AUT Journal of Electrical Engineering

AUT J. Elec. Eng., 55(1) (2023) 11-20
DOI: 10.22060/eej.2022.20857.5444

Gradient-Controlled Gaussian Kernel for Image Inpainting

H. Noori

Department of Electrical Engineering, Vali-e-Asr University of Rafsanjan, Rafsanjan, Iran

ABSTRACT: Image inpainting is the process of filling in damaged or missing regions in an image by using information from known regions or known pixels of the image. One of the most important techniques for inpainting is convolution-based methods, in which a kernel is convolved with the damaged image iteratively. Convolution based algorithms are very quick, but they don't have good results in structures and textural regions and result in blurring. The kernel size in the convolution-based algorithm is a critical parameter. The large size results in edge blurring, and if the kernel size is small, the information may not be sufficient for reconstruction. In this paper, a novel convolution-based algorithm is proposed that uses known gradient of the pixels to construct a convolution mask. In this algorithm, the kernel size is controlled by the gradient of the image in the known regions. The algorithm computes the weighted sum of the known pixels in a neighborhood around a damaged pixel and replaces the value in the place of that damaged pixel. The proposed algorithm is fast and results in good edges and smooth regions reconstruction. It is an iterative algorithm and its implementation is very simple. Experimental results show the effectiveness of our algorithm.

Review History:

Received: Dec. 06, 2021
Revised: Jul. 28, 2022
Accepted: Aug. 01, 2022
Available Online: Mar. 01, 2023

Keywords:

Image inpainting
image reconstruction
image interpolation

1- Introduction

Image inpainting is the process of filling in damaged or missing regions in an image using information from known regions of the image. Fundamental applications of digital image inpainting include text, logo removal and all unwanted patterns removal from still images and videos, removing scratches or stains creating special effect in general. Most of the existing techniques follow a two-stage process: first damaged regions to be inpainted are selected (manually), then information is propagated inward from the surrounding area. Digital image inpainting was introduced by Bertalmio et al. [1]. It has attracted much attention in the research community and many different algorithms are proposed by researchers for this. The proposed methods in the state of the art can be divided into four main categories:

1- 1- Partial Differential Equation (PDE) based Algorithms

PDE based algorithms are designed to connect edges or extent isophotes better into damaged regions [1]-[9]. Bertalmio et al. [1] proposed a model inspired by artists who inpainted an image, based on a nonlinear PDE. The algorithm [1] propagates edges into missed regions and smooths uniform regions. Chan [2] proposed an algorithm using Euler elastica curves to connect appropriate level lines. The algorithm in [2] is a generalization of Total Variation (TV) and Bertalmio's algorithm [1]. Bertalmio et al. [3] used the similarities between equations in image processing and Navier-Stokes equations in dynamic fluid to propose a PDE based algorithm for image inpainting. Grossauer [4] proposed to use Ginzburg-Landau equation for image inpainting. This equation describes the phase transition in superconductors, and provides high contrast and low color smearing. A fourth order PDE based algorithm is presented in [5] based on minimizing the TV norm of an image. Li [6] proposed a nonlocal algorithm that minimizes the TV norm of images only in similar regions, using a texture synthesis method. In [7], Tai presented a two-step method for inpainting. First, the algorithm propagates isophote directions into damaged domains, using TV-Stokes equation. Afterwards, the image is restored to fit the constructed directions. In both steps nonlinear PDEs need to be solved. In [8], Telea presented an algorithm that can be looked as a PDE based method. The author calculated smoothness of an image in a known neighborhood of the current damaged pixel as a weighted average to inpaint current pixel. This method is fast, simple, and provides better results. Li et al. [9] proposed two fourth-order Partial Differential Equations (PDEs) to inpaint the image. By analyzing these two PDEs and comparing their diffusion images, the authors can confirm if they are forward diffusion or backward diffusion. Finally, PDE-based

*Corresponding author's email: h.noori@vru.ac.ir

algorithms need to solve a differential equation, which is very time consuming. In addition, PDE-based algorithms result in discontinuity and blurring at the edges when the damaged region is large.

