EUROPEAN JOURNAL OF PURE AND APPLIED MATHEMATICS
2025, Vol. 18, Issue 3, Article Number 6192
ISSN 1307-5543 – ejpam.com
Published by New York Business Global

A Lattice Boltzmann Method for Image Inpainting Inspired by Fluid Dynamics

Yassine Douich[1,*], Hassan Silkan[1], Youssef Hanyf[2]

[1] *Department of Computer Science, Laboratory LAROSERI, Faculty of Sciences, Chouaib Doukkali University, El Jadida, Morocco*
[2] *Research Laboratory in Management and Decision Support, AI Data SEED Team, Ibn Zohr University, Dakhla, Morocco*

Abstract. This paper presents a novel image inpainting algorithm based on the Lattice Boltzmann Method (LBM), inspired by the vorticity-stream formulation of the Navier-Stokes equations. The proposed method, employing D2Q5 and D2Q9 lattice models, effectively reconstructs missing or damaged image regions by propagating smoothness information along isophote directions. Experimental evaluations demonstrate that LBM achieves superior results in terms of PSNR, SSIM, and reduced CPU time compared to classical inpainting techniques, such as Total Variation (TV) regularisation and the Bertalmio-Sapiro-Caselles-Ballaster (BSCB) algorithm. The method's efficiency in handling complex boundary conditions and preserving image details highlights its robustness for advanced image restoration tasks.

2020 Mathematics Subject Classifications: 65N20, 76-XX

Key Words and Phrases: LBM Inpainting, Navier-Stokes Inpainting, Image Reconstruction, Edge-Preserving Diffusion, Vorticity-Stream Function

1. Introduction

Image inpainting plays a crucial role in image restoration, aiming to accurately and seamlessly fill in missing or damaged regions of an image. It is widely used for tasks such as removing unwanted objects, repairing degraded images, and correcting issues like cracks, scratches, stains, and red-eye effects.

The lattice boltzmann method (LBM), grounded in the principles of microscopic statistical physics, has emerged as an effective approach for simulating fluid flows and capturing complex fluid behavior. Unlike traditional fluid dynamics solvers, LBM offers a more straightforward programming model by simulating particle interactions and deriving partial differential equations (PDEs) through moment operations on particle distributions. Although LBM's application in image processing is relatively recent, its precursor, cellular

*Corresponding author.
DOI: https://doi.org/10.29020/nybg.ejpam.v18i3.6192

Email addresses: y.douich@ucd.ac.ma (Y. Douich)

automata, has been extensively applied in this field. Notable examples include studies by Preston et al. [1], Hernandez et al. [2], Popovici et al. [3], Wongthanavasu et al. [4], and Rosin [5], which primarily target binary image processing due to the challenges in designing cellular automata rules for 256 gray-level images. LBM, however, does not face this restriction, and it has been used to solve equations such as the diffusion and diffusion-convection equations, simplifying rule design for processing images with 256 gray levels.

The lattice Boltzmann method (LBM) has shown great promise in image processing. Jawerth et al. (1999) applied LBM to anisotropic diffusion using the Perona–Malik model [6], followed by our 2008 study [6, 7], and Chang's work on denoising [8]. In this paper, we propose a new LBM framework for image inpainting based on the vorticity–stream formulation of the Navier–Stokes equations, treating the image as a fluid medium.

Our method couples two LBM schemes for vorticity transport and intensity evolution, incorporates adaptive anisotropic diffusion for edge preservation, and uses non-equilibrium boundary conditions to handle complex contours.

The main contribution is a fast, robust LBM scheme tailored for inpainting, delivering coherent and high-quality restorations beyond traditional methods.

This work is structured as follows: Section 2 reviews related work in image inpainting, categorizing methods into PDE-based and learning-based approaches while highlighting gaps in fluid dynamics-inspired techniques. Section 3 formalizes the mathematical foundation, presenting the vorticity-stream formulation of Navier-Stokes equations adapted for inpainting. Section 4 details the proposed Lattice Boltzmann Method (LBM) framework, including D2Q5/D2Q9 implementations and boundary condition treatments. Section 5 evaluates performance through quantitative metrics (PSNR/SSIM) and comparative benchmarks against TV and BSCB methods. Section 6 concludes with contributions and outlines future research directions, notably LBM-CNN hybridization.

2. Related Work for Image Inpainting

In this section, we examine two main categories of research in this field: non-learning-based methods and learning-based methods.

2.1. Non-Learning-Based Methods

Non-learning-based approaches do not require training data and are typically grounded in mathematical models. Examples include partial differential equations (PDEs) for isophote propagation [9], computational fluid dynamics-based isophote propagation [10], and total variation (TV) minimisation [11] and more traditional methods in [12] . These foundational techniques aim to preserve the structural integrity of the image during the inpainting process.

A notable subset of these methods, known as **diffusion-based approaches**, incorporates concepts such as intra-channel and inter-channel local variances [13], the distance and direction between the damaged pixel and its neighbouring pixels [14], and the applica-

tion of fractional-order derivatives combined with Fourier transforms [15]. These methods excel in handling diverse inpainting challenges while maintaining computational efficiency.

2.2. Learning-Based Methods

In recent years, deep convolutional neural networks (CNNs) have demonstrated remarkable success in various computer vision tasks, including image inpainting. Unlike non-learning-based methods, CNN-based approaches leverage large-scale datasets and advanced architectures to achieve high-quality results. These methods are particularly effective for texture reconstruction and ensuring global structural consistency.

Several specialised CNN architectures have been developed to address specific inpainting challenges, including U-Net [16], Shift-Net, PEN-Net [17], and BICNN [18]. These architectures are designed for tasks such as texture completion, fine-detail restoration, and filling large gaps. Additional applications include inpainting X-ray medical images [19], forensic image restoration [20], video inpainting for object removal [21], and scanned data recovery [22]. Techniques such as coherent semantic attention layers [23] and pyramidal-context architectures [24] further enhance the capabilities of CNN-based methods.

Furthermore, specialised models have been introduced for unique applications. For instance, Artist-Net [25] focuses on high-quality artistic restoration, VGG-based approaches are tailored for UAV data recovery [26], and GRNN-based methods are effective for non-texture inpainting [27]. Other applications include text removal [28], artwork enhancement [29], and semantic object removal [30], showcasing the versatility of CNN-based methods. Emerging techniques, such as blind inpainting [18] and advanced encoder-decoder architectures that integrate super-resolution, denoising, and inpainting capabilities [31, 32], continue to push the boundaries of image restoration.

The development and evaluation of image inpainting algorithms rely heavily on publicly available and large-scale datasets. The choice of dataset categories significantly impacts the effectiveness of the proposed methods. Common categories include natural images, artificial images, face images, and others.

This work highlights some of the most widely used datasets for image inpainting, such as Paris StreetView [33], Places [34], depth image datasets [35], Foreground-aware datasets [36], Berkeley segmentation [37], and ImageNet [38], among others.

3. Mathematical Formulation

3.1. Navier-Stokes Equations

Building upon the ideas presented in [10], we consider the incompressible, Newtonian fluid flow governed by the Navier-Stokes equations:

$$\frac{\partial \vec{u}}{\partial t} + (\vec{u} \cdot \nabla)\vec{u} + \nabla P = \nu \Delta \vec{u}, \quad \nabla \cdot \vec{u} = 0$$

where $\vec{u}$ is the velocity vector, P is the pressure, and ν is the viscosity. For two-dimensional flows, we introduce a stream function Ψ, such that $\vec{u} = \nabla^{\perp}\Psi$. Defining the vorticity

$\omega = \nabla \times \vec{u}$, we obtain the vorticity-stream function formulation of the Navier-Stokes equations:

$$\omega_t + \vec{u} \cdot \nabla\omega = \nu\Delta\omega \tag{1}$$

In the limit of negligible viscosity, i.e., when $\nu \approx 0$, these equations reduce to the Euler equations for inviscid flow. The steady-state solution of Eq. (1) then approaches:

$$\vec{u} \cdot \nabla\omega = \nabla^{\perp}\Psi \cdot \nabla\Delta\Psi \approx 0 \tag{2}$$

Note that, in two dimensions, the vorticity ω becomes a scalar quantity related to the stream function via the Laplace operator: $\Delta\Psi = \omega$. The inpainting problem in [9] focuses on propagating smoothness information ΔI in the direction of the isophotes $\nabla^{\perp}I$ (see 1). Let I_0 denote the original image, and let Ω represent a region within the image where inpainting is desired based on the surrounding data. For an image intensity function $I : \Omega \to \mathbb{R}$, the grayscale values within Ω are determined by the smoothness criterion within Ω and boundary values on $\partial\Omega$, rather than by restoration methods. Mathematically, this inpainting criterion is expressed as:

$$\nabla^{\perp}I \cdot \nabla\Delta I = 0 \quad \text{on} \quad \Omega \tag{3}$$

with boundary condition

$$I = I_0 \quad \text{on} \quad \partial\Omega \tag{4}$$

A generalization of Eq.(3) was proposed in [10], introducing an anisotropic diffusion term in the image I. This generalization is formulated as:

$$I_t = \nabla^{\perp}I \cdot \nabla\Delta I + \nu\nabla \cdot (g(|\nabla I|)\nabla I) \tag{5}$$

where g allows for anisotropic diffusion, preserving edges while diffusing smoothness ΔI.

3.2. Inpainting Analogy

The notable resemblance between Eq.(2) and the inpainting criterion Eq.(3) suggests an analogy between 2D incompressible fluid flow and image inpainting. This analogy is summarized in the table 1, where Ψ in fluid flow is analogous to the image matrix I in inpainting.

In image processing terms, we now present an analogue to the vorticity-stream function formulation (1):

$$\frac{\partial\omega}{\partial t} + \vec{u}.\nabla\omega = \nu\nabla.(g(|\nabla\omega|)\nabla\omega) \quad \text{on} \quad \Omega, \tag{6}$$

where g provides anisotropic diffusion of the smoothness ω or enables edge-preserving diffusion.

