COS791 Assignment 1 Report
Image Analysis and Understanding

Ruan Carlinsky
u20416823

***Abstract*—This paper applies metaheuristic optimisation to grayscale image enhancement by evolving variable-length (minimum length four) pipelines drawn from gamma correction, Gaussian blur, unsharp masking, histogram equalisation, and contrast stretching. Two optimisers are considered: a Genetic Algorithm (GA) and Variable Neighbourhood Search (VNS). Optimisation is conducted on 10 distorted/ground-truth training pairs using the Structural Similarity Index (SSIM) as the objective, with fitness $F = 1 - \overline{\text{SSIM}}$, and evaluation is performed on 5 test pairs. Furthermore we report, for each test image, the MSE, PSNR, and SSIM achieved by GA and VNS, and we record the best individual (pipeline) and seed value for each method.**

***Index Terms*—Image enhancement, metaheuristic optimisation, enhancement techniques, genetic algorithm (GA), variable neighbourhood search (VNS), grayscale images**

I. Introduction

Image enhancement is a fundamental task in the field of image processing and computer vision. Its primary goal is to improve the visual quality of an image or to transform it into a form that is more suitable for analysis and interpretation. Enhanced images can assist both human observers—by improving clarity, contrast, and detail—and machine-based systems, which rely on high-quality inputs for tasks such as object detection, recognition, and classification. Because digital images often suffer from degradation due to noise, poor lighting conditions, or limitations in acquisition devices, enhancement techniques play a vital role in restoring or improving image quality.

Traditional image enhancement techniques, such as histogram equalization, filtering, and contrast adjustment, have been widely studied and applied with notable success. However, many of these approaches rely on fixed transformations and may fail to generalize across diverse image conditions. As a result, optimization-based techniques have gained increasing attention. These methods frame image enhancement as a search for optimal parameter values or transformations, allowing for adaptive solutions tailored to the characteristics of specific images.

In this study, we investigate two metaheuristic optimization approaches—Genetic Algorithms (GA) and Variable Neighborhood Search (VNS)—for the task of image enhancement. Both methods offer powerful global search capabilities and can explore large solution spaces efficiently. The GA applies evolutionary principles such as selection, crossover, and mutation to evolve high-quality enhancement parameters, while VNS systematically explores multiple neighborhood structures to escape local optima and refine solutions.

The objective of this work is to evaluate the performance of GA and VNS in enhancing image quality, using quantitative metrics such as Mean Squared Error (MSE), Peak Signal-to-Noise Ratio (PSNR), and Structural Similarity Index (SSIM). By comparing the two approaches, we aim to highlight their relative strengths, weaknesses, and applicability to real-world image enhancement problems.

The remainder of this paper is structured as follows: Section II provides the necessary background on image enhancement and optimization techniques. Section III describes the methodologies and experimental setup in detail. Section IV presents and discusses the results obtained. Finally, Section V concludes the study and outlines potential directions for future work.

II. Background

A. Image enhancement techniques

Classical global methods include histogram equalization (HE [1] and its brightness-preserving variants (e.g., BBHE, DSIHE, MMBEBHE), which redistribute intensities but can over-enhance or shift brightness on some inputs. Local/edge-aware approaches (e.g., Retinex, weighted thresholded HE) target illumination while limiting noise and halo artifacts. Adaptive gamma [2] correction with weighting distribution (AGCWD) blends gamma mapping with a histogram-derived weight to improve contrast with better brightness control, though it may under-enhance scenes with few bright pixels. Enhancement is often paired with sharpening (e.g., unsharp masking) [3] in a pipeline to boost perceived detail after contrast adjustment.

