Image Inpainting for Object Removal using Diffusion-Based Technique

Sakina Jambughodawala
Prina Patel
Aneri Maniar
Nidhee Adesara

***Abstract*—Image inpainting is the process of reconstructing missing or damaged regions of an image in a visually natural manner. In this project, we implement a hybrid inpainting method that combines Telea's fast marching method with anisotropic diffusion. First, the damaged region is detected using a binary mask generated from intensity thresholding. The filling order is computed using a Euclidean distance transform, ensuring that pixels closest to the known region are processed/filled first. Telea's directional weighted model uses nearby pixel values and image gradients to transmit surrounding information into the unknown area by smoothly extending colours and edges. After the initial fill, we apply Perona–Malik anisotropic diffusion PDE to refine the inpainted region, enhance edge continuity, and remove artefacts. This proposed method achieves smooth reconstruction while preserving major structural features.**

***Index Terms*—Image Inpainting, Telea Method, Fast Marching, Anisotropic Diffusion, Perona–Malik Model, Partial Differential Equations, Image Restoration, Mask Detection.**

I. Introduction

Digital images often suffer from damage, unwanted objects, or missing information due to factors such as ageing, noise or object removal. Image inpainting aims to restore such areas by filling in missing pixels, making the result appear natural to the human eye. The method we chose relies on partial differential equation (PDE)-based models, which propagate surrounding structures into the missing region by following isophotes (lines of equal intensity).

In this project, we have designed an inpainting system that utilises Telea's fast marching method, combined with anisotropic diffusion, to enhance the results. First, the damaged part of the image is detected using a binary mask. Then, the missing pixels are filled in step by step using information from the nearby known pixels. In this method, the edges are smoothly extended and the missing region is filled by using the information of surrounding regions, so that the missing patch is restored in a natural and visually consistent way.

II. Methodology

A. Mathematical model

1) Problem Definition: Let the given input image be:

$$I : \Omega \subset R^2 \to R^3$$

where:

Ω is the set of all pixel coordinates (x, y) in the image.

R^2 - In images, this corresponds to pixel coordinates (x, y).
R^3 - For images, this is usually the RGB colour space.

Let, missing or damaged region be:

$$D \subset \Omega,$$

we need to compute an inpainted image

$$\tilde{I} : \Omega \to R^3,$$

such that

$$\tilde{I}(x) = I(x) \quad \forall\, x \notin D,$$

and the structures and edges continue smoothly inside the missing region D

2) Mask and Region Classification: Let a binary mask be:

$$M(x) = \begin{cases} 1, & x \in D, \\ 0, & x \notin D \end{cases}$$

Each pixel x belongs to one of the following sets:

- Known region: $M(x) = 0$
- Band region (boundary):

$$\partial D = \{x \in D : \exists\, y \notin D,\ \|x - y\| = 1\}$$

- Inside region: deep unknown pixels in D

3) Distance Transform (Fast Marching Ordering): Euclidean distance transform will create a filling order:

$$d(x) = \text{DistanceTransform}(M).$$

Pixels are processed in ascending order of distance:

$$(x_1, x_2, \ldots) = \text{argsort}_x\ d(x)$$

This forms the marching front used in Telea's Fast Marching Method.

4) Telea's Weighted Averaging Inpainting Model: For each unknown pixel p, the value is computed using known neighbors q.

4.1 Direction Vector

$$\vec{r}_{pq} = p - q$$

p is the unknown pixel inside the missing region (the mask).
q is a known neighboring pixel near the boundary of the missing region.

4.2 Image Gradient at Known Pixel

$$\nabla I(q) = \begin{bmatrix} g_x(q) \\ g_y(q) \end{bmatrix}$$

4.3 Telea Directional Weight The Telea weight combines:

- distance preference,
- gradient alignment.

$$w(p,q) = \frac{\vec{r}_{pq} \cdot \nabla I(q)}{\|\vec{r}_{pq}\|^3}$$

$\vec{r}_{pq}$ is the direction vector from pixel q (known pixel) to pixel p (unknown pixel).

