Quality Inpainting and Region Filling Using a Complete-Composite Patch Sampling Scheme (CCPSS) Based on Statistical Estimation

Sujata G. Bhele [1,*], Shashank Shriramwar [1], and Poonam T. Agarkar [2]

[1] Department of Electronics Engineering, Priyadarshini College of Engineering, Digdoh Hills, Hingna, Nagpur, Maharashtra, India

[2] Department of Electronics and Telecommunication Engineering, Yeshwantrao Chavan College of Engineering, Hingna Road, Wanadongri, Nagpur, Maharashtra, India

Email: sujata_bhele@yahoo.co.in (S.G.B.); sshriramwar2@gmail.com (S.S.); poonamagarkar71@gmail.com (P.T.A.)

*Corresponding author

***Abstract*—Research in the field of Inpainting and Region filling has nowadays gained immense momentum to account for historical and ancient mural conservation and the technological advancements as regards the need for image processing applications. This work proposes a complete Composite Patch Sampling Scheme (CCPSS) incorporating four refinements over Criminisi and other state-of-the-art techniques. The earlier Criminisi patch matching approach is supported by adding alien patches generated from independent color components of the existing known or valid patches. The dependency of confidence and data terms used in the priority function had been removed by selecting unknown pixels based on their neighborhood unknowns. The proceeding estimated unknown pixel patches are considered in estimating successive invalid pixel patches. Lastly, the highest similar or consistent known patch concerning an unknown patch is not finalized but retained until all the unknown patches are estimated. The true value of any unknown pixel is computed using the mean and median of all retained values. The inpainting, region filling, and reconstruction quality of the CCPSS scheme is superior in terms of visual perception when tested over high-resolution texture-structure images at the cost of computational overhead.**

***Keywords*—inpainting, region filling, complete-composite patch matching, Composite Patch Sampling Scheme (CCPSS), patch matching, texture-structure**

I. Introduction

Recent technological developments in image painting have played a great role in computer vision applications in reconstructing images, removing targeted regions, and restoring objects including texture regeneration. Quality inpainting is limited by many factors which include image resolution, image noises, amount of area to be inpainted, the similarity between the known and unknown patches, structure-texture contents of the image, and finally the inpainting algorithm. Most often, the algorithm becomes image-dependent and generates blurred regions, introduces artifacts, or is even pruned to produce meaningless patch components inconsistent with the target region. The paint mechanism lies in filling the lost areas nearer and closer to the reasonable real semantic details of the image. The problem is typical in that the defects are often unclear and usually the inpainting methods are based on mathematical and physical approaches which are analyzed and worked out according to human cognitive states or rules.

Many different schemes are suggested in the literature to deal with texture synthesis for generating substantial regions and inpainting for region filling. Few methods such as Criminisi *et al.* [1], deal with both aspects and concentrate on the order of pixels to be in which they are estimated. The existing methods consider only single-pass estimated values of the pixels and do not reuse the inpainted region at the next inpainting iteration. The single pass estimated values are accepted as the algorithm is unable to find a better match from the known patches and on the other hand, the inpainted values are not treated to be useful. Such methods are therefore limited to inpaint missing regions from simple texture-structure images but their performance drastically degrades for complex structures.

Bertalmio *et al*. [2] worked on processing small missing regions and small texture and color disparities between the unknown area and neighborhood pixels. The region of interest to be inpainted is selected manually by a user which is then automatically filled using the details in the vicinity of the region under construction. The idea is to complete the isophote lines reaching the region boundaries from the inside. The time complexity of the region-filling approach is low and able to complete missing areas comprising different structures and backgrounds. The proposed inpainting algorithm is carried out independently on each color component which alleviates the problem of spurious color by using a color space model like LUV where one of the components represents the luminance and the rest of the two indicates the chrominance. A similar approach was carried out in [3, 4] to find optimal texture pixels in immediately surrounding areas for filling missing

Manuscript received December 1, 2023; revised February 28, 2024; accepted April 28, 2024; published October 8, 2024.

or defective areas. Chen *et al.* [3] carried out tongue texture analysis by isolating the tongue coating and the body by a Gaussian Mixture Model. The color change of the tongue body image concerning texture continuity was ensured using a generative image inpainting along with contextual attention. The work aimed to classify tough and tender tongues based on a Deep Neural Network utilizing the ResNet101 residual model. The work introduced in [4] concentrated on the mismatch between the features and intensity of inpainted regions using the neighborhood. The former was handled using a feature-based sparsity while the latter used an adaptive intensity technique suitably matching the intensity labels. False matching and excessive texture region extensions were avoided using a patch-based sparse approach. Criminisi *et al.* [1] accelerated the restoration process by finding the target blocks around the missing region. An adaptive technique using sparse reconstruction was suggested by Guleryuz *et al.* [5] which obtained the best estimation of defective areas. The author focused on inpainting missing areas primarily concerned with textures, image features, and edges which were not handled properly by other inpainting methods. The area to be restored was transformed to provide sparse decomposition so that the transform coefficients are zero or approximately zero. Further, the small value coefficients were determined using an adaptive thresholding mechanism. The inpainting method suggested by the author was a simple denoising process without requiring any complex preconditions. An iterative method to restore the background of an image by searching the most suitable patch was the work concerned by Barnes *et al*. [6]. The novel work to reduce the time for finding the best patch match in the image was simplified using a random sampling technique. They concentrated on patch natural coherence and used them to propagate such coherent patches quickly in the surrounding areas or regions. They conducted experiments to show that their interactive editing tool offered high-quality inpainting with reduced time. The fast patch match nearest neighbor field technique reduced the computational complexity by using the continuity of the picture and range of computing the similarity. The valid part of the image was extended to recover the invalid part for small or narrow areas [7]. A complex patch-based approach relying on finding the similarity between the background and the area under repair and copying similar parts to the defective regions was proposed in [8, 9].

Concerning the traditional (non-learning) approach, we used the patch search and match scheme for image inpainting, region filling, and generating texture patterns. The modifications incorporated in the proposed CCPSS scheme are summarized in the succeeding paragraphs.

Almost all methods found in the literature used distance measure between the target patch around any pixel P in region Ω and patch Q in region $\emptyset$, the proposed work introduces in addition an independent color patch matching mechanism that not only creates new patches (composite patches) but also improves the quality of inpainting for complex textures and structures. This was advantageous when the area of region $\emptyset$ was scarce as compared to the area of region Ω.

Also, the most similar patch to estimate the target patch was found in region $\emptyset$ only, and the pixels estimated during the inpainting process in the Ω region were not included or considered for subsequent matching. That is, the look-up table consists of patches only from the known region and the look-up table remains un-updated with the patch that had been estimated in the preceding epoch(s). We updated the list of known patches at each iteration whenever a new pixel in the Ω region was estimated. The scheme improved correlation in the neighborhood and decreased the inpainting errors.