B. Image enhancement and optimization

Because enhancement involves competing goals—contrast increase, detail revelation, brightness preservation, artifact suppression—the problem is naturally cast as (often multi-objective) optimization over operator choices and parameters. Surveys of nature-inspired optimization [4] in enhancement report extensive use of GA/PSO/DE and related metaheuristics to tune tone curves, histogram thresholds, and filter/sharpening parameters using objectives built from entropy, edge measures, AMBE, PSNR/SSIM, and color-quality metrics. Multi-objective formulations explicitly balance such criteria rather than relying on a single index. [5]

1) Genetic Algorithm for Image enhancement: Genetic Algorithms (GAs) [6] search the parameter space of an enhancement pipeline—e.g., histogram partition thresholds, tone-curve coefficients, or sharpening weights—guided by a fitness that encodes desired perceptual/quantitative properties. In a representative GA-AHE design (GAAHE), thresholds for exposure-based sub-histograms are optimized by GA while each sub-histogram []is equalized via its own mapping; the multi-objective fitness aggregates edge content, contrast, PSNR, entropy, and energy. Fitness components are often computed from GLCM-based texture statistics and information measures, reflecting a balance between detail revelation and structural regularity. Empirically, GA-tuned models [7] have outperformed fixed HE/GLG baselines and other classical methods across local/global contrast tasks and specialized domains (e.g., retinal imagery).

2) Variable Neighbourhood Search for Image enhancement: Variable Neighbourhood Search (VNS) is a metaheuristic that systematically changes the neighbourhood [8] structure of the search to escape local optima: it alternates diversification ("shaking") with local search and follows prescribed neighbourhood-change rules (e.g., sequential or probabilistic neighbourhood change) until convergence. In general VNS practice, exploring multiple neighbourhoods (i.e., different perturbation scales or operator subsets) improves the chance of finding high-quality solutions in rugged spaces. For image enhancement, VNS can be instantiated by defining neighbourhoods over: (i) histogram strategy choices (global/local/bi-histogram, clipping), (ii) tone-curve parameters (gamma, sigmoid, piecewise linear knots), and (iii) sharpening/smoothing controls (unsharp radius/amount). A local search evaluates candidates via composite objectives (e.g., minimize AMBE and LOE while maximizing entropy/EME/SSIM), and neighbourhood [9] change governs diversification vs. intensification when improvements stall. (Objectives and indices:) Although VNS is best known in combinatorial optimization, its mechanics align well with enhancement parameter search where multiple competing goals create many local optima.

III. Proposed Approach

We formulate enhancement as a search over *variable-length* pipelines composed from the five operators: Gamma correction, Gaussian blur, Unsharp masking, Histogram equalisation, and Percentile contrast stretching. Each candidate pipeline $P = [s_1, \ldots, s_L]$ is an ordered list of steps ($L \in [4, 6]$) applied sequentially to a grayscale input; outputs are clipped to $[0, 1]$. Training uses the provided paired distorted/ground-truth images; once the search converges, the best pipeline per algorithm (GA or VNS) is applied to the test set for quantitative comparison using MSE, PSNR, and SSIM.

Operator Set and Bounds

We expose the following parameters to optimisation:

- **Gamma**: exponent $g \in [0.40,\ 1.80]$.
- **Gaussian blur**: $\sigma \in [0.30,\ 3.00]$, kernel size $k \in \{3, 5, 7, 9\}$.
- **Unsharp masking**: radius $\in [0.50,\ 2.50]$, amount $\in [0.10,\ 1.50]$, threshold $\in [0,\ 10]$.
- **Histogram equalisation**: no parameters.
- **Percentile contrast stretch**: $p_{\text{low}} \in [0,\ 10]$, $p_{\text{high}} \in [90,\ 99]$ with constraint $p_{\text{high}} \geq p_{\text{low}} + 5$.

Objective and Metrics

For a candidate P, we compute mean metrics over all training pairs:

$$\overline{\text{MSE}}, \quad \overline{\text{PSNR}}, \quad \overline{\text{SSIM}}.$$

Two objective modes are supported (both are minimised):

$$F_{\text{SSIM}} = 1 - \overline{\text{SSIM}}, \tag{1}$$

$$F_{\text{w}} = \alpha\, \overline{\text{MSE}} - \beta\, \overline{\text{PSNR}} + \gamma\, (1 - \overline{\text{SSIM}}), \tag{2}$$

with default weights $(\alpha, \beta, \gamma) = (1.0,\ 0.1,\ 0.5)$. To avoid redundant computation during search, we memoise evaluations via a stable hash of (pipeline, dataset IDs, objective settings). The best pipeline and its seed are persisted for test-time evaluation and reporting.