Table 1: Analogy between Fluid Dynamics and Image Inpainting
c

Aspect	**Fluid Dynamics**	**Image Inpainting**
Physical Quantity	Stream function (Ψ)	Image intensity (I)
Directional Flow	Fluid velocity ($u = \nabla^{\perp}\Psi$)	Isophote direction ($\nabla^{\perp} I$)
Smoothness Operator	Vorticity ($\omega = \Delta\Psi$)	Smoothness ($\omega = \Delta I$)
Diffusion Coefficient	Viscosity (ν)	Anisotropic diffusion (ν)

The image intensity I, which determines the velocity field $\vec{u} = \nabla^{\perp} I$, can be recovered by simultaneously solving the Poisson problem:

$$\Delta I = \omega, \quad \text{on} \quad \Omega, \quad I|_{\partial\Omega} = I_0. \tag{7}$$

For fluid problems with low viscosity ν, stabilising this formulation can be time-consuming, which may restrict its practical applicability. A viable alternative is to adopt a pseudo-steady approach, replacing the Poisson equation (7) with a dynamic relaxation method based on the heat equation:

$$\frac{\partial I}{\partial t} - \alpha(\Delta I + \omega) = 0, \quad \alpha > 0, \quad I|_{\partial\Omega} = I_0, \tag{8}$$

where α, the thermal diffusivity, serves as a relaxation parameter that regulates the rate of convergence.

3.3. Boundary Conditions

In the Navier-Stokes-based inpainting method, continuity across the boundary is inherently preserved. For instance, when the boundary condition is defined by setting the velocity field as $\vec{u} = \nabla^{\perp} I$ along $\partial\Omega$, solving the inpainting equation 6 under this constraint results not only in continuous isophotes but also in an image intensity function that remains smooth across the boundary $\partial\Omega$. Moreover, a second boundary condition of Dirichlet type is applied for the heat equation 8. Both types of boundary conditions are treated at the microscopic level using distribution functions, as described in subsection 4.4.

In the subsequent section, we rigorously recover the vorticity–stream function formulation (6) and the parabolic equation (8) via the Chapman–Enskog asymptotic expansion, utilizing the lattice Boltzmann models $\{D2Qm\}_{m=5,9}$ (refer to Figure 3).

4. Lattice Boltzmann Method Framework

The Lattice Boltzmann Method (LBM) operates on a discretized phase space defined over a uniform $DnQm$ lattice, where m denotes the discrete velocity directions in an n-dimensional spatial domain. For two-dimensional problems, the $D2Q5$ and $D2Q9$ lattice

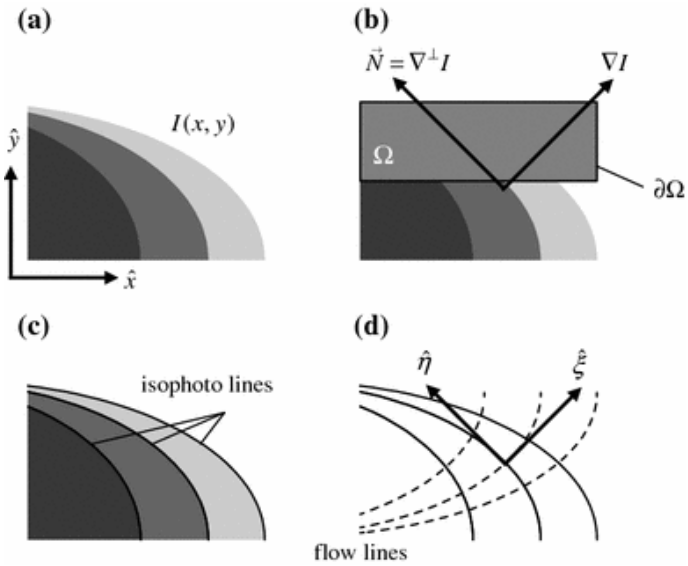


Figure 1: Illustration of the inpainting mechanism where the smoothness information ΔI propagates along the isophote directions $\nabla^{\perp} I$, from the known boundary $\partial\Omega$ toward the interior of the missing region Ω.

configurations are employed, as they are well-suited for modeling advection-diffusion and heat transfer phenomena. A summary of these discrete velocity models is presented in Table 3.

4.1. Macroscopic Equations Derived via Chapman–Enskog Expansion

To derive the macroscopic advection-diffusion equation for vorticity from the mesoscopic lattice Boltzmann formulation, we begin by considering the target form:

$$\frac{\partial \omega(x,t)}{\partial t} + \vec{u} \cdot \nabla \omega(x,t) = \nu \nabla \cdot (k \nabla \omega(x,t)), \tag{9}$$

where $\omega(x,t)$ represents the vorticity field, $\vec{u}$ is the velocity, ν is the kinematic viscosity, and $k(x)$ is a positive, spatially varying diffusion coefficient.

The underlying evolution of the distribution function f_i in the lattice Boltzmann method is governed by:

$$f_i(x + c_i \Delta t, t + \Delta t) = f_i(x,t) - \frac{1}{\tau}(f_i(x,t) - f_i^{(eq)}(x,t)), \quad i = 0, \ldots, m-1, \tag{10}$$

which can be decomposed into two separate steps: **Collision:**

$$f_i^*(x,t) = f_i(x,t) - \frac{1}{\tau}(f_i(x,t) - f_i^{(eq)}(x,t)), \tag{11}$$

Propagation:

$$f_i(x + c_i \Delta t, t + \Delta t) = f_i^*(x,t). \tag{12}$$

Here, f_i and f_i^* denote the pre- and post-collision distribution functions, respectively, c_i is the discrete lattice velocity in direction i, and τ is the relaxation time.

The equilibrium distribution function is defined as:

$$f_i^{(eq)} = \delta_i \omega(x,t) \left(1 + \frac{c_i \cdot \vec{u}}{c_s^2}\right), \tag{13}$$

where δ_i denotes the lattice weights and c_s is the lattice sound speed.

The weights δ_i are given as follows:

- **D2Q5 model:** $\delta_i = \frac{1}{5}, \quad i = 0, 1, \ldots, 4$
- **D2Q9 model:**

$$\delta_i = \begin{cases} \frac{4}{9} & i = 0, \\ \frac{1}{9} & i = 1,2,3,4, \\ \frac{1}{36} & i = 5,6,7,8. \end{cases} \tag{14}$$

The discrete velocities c_i are defined as:

- **D2Q5 model:**

$$c_i = \begin{cases} (0,0) & i = 0, \\ \sqrt{2}c\left(\cos\left[(2i-1)\frac{\pi}{4}\right], \sin\left[(2i-1)\frac{\pi}{4}\right]\right) & i = 1,2,3,4. \end{cases} \tag{15}$$

- **D2Q9 model:**

$$c_i = \begin{cases} (0,0) & i = 0, \\ c\left(\cos\left[(i-1)\frac{\pi}{2}\right], \sin\left[(i-1)\frac{\pi}{2}\right]\right) & i = 1,2,3,4, \\ \sqrt{2}c\left(\cos\left[(2i-9)\frac{\pi}{4}\right], \sin\left[(2i-9)\frac{\pi}{4}\right]\right) & i = 5,6,7,8. \end{cases} \tag{16}$$

We also enforce the following moment relations:

$$\sum_i f_i = \sum_i f_i^{(eq)} = \omega(x,t), \tag{17}$$

$$\sum_i c_i f_i^{(eq)} = \omega(x,t)\vec{u}, \tag{18}$$

$$\sum_i c_i c_i f_i^{(eq)} = c_s^2 \omega(x,t) I, \tag{19}$$

where I denotes the identity matrix and $c_s^2 = \frac{2}{5}c^2$.

To obtain the macroscopic behavior, we employ the Chapman–Enskog expansion:

$$f_i = f_i^{(0)} + \varepsilon f_i^{(1)} + \varepsilon^2 f_i^{(2)}, \quad \partial_t = \varepsilon \partial_{t_1} + \varepsilon^2 \partial_{t_2}, \quad \nabla = \varepsilon \nabla_1, \tag{20}$$

with $f_i^{(0)} = f_i^{(eq)}$.

Applying a Taylor expansion to Eq. (10) yields:

$$f_i(x + c_i\Delta t, t + \Delta t) - f_i(x,t) = \sum_{n=1}^{2} \frac{\Delta t^n}{n!}(\partial_t + c_i \cdot \nabla)^n f_i(x,t) + \mathcal{O}(\Delta t^3). \tag{21}$$

Substituting Eqs. (20) and (21) into Eq. (10) and grouping terms by orders of ε, we obtain:

$$\partial_{t_1} f_i^{(0)} + c_i \cdot \nabla_1 f_i^{(0)} = -\frac{1}{\tau\Delta t} f_i^{(1)}, \tag{22}$$

$$\partial_{t_2} f_i^{(0)} + (\partial_{t_1} + c_i \cdot \nabla_1) f_i^{(1)} + \frac{\Delta t}{2}(\partial_{t_1} + c_i \cdot \nabla_1)^2 f_i^{(0)} = -\frac{1}{\tau\Delta t} f_i^{(2)}. \tag{23}$$

Defining $\mathcal{D}_{1i} = \partial_{t_1} + c_i \cdot \nabla_1$, Eq. (22) becomes:

$$\mathcal{D}_{1i} f_i^{(0)} = -\frac{1}{\tau\Delta t} f_i^{(1)}. \tag{24}$$

Substituting Eq. (24) into Eq. (23) yields:

$$\partial_{t_2} f_i^{(0)} + \left(1 - \frac{1}{2\tau}\right) \mathcal{D}_{1i} f_i^{(1)} = -\frac{1}{\tau\Delta t} f_i^{(2)}. \tag{25}$$

Summing Eq. (25) over i, and using Eq. (18), we derive:

$$\partial_{t_2}\omega + \nabla_1 \cdot \left[\left(1 - \frac{1}{2\tau}\right) \sum_i c_i f_i^{(1)}\right] = 0. \tag{26}$$

Applying Eq. (24) and using moment properties, we obtain:

$$\sum_i c_i f_i^{(1)} = -\tau\Delta t \partial_{t_1}\omega\vec{u} - \tau\Delta t \nabla_1(c_s^2\omega). \tag{27}$$