The pixels estimated during the process were used to update the look-up table and were not finalized and set for the final or true value. Instead, all the estimations for any pixel (Estimated being a center pixel or estimated during estimation of any other unknown pixel as a part of the neighborhood) were stored until all the unknown pixels were exhausted from the Ω region. The final value was calculated using the average and median of all such estimations. It eliminated the diffusion error and improved the perceptual quality of the inpainted image.

A new priority function was introduced and the dependency of the confidence and data term was eliminated. The pixel P patch ϕ_P neighborhood was scanned and the pixel with minimum unknown (zero values) was selected on a priority basis for estimation. For the presence of multiple pixels with the same priority, the scheme considers pixel patches on a first come first served basis.

III. Related Work

Most of the past work is focused on non-learning techniques for region filling and reconstruction which is based on the idea of patch matching and diffusion filling approaches. The patch matcher scheme is clear and relies on finding a high similarity patch in the valid region whereas the diffusion filler spreads the extremes of the complete area to compensate for the holes. These methods were based on partial differential equations which included the Mumford-Shah model [10], Eulers model [11], total variation models [12], etc. Patch matching based on Markov models [13], annihilation filter and low-rank structured matrix [14], two stages [15] and gradient-based low-rank approximation [16], low gradient regularization [17], statistical regularization-Markov random models [18], non-local matching and nonlinear filtering [19], sum of squared difference [20] are some of the techniques used for image inpainting. Diffusion-based inpainting models are based on Fourier transform and fractional order derivatives [21], neighborhood distance-direction-based coefficients between valid and invalid pixels [22], and inter-intra channel variance-based features [23].

Texture-synthesis-based inpainting gained momentum after the texture-based non-parametric method was introduced in [24]. New texture images were generated by sampling and finding similar pixels from the same image known as neighborhood pixels. Efros and Leung [24] in their work effectively synthesized textures that showed a higher resemblance to the input. Their approach was simple and treated as the basis of many other texture-

synthesis-based inpainting due to high-quality results. The main objective of the work proposed in [24] was to fill in the missing region with enhanced reconstruction. However, texture synthesis-based inpainting brought different difficulties as compared to classical inpainting schemes. One of the outstanding works that gained immense attention was introduced by Criminisi *et al.* [1] which contributed to removing large objects from images.

Criminisi's work [1] governed linear structures' repetition of two-dimensional and one-dimensional textures and structures. The destroyed pixels were filled on a priority basis and the selection were a function of the multiplication of confidence and data terms. The dependency of both the components limits their value, i.e., when one reaches zero the other attains a small value [23, 25]. The issue of dependencies was worked out in [23, 26]. Criminisi *et al.* [1] represented the valid region by $\emptyset$, the invalid region by Ω, and $d\Omega$ to represent the boundary between $\emptyset$ and Ω. The pixel P to be inpainted and its rectangular patch is shown in the Fig. 1 below where ∇p and np are the tangents and normal to the boundary surface.

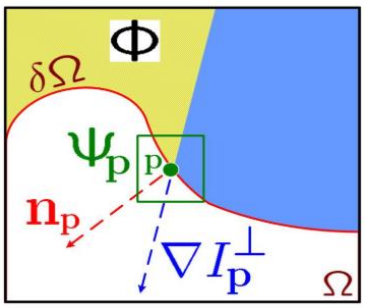


Fig. 1. Criminisi *et al.* [1] inpainting scheme denoting region $\emptyset$, region, pixel P, and patch ϕ_P.

Criminisi *et al.* [1] expressed the priority function as the product of confidence and data term by the following Eq. (1). Eqs. (2) and (3) provide the confidence and the data term with area $|\phi_P|$ of the patch around P. The factor α is the normalization factor.

$$P(p) = C(p) \times D(p) \quad (1)$$

$$C(p) = \frac{\sum_{q \epsilon \varphi_p \cap \varphi} C(q)}{|\varphi_p|} \quad (2)$$

$$D(p) = \frac{|\nabla I_p^{\perp} \cdot n_p|}{\alpha} \quad (3)$$

For pixels in $\emptyset$, confidence is $C(p) = 1$ and $C(p) = 0$ in region Ω (Eq. (4)). The pixel to be estimated over the boundary between $\emptyset$ and Ω is selected on priority based on its confidence value. The patch around P is then compared with all other patches in $\emptyset$. The highest similar patch $\phi'_P = \emptyset_q$ from the available patches is found using the SSD distance metric given by Eqs. (5) and (6).

$$C(p) = \begin{cases} 0, & \forall_p \epsilon \Omega \\ 1, & \forall_p \epsilon \emptyset \end{cases} \quad (4)$$

$$SSD(p, q) = arg \min_{\emptyset_q \in \emptyset} SSD(\emptyset_p, \emptyset_q) \quad (5)$$

$$SSD(\emptyset_p, \emptyset_q) = \sqrt[3]{\sum_{i=1}^{m} \sum_{j=1}^{n} \left[\left(p_{ij}^R - q_{ij}^R\right)^2 + \left(p_{ij}^G - q_{ij}^G\right)^2 + \left(p_{ij}^B - q_{ij}^B\right)^2 \right]} \quad (6)$$

Three terms correspond to the three-color components (R, G, and B), and (i, j) cover the patch window rows m and columns n. The center pixel of the winning patch $\emptyset_q$ obtained corresponding to minimum SSD is then substituted at the target pixel of patch ϕ_P. Subsequently, the $C(p)$ of the target pixel is changed and the confidence of the whole image is updated. The procedure is repeated until all pixels in Ω are estimated.

In recent times, complex techniques exploiting various artifacts within images and in-depth structure analysis in the vicinity of the area to be inpainted were introduced. A patch-matching method was presented in [13] which used a Markov Random field while the same was done using an annihilation property filter and a low-rank structured matrix in [14]. Jin and Ye [14] limited the search process to the surrounding background of the unpainted region in. A similar approach was used in [27] using a target object selection method again restricting the search operation to the background. On the other hand, the work introduced in [15] and [16] focused on recovering corrupted regions (blocks) using low-rank approximation and gradient-based extended schemes respectively.

Some authors worked on another sub-area of representing the image by translating or transforming it to some other representation which primarily included the DWT domain, DFT, DCT, etc. Using the transformation technique and then inpainting produced remarkable results as in [28, 29] and retaining the uniformity of the inpainted area, however, they were prone to ugly artifacts at the textural edges and mad such techniques garbage. Transforming and using conventional inpainting or mixed-mode solutions were adopted in [30, 31]. The mixed-mode techniques included diffusion schemes and texture synthesis. Work explored in [32] combined a PDE-based solution in collaboration with patch synthesis and coherence map. A similar approach was used in [33, 34] where local features like edges were reconstructed using exemplar-based techniques and anisotropic diffusion with transport equation respectively.