1- 2- Texture Synthesis Algorithms

Texture synthesis algorithms use a sample region around a damaged pixel, compare all regions in the image with the sample region to find more similar regions in known parts of the image, and copy the best-founded region into the region to be inpainted [10]-[20]. Efros [14] proposed a texture synthesis method for texture inpainting. The algorithm first finds a set of pixels in an image whose neighborhoods are similar to the neighborhood around the current damaged pixel. The algorithm then keeps K most similar regions and chooses one of the central pixels of candidate regions randomly and copies its value to the current damaged pixel. Criminisi [15] exploits a patch based (exemplar based) algorithm in which the filling order is considered by a priority function ensuring that linear structures are preserved. In [16], a new priority function is introduced that improves Criminisi's work. Wong [17] proposed a nonlocal method for image inpainting. The algorithm uses multiple similar samples in an image and the contribution of each sample in construction damaged pixels is determined by means of a weighting similarity function. The algorithm in [18] automatically guides patch-based image inpainting. For each patch, the algorithm estimates planar projection parameters, then segments the known region into planes. Finally, the planes in the known regions are copied to the related damaged planes. In [19], Deng proposed a new function to calculate the priority of patches. Recently, Ding [20] proposed a new algorithm in which a priority function is first calculated to fill in the damaged region around the edges (to preserve the edges). In addition, a Gaussian-weighted nonlocal texture similarity measure is proposed to obtaine multiple candidate patches for each target patch. Additionally, the authors used α-trimmed mean filters to calculate the mean of the candidate patches.

There are several other works that gain both PDE-based 1-1 and texture synthesis 1-2 categories. Bertalmio [21] proposed to decompose an original image into two components with different characteristics: textures and structures. One is reconstructed by a texture synthesis algorithm and the other component is inpainted by a structure algorithm. In [22], the method utilizes image regularity statistics to extract dominant linear structures of the target image to reconstruct the edges. Guided by these structures, homography transformations are estimated and combined to globally repair the missing regions using the Markov random field model. Finally, all exemplar based (texture synthesis) algorithms need to search in all regions of the image to find candidate patches, therefore are too time consuming. To solve this, some papers proposed to copy the whole patch, which leads to edge discontinuity; but a large time is still needed to search all over the image.

1- 3- Deep Network Based Algorithms

Recently, several algorithms are proposed to inpaint damaged regions of images based on deep learning methods. In these algorithms, a network is first trained with several images. Next, a damaged region is given to the trained network and estimates the damaged pixels. In [23], a new energy function is considered to compute the error of the network. The weights change to minimize this energy function. In [24], an encoder-decoder model is proposed to video inpainting, the encoder provides visible pixels revealed from the scene dynamics. These pixels are fed into the encoder. Afterwards, using temporal consistency, two architectures are proposed. The first model fills the text on the frames of video. The second architecture is designed to fill large holes. In [25], semantic conditions are explored to give better restored face. By leveraging the advantages that Bayesian decision theory deals with uncertainty, the proposed framework exploits deep representation into Bayesian decision theory and derives a deep representation calibrated evidence lower bound. Finally, all deep network-based methods need to train with a large database and this process is very time consuming. Additionally, the network should be trained by the related database for different scenes restoration.

1- 4- Convolution-based Algorithms

All the mentioned categories are quite slow. In contrast, convolution-based methods are fast and simple to implement [26]-[30]. The authors [26] proposed a new algorithm that automatically distinguishes the damaged pixels from the known pixels and inpaints them. The gated convolution is trained by millions of images. Oliveira [27] proposed a fast image inpainting algorithm by convolving the borders of the damaged areas with a 3×3 (fixed) mask, and shrinking the degraded regions. In [28], Hadhood modified the Oliveira's method by considering only known pixels for averaging (fixed mask). Both methods result in blurring the edges. In [29], a convolution-based method is proposed in which the convolution kernel is constructed based on the difference of the pixels value distance (mask with variable weights). The proposed method has two parameters to be determined by the user. In [30], the authors improved the work in [29] by minimizing an energy function to find the patches. Finally, all convolution-based methods result in blurring when damaged region is large or curved. To overcome this drawback, we propose a novel convolution-based algorithm in which the gradient of the image is utilized to compute the weights in the kernel, which will preserve the edges. The proposed algorithm uses both vertical and horizontal gradient to calculate weights in a Gaussian convolving mask. The proposed algorithm is iterative, very simple to implement, and provides better results.

The paper is organized as follows. Section 2 presents the proposed algorithm. To evaluate the performance of the proposed method, some experimental results are given in Section 3. Finally, we conclude the paper in section 4.