Combining the results, the equations at successive time scales become:

$$\partial_{t_1}\omega + \nabla_1 \cdot (\omega\vec{u}) = 0, \tag{28}$$

$$\partial_{t_2}\omega = \nabla_1 \cdot \left[(\tau - \frac{1}{2})\Delta t \partial_{t_1}\omega\vec{u}\right] + \nabla_1 \cdot \left[c_s^2(\tau - \frac{1}{2})\Delta t \nabla_1\omega\right]. \tag{29}$$

Combining both contributions and setting $\nu k = c_s^2(\tau - \frac{1}{2})\Delta t$, we recover:

$$\partial_t\omega + \vec{u} \cdot \nabla\omega = \nu\nabla \cdot (k\nabla\omega) + (\tau - \frac{1}{2})\Delta t \nabla \cdot (\partial_t\omega\vec{u}). \tag{30}$$

Since the flow is incompressible ($\nabla \cdot \vec{u} = 0$), the convective term simplifies as:

$$\nabla \cdot (\omega\vec{u}) = \vec{u} \cdot \nabla\omega. \tag{31}$$

Thus, the recovered advection equation reads:

$$\partial_t\omega + \vec{u}\cdot\nabla\omega = \nu\nabla\cdot(k\nabla\omega) + \underbrace{(\tau-\frac{1}{2})\Delta t\nabla\cdot(\partial_t\omega\vec{u})}_{\text{additional term}}. \tag{32}$$

In the advection-diffusion equation provided in (32), only terms of order $\mathcal{O}(u)$ are considered. An additional term, $(\tau-\frac{1}{2})\Delta t\,\nabla\cdot(\partial_t\omega\vec{u})$, appears in the formulation. For incompressible flows, where the velocity field satisfies the condition $\nabla\cdot\vec{u}=0$, this term simplifies further to $(\tau-\frac{1}{2})\Delta t\,\partial_t(\nabla\cdot(\omega\vec{u}))$. Under steady-state conditions in incompressible flows, this term naturally becomes zero [39].

In the case of unsteady incompressible flows, the magnitude of this additional term scales as $\mathcal{O}(\frac{u^2}{c_s^2})$. Given that simulations of incompressible flows are typically conducted with $\frac{\|\vec{u}\|}{c_s} < 0.1$, the term represents a higher-order correction that contributes negligibly to the overall dynamics. As such, it is commonly neglected without significant impact on the accuracy of the simulation [39].

4.2. Recovered Heat Equation

To derive the macroscopic heat equation from the underlying lattice Boltzmann formulation, we consider the mesoscopic evolution equation for the temperature-related distribution function $g_i(x,t)$:

$$g_i(x+c_i\Delta t, t+\Delta t) = g_i(x,t) - \frac{1}{\tau_g}(g_i(x,t)-g_i^{(eq)}(x,t)) + \alpha\Delta t S_i, \quad i=0,\ldots,m-1, \tag{33}$$

where τ_g is the relaxation time, α a scaling coefficient, and $S_i=\varpi_i\omega(x,t)$ is a source term associated with the vorticity $\omega(x,t)$. The equilibrium distribution function is taken as:

$$g_i^{(eq)}(x,t) = \varpi_i I(x,t), \quad i=0,\ldots,m-1, \tag{34}$$

with $I(x,t)$ denoting the image intensity or temperature field and ϖ_i the lattice weights.

The moments of the distribution function satisfy the following relations:

$$\sum_{i=0}^{m-1} g_i(x,t) = \sum_{i=0}^{m-1} g_i^{(eq)}(x,t) = I(x,t), \tag{35}$$

$$\sum_{i=0}^{m-1} c_i g_i^{(eq)}(x,t) = 0, \tag{36}$$

$$\sum_{i=0}^{m-1} c_i c_i g_i^{(eq)}(x,t) = c_s^2 I(x,t) I_2, \tag{37}$$

where I_2 is the identity matrix in two dimensions.

To derive the macroscopic behavior, we use a Chapman–Enskog multiscale expansion:

$$g_i = g_i^{(0)} + \varepsilon g_i^{(1)} + \varepsilon^2 g_i^{(2)}, \quad \partial_t = \varepsilon\partial_{t_1} + \varepsilon^2\partial_{t_2}, \quad \nabla = \varepsilon\nabla_1, \quad S_i = \varepsilon^2 S_i^{(2)}. \tag{38}$$

Expanding Eq. (33) using Taylor series:

$$g_i(x + c_i\Delta t, t + \Delta t) - g_i(x,t) = \sum_{n=1}^{2} \frac{\Delta t^n}{n!}(\partial_t + c_i \cdot \nabla)^n g_i(x,t) + \mathcal{O}(\Delta t^3). \tag{39}$$

Substituting Eqs. (38) and (39) into Eq. (33) and collecting terms by ε-order yields:

$$\partial_{t_1} g_i^{(0)} + c_i \cdot \nabla_1 g_i^{(0)} = -\frac{1}{\tau_g \Delta t} g_i^{(1)}, \tag{40}$$

$$\partial_{t_2} g_i^{(0)} + (\partial_{t_1} + c_i \cdot \nabla_1) g_i^{(1)} + \frac{\Delta t}{2}(\partial_{t_1} + c_i \cdot \nabla_1)^2 g_i^{(0)} = -\frac{1}{\tau_g \Delta t} g_i^{(2)} + S_i^{(2)}. \tag{41}$$

Using the operator $\mathcal{D}_{1i} = \partial_{t_1} + c_i \cdot \nabla_1$, Eq. (40) becomes:

$$\mathcal{D}_{1i} g_i^{(0)} = -\frac{1}{\tau_g \Delta t} g_i^{(1)}. \tag{42}$$

Substituting into Eq. (41), we get:

$$\partial_{t_2} g_i^{(0)} + \left(1 - \frac{1}{2\tau_g}\right) \mathcal{D}_{1i}^2 g_i^{(0)} = -\frac{1}{\tau_g \Delta t} g_i^{(2)} + S_i^{(2)}. \tag{43}$$

Summing over all i and using Eq. (35) and properties of the equilibrium distribution, we obtain:

$$\partial_{t_1} I = 0, \tag{44}$$

$$\partial_{t_2} I + \left(1 - \frac{1}{2\tau_g}\right) \nabla_1 \cdot \left(\sum_{i=0}^{m-1} c_i g_i^{(1)}\right) = S^{(2)}, \tag{45}$$

where $S^{(2)} = \sum_i S_i^{(2)}$.

To evaluate the flux term, we use Eq. (42):

$$\sum_i c_i g_i^{(1)} = -\tau_g \Delta t \sum_i c_i \mathcal{D}_{1i} g_i^{(eq)} = -\tau_g \Delta t c_s^2 \nabla_1 I. \tag{46}$$

Substituting into Eq. (45) gives:

$$\partial_{t_2} I = c_s^2(\tau_g - \frac{1}{2})\Delta t \nabla_1^2 I + S^{(2)}. \tag{47}$$

Combining Eqs. (44) and (47), and setting $\alpha = c_s^2(\tau_g - \frac{1}{2})\Delta t$, we recover the macroscopic nonlinear heat equation:

$$\partial_t I = \alpha \nabla^2 I(x,t) + \alpha\omega(x,t). \tag{48}$$

4.3. Computation of $\nabla\omega$, τ, τ_g, and $\nabla^{\perp}I$

In the macroscopic equation (6), the coefficient $g(s)$ represents a positive, non-increasing diffusion function. Inspired by the work of Perona and Malik [40], typical choices for $g(s)$, where $s = \|\nabla\omega\|$, include:

$$g(s) = e^{-ls^2}, \quad \text{or} \quad g(s) = \frac{1}{1+ls^2}, \tag{49}$$

where $l > 0$ is a tunable parameter. In this work, we adopt the second form:

$$g(s) = \frac{1}{1+l\|\nabla\omega\|^2}. \tag{50}$$

Under the anisotropic diffusion framework where $k = g(\|\nabla\omega\|)$, the relaxation times τ and τ_g are obtained from the macro-micro relations:

$$\nu k = c_s^2\left(\tau - \frac{1}{2}\right)\Delta t, \qquad \alpha = c_s^2\left(\tau_g - \frac{1}{2}\right)\Delta t. \tag{51}$$

Hence, the expressions for τ and τ_g are:

$$\tau = \frac{1}{2} + \frac{\nu k}{c_s^2\Delta t} = \begin{cases} \dfrac{1}{2} + \dfrac{5\Delta t\nu}{2(\Delta x)^2(1+l\|\nabla\omega\|^2)} & \text{for D2Q5 lattice,} \\ \dfrac{1}{2} + \dfrac{3\Delta t\nu}{(\Delta x)^2(1+l\|\nabla\omega\|^2)} & \text{for D2Q9 lattice.} \end{cases} \tag{52}$$

$$\tau_g = \frac{1}{2} + \frac{\alpha}{c_s^2\Delta t} = \begin{cases} \dfrac{1}{2} + \dfrac{5\Delta t}{2(\Delta x)^2}\alpha & \text{for D2Q5 lattice,} \\ \dfrac{1}{2} + \dfrac{3\Delta t}{(\Delta x)^2}\alpha & \text{for D2Q9 lattice.} \end{cases} \tag{53}$$

To compute τ from Eq. (52), one needs to first evaluate the vorticity gradient $\nabla\omega$. This is computed using a central finite difference scheme depending on the lattice model:

$$\nabla\omega(x,t) = \begin{cases} \dfrac{1}{2\Delta x}\displaystyle\sum_{i=0}^{4} e_i\omega(x+c_i\Delta t), & \text{for D2Q5 lattice,} \\ \dfrac{1}{6\Delta x}\displaystyle\sum_{i=0}^{8} e_i\omega(x+c_i\Delta t), & \text{for D2Q9 lattice,} \end{cases} \tag{54}$$

where $e_i = \frac{c_i}{c}$.

To evaluate the image gradient ∇I, we recall from Eq. (46) that:

$$\nabla_1 I = -\frac{1}{\tau_g\Delta t c_s^2}\sum_{i=0}^{m-1} c_i g_i^{(1)}. \tag{55}$$

Assuming $\varepsilon g_i^{(1)} \approx g_i - g_i^{(eq)}$, and substituting from Eq. (34), we obtain:

$$\nabla I = -\frac{1}{\tau_g \Delta t c_s^2} \sum_{i=0}^{m-1} c_i g_i(x), \tag{56}$$

where we used the fact that $\sum_i c_i g_i^{(eq)} = 0$ due to symmetry.