A. Genetic Algorithm for Image Enhancement Pipeline

Representation and Initialisation. Each GA individual encodes a pipeline of length $L \in [4, 6]$. The initial population is sampled uniformly over the operator set and parameter ranges; the percentile-stretch constraint $p_{\text{high}} \geq p_{\text{low}} + 5$ is enforced at sampling and mutation time.

Selection, Crossover, Mutation. We use tournament selection (size $k{=}3$). One-point crossover is performed at random cut positions to produce two children while keeping L within $[4, 6]$. Mutation consists of:

1) *Parameter jitter:* reflected Gaussian noise on continuous parameters, and a discrete hop for k (Gaussian kernel size).
2) *Structural edits:* swap two neighbouring steps or replace the operator at a position (resampling valid parameters).
3) *Length edits:* insert or delete a step, with hard limits $L \in [4, 6]$.

Elitism copies the top $e{=}2$ individuals each generation. Default hyperparameters: population $= 40$, generations $= 60$, crossover $p_c{=}0.90$, mutation $p_m{=}0.25$.

Evaluation, Logging, Persistence, Reproducibility. Fitness is computed on the training set via the chosen objective. We log per-generation best/mean fitness and best per-metric values; the best pipeline is serialised to `results/ga_best.json`. All pseudorandomness (NumPy and Python) is driven by a fixed seed for reproducibility.

B. VNS for Image Enhancement Pipeline

High-Level Scheme. Variable Neighbourhood Search (VNS) alternates a *shaking* step in neighbourhood N_k with a short *first-improvement* local search, cycling $k = 1, \ldots, K$ and resetting to $k{=}1$ upon improvement. Configuration: $K{=}4$, total iterations $= 300$, local-search trials $= 10$, jitter strength $= 0.12$, $L \in [4, 6]$.

Neighbourhoods.

- N_1: jitter *one* parameter of a random step (bounded with reflection; discrete hop for k).
- N_2: jitter *all* parameters of one step.
- N_3: structural move (swap adjacent steps or replace the operator at a position with freshly sampled valid parameters).
- N_4: change length by inserting or deleting a step (respecting L limits and the stretch constraint).

Local Search, Acceptance, Warm Start. From the shaken candidate, local search attempts reduced-strength parameter jitters for up to 10 trials and accepts the *first* improving neighbour under the objective. On improvement, the current solution is updated and neighbourhood index resets to N_1; otherwise we advance to N_{k+1} (cycling after K). The best-so-far solution and its metrics are tracked and logged; at the end, the best pipeline is saved to `results/vns_best.json`. When available, VNS warm-starts from the GA best (`ga_best.json`); otherwise it starts from a random pipeline. A fixed seed is used for reproducibility.

Test-Time Evaluation and Reporting

The saved GA/VNS best pipelines are applied to the test set to (i) write enhanced outputs per image and (ii) export a CSV of per-image metrics (MSE/PSNR/SSIM) together with per-algorithm means, enabling the direct GA–VNS comparison.

IV. Experimental Setup

A. Data and Pairing

We use the assignment's grayscale image dataset with 10 training pairs and 5 test pairs (distorted/ground truth). Files follow strict, case-insensitive naming schemes and are loaded from `data/train` and `data/test`:

- Train: distorted `imgN.ext`, ground truth `imgN_gt.ext`
- Test: distorted `img_tN.ext`, ground truth `img_tN_gt.ext`

If multiple extensions exist for the same ID, the preferred order is `.png`, `.jpg`, `.jpeg`, `.bmp`, `.tif`, `.tiff`. Only files directly in each folder are considered (no recursion). A strict ID intersection ensures correctly aligned pairs; images are read as grayscale $\in [0, 1]$ and a shape check prevents misalignment.

