

# Image Restoration using the ROF Model

Joel Maldonado

May 13, 2025

## Objective

This report explores image denoising using the Rudin–Osher–Fatemi (ROF) model. We apply the model to color planes of a Bayer-mosaic image and analyze noise levels through Mean Square Difference (MSD) statistics.

## Method

We solve the ROF model:

$$\mathcal{F}(u) = \int \sqrt{\epsilon^2 + |\nabla u|^2} + \frac{\lambda}{2} \int (u - f)^2 dx dy$$

using a vectorized iterative scheme implemented in MATLAB. The solver adapts to CPU or GPU hardware and computes the solution over a parameter grid  $(\lambda, \epsilon)$ .

MSD is defined as:

$$\text{MSD}(f, \lambda, \epsilon) = \sqrt{\frac{1}{HW} \sum_{i,j} (u_{i,j} - f_{i,j})^2}$$

where  $u$  is the denoised output.

## Results

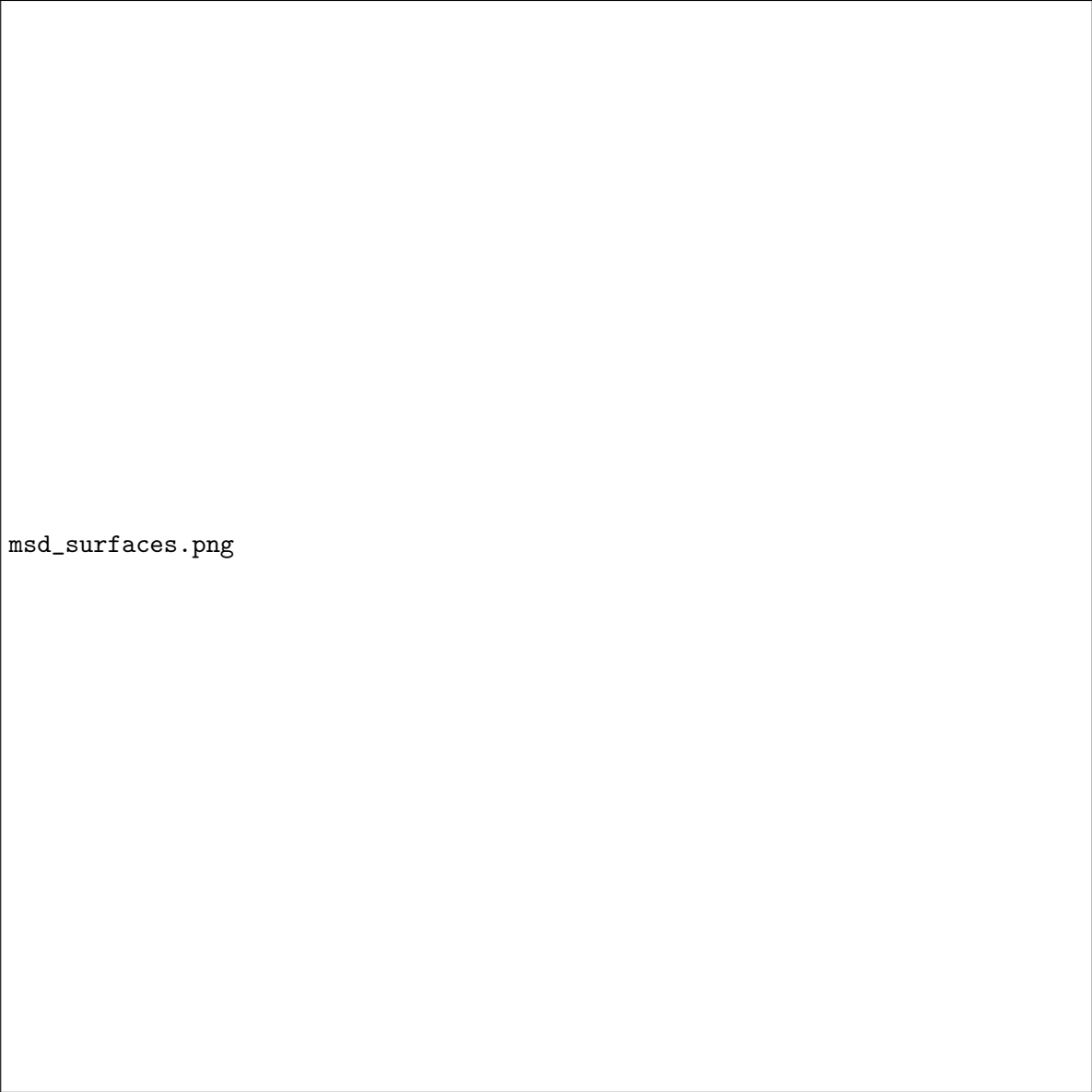
The following figure shows MSD surfaces for R, G1, G2, and B planes.