2- Proposed Convolution based Image Inpainting

In this section, a new method based on convolution is proposed to reconstruct damaged regions of an image. As stated before, in the convolution-based method, a mask (kernel) is convolved by the degraded image. The main drawback of these methods is that they usually utilize the same mask for different regions of an image (i.e. a similar mask (in weights or in size)) is used for both edge and smooth regions. This leads to smoothing the edges or image blurring. Moreover, from the interpolation point of view, the more information is considered in the interpolation process, the more accurate approximation can be obtained. In order to take more information into account for the reconstruction process, the size of the convolving kernel should be increased in convolution-based methods, while results in blurring in the discontinuous regions, such as the edges. To avoid blurring and to be accurate simultaneously when convolving, the kernel size should be large when the damaged region of the image is in a smooth area to be more accurate, while the kernel size should be small in discontinuous regions to avoid blurring. To reach this goal, it is proposed to use the gradient of the image to distinct smooth regions from non-smooth regions.

To reach this goal, it is proposed to use a Gaussian function for computing the weights, while the kernel size is controlled by the image gradient. As we know, in the smooth regions, the gradient of an image is low and approaches to zero, while the gradient of an image in the regions including discontinuity like the edges is high. Therefore, if the gradient is in the exponent of the Gaussian function, it can control the kernel size. Consider pixel i in the boundary of the damaged region and an $m \times m$ neighborhood around it. The information is transferred from known pixels in this neighborhood to the damaged pixel i. The approximated value of the damaged pixel is the weighted average of the known pixels in this neighborhood. The weight for the known pixel j in the kernel is calculated as follows:

$$w_{ij} = e^{\frac{(x_i - x_j)^2 + (y_i - y_j)^2}{\frac{1}{\nabla^j_{max} I}}} \quad (1)$$

where x_i and y_i are the coordinates of pixel i, and $\nabla^j_{max} I$ is the maximum of the gradient of the image in pixel j. $\nabla^j_{max} I$ is calculated as follows:

$$\nabla^j_{max} I = \max\{|\nabla^j_v I|, |\nabla^j_h I|\} \quad (2)$$

Note that norm 2 can be chosen for $\nabla^j_{max} I$ as well. However, it needs more mathematical operations when norm 2 is used, and it is faster to use maximum instead of norm 2 while calculating the gradient or edges. In (2), $\nabla^j_v I$ and $\nabla^j_h I$ show the vertical and the horizontal gradient of the image, which are calculated as follows:

$$\nabla^j_v I = I(x_j, y_j + 1) - I(x_j, y_j - 1) \quad (3)$$

$$\nabla^j_h I = I(x_j + 1, y_j) - I(x_j - 1, y_j) \quad (4)$$

If the weights are calculated according to equation (1), the gradient increases and the value of Gaussian function decreases in the regions including edges (i.e. the weights decrease). Therefore, even if the kernel size (m) is large, the weights assigned to corresponding pixels are low. This way, information is transferred from a little number of pixels, and this avoids averaging in the regions including edges, and therefore avoids image blurring. On the contrary, the gradient becomes low and the weights assigned to corresponding pixels becomes high in the smooth regions of an image. This way, more pixels are taken into account in the averaging process and the approximation is more accurate. When a pixel is on the edge, the kernel size decreases and when a pixel is in the smooth region, the kernel size increases. In fact, when a pixel is in the smooth region (i.e. when the kernel size increases), more pixels are participated in calculating the damaged pixel and this pixel is reconstructed smoothly. If a pixel is on an edge, less number of pixels are participated in the reconstruction and the damaged pixel is reconstructed sharply, which is necessary when pixel is on the edge. Additionally, it is necessary to compare the proposed kernel with the bilateral kernel [29]. In bilateral approach, the kernel is computed by two factors, the first is the gray level difference between two pixels and second is the Euclidean distance of two pixels. However, we used the gradient of pixel for computing the kernel in the proposed method. Where the gradient is high (i.e. edge), the kernel size decreases by assigning zero to more pixels and where the gradient is small (i.e. smooth regions), the kernel size increases and calculates the average of the values of the pixels. However, in the bilateral approach, even if the two values of the pixels are the same but in a large distance, the weight assigned to that pixel for reconstruction becomes low. Finally, the value of the damaged pixel i is calculated as the following weighted sum.