The sequence of inpainting using the patch-based or exemplar-based techniques is to search for the best order of filling, locate the best patch to approximate the target patch, and finally copy the matched patch. The last step can be governed by some pre-processing if required to ensure the textural similarity within the region to be inpainted. However, every such patch match inpainting method introduces artifacts similar to the diffusion schemes. The artifacts were analyzed by Criminisi and more recently in [9]. Various artifacts such as texture mismatch, blur, and staircases are examples of susceptible artifacts. Color or texture mismatch is the result of unfit patches being copied to the missing areas by the inpainting algorithm. Jaggy edges or staircase effects are the cause of the inability of the inpainting scheme to accurately replicate the delicate or fine details, especially along high-frequency edges. Blurs result due to averaging operations during the patch copying process while tackling intricate textures or high-contrast edge regions.

III. Methods and Materials

We propose a CCPSS scheme for inpainting, region filling, and regeneration in images based on a complete color patch matching window and priority-based selection of unknown pixels. The scheme maintains a list of L_k of all known pixels with their 3×3 color neighborhood and another list of L_{uk} of 3×3 patches surrounding unknown pixels that need to be reconstructed. The similarity between the unknown pixel patch (ϕ_p) in L_{uk} and the known patch (ϕ_q) in L_k is estimated using the Euclidean distance on one hand and similarity based on individual color 1D components L_{kR}, $L_{kG,}$ and L_{kB} respectively on the other hand. Almost all the state-of-the-art techniques in the literature follow the former similarity measure including Criminisi *et al.* [1], The latter method finds similar patches in the red, green, and blue component 1D patches independently. This ensures greater depth matching even though there is the possibility of new patch generations since the matched individual 1D patches are concatenated to form a complete 3×3, 3D color patch from individual color 1D patches. We found experimentally that individual 1D color component patch matching scheme is very effective when the list L_k has fewer entries, that is when the image is degraded to a greater extent ($L_{uk} > L_k$). When a known patch is found consistent with the unknown pixel neighborhood patch, the same is retained and not finalized until all the unknown pixels have been estimated from region Ω.

We have considered a 3×3 neighborhood while estimating the similarity measure for greater accuracy. Any pixel ($\emptyset$) is highly correlated to its neighborhood pixel ($\emptyset'$) in the unknown nearest dimension. Considering 3×3 neighborhoods will not suitably correlate the estimated value with other neighborhood candidates. The neighboring candidates may at that instant be known or unknown and the degree of correlation depends on candidate distance and spatial location. Based on 3×3, 3D patch similarity, if the pixel ($\emptyset$) is finalized, the reconstruction may suffer from under or overfitting issues and the quality of reconstruction will be very poor.

Therefore, we estimate two different similarity measures as given by the Eqs. (7)–(10):

$$D_{color} = \sum_{d=1}^{3}\{\sqrt{\sum_{i=1}^{9}[(P_{LkR} - P_R)^2]} + \sqrt{\sum_{i=1}^{9}[(P_{LkG} - P_G)^2]} + \sqrt{\sum_{i=1}^{9}[(P_{LkB} - P_B)^2]}\} \quad (7)$$

where, P_{LkR}, $P_{LkG,}$ and P_{LkB} are individual R, G, and B one-dimensional color patches from list L_k. P_R, $P_{G,}$ and P_B are individual R, G, and B 1D color patches from List $L_{uk.}$ Fig. 2 shows the computation of distance D_{color}.

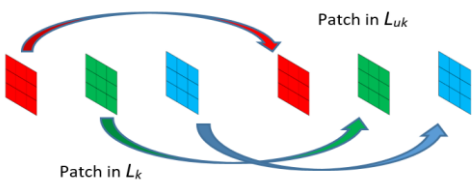


Fig. 2. Computation of *complete* patch distance D_{color}

The best matching patch (*Complete patch*) $\phi_P = \emptyset_q$ is selected corresponding to the minimum distance D_{color} obtained for all patches in L_k.

Eqs. (8)–(10) are used for computing the distance between similar color components. The individual 2D color patch corresponding to each color is scanned respectively in list L_k.

$$D_r = \sqrt{\sum_{i=1}^{9}[(P_{LkR} - P_R)^2]} \rightarrow List_{frameR} \quad (8)$$

$$D_g = \sqrt{\sum_{i=1}^{9}[(P_{LkG} - P_G)^2]} \rightarrow List_{frameG} \quad (9)$$

$$D_b = \sqrt{\sum_{i=1}^{9}[(P_{LkB} - P_B)^2]} \rightarrow List_{frameB} \quad (10)$$

The 2D color patch having a minimum distance in L_k is then selected and concatenated using the following Eq. (11).

$$P_{new} = C\{D_r, D_g, D_b\} \quad (11)$$

Here, C is concatenation operator. The patch corresponding to a minimum distance from R, G, and B entries of list L_k are concatenated to form a new patch (*Composite patch*) P_{new} shown in Fig. 3.

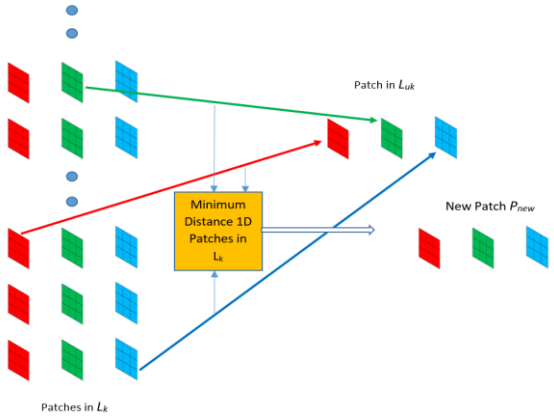


Fig. 3. Construction of new patch (composite patch).

We need to find the new distance concerning the new patch P_{new} with the unknown patch from $L_{uk.}$ The new distance D_{RGB} is given by the following Eq. (12):

$$D_{RGB} = E[P_{new}, P_{uk}] \quad (12)$$

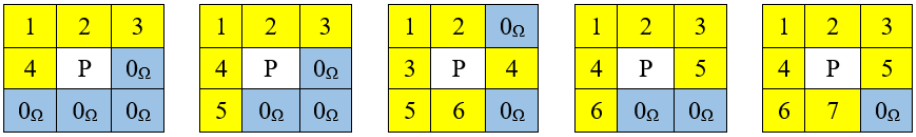
where P_{uk} represents the unknown patch under consideration from L_{uk} and *'E'* is the Euclidian distance. The best match from $\{P_m, P_{new}\}$ is obtained by comparing the distances D_{color} and D_{RGB} as represented by Eq. (13). (Patch having Minimum distance)

$$P = Patch \in Minimum\{D_{color,} D_{RGB}\} \quad (13)$$

The pixels in patch P (center pixel and its 3×3 neighborhood) are not final and are therefore temporarily stored but the list L_k is updated with this new entry. The estimated unknown entry P_{uk} under consideration for

which the estimation is carried out is deleted from the list L_{uk}. The list L_{uk} is sorted to find the new unknown patch to be estimated on priority. The process is repeated until all unknown pixels in list L_{uk} are estimated.