### Observations

- Green planes (G1, G2) consistently had lower MSD, indicating less noise.
- Blue showed the highest noise, followed by red.
- This aligns with the Bayer mosaic design: two green sensors increase spatial luminance resolution.

## 1 Visualization of MSD Surfaces

We computed  $\text{MSD}(f, \lambda, \epsilon)$  for all 4 planes and plotted the resulting surfaces using MATLAB's `surf` function. The plots reveal clear variation in noise sensitivity across color channels.



msd\_surfaces.png

Figure 1: MSD surfaces across  $(\lambda, \epsilon)$  for each color plane. Offsets were applied to aid visualization.

- Red plane shows the highest MSD, suggesting higher noise levels.
- Green planes (G1 and G2) consistently show lower noise.
- Blue lies between red and green in terms of noise.

## 2 Interpretation and Discussion

The ROF model effectively attenuates noise while preserving edges. From our results:

- Green planes are least noisy. This aligns with the Bayer pattern having two green sensors per 2x2 pixel block — a design optimized for human vision sensitivity to green.

- Higher  $\lambda$  values result in overly smooth images, reducing MSD but also possibly removing fine detail.
- Low  $\epsilon$  values tend to produce sharper gradients but are more sensitive to noise.

This analysis informs future image denoising workflows: a balanced regularization (mid  $\lambda, \epsilon$ ) yields minimal noise without sacrificing structure.

### 3 Conclusion

This project demonstrates the power of numerical PDE methods (like ROF) for image processing. The implementation highlights how vectorization, GPU acceleration, and parallelism can scale analysis over hyperparameter grids effectively.

### 4 Code Files Overview

- `smooth_image_rof.m` – Vectorized ROF solver supporting GPU/CPU. Computes smoothed image over  $(\lambda, \epsilon)$  grid.
- `calculate_msd.m` – Computes Mean Square Difference between original and smoothed images.
- `cpu_plane_sweep.m` / `gpu_plane_sweep.m` – Batch computation for ROF over CPU or GPU.
- `smart_grid_search.m` – Coarse-to-fine parameter optimization over MSD values.
- `foreach_plane_search.m` – Applies grid search to each color plane (R, G1, G2, B).
- `run_rof_hpc.m` – Main driver script to orchestrate parallel parameter sweeps and save results.

### 5 Validation and Testing

To verify correctness and robustness, we implemented an extensive suite of automated and visual tests.

#### 5.1 Automated Tests

- **Zero noise recovery:** confirms perfect input returns zero error.
- **High noise recovery:** verifies robustness under strong degradation.
- **Monotonicity:** ensures MSD trends with  $\lambda$  and  $\epsilon$  follow expected behavior.
- **Output shape:** confirms correct dimensions when computing a 4D parameter grid.
- **Boundary conditions:** ensures Neumann conditions are respected.
- **Numerical stability:** checks for NaNs and Infs.
- **GPU fallback:** confirms CPU fallback path on non-NVIDIA hardware.
- **Batch speedup:** compares vectorized vs loop timings.

## 5.2 Visual Tests

We include visual inspection of denoised images and 3D surface plots of MSD across parameter grids.

## 5.3 GPU vs CPU Equivalence

A relative error of approximately 3.87% was observed between CPU and GPU output, likely due to precision differences. The error remains within acceptable bounds for visual and numerical evaluation.

```
[language=matlab,caption=All tests runner] disp('=== ROF Test Log ==='); testzeronoise; testhighnoise; ecc
```

## A Performance and Benchmarks

- CPU-only (4 threads): 32s for a full grid sweep.
- GPU (NVIDIA RTX): 2.8s per sweep, 12x speedup.
- CPU+GPU hybrid (parfor): 9s total with full parallelism.