$$\tilde{I}(x_i, y_i) = \frac{1}{W} \sum_{j \in \mathcal{N}} w_{ij} I(x_j, y_j) M(x_j, y_j) \quad (5)$$

where $\mathcal{N}$ is the $m \times m$ neighborhood of pixel i, W is the normalizing factor $W = \sum_{j \in \mathcal{N}} w_{ij} M(x_j, y_j)$, and $M(x_j, y_j)$ is a mask that shows whether the pixel is known or not (i.e. $M(x_j, y_j) = 1$ if the pixel is known and $M(x_j, y_j) = 0$ if

Table 1. Inpainting Algorithm

Algorithm 1
1: Get the damaged region Ω (get damaged region mask as a binary image) and the input image $I(x, y)$.
2: $\Omega_1 = \Omega$
3: **For** iteration = 1 to T
4: $\Omega = \Omega_1$
5: **While** $\Omega \neq \emptyset$ **do**
6: $\partial\Omega \leftarrow$ the boundary of damaged regions
7: **For** all pixels in $\partial\Omega$ **do**
8: $\mathcal{N} \leftarrow$ neighborhood of each pixel
9: Calculate the weights using (1)
10: Replace the pixel value of the image $I(x, y)$ in $\partial\Omega$ using (5)
11: **End For**
12: Shrink the damaged region Ω
13: **End While**
14: **End For**

the pixel is damaged). We consider the mask M to avoid transferring information from damaged regions. Therefore, the algorithm should start from where the information is known. To this goal, first the pixels in the boundary of the damaged region are selected and the weighted sum is calculated. Next, a value is calculated for each damaged pixel in the boundary of the damaged region. Afterwards, the damaged region is shrunk. This process continues until damaged region vanishes. All the above process is repeated several times to result in a better result. The proposed algorithm is shown in Table 1. In all iterations, the damaged region is given to the algorithm, and is filled by the algorithm. In the first iteration, the information is propagated only from the known surrounding pixels; but in the other iterations, the information is propagated from all the surrounding pixels. This is because in iteration 1 the damaged pixels are unknown, but in other iterations the damaged pixels are reconstructed in the iteration 1. In the proposed algorithm, first the damaged region (which we want to inpaint) is detected manually in mask Ù, and copied to $Ù_1$ as lines 1 and 2 of the algorithm illustrate. Next, the algorithm goes to line 6 and the boundary of the damaged region is extracted. Afterwards, the algorithm finds values for damaged pixels in lines 7 to 9 and all the damaged pixels in the boundary of damaged region in the original image is replaced in line 10 and ends for the loop; next in line 11 of the algorithm, the damaged region is shrunk. After that, the algorithm returns to line 5 and the steps are repeated till all the damaged pixels are filled. It is necessary to note that Ù, which is a binary mask that separates damaged pixels and known pixels with zeros and ones respectively, is updated in each iteration, while loop in line 12 and the locations according to the damaged pixels of the original image in Ù change to one. Therefore, the value of Ù (i.e. location of damaged pixels) is lost in each iteration (the loop) and should be re-initialized for the next iteration, which is done in line 4. Note that locations according to damaged pixels of the original image in Ù get zero value before inpainting while all locations according to damaged pixels of the original image in Ù get value one after one iteration, therefore Ù is fully filled with ones. At this time, one iteration of the algorithm is completed and the algorithm returns to line 3 and the next iteration starts. Fig. 6 shows how MSE decreases when the number of iterations increases.In the next section, the proposed algorithm is evaluated and compared to other algorithms.

3- Simulation Results

In this section, the proposed algorithm is evaluated and compared with several state-of-art algorithms ([9],[20],[28],[29],[30]) in both qualitative and quantitative. Usually, the original image doesn't exist in the image inpainting, therefore there is no well-defined criterion for quantitative comparison. However, to have a fair comparison, we degrade several images intentionally and reconstruct them by the proposed algorithms and some algorithms in the state of art and compute the Mean Square Error (MSE) by eq. (6) for each algorithm. Additionally, we compute the implementation time of each algorithm to compare the speed of each one. All the algorithms are implemented by Matlab R2019b on an Intel core i7-6500 CPU 2.50 GHz with 8 GB RAM on a 64-bit operating system. In all simulations, the kernel size of Gaussian in the proposed method is 13×13 and the number of iterations is fixed to 600 iterations, which may not be needed in all figures.

$$MSE = \frac{1}{NK}\sum_{i=1}^{NK}\left(f(x_i, y_i) - \hat{f}(x_i, y_i)\right)^2 \qquad (6)$$

Where N and K are the number of rows and columns of the image, f shows the original image and $\hat{f}$ shows the restored image.