B. Preprocessing and Operator Set

All inputs are coerced to grayscale float32 in $[0, 1]$; intermediate results are clipped after each step. The enhancement pipeline draws from the five techniques required by the spec: Gamma, Gaussian blur, Unsharp masking, Histogram equalisation, and Percentile contrast stretching. Parameter bounds (enforced throughout sampling, mutation, shaking, and local search) are:

- **Gamma:** $g \in [0.40, 1.80]$ (applies x^g; small epsilon avoids 0^g edge cases).
- **Gaussian:** $\sigma \in [0.30, 3.00]$, kernel $k \in \{3, 5, 7, 9\}$ (odd; derived from σ if unspecified).
- **Unsharp:** radius $[0.50, 2.50]$, amount $[0.10, 1.50]$, threshold $[0, 10]$ with threshold applied on a $0 \ldots 1$ mask via division by 255.
- **Histogram equalisation:** no parameters (global HE).
- **Percentile contrast stretch:** $p_{\text{low}} \in [0, 10]$, $p_{\text{high}} \in [90, 99]$ with the hard constraint $p_{\text{high}} \geq p_{\text{low}} + 5$.

C. Pipeline Representation and Sampling

A candidate is a variable-length pipeline $P = [s_1, \ldots, s_L]$ with $L \in [4, 6]$. Each step stores an operator name and its parameters; pipelines are applied sequentially and clipped to $[0, 1]$. Random initial pipelines are sampled uniformly over operators and within bounds, while honouring the contrast-stretch constraint. Pipelines are serialised/deserialised to/from JSON for persistence and evaluation.

D. Objective and Metrics

For a pipeline P, we compute the mean metrics over the training pairs: $\overline{\text{MSE}}$, $\overline{\text{PSNR}}$, $\overline{\text{SSIM}}$. Two objective modes are supported (both minimised):

$$F_{\text{SSIM}} = 1 - \overline{\text{SSIM}}, \qquad F_{\text{w}} = \alpha\,\overline{\text{MSE}} - \beta\,\overline{\text{PSNR}} + \gamma\,(1 - \overline{\text{SSIM}}),$$

with default $(\alpha, \beta, \gamma) = (1.0, 0.1, 0.5)$. We guard against $\text{PSNR} = \infty$ (zero MSE) with a cap for stability. The metric implementations use standard definitions with `data_range` $= 1$ for SSIM. A memoised cache keyed by a stable hash of (pipeline, dataset IDs, objective mode/weights) avoids re-evaluating the same candidate.

E. Genetic Algorithm Configuration

We use a *variable-length* GA whose individuals encode pipelines with $L \in [4, 6]$.

- **Initialisation:** Uniform over operators and bounds, enforcing $p_{\text{high}} \geq p_{\text{low}} + 5$.
- **Selection:** Tournament, $k = 3$.
- **Crossover:** One-point ($p_c = 0.90$), with length repair to keep $L \in [4, 6]$.
- **Mutation:** (i) Parameter jitter via reflected Gaussian noise for continuous params and discrete hops for k; (ii) structural edits (swap two steps or replace an operator with resampled params); (iii) length edits (insert/delete) while respecting bounds and constraints.
- **Elitism:** top $e = 2$ copies per generation.
- **Hyperparameters (real run):** population $= 40$, generations $= 60$; objective=`‘‘ssim’’` by default (weighted mode available); seed $= 1$ for reproducibility.
- **Logging/Persistence:** Per-generation CSV logs (`results/ga_log.csv`) include best/mean fitness and best SSIM/PSNR/MSE; best pipeline is saved to `results/ga_best.json`.

F. VNS Configuration

We adopt a 4-neighbourhood VNS with shaking and first-improvement local search.