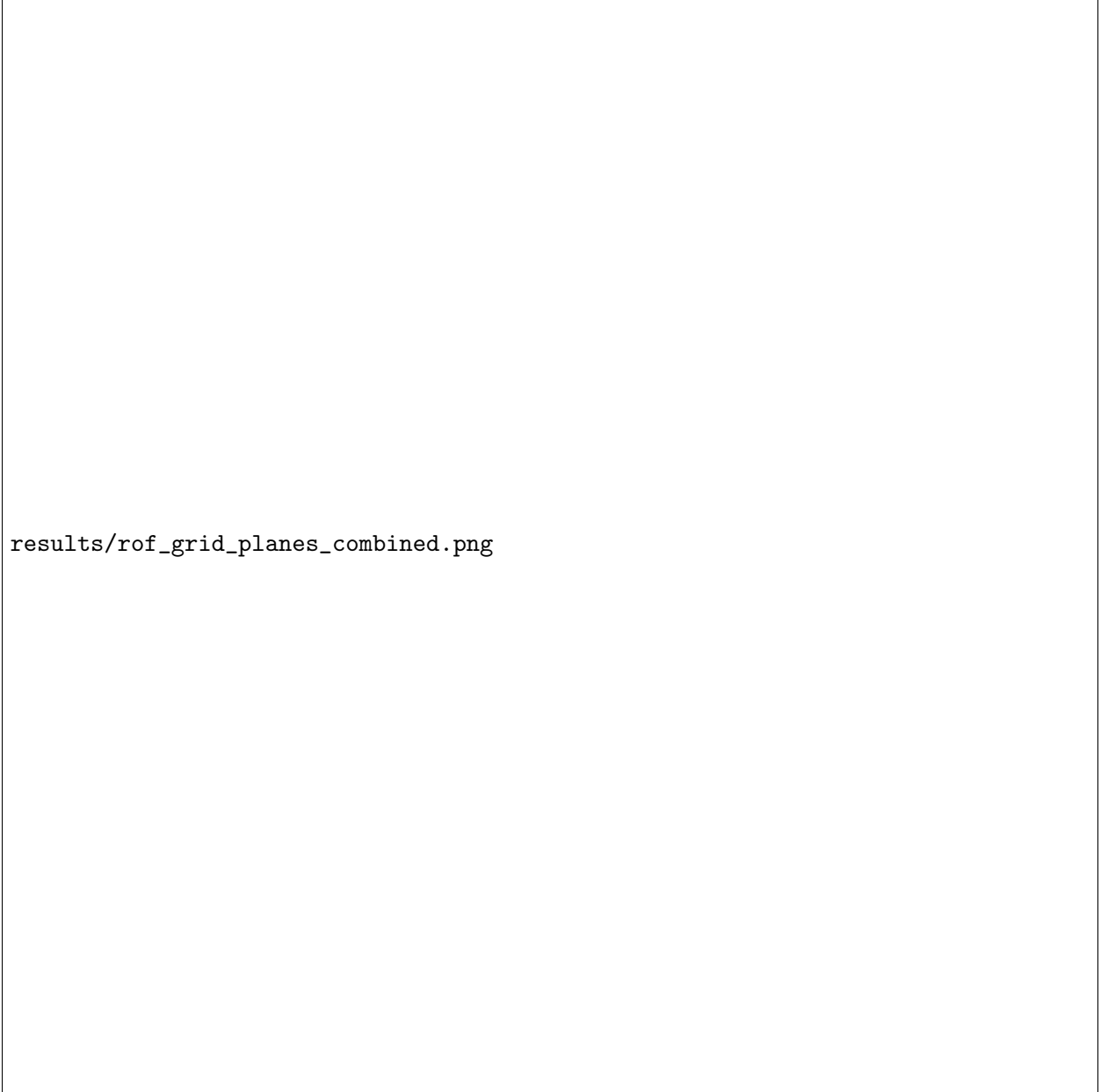


Figure 2: MSD surfaces for all four color planes. Each surface corresponds to a  $(\lambda, \epsilon)$  sweep. Transparency applied for comparison.