The perpendicular gradient $\nabla^\perp I$ is then given by:

$$\nabla^\perp I = -\frac{1}{\tau_g \Delta t c_s^2} \sum_{i=0}^{m-1} c_i^\perp g_i(x), \quad x \in \Omega, \tag{57}$$

where $c_{\bar{i}}^\perp$ denotes the velocity vector perpendicular to c_i.

4.4. Curved Boundary Conditions

In partial differential equation (PDE)-based image inpainting, boundary conditions play a pivotal role in governing how information propagates into the inpainting domain. In this work, the region to be inpainted, denoted Ω, is manually selected and often features a nontrivial, curved boundary $\partial\Omega$ that conforms closely to the underlying image structures.

To ensure a coherent and stable reconstruction, two boundary conditions are imposed. First, a velocity field is prescribed on the boundary by enforcing $\vec{u} = \nabla^\perp I|_{\partial\Omega}$, which serves as a guiding advection flow along $\partial\Omega$. This condition regulates the directional transport of isophote information into the missing region. Second, a Dirichlet condition $I|_{\partial\Omega} = I_0$ is applied to the nonlinear heat equation, thereby anchoring the intensity values on the boundary to those of the known image.

Together, these boundary constraints ensure that the inpainting process yields a smooth and seamless transition between the reconstructed and known regions, preserving the structural and tonal coherence of the original image.

4.5. Non-Equilibrium Extrapolation Method for Curved Boundaries

Curved boundary conditions are treated using the non-equilibrium extrapolation technique. In this approach, the distribution functions at boundary nodes x_b are decomposed into equilibrium and non-equilibrium components:

$$f_{\bar{i}}(x_b, t) = f_{\bar{i}}^{(eq)}(x_b, t) + f_{\bar{i}}^{(neq)}(x_b, t), \tag{58}$$

where the equilibrium part $f_{\bar{i}}^{(eq)}(x_b, t)$ is approximated by:

$$f_{\bar{i}}^{(eq)}(x_b, t) \approx f_{\bar{i}}^{*}(x_b, t) \equiv \delta_{\bar{i}}\, \omega(x_b) \left(1 + \frac{\vec{c}_{\bar{i}} \cdot \vec{u}(x_b)}{c_s^2}\right), \quad i = 0, \ldots, m-1, \tag{59}$$

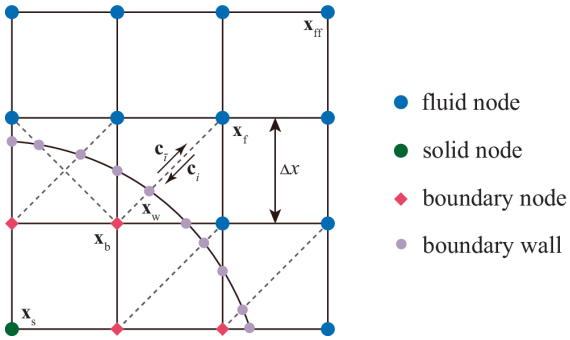


Figure 2: Schematic diagram of the node structure used in the Lattice Boltzmann framework. The domain includes fluid nodes (x_f, x_{ff}), solid nodes (x_s), boundary nodes (x_b), and wall boundary nodes (x_w). The lattice spacing is denoted by Δx, and $\vec{c}_i$ represents the discrete velocity in the i-th direction, with $\vec{c}_{\bar{i}}$ as its opposite counterpart.

where $\omega(x_b)$ and $\vec{u}(x_b)$ denote the vorticity function and velocity vector at node x_b, respectively. Since these quantities are generally not known a priori, Guo et al. [41] proposed the following extrapolation scheme:

$$f_{\bar{i}}^{(neq)}(x_b,t) = \begin{cases} f_{\bar{i}}(x_f,t) - f_{\bar{i}}^{(eq)}(x_f,t), & q \geq \dfrac{3}{4}, \\ q\left(f_{\bar{i}}(x_f,t) - f_{\bar{i}}^{(eq)}(x_f,t)\right) + (1-q)\left(f_{\bar{i}}(x_{ff},t) - f_{\bar{i}}^{(eq)}(x_{ff},t)\right), & q < \dfrac{3}{4}, \end{cases} \tag{60}$$

where $q = \dfrac{|x_f - x_w|}{|x_f - x_b|} = \dfrac{|x_f - x_w|}{\Delta x}$, and $\omega(x_b) = \omega(x_f)$. The velocity $\vec{u}(x_b)$ is estimated by:

$$\vec{u}(x_b) = \begin{cases} \dfrac{1}{q}\vec{u}(x_w) + \dfrac{q-1}{q}\vec{u}(x_f), & q \geq \dfrac{3}{4}, \\ \dfrac{3-q}{q+1}\vec{u}(x_w) + (q-1)\vec{u}(x_f) + (1-q)\dfrac{q-1}{q+1}\vec{u}(x_{ff}), & q < \dfrac{3}{4}. \end{cases} \tag{61}$$

In the fluid region Ω, the velocity field $\vec{u}$, corresponding to the inpainted region, is computed from the gradient of the auxiliary function g_i as follows:

$$\vec{u}(x_f) = -\frac{1}{\tau_g \Delta t c_s^2} \sum_{i=0}^{m-1} \vec{c}_i^{\perp} \, g_i(x_f), \tag{62}$$

$$\vec{u}(x_{ff}) = -\frac{1}{\tau_g \Delta t c_s^2} \sum_{i=0}^{m-1} \vec{c}_i^{\perp} \, g_i(x_{ff}). \tag{63}$$

At the wall boundary $\partial\Omega$, the velocity $\vec{u}(x_w)$ is evaluated via a discrete transverse gradient of the initial image I_0:

$$\vec{u}(x_w) = \nabla^{\perp} I|_{\partial\Omega} = \begin{cases} \dfrac{1}{2\Delta x} \displaystyle\sum_{i=0}^{4} \vec{e}_i^{\perp} I_0(x_w + \vec{c}_i \Delta t), \\ \dfrac{1}{6\Delta x} \displaystyle\sum_{i=0}^{8} \vec{e}_i^{\perp} I_0(x_w + \vec{c}_i \Delta t), \end{cases} \tag{64}$$

depending on whether a $D2Q5$ or $D2Q9$ stencil is used.

Using the above approximations, the post-collision distribution function at the boundary node x_b is given by:

$$f_{\bar{i}}^{'}(x_b, t) = f_{\bar{i}}(x_b, t) - \frac{1}{\tau} f_{\bar{i}}^{(neq)}(x_b, t) = f_{\bar{i}}^{*}(x_b, t) + \frac{\tau - 1}{\tau} f_{\bar{i}}^{(neq)}(x_b, t). \tag{65}$$

For scalar transport with Dirichlet boundary conditions, the distribution function $g_{\bar{i}}(x_b, t)$ is treated analogously:

$$g_{\bar{i}}(x_b, t) = g_{\bar{i}}^{(eq)}(x_b, t) + g_{\bar{i}}^{(neq)}(x_b, t), \tag{66}$$

where the equilibrium part is approximated by:

$$g_{\bar{i}}^{(eq)}(x_b, t) \approx g_{\bar{i}}^{*}(x_b, t) \equiv \varpi_i I(x_f), \quad i = 0, \ldots, m-1, \tag{67}$$

and the non-equilibrium part is extrapolated as:

$$g_{\bar{i}}^{(neq)}(x_b, t) = \begin{cases} g_{\bar{i}}(x_f, t) - g_{\bar{i}}^{(eq)}(x_f, t), & q \geq \dfrac{3}{4}, \\ q\left(g_{\bar{i}}(x_f, t) - g_{\bar{i}}^{(eq)}(x_f, t)\right) + (1 - q)\left(g_{\bar{i}}(x_{ff}, t) - g_{\bar{i}}^{(eq)}(x_{ff}, t)\right), & q < \dfrac{3}{4}. \end{cases} \tag{68}$$

Thus, the post-collision distribution for the scalar field becomes:

$$g_{\bar{i}}^{'}(x_b, t) = g_{\bar{i}}^{*}(x_b, t) + \frac{\tau_g - 1}{\tau_g} g_{\bar{i}}^{(neq)}(x_b, t). \tag{69}$$

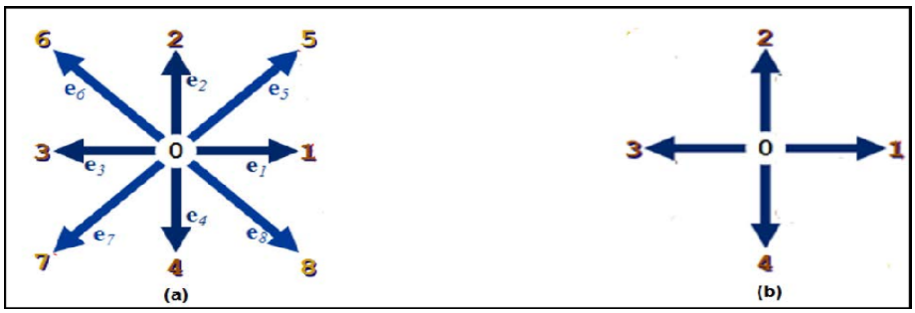


Figure 3: Lattice configurations: (a) $D2Q9$, (b) $D2Q5$.

4.6. implementation of the Algorithm

To facilitate the understanding of the proposed Lattice Boltzmann inpainting framework, all relevant parameters and their respective roles are summarised in the following two tables. Table 2 lists structural and numerical variables, while Table 3 gathers physical quantities and functional coefficients involved in the image reconstruction process.

Table 2: Structural and numerical parameters involved in the LBM-based inpainting algorithms.