- **Initial solution:** GA best from `ga_best.json` if present; otherwise a random pipeline.
- **Neighbourhoods:** N_1 jitter one parameter; N_2 jitter all parameters of one step; N_3 structural move (swap adjacent steps or replace an operator); N_4 insert or delete a step. All moves enforce operator bounds and $p_{\text{high}} \geq p_{\text{low}} + 5$.
- **Local search:** Up to 10 reduced-strength jitters; accept the first improving neighbour.
- **Search control:** If improved, reset $k \leftarrow 1$; else advance to $k{+}1$ (cycle after $K = 4$).
- **Hyperparameters (real run):** total iterations $=$ 300, $K = 4$, jitter strength $=$ 0.12; objective=``ssim'' (weighted mode available); seed $=$ 1.
- **Logging/Persistence:** CSV logs (`results/vns_log.csv`) track iter, k, best fitness and metrics; best pipeline saved to `results/vns_best.json`.

G. Evaluation Protocol

During search, GA/VNS fitness uses only the training set. After convergence, we load the saved best pipelines and evaluate on the 5 test pairs:

1) Apply the best pipeline for each algorithm to every test image and save outputs to `outputs/test/ga` and `outputs/test/vns`.
2) Compute per-image MSE/PSNR/SSIM and write `results/test_metrics.csv`.
3) Report per-algorithm mean MSE/PSNR/SSIM for the test set (as required by the spec), and include the objective mode, the best individual (pipeline string/JSON), and the seed value in the report tables.

H. Reproducibility and Environment

All randomness is controlled via a fixed seed that sets NumPy and Python's RNGs; we also use stable hashing for memoisation keys. The code relies on Python (NumPy, OpenCV, scikit-image; `pandas` for test summarisation; optional `tqdm` for progress).

V. Results

A. Objective, Best Individuals, and Seed

Both GA and VNS optimised the SSIM-centric objective $F_{\text{SSIM}} = 1 - \overline{\text{SSIM}}$. Table I lists the best pipelines (one step per line) discovered by each method, together with the seed used (common to both runs).

B. Training Behaviour (GA vs VNS)

The GA (population 40, 60 generations) improved best fitness from F=0.2794 (gen 0) to F=0.2392 (gen 55), corresponding to SSIM=0.7608, PSNR=19.79 dB, MSE=0.02644. The VNS was warm-started from the GA best and further reduced the objective to F=0.2286 at the end of 300 iterations, i.e., SSIM=0.7714, PSNR=20.51 dB, MSE=0.02579. Overall, VNS gained $+0.0106$ SSIM, $+0.72$ dB PSNR, and lowered MSE by $6.5{\times}10^{-4}$ over the GA best.

TABLE I
OBJECTIVE, BEST PIPELINE, AND SEED FOR GA/VNS.

Algorithm	Objective	Best individual (one step per line)
GA	F_{SSIM}	unsharp(radius=0.808, amount=1.239, thresh=2.507) unsharp(radius=0.868, amount=0.389, thresh=1.305) cstretch(p_low=6.724, p_high=98.772) cstretch(p_low=5.948, p_high=98.827) gauss(sigma=0.978, ksize=7) gamma(g=1.156)
VNS	F_{SSIM}	unsharp(radius=0.755, amount=1.350, thresh=0.162) unsharp(radius=0.628, amount=0.921, thresh=0.152) cstretch(p_low=7.606, p_high=98.994) gamma(g=0.683) gauss(sigma=1.092, ksize=9) gamma(g=1.565)
Seed used for both runs: **1**		

TABLE II
TEST-SET METRICS PER IMAGE (HIGHER PSNR/SSIM AND LOWER MSE ARE BETTER).

Image	MSE		PSNR		SSIM	
	GA	VNS	GA	VNS	GA	VNS
img_t1	0.049850	0.043332	13.02	13.63	0.748830	0.749105
img_t2	0.025981	0.029470	15.85	15.31	0.242010	0.209324
img_t3	0.001472	0.001068	28.32	29.71	0.880660	0.910865
img_t4	0.072443	0.071568	11.40	11.45	0.674207	0.668720
img_t5	0.001696	0.001219	27.70	29.14	0.923790	0.953194
Mean	0.030288	0.029331	19.26	19.85	0.693900	0.698242

C. Test-Set Performance

We report per-image MSE (lower is better), PSNR (higher is better), and SSIM (higher is better) on the 5 test pairs. Means across the test set appear in the last row.

