AAI-521 Final Team Project\_Extra Credit: Image Restoration

Syed Sirajuddin, Ian Rebmann, Carlos Ortiz, Tadbhir Singh

University of San Diego

AAI-521: Applied Computer Vision for AI

Dr. Roozbeh Sadeghian

December 7, 2025

Table of Contents

[Project Selection & Setup 3](#_Toc216127501)

[Pre-Processing: Artificially Damaged Images 3](#_Toc216127502)

[Model Methods 4](#_Toc216127503)

[Super-Resolution 4](#_Toc216127504)

[Base Model 4](#_Toc216127505)

[Fine-Tuning 4](#_Toc216127506)

[Limitations and Improvements 5](#_Toc216127507)

[Planned improvements include: 5](#_Toc216127508)

[Colorization 5](#_Toc216127509)

[Base Model 5](#_Toc216127510)

[Fine-Tuning Limitations 5](#_Toc216127511)

[Inpainting 6](#_Toc216127512)

[Base Model 6](#_Toc216127513)

[Fine-Tuning 6](#_Toc216127514)

[Validation Strategy and Metrics 7](#_Toc216127515)

[Results and Analysis 7](#_Toc216127516)

[Fine-Tuning Limitations and Next Steps 8](#_Toc216127517)

[Denoising 8](#_Toc216127518)

[Method 8](#_Toc216127519)

[Validation Considerations 9](#_Toc216127520)

[GUI: Web Application 9](#_Toc216127521)

[Validation: 9](#_Toc216127522)

[Conclusion 10](#_Toc216127523)

[References 11](#_Toc216127524)

[Denoising 11](#_Toc216127525)

[Super Resolution 11](#_Toc216127526)

[Colorization 12](#_Toc216127527)

[Appendix (Report) 13](#_Toc216127528)

[Preprocessing (Code in Addendum Appendix) 13](#_Toc216127529)

[P.1 13](#_Toc216127530)

[Super Resolution 14](#_Toc216127531)

[SR.1 14](#_Toc216127532)

[SR.2 15](#_Toc216127533)

[Inpainting 15](#_Toc216127534)

[I.1 Inpainting Pipeline 15](#_Toc216127535)

[I.2 Image Inference 16](#_Toc216127536)

[GUI Web Application 17](#_Toc216127537)

[G.1 Screenshot of Streamlit WebApp 17](#_Toc216127538)

# Project Selection & Setup

This project developed a unified computer vision restoration system that supports four core tasks: image denoising, super-resolution, colorization, and inpainting. The goal was to deliver a practical, user-facing toolkit that combines strong pretrained baselines with lightweight fine-tuning where feasible, then evaluate each task using appropriate metrics and controlled testing conditions. Rather than treating each task as an isolated demonstration, the project frames restoration as a realistic multi-step workflow. In practice, an image may require denoising first, localized repair next, and resolution enhancement last. This end-to-end perspective shaped both the backend architecture and the Streamlit-based interface, which provides consistent preprocessing, reproducible inference settings, and straightforward task switching across the four pipelines.

# Pre-Processing: Artificially Damaged Images

COCO 2017 served as the primary image source across tasks due to its diverse scenes, objects, and lighting conditions at large scale (100K+ images). To enable reliable ground-truth comparisons in restoration, artificially damaged versions of images were generated where needed, producing aligned inputs and targets for fair, repeatable evaluation.

Dataset preparation was tailored to each task:

* **Denoising** used clean COCO images as references and introduced synthetic noise to create paired inputs and targets. This design enabled controlled testing across known degradation types while preserving the correct clean image for comparison.
* **Super-resolution** used DIV2K HR images (800 train, 100 val, 100 test), paired with bicubic ×4 LR counterparts. Training on random 64×64 LR patches (256×256 HR) expanded the effective dataset while keeping runtime manageable.
* **Colorization** used filtered COCO images converted to grayscale inputs, with original color images as targets. Full training is known to be lengthy (~20+ hours on Colab T4 GPUs), and we did not fully reproduce that training under current constraints.
* **Inpainting** used paired triples—original images, synthetically damaged images, and binary masks—aligned by filename and resized to 512×512. Two controlled mask regimes using random rectangles were created:
  + *Small masks* (≈0.05–0.15 of image dimensions)
  + *Large masks* (≈0.10–0.40)  
    These regimes enabled a consistent comparison between the base and LoRA-adapted inpainting models.