$w(p,q)$ is the weight assigned to pixel q when estimating the value at pixel p.

Expanded form:

$$w(p,q) = \frac{(p_x - q_x)g_x(q) + (p_y - q_y)g_y(q)}{\left[(p_x - q_x)^2 + (p_y - q_y)^2\right]^{3/2}}$$

5) Telea Inpainting Update Rule: For each color channel $c \in \{R, G, B\}$,

$$I_c(p) = \frac{\sum\limits_{q \in N(p)} w(p,q)\, I_c(q)}{\sum\limits_{q \in N(p)} w(p,q)}$$

$N(p)$ denotes the neighborhood of pixel p.

If the denominator is zero then quation becomes:

$$I_c(p) = \text{median}_{q \in N(p)}\, I_c(q)$$

This step ensures stable texture continuation.

6) PDE Refinement (Perona–Malik Diffusion): After Telea filling, anisotropic diffusion smooths shading while preserving edges to recover the texture of that area.

6.1 Gradient Magnitude will be computed as:

$$|\nabla I| = \sqrt{I_x^2 + I_y^2}$$

I_x denotes the partial derivative of the image intensity with respect to the horizontal (x) direction.

I_y denotes the partial derivative of the image intensity with respect to the vertical (y) direction.

6.2 Conductivity Function will be:

$$c(x) = \exp\left[-\left(\frac{|\nabla I(x)|}{K}\right)^2\right]$$

K is the edge threshold used in the Perona–Malik conductivity function. Only gradients larger k should be protected from diffusion.
$\nabla I(q) > K$ preserves sharp edges.
$\nabla I(q) < K$ preserves smoothening.

Edges yield $c \approx 0$ i.e., diffusion stop and edge detected, flat regions yield $c \approx 1$ i.e., diffusion continues and smoothness occurs.

6.3 Discrete PDE Update

Let N, S, E, W be 4-connected neighbors. Flux will be calculated as:

$$F = c_N(I_N - I) + c_S(I_S - I) + c_E(I_E - I) + c_W(I_W - I)$$

I is the intensity of that pixel.
Final update:

$$I(t+1) = I(t) + \lambda F$$

Here, let:

$$\lambda = 0.18, \qquad \text{40 iterations}$$

7) Complete Mathematical Flow:

1) **Pre-processing:** convert to grayscale, compute mask, gradient ∇I.
2) **Region Separation:**
$$\Omega = \text{KNOWN} \cup \text{BAND} \cup \text{INSIDE}.$$
3) **Compute Distance Map:**
$$d(x) = \text{EDT}(M),$$
sorted into a priority queue.
4) **Telea Inpainting:**
$$w(p,q) = \frac{\vec{r} \cdot \nabla I(q)}{\|\vec{r}\|^3}, \qquad I(p) = \frac{\sum wI}{\sum w}$$
5) **PDE Refinement:**
$$I(t+1) = I(t) + \lambda\, \nabla \cdot (c \nabla I).$$

B. Algorithm Steps

1) Input: Original RGB image or gray image I.
2) Convert I to grayscale and generate a binary mask M where missing pixels are marked as 1 and valid pixels as 0.
3) Initialize masked pixels as *INSIDE*, valid pixels as *KNOWN*, and detect the *BAND* boundary around the mask.
4) Compute the distance transform on M and insert all BAND pixels into a priority queue, where priority is based on distance from known pixels.
5) Compute grayscale gradients using Sobel filters to obtain directional information for filling.
6) While the priority queue is not empty:
   a) Extract the BAND pixel with the smallest distance.
   b) For each color channel, gather known neighbors and compute direction-based weights using:
   $$w = \frac{(p_x - q_x)g_x(q) + (p_y - q_y)g_y(q)}{r^2 \cdot r}$$
   c) If positive weights exist, assign a weighted value; otherwise use the median of neighbours.

d) Mark the pixel as KNOWN and promote its INSIDE neighbors to BAND.