Symbol	**Name**	**Description**
I_0	Initial image	Input grayscale or RGB image to be restored
M	Inpainting mask	Binary matrix: 1 inside the missing region, 0 elsewhere
I	Inpainted image	Output image updated through iterative reconstruction
f_i	Vorticity distribution	LBM distribution function for vorticity
g_i	Intensity distribution	LBM distribution function for intensity
$f_i^{(eq)}$	Equilibrium (vorticity)	Local equilibrium for ω
$g_i^{(eq)}$	Equilibrium (intensity)	Local equilibrium for I
c_i	Lattice velocities	Discrete propagation directions (D2Q5/D2Q9)
c_s	Sound speed	LBM constant for viscosity/diffusivity scaling
δ_i, ϖ_i	Lattice weights	Weights in equilibrium formulations
N	Number of iterations	Fixed stopping criterion for simulation
Δt	Time step	Time increment (set to 1)
Δx, Δy	Grid spacing	Spatial resolution (set to 1)

Table 3: Physical and functional parameters governing the behaviour of the LBM inpainting model.

Symbol	**Name**	**Description**
ω	Vorticity field	Encodes local image smoothness flow
$v = \nabla^{\perp} I$	Guidance field	Isophote direction used in transport
S_i	Source term	Forcing term derived from ω for the heat equation
τ	Relaxation time for ω	Controls diffusion rate via viscosity ν
τ_g	Relaxation time for I	Controls thermal behaviour via diffusivity α
ν	Kinematic viscosity	Regulates the smoothing of vorticity
α	Thermal diffusivity	Controls speed of convergence in I evolution
k	Diffusion function	Anisotropic weight for edge preservation

Algorithm 1 Proposed Lattice Boltzmann Method Algorithm for Grayscale Image Inpainting

Data: Initial image I_0, binary mask M indicating the inpainting region
Result: Reconstructed image I

Initialization:

Define the parameters of lattice DnQm ; // Δx, Δt, $c = \frac{\Delta x}{\Delta t}$, velocities c_i, w_i and c_s

$I \leftarrow I_0$; // Copy initial image

$g_i(x)^{(0)} \leftarrow g_i^{eq}(x, 0)$; // Initialize g with equilibrium

$\omega^{(0)} \leftarrow 0$, $f_i^{(0)} \leftarrow 0$; // Initialize distributions

$v \leftarrow \nabla^{\perp} I$; // Perpendicular gradient (guidance field)

for $n = 1$ **to** N **do**

  foreach *pixel* x *where* $M(x) = 1$ **do**

    Compute $\omega^{(n)}(x) \leftarrow \sum_{i=0}^{4} f_i^{(n)}(x)$ Compute $\nabla\omega^{(n)}(x)$ using Eq. (54) Compute relaxation time $\tau^{(n)}(x)$ using Eq. (52)

  end

  Apply boundary conditions using Eq. (65) and Eq. (69)

  foreach *pixel* x *where* $M(x) = 1$ **do**

    $f_i^{(n)}(x) \leftarrow f_i^{eq(n)}(x)$; // Equilibrium initialization

    Collision step:

    $f_i^{(n)}(x) \leftarrow f_i^{(n)}(x) - \frac{1}{\tau^{(n)}(x)} \left(f_i^{(n)}(x) - f_i^{eq(n)}(x) \right)$ $g_i^{(n)}(x) \leftarrow g_i^{(n)}(x) - \frac{1}{\tau_g} \left(g_i^{(n)}(x) - g_i^{eq(n)}(x) \right) + \alpha \Delta t \varpi_i \omega^{(n)}(x)$

    Streaming step:

    $f_i^{(n+1)}(x + c_i \Delta t) \leftarrow f_i^{(n)}(x)$ $g_i^{(n+1)}(x + c_i \Delta t) \leftarrow g_i^{(n)}(x)$

    Reconstruction:

    $I^{(n+1)}(x) \leftarrow \sum_{i=0}^{4} g_i^{(n+1)}(x)$

  end

end

$I \leftarrow I^{(N+1)}(x)$; // Final inpainted image

Algorithm 2 Proposed Lattice Boltzmann Method Algorithm for Inpainting Colored Images

Input: Initial color image I_0 with 3 channels (RGB), mask M for each channel
Output: Inpainted color image I

Initialization:

Define the parameters lattice DnQm ; // Δx, Δt, $c = \frac{\Delta x}{\Delta t}$, velocities c_i, w_i and c_s

$I \leftarrow I_0$; // Copy initial image into working variable

$g_i(x)^{(0)} \leftarrow g_i^{eq}(x, 0)$; // Initialize auxiliary distribution function with equilibrium

$\omega^{(0)} \leftarrow 0$, $f_i^{(0)} \leftarrow 0$; // Initialize macroscopic and microscopic distributions

$v \leftarrow \nabla^{\perp} I$; // Compute perpendicular gradient (guidance field)

for $k = 1$ **to** 3 // Loop over R, G, B channels **do**

  Apply Algorithm 1 to channel $I(:, :, k)$; // Inpaint current channel using LBM

end

return Inpainted color image I ; // Final reconstructed RGB image

5. Experimental results and discussion

The proposed image inpainting algorithm was evaluated using a total of six test images, including three grayscale images (*Cameraman*, *Clown*, *Girlface*) and three colour images (*Boat*, *Bridge*, *El Jadida City*). To simulate degradation, random lines were superimposed on the original images, mimicking scratches or missing regions. The inpainting was performed using the Lattice Boltzmann Method (LBM) based on the D2Q5 and D2Q9 lattice models, (see Fig.3).

To benchmark the performance of the proposed LBM-based inpainting method, we compare it against two well-established classical approaches: the **Total Variation (TV)** regularisation method and the **Bertalmio–Sapiro–Caselles–Ballester (BSCB)** algorithm. The TV model, introduced by Rudin, Osher, and Fatemi, represents a **variational framework** that effectively preserves edges while smoothing homogeneous regions. Meanwhile, the BSCB method is one of the earliest inpainting techniques inspired by **fluid dynamics**, using Navier–Stokes-based transport along isophotes. These two methods have become canonical baselines in the literature, with TV exemplifying **geometric regularisation** and BSCB reflecting the **physical modeling perspective**. Their inclusion provides a meaningful contrast for evaluating the **reconstruction quality**, **computational efficiency**, and **structural coherence** of our LBM framework.

The restored images were compared with those obtained using BSCB and TV approaches. All implementations were carried out in Matlab 2018 on a Windows 10 operating system. The quality of the restored images was quantitatively assessed using the Peak Signal-to-Noise Ratio (PSNR) and the Structural Similarity Index Measure (SSIM) . The PSNR is a standard metric that evaluates the fidelity of the restored image relative to the

original, defined by:

$$\mathrm{PSNR}(t) = 10\log_{10}\left(\frac{mn}{\sum_{(i,j)=(1,1)}^{(m,n)}[I((i,j),t) - I_0(i,j)]^2}\right),$$

where m and n denote the dimensions of the image, $I((i,j),t)$ is the pixel value of the inpainted image at time t, and $I_0(i,j)$ is the pixel value of the original image.

The SSIM metric evaluates the similarity between two images by incorporating perceptual factors such as luminance, contrast, and structural information. It is given by:

$$\mathrm{SSIM}(X,Y) = \left[\frac{2\mu_X\mu_Y + C_1}{\mu_X^2 + \mu_Y^2 + C_1}\right] \cdot \left[\frac{2\sigma_{XY} + C_2}{\sigma_X^2 + \sigma_Y^2 + C_2}\right] \cdot \left[\frac{\sigma_{XY}}{\sqrt{\sigma_X^2\sigma_Y^2}}\right],$$

where μ_X and μ_Y are the means, σ_X^2 and σ_Y^2 are the variances, and σ_{XY} is the covariance of local image patches from images X and Y. The constants C_1 and C_2 are stabilizing terms defined as $C_1 = (K_1L)^2$ and $C_2 = (K_2L)^2$, with $L = 255$ representing the dynamic range of pixel intensities, and $K_1 = 0.01$, $K_2 = 0.03$.

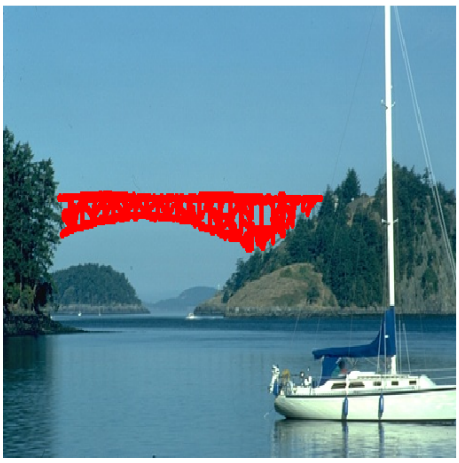
In our numerical experiments, the proposed algorithm was applied to both grayscale and colour images and compared against two classical inpainting techniques: Total Variation (TV) regularisation and the Bertalmio–Sapiro–Caselles–Ballester (BSCB) algorithm. To ensure consistency across all methods, a uniform spatial discretisation was adopted, with $\Delta x = \Delta y = 1$. For the TV-based inpainting approach, the time step Δt was selected from the set $\{0.5, 1\}$, and the stopping criterion was based on a convergence tolerance $\epsilon \in \{0.01, 0.001, 0.0008\}$, depending on the required reconstruction accuracy. In contrast, the BSCB method used a time step $\Delta t \in \{0.1, 0.5\}$, chosen to ensure numerical stability and structural coherence. For the Lattice Boltzmann Method (LBM), a standard configuration with $\Delta x = \Delta y = 1$ and $\Delta t = 1$ was employed, as commonly adopted in LBM-based schemes. Unlike energy-minimisation-based iterative methods, the stopping condition for the LBM algorithm was based on a fixed number of iterations N_{iter}, generally ranging between 100 and 2000 depending on the image resolution and the extent of the inpainting region. In the LBM framework, the parameters ν and α play crucial roles in controlling the diffusion dynamics. The kinematic viscosity ν, which governs the spread of vorticity in the advection-diffusion equation, depends on both the relaxation time τ and the spatially varying diffusion function k, which is often designed to preserve edge information through anisotropic smoothing. On the other hand, the thermal diffusivity α, which regulates the temporal evolution of the image intensity field, is determined by the relaxation time τ_g in the thermal LBM scheme. For numerical stability, both relaxation times must satisfy the condition $\tau > \frac{1}{2}$ and $\tau_g > \frac{1}{2}$, which ensures the positivity of the physical diffusion coefficients and prevents unphysical oscillations during the simulation. Proper calibra-

tion of these parameters is essential to achieve a balance between fast convergence, detail preservation, and stable inpainting dynamics.