(See Figure P.1 for examples of “Damaged Images”)

# Model Methods

## Super-Resolution

### Base Model

The super-resolution pipeline uses the 4× real-world Swin2SR checkpoint caidas/swin2SR-realworld-sr-x4-64-bsrgan-psnr. This model is a ~12M parameter Swin Transformer v2–based architecture designed to handle practical real-world degradations. Swin2SR follows an encoder-decoder structure with shallow convolutional feature extraction, multiple SwinV2 transformer blocks (local window self-attention), and an upsampling head that outputs a 4× higher-resolution RGB image. This model supports the project’s broader restoration theme by enhancing detail after upstream corrections such as denoising and inpainting while remaining computationally feasible within a unified application.

### Fine-Tuning

Starting from the pretrained checkpoint, we performed compact fine-tuning on random DIV2K patches using **L1 reconstruction loss** between predicted and ground-truth HR patches. The intent was to reduce overly smooth outputs on very blurry or out-of-distribution images. Training ran for **1000 steps** on an **A100-80GB GPU**, using Adam optimization and a small learning rate to preserve pretrained knowledge (see appendix SR.1 & SR.2 for the image comparison and L1 loss curve).

### Limitations and Improvements

Fine-tuning used a small subset of DIV2K and only 1000 steps due to compute constraints. With L1-only optimization and no perceptual/adversarial components, texture realism may lag behind the pretrained baseline in some cases.

### Planned improvements include:

* Train longer on full DIV2K plus additional datasets (e.g., Flickr2K) and test mixed losses (L1 + perceptual + adversarial).
* Add UI-level pre/post-processing options (denoising, sharpening, face-aware upscaling) and support batch inference.
* Evaluate newer transformer SR variants such as Swin2-MoSE or ConvSwin2SR.
* Explore different patch sizes, batch sizes, learning-rate schedules, and robustness augmentations (flips, rotations, compression).

## Colorization

### Base Model

The colorization pipeline uses a Hugging Face pretrained model with a **ResNet encoder** and **UNet decoder**. Images are converted from RGB/BGR into grayscale inputs with pixel values scaled to 0–1. The model performs colorization in the **Lab color space**, using the **L (lightness)** channel as input and predicting the **a (green–red)** and **b (yellow–blue)** channels. This framework reduces prediction complexity by requiring two chrominance channels instead of three RGB channels. The predicted a and b channels are returned in a normalized range (0–1), requiring reconstruction into valid Lab channel values before conversion back to RGB/BGR. We used the author’s normalization utility while implementing the remaining preprocessing and Lab→RGB conversion using **OpenCV** rather than Kornia.

### Fine-Tuning Limitations

During training, images were resized to **224×224**. At inference time, the architecture required input dimensions that are multiples of 224. To handle arbitrary user images, we added a preprocessing step that resizes inputs to the next lowest multiple of 224. Because the task is grayscale-to-color prediction and preprocessing is substantial (resizing + L-channel extraction), most general color image datasets can be used for training after filtering out low-quality or already monochrome samples. This model uses a combined **L1 + adversarial** loss structure. We explored alternative fine-tuning routes—including streamlined libraries, classical baselines, and a Stable Diffusion Img2Img-style approach—but these experiments confirmed that the current ResNet+UNet pipeline was the most reliable option within time and compute constraints.

Model performance can be found on the appendix.