In Fig. 1, an image (384×256) with very sharp straight edges is degraded with a text (1b). As Figs. 1c-1g show the performance of the algorithm of Ding [20] suppress the other the state-of-the-art algorithms; but the proposed algorithm performs better than Ding's algorithm [20], as it can be seen from the above of color pencils.

In Fig. 2, an image (267×182) with smooth edges is degraded with some lines with different sizes (2b). In this figure, all algorithms perform similarly. The algorithm proposed by Ding [20] is wrong in finding similar patches due to the rapid change in gradient of the image, which results in bad restoration. All the convolution-based algorithms result in blurring the image, especially at the edge of lemon. In this image, the edges are not sharp, therefore the obtained gradient is low and cannot guide neither the exemplar-based approaches nor gradient based methods.

In Fig. [3], an image (384×256) including curved sharp edges and some textural regions is degraded with some lines (3b). It can be seen from this figure that all algorithm result in discontinuity in the edges, but the proposed algorithm has the best performance. For example, in the sharp edge in the lowest orange, no algorithm can restore the edges without artifact except the proposed method (see the lowest degraded region in the edge).

In Fig. 4, degraded region is on the elbow of the cameraman (256×256), which there is no similar other region and is a curved sharp edge. As can be seen in the Figs. 4c,4d, and 4e, all convolution-based methods [28], [29], and [30] cannot restore this kind of edge. As Fig. 4g shows, PDE based methods ([9]) result in blurring in this kind of edges. Additionally, as Fig. 4f shows, Ding's algorithm [20] doesn't have a good performance since there is no similar edge in this figure, consequently, the Ding's algorithm [20] cannot find the similar patch and doesn't result in good restoration. However, the proposed method controlled by the local gradient of the image can restore the elbow of cameraman well. Although the restored image is not completely the same, it is not blurred and has no artifact (i.e., the restored image is satisfactory). In Fig. 5, an object removal is considered in which we try to remove the man from the sunset image (300×168). Fig. 5b shows the mask for removing the man. As it can be seen from this figure, no convolution-based algorithm and no PDE based method can restore the image satisfactorily. This is because the gradient in the textural area changes very rapidly, therefore the information cannot be transferred from known pixels to unknown pixels. However, it as can be seen from Fig. 5f that Ding's algorithm [20] can restore the image very well since it can find similar patches in this image. Moreover, this algorithm results to discontinuity at the edge of the road.To compare the performance of the proposed algorithm with the state of the art, the MSE between the original image and the restored image is shown in Table 2. As this table shows, the proposed method results in less MSE among the algorithms, and has a good visible quality. Therefore, the simulation results show the effectiveness of the proposed method. Additionally, in Table 3, the implementation time of the algorithms is compared. The PDE based algorithms and the exemplar-based algorithms need too much time for restoration, while the convolution-based algorithms are nearly real time. As can be seen, the proposed method has a comparable run time with convolution based like Bilateral [29]. It should be noted that in figure 1 to 4 the algorithm tries to reconstruct a damaged image whose original image is available and comparison between the reconstructed and the original image is feasible. However, in figure 5, the algorithm tries try to remove the person from image, therefore the reconstructed image is not similar to the original and therefore the MSE could not be computed for Fig. 5.As stated before, there is no criterion for image inpainting. In this paper, MSE is considered to show the performance of the proposed method. Another important factor in the inpainting algorithms is the implementation time. Considering both the implementation time and MSE, the results shows that the proposed algorithm has better MSE and an intermediate implementation time.