The selection of an unknown patch from the list L_{uk} is governed by the concentration of holes in the patch. The patch having the lowest number of unknown pixels is estimated on priority concerning others. Fig. 4 demonstrates the priority mechanism, shows an example with 5 different patches available in the list L_{uk}. The first, second, third, fourth, and fifth patches have respectively 4, 3, 2, 2, and 1 unknown elements. Our patch on priority will be the last (Fig. 4(e)) patch having 1 unknown element first. Patch (c) and (d) will be next, followed by patch (b) and then patch (a) since it has the highest unknown (4) elements.



(a) (b) (c) (d) (e)

Fig 4. Patch configurations. (a) shows a patch with 4 holes (zero/unknown elements) in the list L_{uk}. (b) shows a patch with 3 unknowns. (c) & (d) depict a patch with 2 unknowns. (e) contains a patch with one unknown from region Ω..

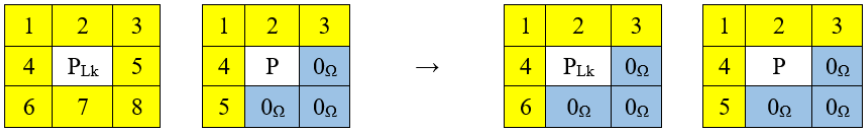
The criteria to select the patch ϕ_P on priority is given by the following Eq. (14),

$$Min\ (R) = \frac{Z_E}{NZ_E} \tag{14}$$

where R represents the ratio of several holes (zero elements (Z_E)) to non-zero (NZ_E) elements in P.

This controls the flow of error across neighboring unknown pixels to a considerable extent and thus improves the quality of inpainting.

The distance between the unknown patch and the patches in the list L_k is computed after matching their element pattern. Matching element pattern here means matching holes in L_k patches concerning the unknown patch from L_{uk} under consideration. This is done to minimize the effect of extra overhead in distance calculation and prevent false patch hits. Fig. 5 demonstrates the pattern matching of L_k patches in correspondence to an unknown patch. Fig. 5(a) is the list L_k entry and Fig. 5(b) is the unknown patch to be estimated from L_{uk}. The patch in Fig. 5(b) has three unknown elements (holes) so it exhibits a pattern having 5 known and 3 unknown elements around the center pixel P. Therefore, to match patch P_{Lk} concerning P, 3 elements in P_{Lk} are made zero. Fig. 5(a) will be converted to Fig. 5(c) while patch P remains the same, the distance between P_{Lk} and P will be computed then.



(a) (b) (c) (d)

Fig. 5. Patch adjustment for matching. (a) shows the Patch entry from list L_{uk}. (b) shows a Patch p with 3 holes (unknown neighbors) to be reconstructed. (c) depicts a List L_k patch with substituted zero elements corresponding to indices of zero elements of patch P. (d) shows an Unaltered Patch P.

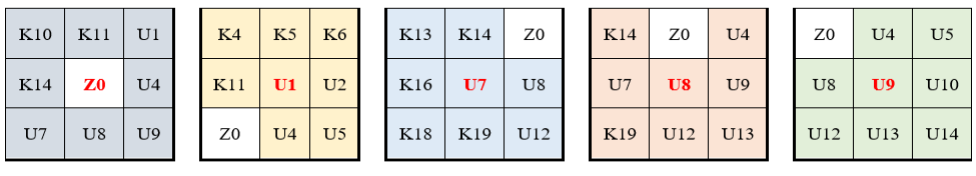
The following Eq. (15) explains the mechanism.

$$P_{Lk\,(N)} = 0\ (if,\ P_{N\,\epsilon\,\Omega} = 0) \tag{15}$$

The intensity values 5, 7, and 8 in the P_{Lk} entry have been assigned value 0 as per the positions of zero elements Z_E of patch P. The assignment is done for all three color components of the patch P_{Lk} entry in list L_k.

The estimated pixel value is temporary and limited to the use of updating the list L_k. The estimated value for a certain unknown pixel (as principal or center pixel) and when the same pixel is estimated as a neighboring pixel for any other principal pixel, is stored until all the unknown pixels in L_{uk} are estimated likewise or the list L_{uk} is completely emptied. The estimated values for each unknown pixel are saved till the last iteration.

Considering Z0 as the unknown pixel to be estimated, the patches containing Z0 are represented in Fig. 6. In patch p1, Z0 is the unknown pixel and while targeting U1, U7, U8, and U9 different values of Z0 are expected from the matched list entries.



P1 P2 P3 P4 P5

Fig. 6. Unknown patches P1–P5 to be estimated which includes Z0.

We experimented using different statistical approaches for finalizing the value from available estimated values. The objective was to paint for the best match and preserve the textural and structural aspects of the given image regarding the missing elements. We succeeded in finding the best statistic over the estimated values which proved accurate for inpainting the missing values. The best accurate match was found by substituting the average and median of all estimated values as the final substitute for any unknown pixel. For inpainting, the median value proved to be much better than the average value while for region filling the average value superseded the median value. We used an array of dimension 50 (higher side) for each unknown pixel to store estimated values during the inpainting process. The dimension was assumed on a test conducted on several images. It was seen that any unknown pixels are estimated at least twice. We used the MATLAB feature of the sparse array to save the amount of memory for the allocation to store estimated values.

For texture repetition, we used the proposed inpainting or region-filling technique with some modifications in the initial stage. In the inpainting case, the region to be reconstructed is selected and cropped using a polygon. The pixel corresponding to the cropped region for our reconstruction scheme is set to *R = 0, G = 0, and B = 0* for all the three color components, that represent a black mask as shown in Fig. 7 below.

Our proposed inpainting Algorithm 1 is listed below.

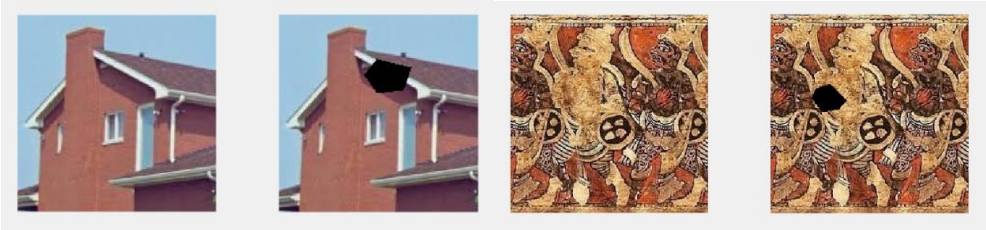


Fig. 7. Original images and their inpainting regions. The original house and the mural image. black polygonal regions are to be imprinted in both cases.