## Inpainting

### Base Model

The inpainting component aimed to restore missing or damaged regions in a visually natural way. Practically, the model should fill masks without visible seams, texture discontinuities, or obvious hallucinations such as random text or watermark-like artifacts. In addition to perceptual quality, we evaluated similarity to the original images using objective metrics. The system uses **Stable Diffusion v1.5 Inpainting** via Hugging Face Diffusers. This latent diffusion architecture employs:

A **VAE** to encode images into latent space,

A **text encoder** for prompt conditioning, and

A **UNet** to perform iterative denoising across the diffusion schedule.

The inpainting UNet accepts a **9-channel latent input** composed of noisy latents (4), mask latents (1), and masked-image latents (4).

### Fine-Tuning

To test parameter-efficient improvement, **LoRA adapters** were inserted into key attention projections (to\_q, to\_k, to\_v, to\_out.0). The base UNet, VAE, and text encoder remained frozen. Core training settings were: **512×512 images**, **batch size 2**, **~2000 steps**, **learning rate 1e-4**, **rank 8 with alpha 8**, **gradient clipping 1.0**, and **fp16 mixed precision**. (See Appendix I.1 for visualization of Inpainting Pipeline)

### Validation Strategy and Metrics

Because global metrics can hide meaningful differences when masked regions are small, evaluation emphasized **masked PSNR** and **masked SSIM**, computed only on the white mask area. Global PSNR/SSIM were retained as secondary references.

### Results and Analysis

Across both synthetic damage regimes, the **base inpainting model** generally outperformed the LoRA-adapted variant under the current training budget.

**Small masks:**

LoRA masked PSNR win-rate: **32%**

LoRA masked SSIM win-rate: **25%**

Base mean masked PSNR: **17.998** vs. LoRA **17.541**

**Large masks:**

LoRA masked PSNR win-rate: **26%**

LoRA masked SSIM win-rate: **25%**

Base mean masked PSNR: **13.963** vs. LoRA **12.919**

These results suggest LoRA is not inherently ineffective, but the current configuration may be underpowered relative to the strength of the pretrained baseline. The limited training duration and rank-8 capacity likely constrained learning of large-region structural priors. In addition, diffusion inpainting can produce perceptually convincing reconstructions that remain pixel-different from the ground truth; thus PSNR/SSIM may understate real visual quality. A distinct practical contribution of this work was the explicit emphasis on reducing hallucinated text or label-like artifacts—a common real-world failure mode of generative restoration models that is not fully captured by similarity metrics alone. (See Appendix I.2 for example images)

### Fine-Tuning Limitations and Next Steps

Future improvements should include:

* Formal **train/validation/test** splits,
* More realistic mask diversity (irregular shapes),
* Higher LoRA ranks (e.g., **16–32**),
* Longer training,
* Results stratified by mask area and image category.
* System Architecture

To support all four tasks as a cohesive product, the system uses a shared, configuration-driven design that routes inputs to task-specific pipelines for denoising, super-resolution, colorization, and inpainting. Each pipeline follows consistent device and precision handling and exposes a lightweight standardized inference interface. This modular structure supports both single-task execution and multi-step restoration chains, aligning with the project’s premise that real images often require multiple forms of enhancement. The architecture reflects practical engineering choices. Stable Diffusion is used for both denoising (Img2Img) and inpainting, enabling shared dependencies and consistent prompt-driven control. Swin2SR and the ResNet+UNet colorization model remain focused, task-optimized components. This balance improves maintainability and provides a clear path for integrating newer model variants in future iterations.

## Denoising

### Method

The denoising component was implemented using a Stable Diffusion Img2Img baseline with runwayml/stable-diffusion-v1-5. This approach treats denoising as controlled generative refinement rather than direct pixel regression. The main inference controls—**strength**, **guidance scale**, **inference steps**, and optional **seed**—enable systematic experimentation and help balance noise removal against the risk of over-generation.

### Validation Considerations

Because this baseline is inference-driven and generative, a dedicated noisy-clean training set was not required to demonstrate functional capability within the unified system. However, the design remains compatible with future quantitative benchmarking. Introducing standardized synthetic noise patterns (e.g., Gaussian noise, compression artifacts, or motion blur) would support objective PSNR/SSIM comparisons across defined noise regimes and strengthen validation beyond qualitative inspection.