D. Analysis

On average, VNS outperforms GA across all three metrics on the test set: ΔMSE$=-9.6{\times}10^{-4}$, ΔPSNR$= +\,0.59$ dB, ΔSSIM$= +\,0.0043$. Per-image SSIM wins: VNS leads on *img_t1*, *img_t3*, and *img_t5*; GA is slightly better on *img_t2* and *img_t4*. These results corroborate the training behaviour: VNS's neighbourhood changes and local search provide additional improvements over the GA-discovered baseline pipeline.

Aggregate performance. On the 5-image test set, VNS improves the mean across all metrics relative to GA: $\Delta\overline{\text{MSE}} = -9.57{\times}10^{-4}$ (lower is better), $\Delta\overline{\text{PSNR}} = +0.588$ dB, $\Delta\overline{\text{SSIM}} = +0.0043$. VNS wins on 4/5 images for MSE and PSNR, and on 3/5 for SSIM (see Table II).

Per-image effects and variance. Per-image gains are uneven, reflecting dataset heterogeneity:

- *Large gains (img_t3, img_t5).* VNS reduces MSE by 27.4% and 28.2%, respectively, and raises PSNR by $+1.39$ dB and $+1.44$ dB, with SSIM increases of $+0.030$ and $+0.029$. These images likely benefit from VNS's joint tone and structure adjustments (two-stage gamma with strong unsharp and a larger blur kernel).
- *Moderate gains (img_t1, img_t4).* VNS yields small but consistent improvements in MSE (-13.1% and -1.2%) and PSNR ($+0.61$ dB and $+0.05$ dB). SSIM is essentially unchanged for img_t1 and slightly lower (-0.0055) for img_t4, indicating minor local-structure trade-offs.
- *Regression (img_t2).* VNS underperforms GA (MSE $+13.4\%$, PSNR -0.55 dB, SSIM -0.0327). This suggests over-smoothing or an overly aggressive tone mapping on an image where GA's pipeline preserved informative local contrast better.

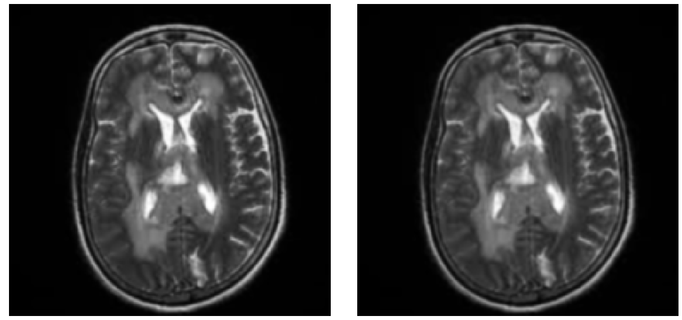
Dispersion also shifts: the standard deviation across images for PSNR increases from 8.15 dB (GA) to 8.85 dB (VNS), and for SSIM from 0.272 to 0.297, indicating that VNS amplifies both improvements on easy cases and sensitivity on harder ones.

Metric relationships. Across all (algorithm, image) pairs, PSNR is strongly negatively correlated with MSE (as expected from the definition): $\rho(\mathrm{MSE}, \mathrm{PSNR}) \approx -0.93$. SSIM shows moderate positive association with PSNR ($\rho \approx 0.60$) but only weak negative association with MSE ($\rho \approx -0.34$), illustrating that structural fidelity (SSIM) is related to, but not fully determined by, pixelwise error. The img_t2 case exemplifies this: both PSNR and SSIM drop under VNS, consistent with a structure-damaging smoothing or tone choice.