Algorithm 1: Inpainting

Input: Original Image **A**
Output: Reconstructed Image **F**, **G** using Median Value & the Average Value

1 Select the portion to reconstruct
2 Prepare Mask of the region to reconstruct - **Mask**
3 Construct the list of indices of the region to reconstruct - $\mathbf{R_{idx}}$

4 Sort the patches of unknown pixels in L_{uk} to find the Priority - **priority_sort**
5 Create L_k using the known pixels with complete neighborhood - **prepare_Lk**

6 Sort the list, L_{uk}
7 While L_{uk} becomes empty
8 Get the highest-priority patch
9 Find the indices of zero elements in the patch
10 Mask all elements in L_k entries corresponding to indices
11 Reshape patch as per L_k entry for mathematical calculations
12 Repeat patch L_{uk} as per size of L_k
13 Calculate Distance for Similarity – Complete and Composite color component match
14 Find the best patch match from two approaches
15 Copy target pixel value and neighboring pixel values from matched patch
16 Store values of all unknown pixels for final estimation
17 Remove entry from L_{uk}
18 Update L_k with new entry and sort as per Original image indices- **prepare_Lk**
19 Sort L_{uk} for priority - **priority_sort**
20 End

21 For m = 1 to total estimated values
22 Get all values of unknown pixels – R, G, and B
23 Find the average/median of values for R, G, and B – $\mathbf{M_R}$, $\mathbf{M_G}$, **and** $\mathbf{M_B}$
24 Substitute the values at pixel location in the images – **F**
25 end; Output G and F

27 *Display the performance parameters*

Fig. 8 below shows the output of inpainting by our proposed algorithm. The third image represents the reconstruction when the average value is considered instead of the median value of the predicted pixel from all predictions. The perception shows that the inpainting result using the average value is inferior to the result

obtained using the median value. The region indicated by the red circle shows the region to be inpainted and the result of our algorithm considers the average and median of the predicted value of each pixel. Fig. 9 shows another image with high texture and its reconstruction using the median value. The proposed CCPSS technique can efficiently paint high-texture structures and degraded regions.

(a) (b) (c) (d)

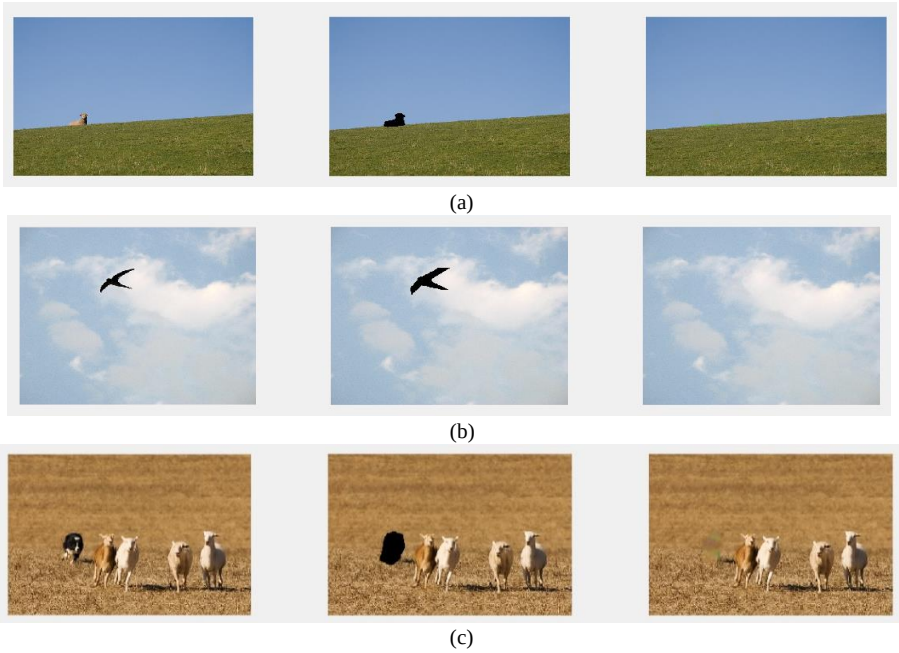
Fig. 8. Results of proposed inpainting scheme. (a) Input image. (b) Polygon region to be inpainted. (c) Reconstructed image using average value. (d) Reconstructed image using median value. (Red circle shows the region of interest).

(a) (b) (c)

Fig. 9. Inpainting results on high textured image. (a) Input image. (b) Polygon region to be inpainted. (c) Reconstructed image using median value.

IV. Results and Discussion

The performance of our CCPSS-based statistical inpainting is shown in Figs. 10 (a)–10(e). The images are part of the well-known high-resolution Pristine Dataset images. The dimensions of images vary and the average dimension is about 500 pixels × 350 pixels (rows × columns). The result shows that CCPSS can reconstruct regions with background when objects are removed. We used a variety of images about homogeneous and heterogeneous backgrounds, the average value of estimations produces perceptual quality reconstructions. Fig. 11 shows how the unknown pixels are selected using the priority mechanism adopted under the CCPSS in this paper and explained earlier for Figs. 10(d) and 10(e) respectively. The pixel index represents the one-value index or location of the unknown pixel. Table I represents the time taken to fill the targeted area using MATLAB 2012b, Intel 11th Generation i5 processor, 2.71 GHz, 16 GB RAM, and 256 GB SSD. We already quoted that the quality of inpainting is achieved at the cost of computation time. Since we used a dual patch matching approach to improve the quality of reconstruction, the computation time is larger and so is the time. Also, the final value of the pixel is set when all the unknown pixels are estimated and during all iterations, the estimated values are temporarily stored which requires extra sparse storage.



(a)

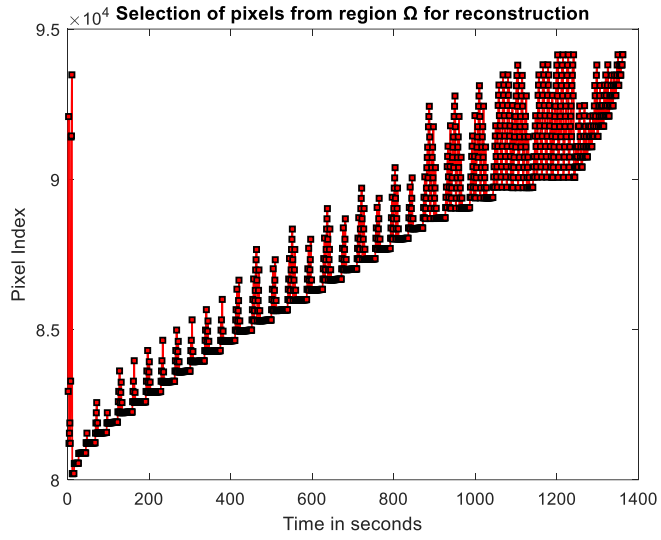
(b)

(c)

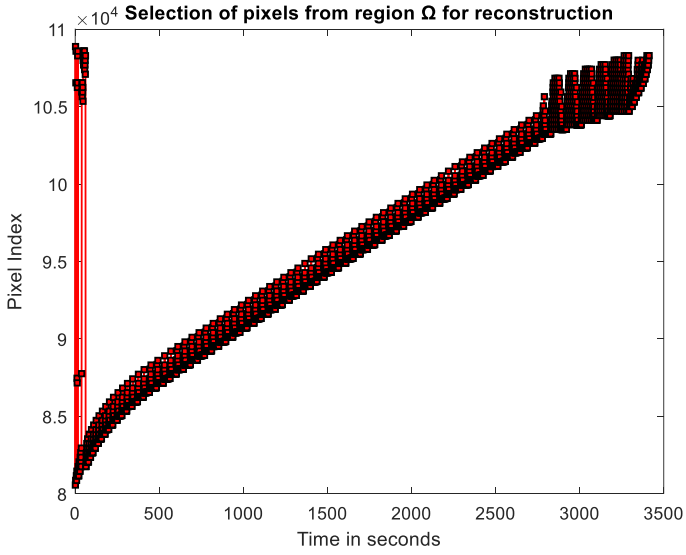
(d)

(e)

Fig. 10. Original image in the first column, marked region (sheep, bird, left sheep, football, helicopter, and left object) to be reconstructed in the second column and Missing region filled the last column, (a) Single sheep image, (b) Bird image, (c) Multiple sheep image, (c) Football image, (e) Helicopter image.