# GUI: Web Application

The Streamlit interface serves as applied validation of the complete system. Users can upload images, select tasks, adjust parameters, and preview results in real time. The inpainting UI provides interactive mask creation with consistent preview sizing, brush controls, and task-aware layout. This deployment bridges offline experimentation with a realistic end-user workflow and demonstrates that the system extends beyond notebooks into a functional restoration tool.

# Validation:

For our evaluation we mainly relied on a qualitative standard, visually inspecting the outputs for each restoration task to verify that the models behaved as expected on real images. For denoising, we compared noisy inputs to denoised outputs and confirmed that grain and speckle artifacts were reduced while key structures and edges remained intact. For colorization, we checked that originally grayscale images were converted to plausible color images with consistent hues and no obvious color bleeding. For inpainting, we verified that the masked regions were filled with realistic content that blended smoothly with the surrounding context and did not introduce glaring artifacts. For super‑resolution, we ensured that the enhanced outputs appeared sharper and less pixelated than the low‑resolution inputs, and additionally used a programmatic approach to confirm that the spatial resolution (height and width) of the super‑resolved images was exactly 4× larger than the corresponding inputs; please refer to the appendix for the verification script

# Conclusion

This project presents a unified restoration framework that integrates four high-impact computer vision tasks under a consistent architecture and user-facing application. The system demonstrates strong pretrained baselines with selective fine-tuning experiments designed to test parameter-efficient adaptation under practical compute constraints. The super-resolution subsystem effectively leverages Swin2SR and a compact patch-based fine-tuning strategy. The colorization pipeline demonstrates careful architectural and preprocessing design grounded in the Lab color space, while acknowledging training and inference constraints. The denoising workflow provides a functional Stable Diffusion Img2Img baseline that fits cleanly into the shared multi-task architecture. Inpainting produced the most controlled experimental findings. Under the current training budget, the base Stable Diffusion v1.5 inpainting model remained robust across both small and large mask regimes. The LoRA-adapted version did not consistently exceed the baseline, indicating that longer training, higher adapter ranks, and more realistic mask diversity will likely be required to achieve reliable gains.

Most importantly, the project balanced metric-driven evaluation with real-world perceptual goals. While masked PSNR and SSIM quantified similarity to ground truth, the system also prioritized reducing hallucinated text and label-like artifacts—an applied restoration objective aligned with real user expectations. Overall, the project meets its objectives with feasible scope, modern model choices, task-appropriate evaluation, and a clear end-to-end implementation that connects research methods with an operational interface.

# References

## Denoising

## Super Resolution

Swin2SR paper

Conde, M. V., Choi, U. J., Burchi, M., & Timofte, R. (2022). Swin2SR: SwinV2 transformer for compressed image super-resolution and restoration (arXiv:2209.11345). arXiv.

<https://doi.org/10.48550/arXiv.2209.11345>

Swin2SR GitHub repository

Conde, M. V., Choi, U. J., Burchi, M., & Timofte, R. (2022). Swin2SR: SwinV2 transformer for compressed image super-resolution and restoration [Computer software]. GitHub.

<https://github.com/mv-lab/swin2sr>

Swin2SR Hugging Face checkpoint

Caidas AI. (2022). swin2SR-realworld-sr-x4-64-bsrgan-psnr [Computer software]. Hugging Face.

<https://huggingface.co/caidas/swin2SR-realworld-sr-x4-64-bsrgan-psnr>

DIV2K dataset

Agustsson, E., & Timofte, R. (2017). NTIRE 2017 challenge on single image super-resolution: Dataset and study. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (pp. 1122–1131).