Linking outcomes to learned pipelines. The *GA* best pipeline ends with `gauss` $\rightarrow$ `gamma`$(g>1)$ after two `cstretch` and two `unsharp` stages, implying a final darkening/contrast-tightening post-blur. The *VNS* best pipeline forms a tone "sandwich": two `unsharp` stages, `cstretch`, `gamma`$(g<1)$ (brighten), then `gauss` with a larger kernel ($k=9$), and a final `gamma`$(g>1)$ (recompress highlights). This sequence can explain the stronger gains on img_t3 and img_t5 (detail emphasis with controlled dynamic range), but also the regression on img_t2 where the combined brighten–smooth–darken could suppress midtones and damage local structures.

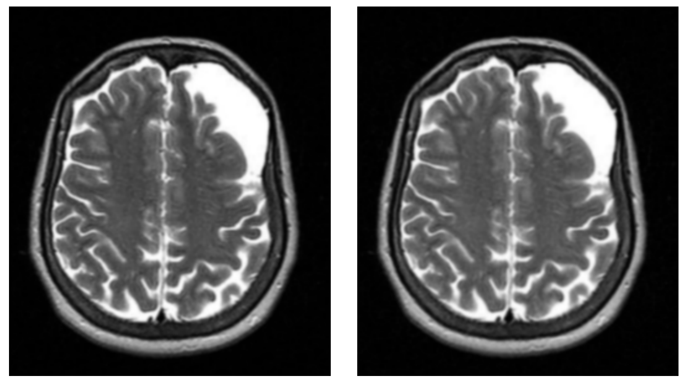
Train $\rightarrow$ test generalisation. During training, VNS improved the best objective from GA's $F=0.2392$ (SSIM$=0.7608$, PSNR$=19.79$ dB, MSE$=0.02644$) to $F=0.2286$ (SSIM$=0.7714$, PSNR$=20.51$ dB, MSE$=0.02579$). On test, gains persist but are smaller ($+0.59$ dB PSNR, $+0.0043$ SSIM on average), which is expected given distribution shift and the compact pipeline hypothesis class. This suggests that the additional neighbourhood exploration and local search in VNS improved the GA baseline without severe overfitting.

Takeaways. (1) VNS reliably refines the GA solution, particularly on images with clear structure and moderate noise; (2) SSIM gains track, but are not guaranteed by, PSNR/MSE improvements; (3) The tone–blur–tone motif in the VNS pipeline is powerful but can overshoot on images like img_t2, motivating either image-conditional pipeline selection or adaptive parameter scheduling (future work).



(a) GA: img_t3 (b) VNS: img_t3

Fig. 1. Qualitative comparison on `img_t3`. VNS yields crisper edges and fewer low-contrast regions, aligning with its higher SSIM/PSNR for this case.



(a) GA: img_t5 (b) VNS: img_t5

Fig. 2. Qualitative comparison on `img_t5`. VNS enhances fine detail and local contrast without overshooting highlights, consistent with its metric gains.

E. *Qualitative Analysis*

We include representative visual comparisons of the best GA and VNS pipelines on two test images. Each figure shows the *enhanced* outputs produced by GA and VNS (left–right). These samples corroborate the quantitative findings: VNS typically sharpens salient structures and controls tone more effectively, while GA can preserve midtones slightly better on some images (cf. Table II).

VI. Conclusion

In this paper, we framed grayscale image enhancement as a metaheuristic search over variable-length pipelines composed of five classical operators (Gamma, Gaussian blur, Unsharp masking, Histogram equalisation, Percentile contrast stretch). Using a training set of paired distorted/ground-truth images, we optimised either an SSIM-centric objective or a weighted scalarisation and evaluated generalisation on a 5-image test set with MSE, PSNR, and SSIM.