(a)



(b)

Fig. 11. Pixel's index inpainted on priority for football and helicopter image, (a) Football image, (b) Helicopter image.

In this paper, we are dealing with those images, where the scenes need to be repeated along any direction viz. horizontally left or right, vertically up or down, or outward concentric in all directions of the given image. The region to be generated in any of the directions mentioned is in priory created using a black mask ([0, 0, 0]) for R, G, and

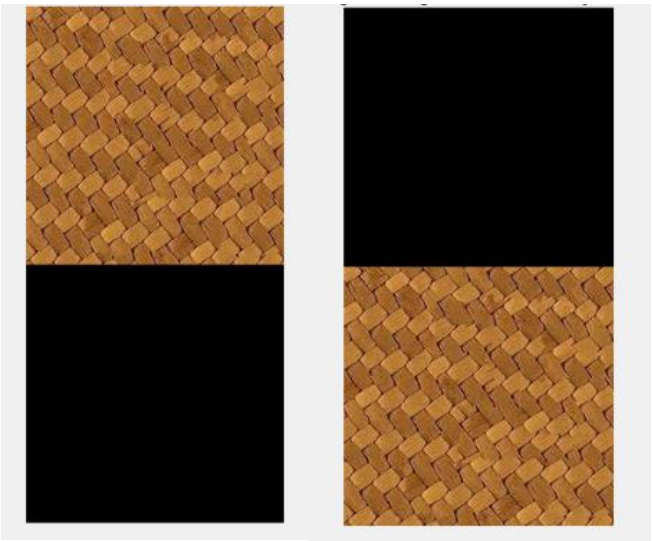
B colors as shown in the figure. It is the requirement of our algorithm that can pinpoint a region based on the known regions from the same image. Fig. 12(a) shows a mask generated on the horizontally left side which is equal to the column size of the given mural image and Fig. 12(b) shows a similar mask on the right side. Fig. 13 shows the regeneration to be followed in vertically up (b) and down direction (a) whereas Fig. 14 demonstrates a concentric construction.

TABLE I. TIME REQUIRED TO FILL THE REGION FOR EACH IMAGE

Sr. No.	Image (Pristine Dataset)	Inpainting time (MATLAB 2021b)
1	Sheep	4915.217109 s
2	Bird	15142.912378 s
3	Multiple sheep	17722.121274 s
4	Football	12058.300869 s
5	Helicopter	19162.968409 s

(a) (b)

Fig. 12. Mask formation. (a) Region to be generated horizontally left. (b) Region to be generated horizontally right.



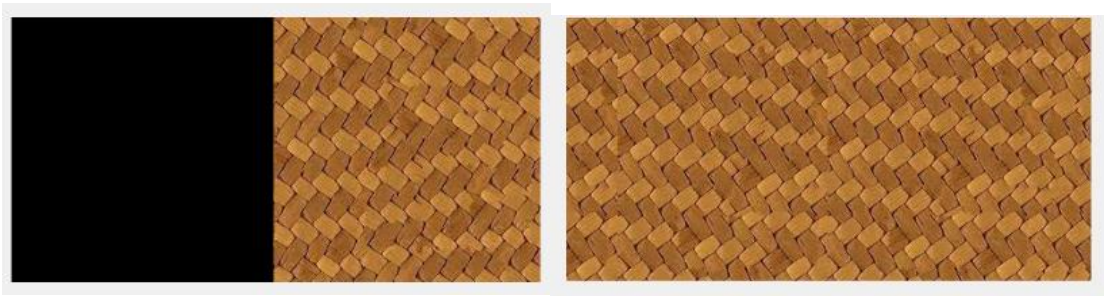
(a) (b)

Fig. 13. Mask formation. (a) Region to be generated downward. (b) Region to be generated vertically upward.

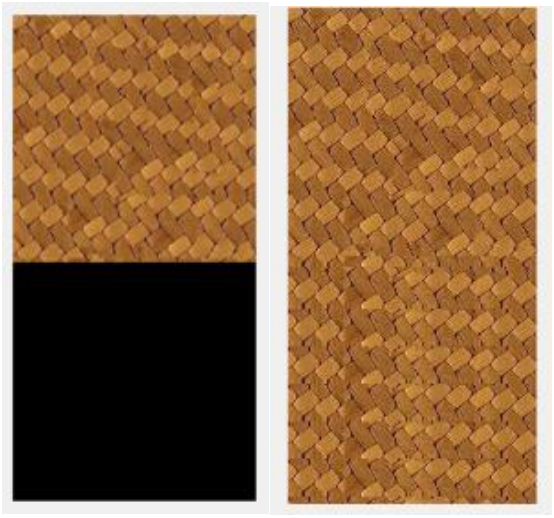
Fig. 14. Mask created to construct the texture in all directions (Concentric).

The length of the mask along the dimension along which it is to be generated is not limited and depends on choice. The generation process will identify the pixel with higher priority and continue likewise after the list L_{uk} is updated and sorted each time depending upon the number of zero elements in the patch. Inpainting repeating texture pattern, the case is different from the within region inpainting case, since all the boundary pixels that are to be constructed will possess almost equal priority. Therefore, the pixels will be served on a first come first basis as they appear in the list L_{uk}. Inpainting time is dependent on the area to be inpainted, the larger the area, the more time will be required. For generating textures, we have selected different patches or window sizes depending upon the class of texture the image holds to overcome time complexity in inpainting. For fine textures, a smaller window (w = 3, 5) works well, while when the texture pattern is quite large, a larger window size (w = 7, 9, 11, 13, and 15) is required to construct or repeat patterns accurately. A smaller window in a later case tends to get stuck at the wrong known patch since larger texture patterns have great similarity to their neighboring patches.

