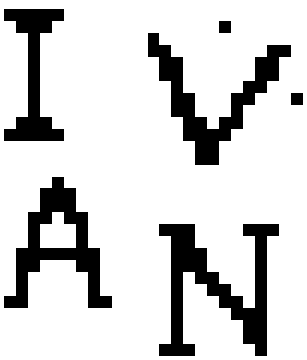




PSNR=27.124 dB (D: NC=0.734, BER=0.407) (U: NC=0.999, BER=0.001)

Gamma Correction (gamma=0.9)



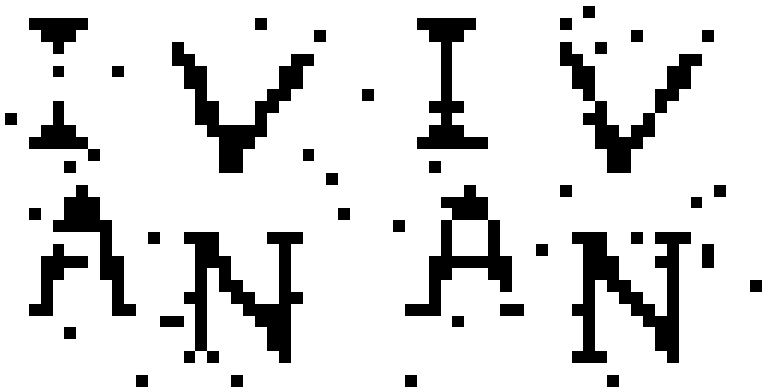


PSNR=32.033 dB (D: NC=0.788, BER=0.331) (U: NC=0.999, BER=0.002)

Row Column Blanking

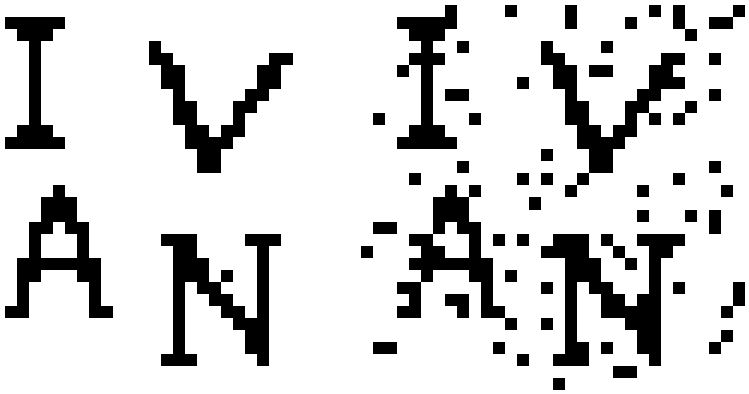
PSNR=22.735 dB (D: NC=0.706, BER=0.443) (U: NC=0.899, BER=0.166)

Row Column Copying



PSNR=39.677 dB (D: NC=0.983, BER=0.029) (U: NC=0.983, BER=0.029)

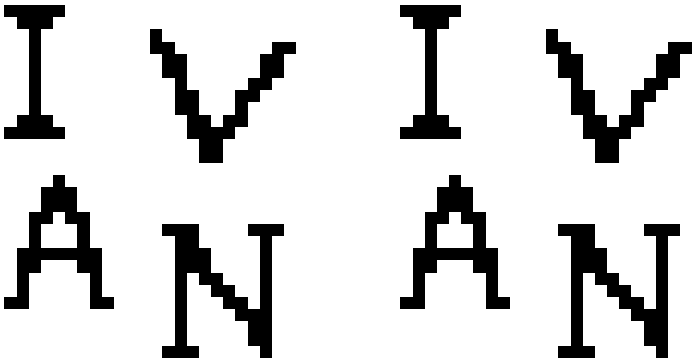
Bit Plane Removal (LSB)



PSNR=40.650 dB (D: NC=0.999, BER=0.001) (U: NC=0.949, BER=0.087)

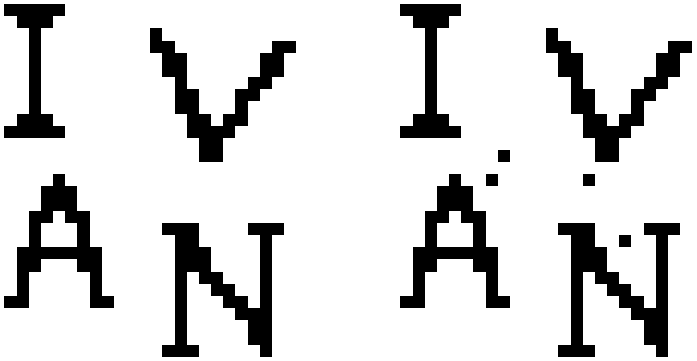
ATTACKS (LENA)