(a) cameraman

(d) boat

(b) clown

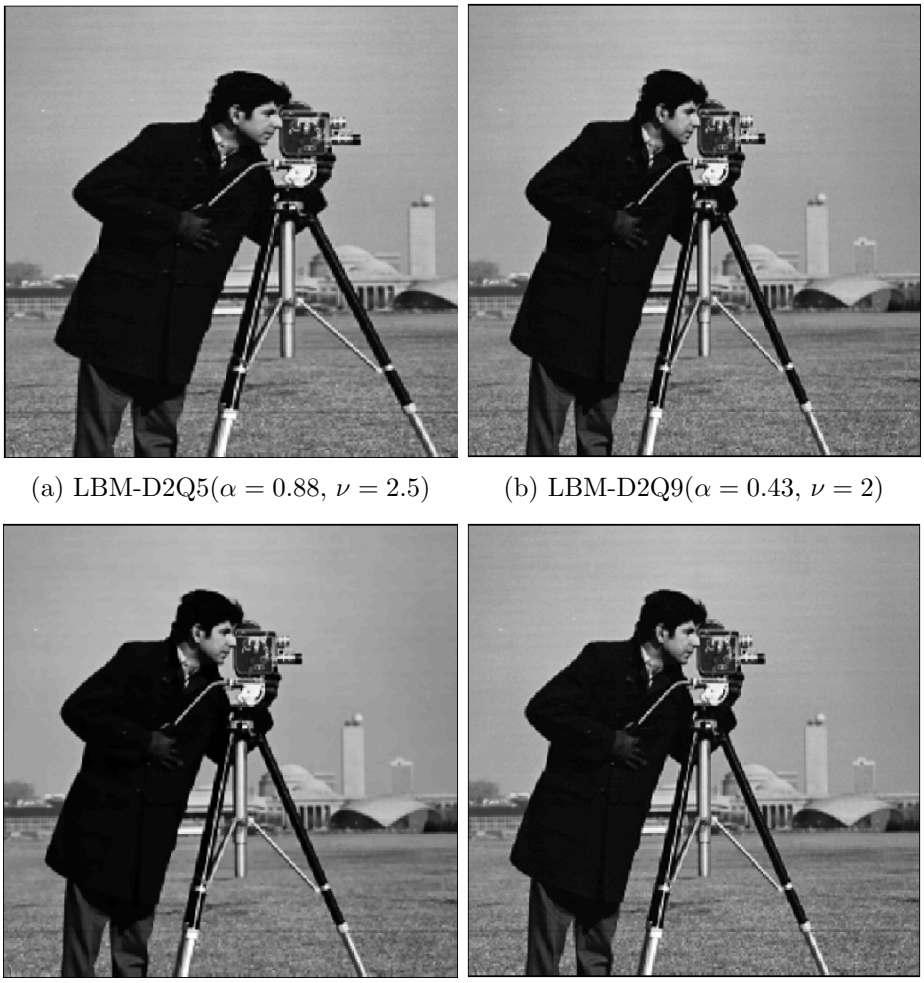


(e) bridge

(c) girlface

(f) El Jadida city

Figure 4: The grid of 3 × 2 masked images includes grayscale images in the left column and color images in the right column.



(a) LBM-D2Q5($\alpha = 0.88$, $\nu = 2.5$) (b) LBM-D2Q9($\alpha = 0.43$, $\nu = 2$)

(c) BSCB Method: $\Delta t = 0.1$ (d) TV Regularization :$\epsilon = 10^{-3}$, $\Delta t = 1$

Figure 5: Inpainting comparison on the *Cameraman* grayscale image corrupted by a synthetic random line simulating a scratch. The missing region was restored using four different techniques. (a)–(b): reconstructions via the proposed LBM method with D2Q5 and D2Q9 lattices; (c): BSCB algorithm; (d): Total Variation (TV) regularisation. LBM-based methods show sharper edge continuity and better restoration of the corrupted region while requiring significantly less computation time.

(a) LBM-D2Q5 ($\alpha = 0.16$, $\nu = 2.7$) (b) LBM-D2Q9($\alpha = 0.23$, $\nu = 0.9$)

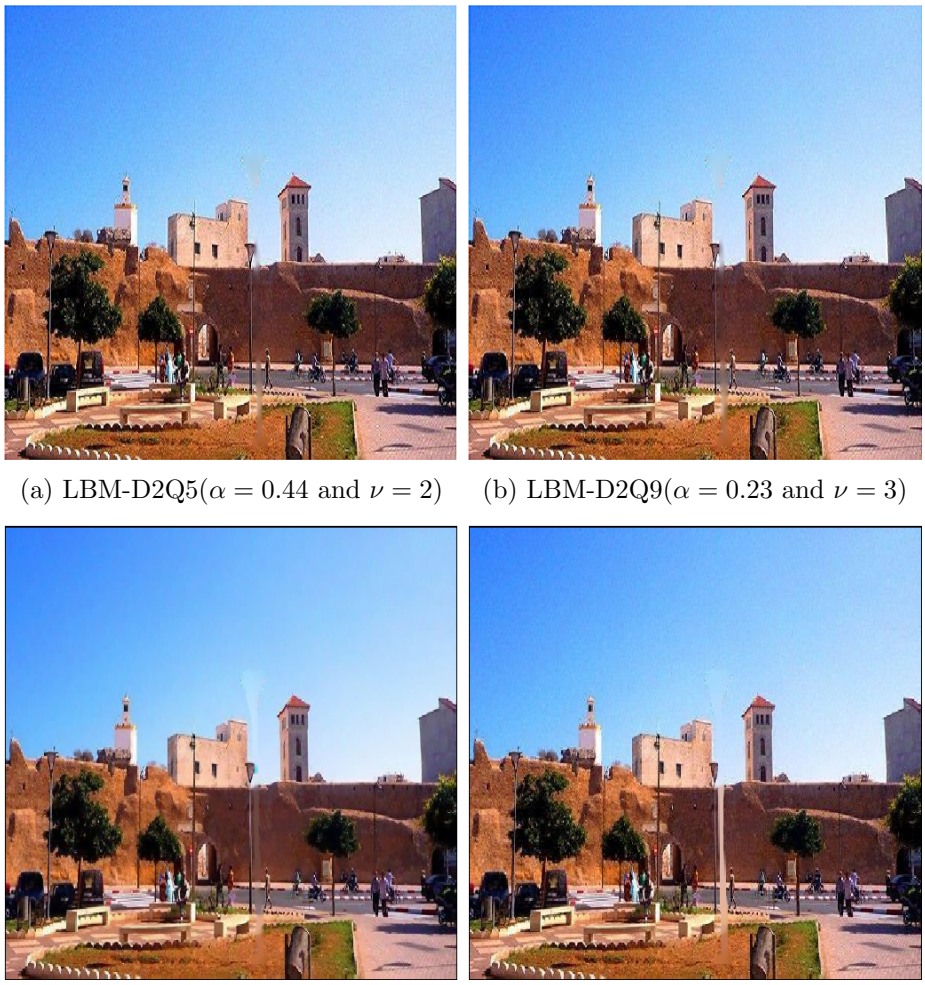
(c) BSCB method: $\Delta t = 0.1$ (d) TV regularization $\epsilon = 10^{-3}$, $\Delta t = 1$

Figure 6: Inpainting results for the *Girlface* grayscale image artificially damaged by randomly oriented lines. The missing regions are reconstructed using four different algorithms. (a)–(b) show the results obtained with the proposed LBM approach using D2Q5 and D2Q9 lattices. (c) and (d) correspond to the BSCB and TV regularisation methods, respectively. The LBM-D2Q9 model offers enhanced restoration quality, particularly in facial contours, with minimal blurring and improved structural consistency.

(a) LBM-D2Q5($\alpha = 0.36$, $\nu = 1$) (b) LBM-D2Q9 ($\alpha = 0.43$, $\nu = 2.7$)

(c) BSCB Method: $\Delta t = 0.1$ (d) TV Regularization: $\epsilon = 8.10^{-4}$ and $\Delta t = 1$

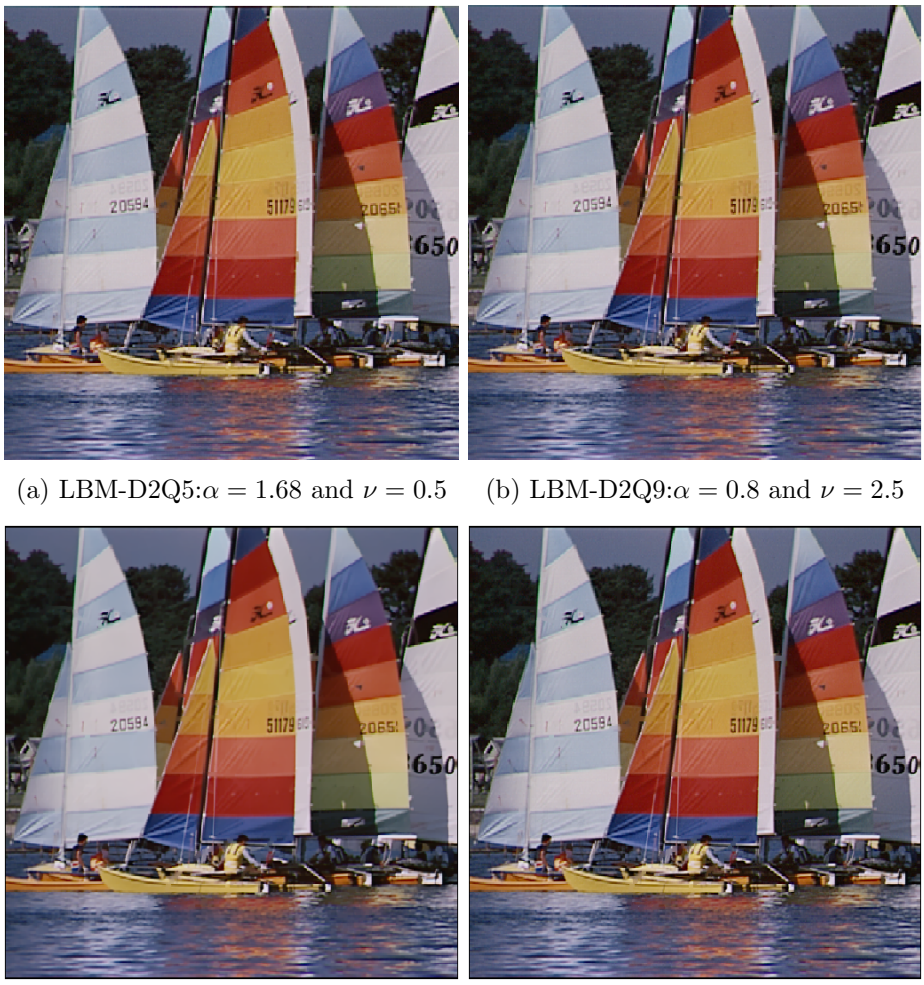
Figure 7: Inpainting results for the *Clown* grayscale image artificially damaged by a magenta-colored line simulating a severe scratch. (a)–(b): reconstructions using the proposed LBM method with D2Q5 and D2Q9 lattices; (c)–(d): results from BSCB and TV-based techniques. LBM approaches, especially D2Q9, exhibit superior capability in restoring textures and curved contours with minimal artifacts.