We have used a medium window size w = 9 for all the texture generation shown below to reduce time and computational complexity. Fig. 15(a) and Fig. 15(b) show the result of the region generated horizontally in the right and left direction, and vertically in the north and south direction whereas Fig. 16 indicates the all-direction (concentric) generation. We have used images with regular (repetitive) and irregular (non-repetitive) textures so that the performance can be evaluated even through human perception. As seen from the results obtained, the quality of generation is acceptable as far as texture pattern is concerned. The generated regions show some amount of inconsistency with the original image when perceived properly. This is due to the irregular texture pattern in the image and the size of the window (w = 9) used. The performance can be improved using a smaller window (say w = 3, 5) increasing overheads (time and computation). The all-direction generation using the bricks in Fig. 16 is accurate since the textural pattern is larger and repetitive. Also, the mural image in Fig. 17 is exactly generated. All three soldiers have been generated on the right side since the texture pattern is regular and large.

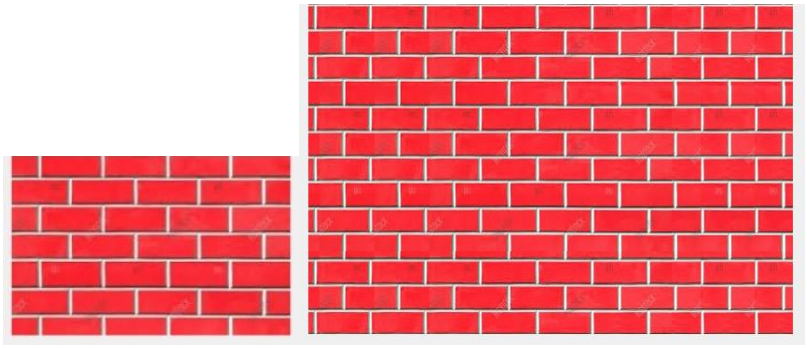


(a)



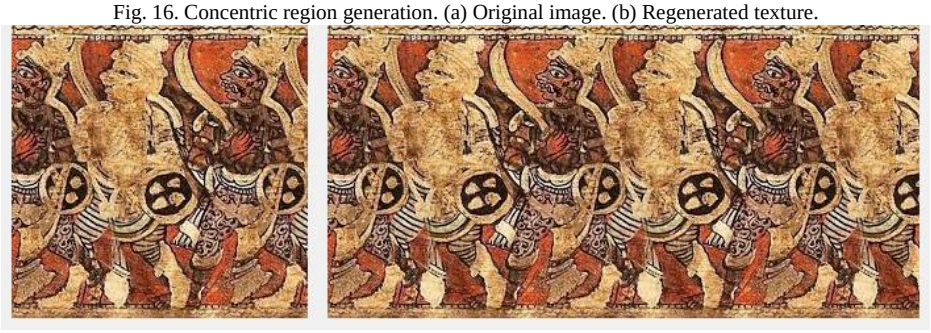
(b)

Fig. 15. Texture generation. (a) shows generated textures in the horizontal direction. (b) shows texture generated in the vertical direction.



(a) (b)

Fig. 16. Concentric region generation. (a) Original image. (b) Regenerated texture.



(a) (b)

Fig. 17. Regeneration using a mural image. (a) Original image. (b) Regenerated texture in horizontally right direction.

As seen from Refs. [1, 35–40], different authors have used different sets of images for the region-filling approach. No standard dataset with the ground truth is available to validate the results based on performance metrics. Authors have claimed the based-on disparities between the original and the resultant image. The background texture replaces the object in the image and decides the visual quality. Therefore, a comparative approach is not possible in this case. The proposed work is based on our TSCPMA (Texture-Structure Conserving Patch Matching Algorithm) [41] approach used for region-filling.

V. Conclusion

TSCPMA can preserve the color, textural, and structural quality of an image which have deteriorated due to various natural factors. We have proposed an efficient inpainting and background generation CCPSS over Criminisi and other state-of-the-art conventional algorithms to generate regular and irregular texture-structure patterns in all possible directions concerning a given image. The overheads can be controlled by using varying matching window sizes at the cost of texture-structure pattern generation quality. Analysis showed that for finer textures, smaller matching windows are suggested while for large textures can be generated using a larger matching window. The matching criteria are improved by including individual color component patch matches based on minimum distance and then combining them to introduce new patches. The minimum distance patch is temporary for any center pixel and finalized when all the unknown pixels are estimated. But the temporary patch is used to update the known patch list for estimating other unknown pixels thus inventing and introducing new patches. The technique eliminates any error that can penetrate and diffuse in the neighboring pixel when estimated. The inpainting achieved by our method outperforms any other method. Our method has the extra burden of comparing individual color component patches and statistical calculations with increased storage which makes the technique somehow time and computationally complex. The CCPSS method introduced in this paper can paint high-resolution, irregular regions, generate quality perceived and acceptable background when the object is removed, and generate the textural pattern.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Sujata Bhele confirms responsibility for the following: study conception and design, data collection, and analysis. She carried out surveys on several research papers, collected data, designed the methodology, and coded the program in MATLAB 2021b. Shashank Shriramwar tested and validated the results. He agreed to be accountable for all aspects of the work ensuring that the questions related to the accuracy or integrity of any part of the work are appropriately investigated and resolved. Poonam Agarkar worked on the interpretation of results and manuscript preparation. She drafted the article or revised it critically for important intellectual content. All authors had lastly approved the final version.

REFERENCES

[1] A. Criminisi, P. Perez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Transaction on Image Processing*, vol. 13, pp. 1200–1212, 2004.

[2] M. Bertalmio, G. Sapiro, and V. Caselles, "Image inpainting," in *Proc. 27th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 2000, pp. 417–424.

[3] Y. J. Chen, B. Guo, R. Zeng, M. Yan, H. Xu, and Y. Wang, "Tongue image texture classification based on image inpainting and convolutional neural network," *Comput. Math. Methods Med*., 6066640, 2022.

[4] A. Pathak, J. Karmakar, D. Nandi, and M. K. Mandal, "Feature enhancing image inpainting through adaptive variation of sparse coefficients," *Signal Image Video Processing*, pp. 1–9, 2022.

[5] O. Guleryuz, "Nonlinear approximation-based image recovery using adaptive sparse reconstructions and iterated denoising-part I: Theory," *IEEE Transaction on Image Processing*, vol. 15, pp. 539–554, 2006.

[6] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "Patch match: A randomized correspondence algorithm for structural image editing," *ACM Trans Graphics*, vol. 28, no. 3, pp. 2–11, 2009.

[7] C. Ballester, M. Bertalmio, V. Caselles, G. Sapiro, and J. Verdera, "Filling-in by joint interpolation of vector fields and gray levels," *IEEE Transactions on Image Processing*, vol. 10, no. 8, pp. 1200–1211, 2001.

[8] S. Darabi, E. Shechtman, C. Barnes, D. B. Goldman, and P. Sen. "Image melding: Combining inconsistent images using patch-based synthesis," *ACM Transactions on Graphics (TOG)*, vol. 31, no. 4, pp. 82–221, 2012.

[9] J. B. Huang, S. B. Kang, N. Ahuja, and J. Kopf., "Image completion using planar structure guidance," *ACM Transactions on Graphics (TOG)*, vol. 33, no. 4, 129, 2014.