No attack



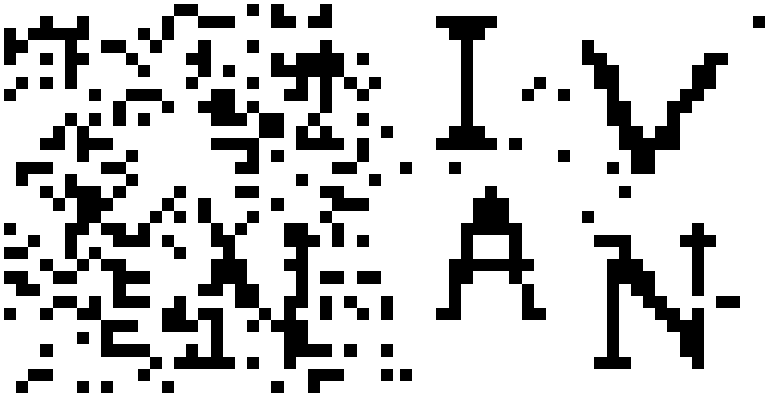
PSNR=39.270 dB (D: NC=1.000, BER=0.000) (U: NC=1.000, BER=0.000)

JPEG Q70



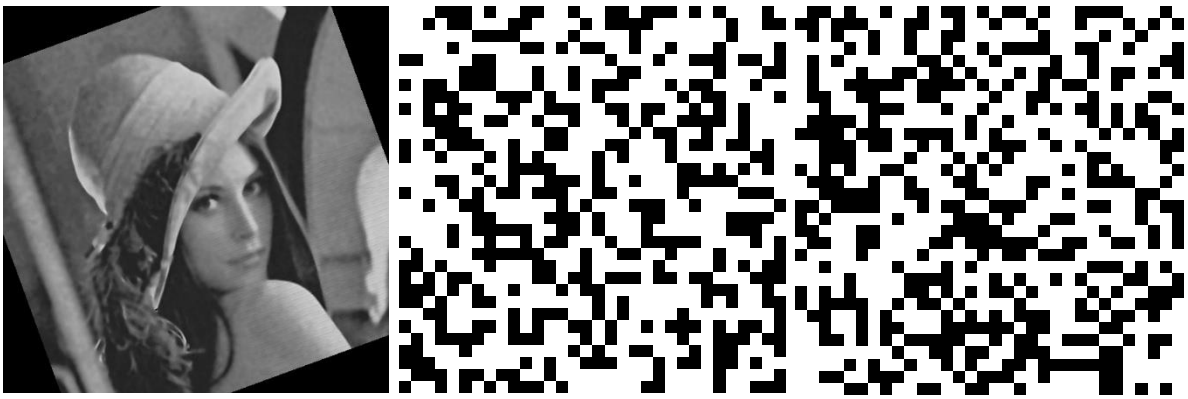
PSNR=38.481 dB (D: NC=1.000, BER=0.000) (U: NC=0.998, BER=0.004)

Gaussian noise 10



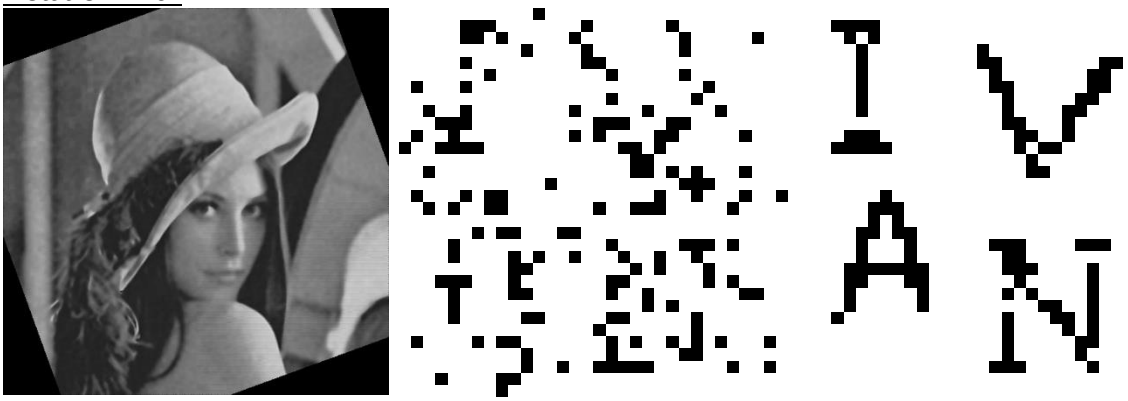
PSNR=27.774 dB (D: NC=0.858, BER=0.231) (U: NC=0.988, BER=0.021)