(a) LBM-D2Q5($\alpha = 0.44$ and $\nu = 2$) (b) LBM-D2Q9($\alpha = 0.23$ and $\nu = 3$)

(c) BSCB Method: $\Delta t = 0.5$ (d) TV Regularization :$\Delta t = 1$ and $\epsilon = 0.01$

Figure 8: Object removal results on the *El Jadida City* colour image. A building structure was manually masked to simulate a missing region. (a)–(b): inpainting results using the proposed LBM method with D2Q5 and D2Q9 lattices. (c): BSCB method; (d): Total Variation (TV) regularisation. The LBM approaches achieve structurally consistent and visually plausible restorations, particularly in complex regions such as architectural lines and sky gradients.



(a) LBM-D2Q5:$\alpha = 1.68$ and $\nu = 0.5$ (b) LBM-D2Q9:$\alpha = 0.8$ and $\nu = 2.5$

(c) BSCB method with $\Delta t = 0.1$ (d) TV Regularization : $\epsilon = 0.01$ and $\Delta t = 0.5$

Figure 9: Inpainting results for the *Boat* color image corrupted by a red-colored line simulating a structural defect. (a)–(b): Reconstructions using the proposed LBM method with D2Q5 and D2Q9 lattice models, respectively. (c): Result obtained using the BSCB algorithm. (d): Output from the TV regularization method. The LBM-D2Q5 model achieves the highest PSNR, effectively restoring fine details and textures with minimal artifacts and significantly lower computation time compared to classical approaches.

(a) LBM-D2Q5: $\alpha = 0.48$ and $\nu = 1$

(b) LBM-D2Q9: $\alpha = 0.13$ and $\nu = 2$

(c) BSCB method: $\Delta t = 0.5$

(d) TV Regularization: $\epsilon = 0.01$ and $\Delta t = 0.5$

Figure 10: Object removal results on the *Bridge* colour image. A structural component was manually masked to simulate a missing region. (a)–(b): inpainting results using the proposed LBM algorithm with D2Q5 and D2Q9 lattice models. (c): BSCB method; (d): Total Variation (TV) regularisation. The LBM-based approaches effectively restore fine contours and background textures, yielding more visually coherent reconstructions than classical methods.

The results in Tables 4 and 5 demonstrate the superior performance of the Lattice Boltzmann Method (LBM) compared to traditional methods like Total Variation regularisation (TV reg) and the Bertalmio-Sapiro-Caselles-Ballaster (BSCB) algorithm. For grayscale images, LBM achieved comparable or slightly better PSNR and SSIM values while significantly reducing computational time. For instance, the Cameraman image processed with the LBM-D2Q5 model achieved a PSNR of **25.30 dB** in just **0.93 minutes**,

Table 4: Quantitative performance comparison for grayscale image inpainting across different methods. Metrics include PSNR (dB), SSIM, and CPU Time (in minutes) for the *Cameraman*, *Girlface*, and *Clown* images. The LBM-based approaches (D2Q5 and D2Q9) consistently yield comparable or superior reconstruction quality while significantly reducing computational cost compared to classical methods (TV regularization and BSCB).

Image	Method	PSNR (dB)	SSIM	CPU Time (min)
4*Cameraman	TV reg	25.28	**0.999**	5.10
	BSCB	25.27	**0.999**	5.54
	LBM-D2Q5	25.30	**0.999**	**0.93**
	LBM-D2Q9	**25.31**	**0.999**	1.21
4*Girlface	TV reg	15.16	**0.9996**	10.73
	BSCB	15.12	**0.9996**	9.31
	LBM-D2Q5	15.16	**0.9996**	**0.62**
	LBM-D2Q9	**15.17**	**0.9996**	1.60
4*Clown	TV reg	16.47	**0.9995**	11.27
	BSCB	16.36	**0.9995**	16.76
	LBM-D2Q5	**16.48**	**0.9995**	**0.64**
	LBM-D2Q9	**16.48**	**0.9995**	1.12

compared to over **5 minutes** for both TV reg and BSCB. This efficiency was also evident for other images, such as Girlface and Clown, where LBM consistently delivered high-quality inpainting results in a fraction of the time required by classical methods.

Similarly, for colour images, the LBM methods showcased their ability to handle complex inpainting tasks efficiently. The D2Q5 model stood out, achieving the highest PSNR values with the shortest processing times. For example, the Boat image inpainted with LBM-D2Q5 reached a PSNR of **37.31 dB** in **1.50 minutes**, compared to **14.36 minutes** for TV reg (PSNR: **35.25 dB**) and **19.48 minutes** for BSCB (PSNR: **33.91 dB**). The D2Q9 model also performed well, particularly for higher relaxation times, highlighting the adaptability of LBM to different conditions. These results confirm LBM as a robust and efficient approach for image inpainting, offering significant improvements in computational speed and quality over traditional methods.

The figure 4 highlights the influence of the thermal diffusivity parameter α on the temporal evolution of PSNR for the "Girlface" image over 100 iterations using the Lattice Boltzmann Method (LBM). Smaller α values enable faster convergence and higher final PSNR, indicating improved inpainting quality due to controlled diffusion that preserves image details. Conversely, larger α values result in slower convergence and reduced inpainting performance, likely due to excessive smoothing that diminishes image fidelity. This underscores the importance of optimising α for achieving high-quality inpainting results.

Table 5: Performance metrics for color image inpainting on three datasets: *El Jadida City*, *Boat*, and *Bridge*. The LBM-D2Q5 method demonstrates superior PSNR values and drastically reduced CPU times. Both LBM variants outperform classical methods (TV reg and BSCB) in terms of efficiency and image fidelity, confirming the robustness of the proposed framework for high-resolution color image restoration.

Image	Method	PSNR (dB)	SSIM	CPU Time (min)
4*City	TV reg	27.23	**1.000**	85.56
	BSCB	28.58	**1.000**	74.90
	LBM-D2Q5	**29.94**	**1.000**	**17.01**
	LBM-D2Q9	29.85	**1.000**	26.07
4*Boat	TV reg	35.25	**1.000**	14.36
	BSCB	33.91	**1.000**	19.48
	LBM-D2Q5	**37.31**	**1.000**	**1.50**
	LBM-D2Q9	36.89	**1.000**	2.10
4*Bridge	TV reg	30.92	**1.000**	52.20
	BSCB	29.97	**1.000**	55.23
	LBM-D2Q5	31.96	**1.000**	**7.73**
	LBM-D2Q9	**31.98**	**1.000**	11.60

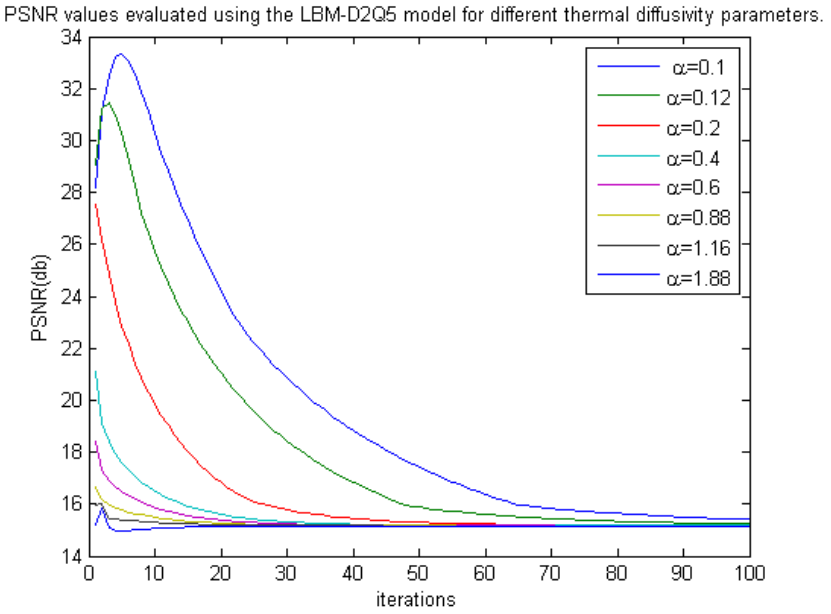
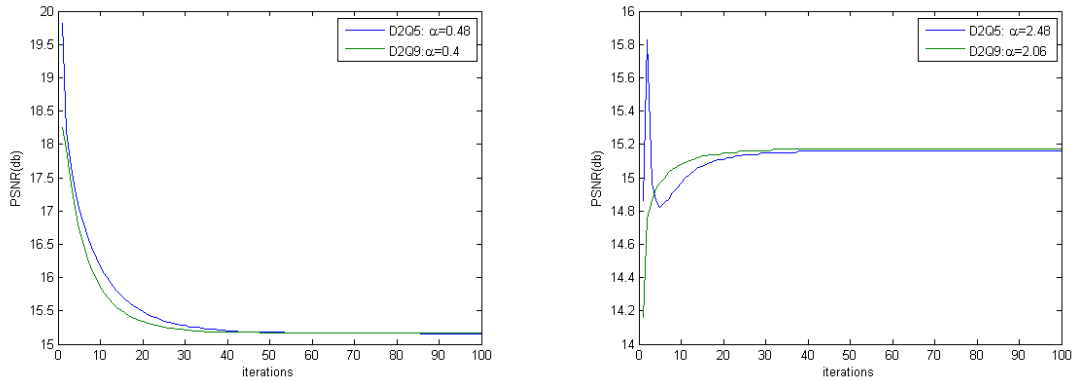


Figure 11: Temporal evolution of the Peak Signal-to-Noise Ratio (PSNR) for the *Girlface* image over 100 iterations, illustrating the impact of the thermal diffusivity parameter α on the inpainting performance. Lower α values lead to faster convergence and higher PSNR, reflecting enhanced reconstruction quality through controlled diffusion. In contrast, higher α values result in slower convergence and diminished fidelity due to excessive smoothing.