[10] A. Tsai, A. Yezzi, and A. S. Willsky, "Curve evolution implementation of the Mumford-Shah functional for image segmentation, denoising, interpolation, and magnification," *IEEE Transaction on Image Processing*, vol. 10, pp. 1169–1186, 2001.

[11] J. S hen, S. H. Kang, and T. F. Chan, "Euler's Elastica and curvature-based inpainting," *SIAM Journal of Applied Mathematics*, vol. 63, pp. 564–592, 2003.

[12] J. Shen and T. F. Chan, "Mathematical models for local non-texture in-paintings," *SIAM Journal of Applied Mathematics*, vol. 62, pp. 1019–1043, 2002.

[13] T. Ruzic and A. Pizurica, "Context-aware patch-based image inpainting using Markov random field modeling," *IEEE Transactions on Image Processing*, vol. 24, no. 1, pp. 444–456, 2014.

[14] K. H. Jin and J. C. Ye, "Annihilating filter-based low-rank Hankel matrix approach for image inpainting," *IEEE Transactions on Image Processing*, vol. 24, no. 11, pp. 3498–3511, 2015.

[15] Q. Guo, S. Gao, X. Zhang, Y. Yin, and C. Zhang, "Patch-based image inpainting via two-stage low-rank approximation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 6, pp. 2023–2036, 2017.

[16] H. Lu, Q. Liu, M. Zhang, Y. Wang, and X. Deng, "Gradient-based low-rank method and its application in image inpainting," *Multimedia Tools and Applications*, vol. 77, no. 5, pp. 5969–5993, 2018.

[17] J. Liu, S. Yang, Y. Fang, and Z. Guo, "Structure-guided image inpainting using homography transformation," *IEEE Transactions on Multimedia*, vol. 20, no. 12, pp. 3252–3265, 2018.

[18] D. Ding, S. Ram, and J. J. Rodríguez, "Image inpainting using nonlocal texture matching and nonlinear filtering," *IEEE Transactions on Image Processing*, vol. 28, no. 4, 2018, pp. 1705–1719.

[19] Q. Fan and L. Zhang, "A novel patch matching algorithm for exemplar-based image in-painting," *Multimedia Tools and Applications*, vol. 77, no. 9, 2018, pp. 10807–10821.

[20] G. Sridevi and S. S. Kumar, "Image inpainting based on fractional-order nonlinear diffusion for image reconstruction," *Circuits, Systems, and Signal Processing*, pp. 1–16, 2019.

[21] K. Li, Y. Wei, Z. Yang, and W. Wei, "Image inpainting algorithm based on TV model and evolutionary algorithm," *Soft Computing*, vol. 20, no. 3, pp. 885–893, 2016.

[22] H. Li, W. Luo, and J. Huang, "Localization of diffusion-based inpainting in digital images," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 12, pp. 3050–3064, 2017.

[23] A. Nan and X. Xi, "An improved criminisi algorithm based on a new priority function and updating confidence," in *Proc. 7th International Conference on Biomedical Engineering and Informatics*, Dalian, China, 2014, pp. 885–889.

[24] A. A. Efros and T. K. Leung, "Texture synthesis by non-parametric sampling," in *Proc. Seventh IEEE International Conference on Computer Vision*, Kerkyra, Greece, 1999, pp. 20–27, 1999.

[25] P. Yuan, X. Gong, S. Cao, J. Y. Guo, C. Y. Wang, and H. M. Zou, "A modified exemplar-based inpainting algorithm," *CRSSC-CWI-CGrC*, vol. 1, 2010.

[26] Z. Hou, "Criminisi image concealment algorithm based on priority function and blocking matching principle," *Technical Magazine of the Faculty of Engineering of the University of Zulia*, vol. 39, no. 9, pp. 203–209, 2016.

[27] N. Kawai, T. Sato, and N. Yokoya, "Diminished reality based on image inpainting considering background geometry," *IEEE Trans. Vis. Comput. Graph*, vol. 22, pp. 1236–1247, 2016.

[28] L. Shen, Y. Xu, and X. Zeng, "Wavelet inpainting with the ℓ0 sparse regularization," *Appl. Comput. Harmon. Anal*., vol. 41, pp. 26–53, 2016.

[29] B. M. Waller, M. S. Nixon, and J. N. Carter, "Image reconstruction from local binary patterns," in *Proc. 2013 International Conference on Signal-Image Technology and Internet-Based Systems*, Kyoto, Japan, December 2–5, 2013, pp. 118–123.

[30] H. A. Li, L. Hu, J. Liu, J. Zhang, and T. Ma, "A review of advances in image inpainting research," *Imaging Sci. J*., vol. 543, 2023.

[31] D. Rasaily and M. Dutta, "Comparative theory on image inpainting: A descriptive review," in *Proc. 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, Chennai, India, August 1–2, 2017, pp. 2925–2930.

[32] A. Bugeau, M. Bertalmío, V. Caselles, and G. Sapiro, "A comprehensive framework for image inpainting," *IEEE Trans. Image Process*, pp. 2634–2645, 2010.

[33] J. F. Aujol, S. Ladjal, and S. Masnou, "Exemplar-based inpainting from a variational point of view," *SIAM J. Math. Anal*., vol. 42, pp. 1246–1285, 2010.

[34] W. Casaca, M. Boaventura, M. P. D. Almeida, and L. G. Nonato, "Combining anisotropic diffusion, transport equation and texture synthesis for inpainting textured images," *Pattern Recognit. Lett*., vol. 36, pp. 36–45, 2014.

[35] A. Criminisi, P. Perez and K. Toyama, "Object removal by exemplar-based inpainting," in *Proc. 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2003, p. 8.

[36] V. K. Alilou and F. Yaghmaee, "Introducing a new fast exemplar-based inpainting algorithm," in *Proc. 2014 22nd Iranian Conference on Electrical Engineering (ICEE)*, Tehran, Iran, pp. 874–878, 2014.

[37] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "PatchMatch: A randomized correspondence algorithm for structural image editing," *Seminal Graphics Papers, Pushing the Boundaries*, vol. 2, no. 65, pp. 619–629, 2023.

[38] K. M. He and J. Sun, "Image completion approaches using the statistics of similar patches," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 12, pp. 2423–2435, 2014.

[39] M. Ghorai, S. Mandal, and B. Chanda, "A group-based image inpainting using patch refinement in MRF framework," *IEEE Transaction on Image Processing*, vol, 27, no. 2, pp. 556–567, February 2018.

[40] R. Bornard and E. Lecan, "Missing data correction in still images and image sequences," in *Proc. 10th ACM International Conference on Multimedia 2002*, 2022, pp. 355–361.

[41] S. B. Bhele, S. Shriramwar, and P. Agarkar, "An efficient texture-structure conserving patch matching algorithm for inpainting mural images," *Multimedia Tools and Applications*, vol. 82, 2023.