Rotation 20°



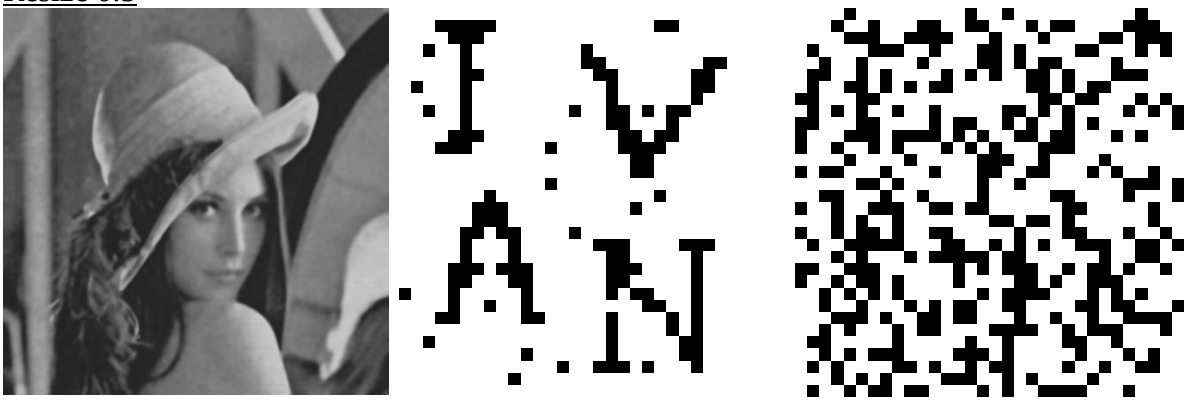
PSNR=12.215 dB (D: NC=0.728, BER=0.418) (U: NC=0.710, BER=0.441)

Rotation2 20°



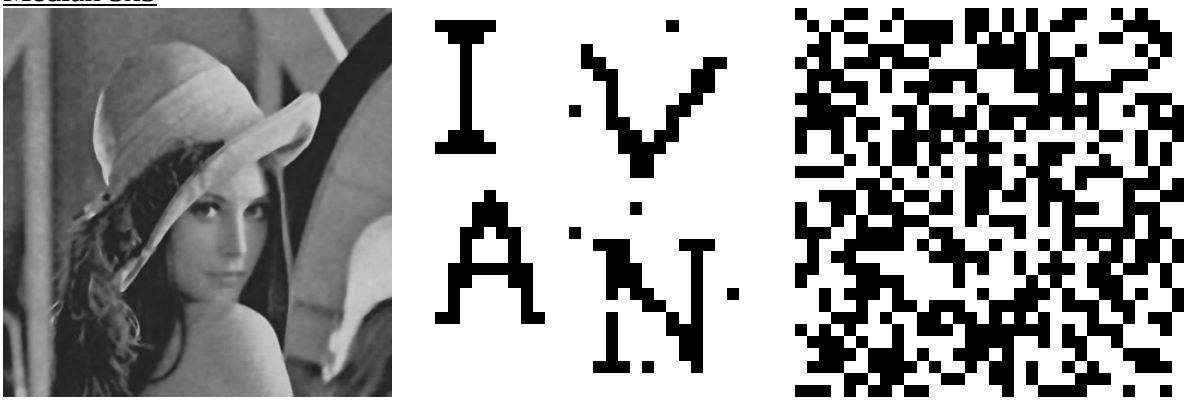
PSNR=16.800 dB (D: NC=0.917, BER=0.143) (U: NC=0.989, BER=0.019)