(a) D2Q5 performs better at $\alpha = 0.48$, $\tau_g = 1.7$ for D2Q5; $\alpha = 0.40$, $\tau_g = 1.7$ for D2Q9

(b) D2Q9 performs better at $\alpha = 2.48$, $\tau_g = 6.7$ for D2Q5; $\alpha = 2.06$, $\tau_g = 6.7$ for D2Q9

Figure 12: Comparison of PSNR evolution for the *Girlface* image using D2Q5 and D2Q9 lattice models over 100 iterations under two thermal diffusivity settings. (a) For moderate diffusivity, D2Q5 shows faster convergence and higher final PSNR, indicating better preservation of image structure. (b) For higher diffusivity, D2Q9 achieves superior PSNR, suggesting improved performance in scenarios requiring stronger smoothing.

6. Conclusions and future work

In this study, we introduced a novel image inpainting algorithm grounded in the Lattice Boltzmann Method (LBM), drawing inspiration from fluid dynamics through a vorticity–stream formulation. The method was validated on six benchmark images—three grayscale and three color—artificially degraded using random scratches. Experimental results show that the proposed LBM approach outperforms classical techniques such as Total Variation (TV) regularization and the BSCB algorithm in both reconstruction quality and computational efficiency.

The LBM-D2Q5 model achieved faster processing times than TV and BSCB, reducing computation from several minutes to under one minute while preserving or improving PSNR and SSIM. This confirms its ability to deliver high-quality inpainting with low computational cost.

Nevertheless, limitations exist. The model may underperform in highly textured regions, where deep learning approaches tend to be more effective. In addition, the method's sensitivity to parameters such as α and ν necessitates empirical tuning to achieve optimal performance.

Future work will explore hybrid frameworks combining LBM with deep neural networks to synergize the physical interpretability and speed of LBM with the representational power of deep learning. Such integration could enable adaptive parameter adjustment, enhanced texture reconstruction, and better generalization across diverse image types and

damage scenarios.

References

[1] K. Preston, M.J.B. Duff, S. Levialdi, P.E. Norgren, and J. Toriwaki. Basics of cellular logic with some applications in medical image processing. *Proceedings of the IEEE*, 67(5):826–856, 1979.

[2] G. Hernandez and H. J. Herrnann. Cellular automata for elementary image enhancement. *Graphical Models and Image Processing*, 58:82–89, 1996.

[3] A. Popovici and D. Popovici. Cellular automata in image processing. *In: MTNS*, 2002.

[4] S. Wongthanavasu and R. Sadananda. A ca-based edged operator and its performance evaluation. *Journal of Visual Communication and Image Representation*, 14:83–96, 2003.

[5] SP. L. Rosin. Training cellular automata for image processing. *IEEE Transaction on image processing*, 15:2076–2087, 2006.

[6] P. Lin B. Jawerth and E. Sinzinger. Lattice boltzmann models for anisotropic diffusion of images. *Journal of Mathematical Imaging and Vision*, pages 231–237, 1999.

[7] Z.Z.Yan Y.Chen and Y.H. Qian. An anisotropic diffusion model for medical image smoothing by using the lattice boltzmann method. *IFMBE Proceeding of APCMBE2008*, pages 255–259, 2008.

[8] Q. S. Chang and Y. Tong. A lattice boltzmann method for image denoising. *IEEE Transactions on Image Processing*, 18:2797–2802, 2009.

[9] V. CASELLES M. Bertamlio, G. SAPIRO and C. BALLESTER. Image inpainting. *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics*, page 417–424, 2000.

[10] A. L. Bertozzi M. Bertalmio and G. Sapiro. Navier-stokes, fluid dynamics, and image and video inpainting. *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, 1:I–I, 2001.

[11] T T. Chan. Local inpainting models and tv inpainting. *SIAM J. Appl. Math*, 62:1019–1043, 2001.

[12] Riya Shah, Anjali Gautam, and Satish Kumar Singh. Overview of image inpainting techniques: A survey. pages 1–6, 2022.

[13] Haodong Li, Weiqi Luo, and Jiwu Huang. Localization of diffusion-based inpainting in digital images. *IEEE Transactions on Information Forensics and Security*, 12(12):3050–3064, 2017.

[14] S. Liu Y. Wei. Domain-based structure-aware image inpainting. *Signal, Image and Video Processing10*, 5:911–919, 2016.

[15] G. Yang D. Zhang, Z. Liang, Q. Li, L. Li, and X. Sun. A robust forgery detection algorithm for object removal by exemplar-based image inpainting 77. *Signal, Image and Video Processing10*, 10:11823–11842, 2018.

[16] M. Li Z. Yan, X. Li, W. Zuo, and S. Shan. Shift-net: Image inpainting via deep feature

rearrangement. *in:Proceedings of the European Conference on Computer Vision*, pages 1–17, 2018.

[17] H. Chao Y. Zeng, J. Fu and B. Guo. Learning pyramid-context encoder network for highquality image inpainting. *in: Proceedings of the IEEE Conference on Computer Visionand Pattern Recognition*, pages 1486–1494, 2019.

[18] Z. Lin N. Cai, Z. Su, H. Wang, Z. Yang, and B. W.-K. Ling. Blind inpainting using thefully convolutional neural network. *The Visual Computer 33*, 2:249–261, 2017.

[19] Ji Zhao, Zhiqiang Chen, Li Zhang, and Xin Jin. Unsupervised learnable sinogram inpainting network (sin) for limited angle ct reconstruction, 2018.

[20] X. Zhao X. Zhu, Y. Qian, B. Sun, and Y. Sun. A deep learning approach to patch-based image inpainting forensicsk. *Signal Processing: Image Communication*, 67:90–99, 2018.

[21] Z. Yu Liu Y.-L. Chang, W. Hsu, and Vornet. Spatio-temporally consistent video inpainting for object removal. *in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019.

[22] S. Iizuka K. Sasaki, E. Simo-Serra, and H. Ishikawa. Joint gap detection and inpainting of line drawings. *in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5725–5733, 2017.

[23] Hongyu Liu, Bin Jiang, Yi Xiao, and Chao Yang. Coherent semantic attention for image inpainting, 2019.

[24] H. Chao Y. Zeng, J. Fu and B. Guo. Learning pyramid-context encoder network for highquality image inpainting. *in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1486–1494, 2019.

[25] J. Xiao L. Liao, R. Hu and Z. Wang. Artist-net: Decorating the inferred content with unified style for image inpainting. *IEEE Access*, 7:36921–36933, 2019.

[26] G. Wang C. Hsu, F. Chen. High-resolution image inpainting through multiple deep networks. *2017International Conference on Vision, Image and Signal Processing (ICVISP), IEEE*, pages 76–81, 2017.

[27] F. Yaghmaee V. K. Alilou. Application of grnn neural network in non-texture image inpainting and restoration. *Pattern Recognition Letters*, 62:24–31, 2015.

[28] K. Yanai T. Nakamura, A. Zhu and S. Uchida. Scene text eraser,. *in: 2017 14th IAPR InternationalConference on Document Analysis and Recognition (ICDAR) IEEE*, 1:832–837, 2017.

[29] J. Cheng P. Xiang, L. Wang, B. Zhang, and J. Wu. A deep network architecture for image inpainting. *in:2017 3rd IEEE International Conferenceon Computer and Communications (ICCC), IEEE*, pages 1851–1856, 2017.

[30] Xiuxia Cai, Song, and Bin. Semantic object removal with convolutional neural network feature-based inpainting approach. *Multimedia Systems*, 24:597–609, 2018.

[31] Oleksii Sidorov and Jon Yngve Hardeberg. Deep hyperspectral prior: Denoising, inpainting, super-resolution, 2019.

[32] J. Donahue D. Pathak, P. Krahenbuhl, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. *n: Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016.

[33] A. Gupta C. Doersch, S. Singh, J. Sivic, and A. A. Efros. What makes paris look like paris? *Communications of the ACM 58*, 12:103–110, 2015.

[34] A. Khosla B. Zhou, A. Lapedriza, A. Oliva, A. Torralba, and Places. A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence 40*, 6:1452–1464, 2017.

[35] S. Zhang H. Xue and D. Cai. Depth image inpainting: Improving low rank matrix completion with low gradient regularization. *IEEE Transactions on Image Processing 26*, 9:4311–4320, 2017.

[36] C. Fowlkes D. Martin, D. Tala J. Malik, and et al. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. *Iccv Vancouver*, pages 4311–4320, 2001.

[37] H. Su O. Russakovsky, J. Deng, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, and et al. Imagenet large scale visual recognition challenge. *International journal of computer vision 115*, 3:211–252, 2015.

[38] X. Wang Z. Liu, P. Luo and X. Tang. Large-scale celeb faces attributes (celeba) dataset. *Retrieved August15 2018*, 2018.

[39] Bastien Chopard, Jean-Luc Falcone, and Jonas Latt. The lattice boltzmann advection-diffusion model revisited. *European Physical Journal: Special Topics*, 171:245–249, 2009.

[40] J PERONA, P.and MALIK. Scale space and edge detection using anisotropic diffusion. *IEEE Trans. Pattern Anal. Mach.Intell*, pages 629–639, 1990.

[41] Leonid Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60:259–268, 11 1992.