7) After initial filling, perform iterative refinement:
   a) Recompute gradients and gradient magnitude.
   b) Compute diffusion conductance:

$$c = \exp\left(-\left(\frac{|\nabla I|}{\kappa}\right)^2\right)$$

   c) Update masked pixels using a flux-based anisotropic diffusion step.

8) Repeat the refinement for multiple iterations to smooth the filled region while preserving edges.
9) Clip pixel values to the valid range and convert to integer format.
10) Output: Final inpainted RGB image.

C. Flowchart

The following flowchart illustrates the complete pipeline of the proposed hybrid image inpainting method, showing the sequence from mask generating to Telea-based filling and PDE refinement.

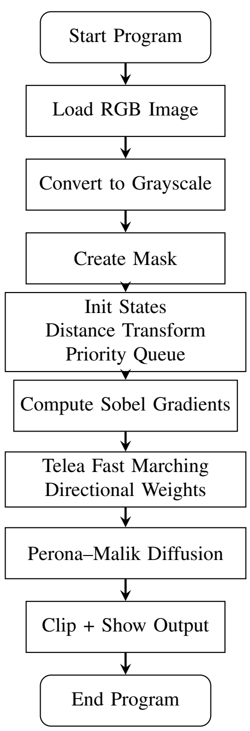

Fig. 1. Inpainting Algorithm Flowchart

D. Pseudocode

```
# Load image
image_rgb = load_image("input.png")

# Convert to grayscale
gray = convert_to_grayscale(image_rgb)

# Create mask (1 = missing pixel,0=known)
mask =
create_threshold_mask(gray,threshold=0.97)

# Smooth mask using morphology
mask = smooth_mask(mask)

# Initialize inpainting image
inpaint = gray.copy()
# initialize hole region
inpaint[mask == 1] = 0

# Set algorithm parameters
dt = 0.1
alpha = 1.0
max_iter = 500
tolerance = 1e-4

for iter in range(max_iter):

    # Compute gradient and Laplacian
    grad_x, grad_y =
        compute_gradients(inpaint)
    lap = compute_laplacian(inpaint)

    # Update only missing pixels
    (PDE evolution)
    update = grad_x + grad_y + alpha*lap
    inpaint[mask == 1]+=dt*update[mask==1]

    # Check convergence
    if has_converged(inpaint, tolerance):
        break

#Merge restored region with original image
final_image = merge_with_original(
    original=image_rgb,
    inpainted=inpaint,
    mask=mask
)

# Save final output
save_image(final_image, "output.png")
```

III. Results

The diffusion-based inpainting successfully filled missing parts of the test images in the following way:
The first image shows the original photos without damaged regions in which the image 1 is taken from an online platform [1], and the others are real-life images.
The second photo shows the output we got from the code. It displays three columns: the first shows an image with the damaged part, the second shows the mask created through code, and the last is the final output we obtained.

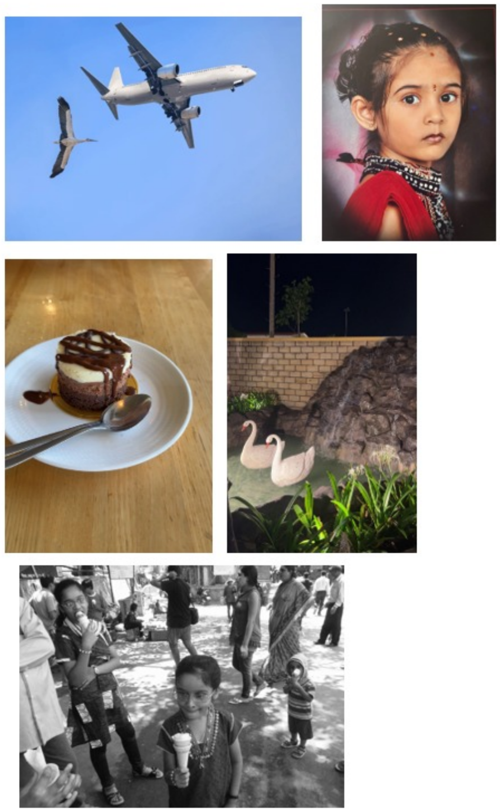

Fig. 2. Original images (without damage)