Resize 0.5



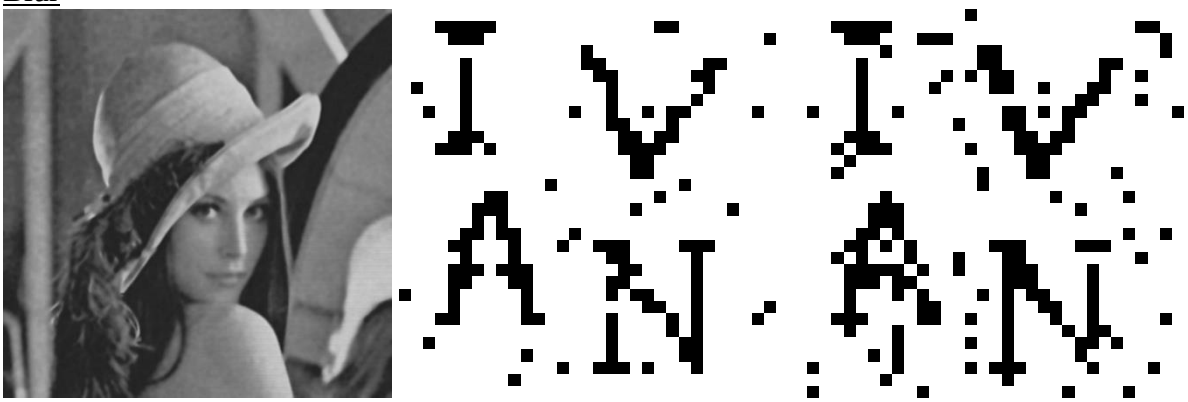
PSNR=36.944 dB (D: NC=0.984, BER=0.028) (U: NC=0.798, BER=0.318)

Median 3x3



PSNR=40.929 dB (D: NC=0.994, BER=0.010) (U: NC=0.674, BER=0.482)

Blur

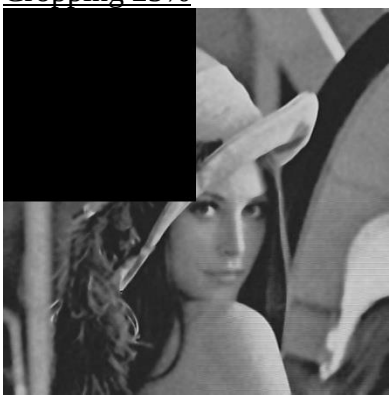


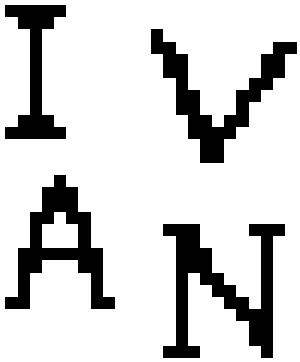
PSNR=38.513 dB (D: NC=0.979, BER=0.037) (U: NC=0.957, BER=0.073)

Salt and Pepper

SNR=25.179 dB (D: NC=0.818, BER=0.290) (U: NC=0.982, BER=0.031)

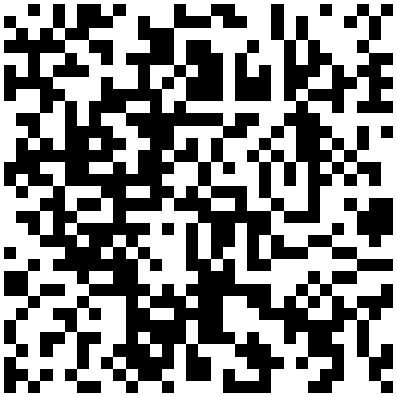
Cropping 25%

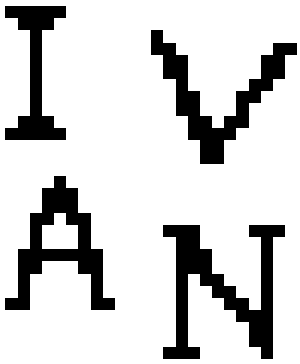




PSNR=13.385 dB (D: NC=0.932, BER=0.132) (U: NC=1.000, BER=0.000)