<https://data.vision.ee.ethz.ch/cvl/DIV2K/>

## Colorization

Curated Coco Dataset

Shah, A. (2025, October 23). *coco-2017-image-colorization-224*. Huggingface.co. <https://huggingface.co/datasets/ayushshah/coco-2017-image-colorization-224>

ImageColorization Hf space

*ayushshah/imagecolorization · Hugging Face*. (2025, August 17). Huggingface.co. <https://huggingface.co/ayushshah/imagecolorization>

ImageColorization Github Reference

ayushpatidar18. (2025). *GitHub - ayushpatidar18/Image-Colorization*. GitHub. <https://github.com/ayushpatidar18/Image-Colorization>

Colorization GitHub repository

Zhang, R. (2024, May 29). *richzhang/colorization*. GitHub. <https://github.com/richzhang/colorization>

DeOldify GitHub repository

jantic. (2025). *GitHub - jantic/DeOldify: A Deep Learning based project for colorizing and restoring old images (and video!)*. GitHub. <https://github.com/jantic/DeOldify?tab=readme-ov-file#getting-started-yourself>

Inpainting

Bernstein, D. A., & Borkovec, T. D. (1973). *Progressive relaxation training: A manual for the helping professions*. Research Press.

# Appendix (Report)

## Preprocessing (Code in Addendum Appendix)

### P.1

From left-> right and top->down:  
Original, Black and White. Low Resolution, Noise, Damaged, Mask

A dog standing in a field with a herd of sheep

AI-generated content may be incorrect.A dog standing in a field with sheep

AI-generated content may be incorrect.

A dog standing in a field with sheep

AI-generated content may be incorrect.A dog standing in a field with sheep

AI-generated content may be incorrect.

A dog standing in a field with sheep

AI-generated content may be incorrect.A black background with white squares

AI-generated content may be incorrect.

## Super Resolution

### SR.1

A screenshot of a computer screen

AI-generated content may be incorrect.

### SR.2

A graph of a graph showing a loss

AI-generated content may be incorrect.

The plot summarizes how the L1 reconstruction loss behaves during an additional 300 step fine‑tuning run on random DIV2K patches. Each point on the graph represents one optimization step and measures the absolute difference between the model’s super resolved patch and its corresponding high resolution ground truth patch. Because every step uses a different random crop, some patches are easier than others, causing the loss to oscillate between roughly 0 and 0.22 rather than forming a perfectly smooth trajectory. Overall, the loss hovers around 0.1 without a clear downward trend, which is expected given that Swin2SR already starts from a strong pretrained checkpoint and this extra 300 step run makes only small refinements instead of large improvements in training error

### SR.3

A screenshot of a computer program

AI-generated content may be incorrect.

Code used to confirm that the spatial resolution (height and width) of the super‑resolved images was exactly 4× larger than the corresponding inputs.

## Inpainting

### I.1 Inpainting Pipeline

A diagram of a solution

AI-generated content may be incorrect.

### I.2 Image Inference

Example of Different behavior which show’s realistic behavior but not matching original picture

Masked Base PSNR=21.60 SSIM=0.896 | LoRA PSNR=14.02 SSIM=0.887 Global Base PSNR=26.42 SSIM=0.743 | LoRA PSNR=21.42 SSIM=0.731 File: test-00009-of-00028\_002789.png Top-left: original | Top-right: damaged | Bottom-left: base | Bottom-right: LoRA

A collage of birds flying over water

AI-generated content may be incorrect.

Masked Base PSNR=16.77 SSIM=0.882 | LoRA PSNR=17.13 SSIM=0.887 Global Base PSNR=23.51 SSIM=0.763 | LoRA PSNR=23.63 SSIM=0.762 File: test-00001-of-00028\_003291.png Top-left: original | Top-right: damaged | Bottom-left: base | Bottom-right: LoRA

A collage of a dog

AI-generated content may be incorrect.

## Colorization

### C.1 Colorization Model Performance

A comparison of waves in different stages of weather

AI-generated content may be incorrect.

A screenshot of a dog

AI-generated content may be incorrect.

## GUI Web Application

### G.1 Screenshot of Streamlit WebApp

A screenshot of a computer

AI-generated content may be incorrect.