Genetic Algorithms (GA) established a strong baseline by discovering high-quality enhancement pipelines. Variable Neighbourhood Search (VNS), warm-started from the GA best, reliably refined this baseline: on the test set VNS improved the means by $\Delta\overline{\mathrm{PSNR}} = +0.59\,\mathrm{dB}$, $\Delta\overline{\mathrm{SSIM}} = +0.0043$, and $\Delta\overline{\mathrm{MSE}} = -9.6\times10^{-4}$, and won on 4/5 images for MSE and PSNR (3/5 for SSIM). Qualitative comparisons corroborated these trends: VNS tended to yield crisper edges and better tone management, though a minority case exhibited midtone attenuation where GA was preferable.

Overall, the results demonstrate that (i) automated pipeline search is effective for enhancement when ground truth is available, and (ii) VNS's neighbourhood exploration plus local search complements GA's global search by extracting additional gains without severe overfitting. Pipelines and seeds were persisted for reproducibility, enabling direct re-evaluation.

Limitations and future work. Our experiments are limited by a small test set, grayscale-only images, and a fixed operator set. Future extensions include Pareto multi-objective optimisation (e.g., SSIM–PSNR–AMBE), per-image adaptive pipeline schedules, broader operator libraries (e.g., CLAHE, bilateral/edge-preserving filters), and more robust validation (cross-validation or bootstrapping) to better characterise stability across datasets.

References

[1] R. D. Putra, T. W. Purboyo, and A. L. Prasasti, "A review of image enhancement methods," *International Journal of Applied Engineering Research*, vol. 12, no. 23, pp. 13 596–13 603, 2017. [Online]. Available: http://www.ripublication.com

[2] S. Rahman, M. M. Rahman, M. Abdullah-Al-Wadud, G. D. Al-Quaderi, and M. Shoyaib, "An adaptive gamma correction for image enhancement," *EURASIP Journal on Image and Video Processing*, vol. 2016, no. 35, 2016. [Online]. Available: https://doi.org/10.1186/s13640-016-0138-1

[3] S. Kansal, S. Purwar, and R. K. Tripathi, "Image contrast enhancement using unsharp masking and histogram equalization," *Multimedia Tools and Applications*, 2018. [Online]. Available: https://doi.org/10.1007/s11042-018-5894-8

[4] K. G. Dhal, S. Ray, A. Das, and S. Das, "A survey on nature-inspired optimization algorithms and their application in image enhancement domain," *Archives of Computational Methods in Engineering*, vol. 26, pp. 1607–1638, 2019. [Online]. Available: https://doi.org/10.1007/s11831-018-9289-9

[5] M. Braik, "Hybrid enhanced whale optimization algorithm for contrast and detail enhancement of color images," *Cluster Computing*, vol. 27, pp. 231–267, 2024. [Online]. Available: https://doi.org/10.1007/s10586-022-03920-9

[6] S. Muniyappan and P. Rajendran, "Contrast enhancement of medical images through adaptive genetic algorithm (aga) over genetic algorithm (ga) and particle swarm optimization (pso)," *Multimedia Tools and Applications*, vol. 78, no. 5, pp. 6487–6511, 2019. [Online]. Available: https://doi.org/10.1007/s11042-018-6355-0

[7] U. K. Acharya and S. Kumar, "Genetic algorithm based adaptive histogram equalization (gaahe) technique for medical image enhancement," *Optik*, vol. 130, pp. 1220–1225, 2017. [Online]. Available: https://doi.org/10.1016/j.ijleo.2016.11.137

[8] I. González-Díaz, M. Birinci, F. D. de María, and E. J. Delp, "Neighborhood matching for image retrieval," *IEEE Transactions on Multimedia*, vol. 19, no. 3, pp. 544–558, 2017. [Online]. Available: https://doi.org/10.1109/TMM.2016.2616298

[9] L. Li, Y. Du, F. Yao, S. Xu, and Y. She, "Learning memetic algorithm based on variable population and neighborhood for multi-complex target scheduling of large-scale imaging satellites," *Computers & Operations Research*, vol. 111, pp. 99–113, 2019. [Online]. Available: https://doi.org/10.1016/j.cor.2019.06.006