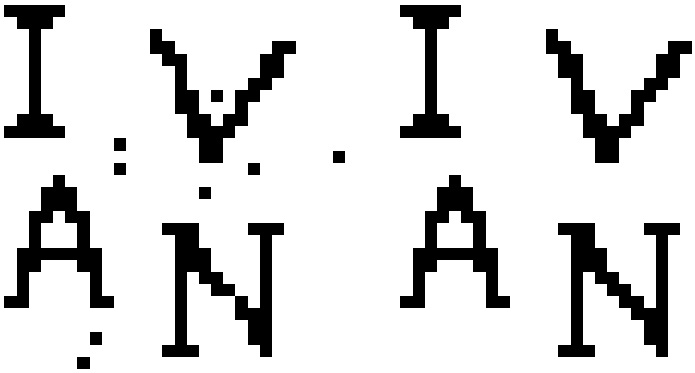
Brightness 20%





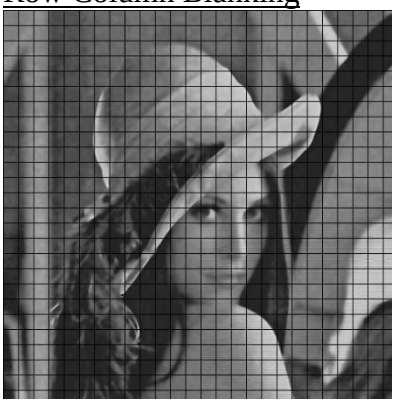
PSNR=21.291 dB (D: NC=0.691, BER=0.459) (U: NC=1.000, BER=0.000)

Gamma Correction (gamma=0.9)



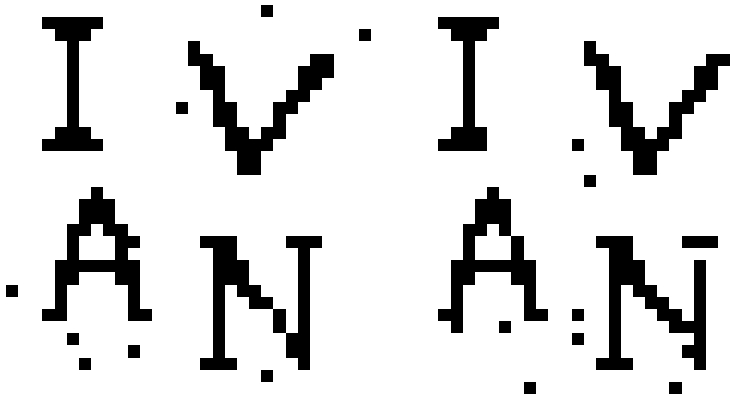
PSNR=29.955 dB (D: NC=0.994, BER=0.010) (U: NC=1.000, BER=0.000)

Row Column Blanking



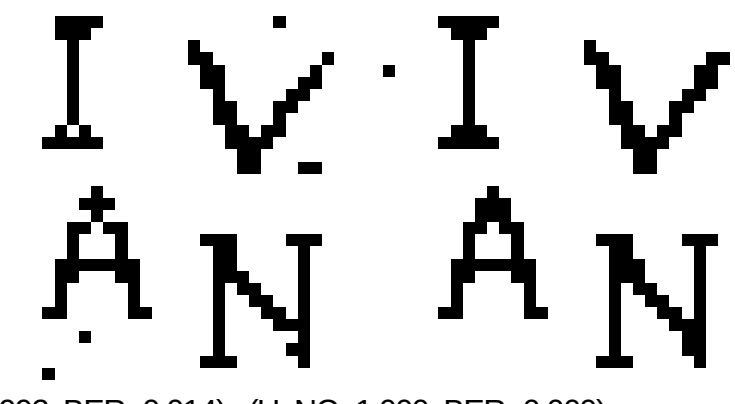
PSNR=17.024 dB (D: NC=0.843, BER=0.253) (U: NC=0.900, BER=0.165)

Row Column Copying



PSNR=37.775 dB (D: NC=0.993, BER=0.012) (U: NC=0.993, BER=0.012)

Bit Plane Removal (LSB)



PSNR=38.709 dB (D: NC=0.992, BER=0.014) (U: NC=1.000, BER=0.000)

Conclusion

In our work, we followed the method from the referenced paper, a watermarking scheme based on SVD. The watermark was embedded in the upper left and bottom right sub image, specifically in the D and U matrices gotten from the decomposition respectively. Our embedding was a bit different from the one in the referenced paper and shown to have slightly better resonance to some of attacks when compared to the known “Lena” image.

The quality of the watermarked image, with chosen parameters, was good in terms of the metric PSNR=41.645 dB. The acknowledged algorithm has shown strength when attacked with JPEG Q70 compression, rotation, scaling, cropping, median filtering, low pass filtering, row-column copying, row-column blanking, bit plane removal, salt and pepper noise and gamma correction, scoring lower for median and resize attack on “Shelby” image, but impressively good on “Lena” image.

All this indicates that an embedded watermark will still be recoverable even after the common image processing operations on the watermarked image and hence highly suitable for the copyright protection. Potential improvements could be made in both the embedding methods and using a large dataset of pictures the further validate the methods used.

References

Robust Image Watermarking Scheme using Singular Value Decomposition, B.Chandra Mohan, S. Srinivas Kumar

ChatGPT. OpenAI, response to a prompt, Dec. 2025.