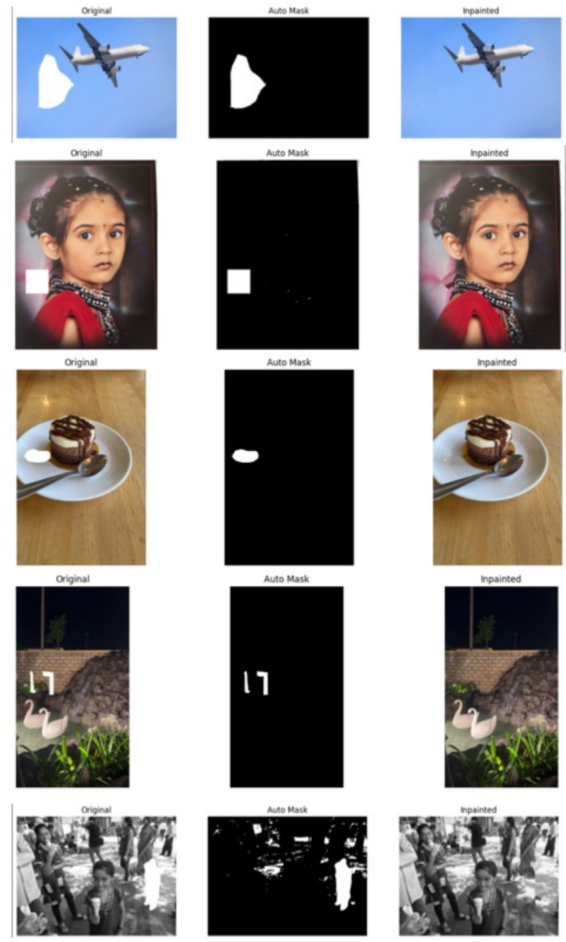

Fig. 3. Input-mask-output after inpainting

IV. Discussion

This developed inpainting model that combines Telea's fast marching with anisotropic diffusion.

Telea's method is used to restore missing pixels by gradually filling them in from the boundary and moving inward using weighted averages of nearby known pixels. It maintains smooth edges and structures while keeping the process fast and efficient.

The Perona–Malik anisotropic diffusion method smooths an image while preserving edges by controlling the amount of diffusion based on gradient strength. The proposed hybrid approach, which integrates mask detection, Telea interpolation, and PDE-based refinement, achieves smooth reconstruction.

Potential solutions are:

- Anisotropic diffusion to improve how edges are preserved.
- Structure-based smoothing continues and shapes it to look more naturally.

V. Conclusion

This method successfully restores missing regions by combining Telea's fast marching approach with Perona–Malik anisotropic diffusion. The distance transform guides the filling order, while gradient-based weighting maintains smooth edge continuation and texture merges from known areas. The PDE equation helps to enhance and preserve the important edges. This method also maintains the geometric continuation by propagating isophotes into the damaged region. The integration of gradient information ensures that the final details are preserved during the image restoration process. Anisotropic diffusion further improves the reconstruction while preserving edges. Overall, this method produces an inpainted image that looks smooth and blends well with the original.

References

[1] Shutterstock, "Birds airplane sky — Stock Images," [Online]. Available: https://www.shutterstock.com/search/birds-airplane-sky

[2] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 7, pp. 629–639, 1990. [Online]. Available: https://www.sci.utah.edu/~gerig/CS7960-S2010/materials/Perona-Malik/PeronaMalik-PAMI-1990.pdf

[3] A. Telea, "An Image Inpainting Technique Based on the Fast Marching Method," *Journal of Graphics Tools*, vol. 9, no. 1, pp. 23–34, 2004. [Online]. Available: https://www.researchgate.net/publication/238183352_An_Image_Inpainting_Technique_Based_on_the_Fast_Marching_Method