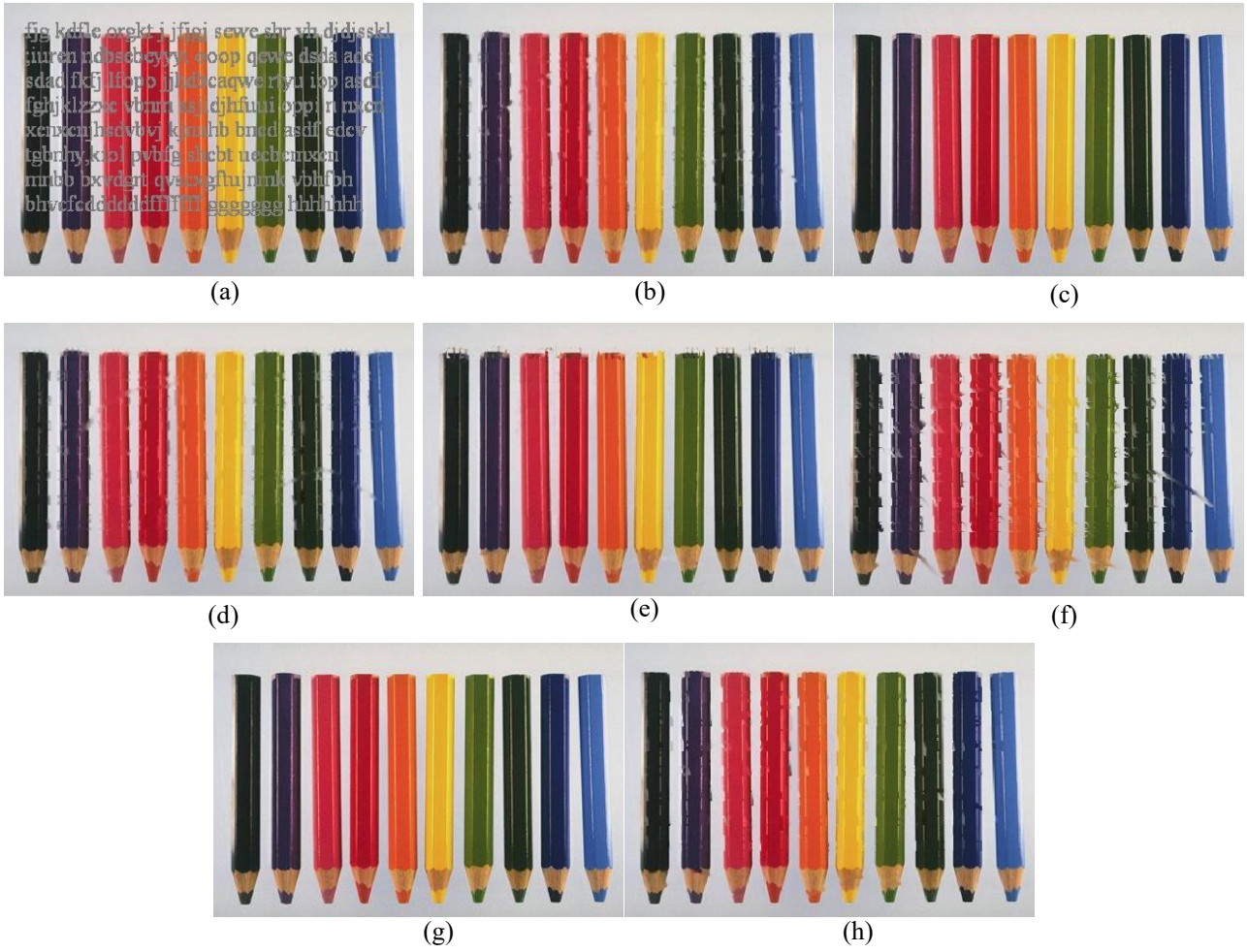
In addition, there are other inainting results for interested reviewer on the web: https://cloud.vru.ac.ir/s/4AQiosBKctDDpb9

Table 2. MSE between the original image and the restored image

Algorithm	Li [9]	Bilateral [29]	Hadhood [28]	Anh [30]	Ding [20]	Proposed
Fig. 1	1.441	0.857	3.825	0.884	0.4	**0.281**
Fig. 2	2.007	0.387	0.725	**0.225**	1.092	0.433
Fig. 3	0.277	0.220	0.661	0.21	0.362	**0.184**
Fig. 4	0.047	0.042	0.262	0.038	0.038	**0.012**

Table 3. Implementation time (s)

Algorithm	Li [9]	Bilateral [29]	Hadhood [28]	Anh [30]	Ding [20]	Proposed
Fig. 1	40381.232	141.153	0.024	4.647	66182.095	1323.940
Fig. 2	665.854	2.802	0.009	1.427	145.477	32.482
Fig. 3	22927.482	43.626	0.126	2.478	35107.882	71.125
Fig. 4	42.705270	12.958	0.007	1.240	31.067	30.984



(a) (b) (c) (d) (e) (f) (g) (h)

Fig. 1. (a) original image (b) degraded image, restored by (c) Bilateral [29], (d) Hadhood [28], (e) Anh [30], (f) Ding [20], (g) Li [9], (h) Proposed method

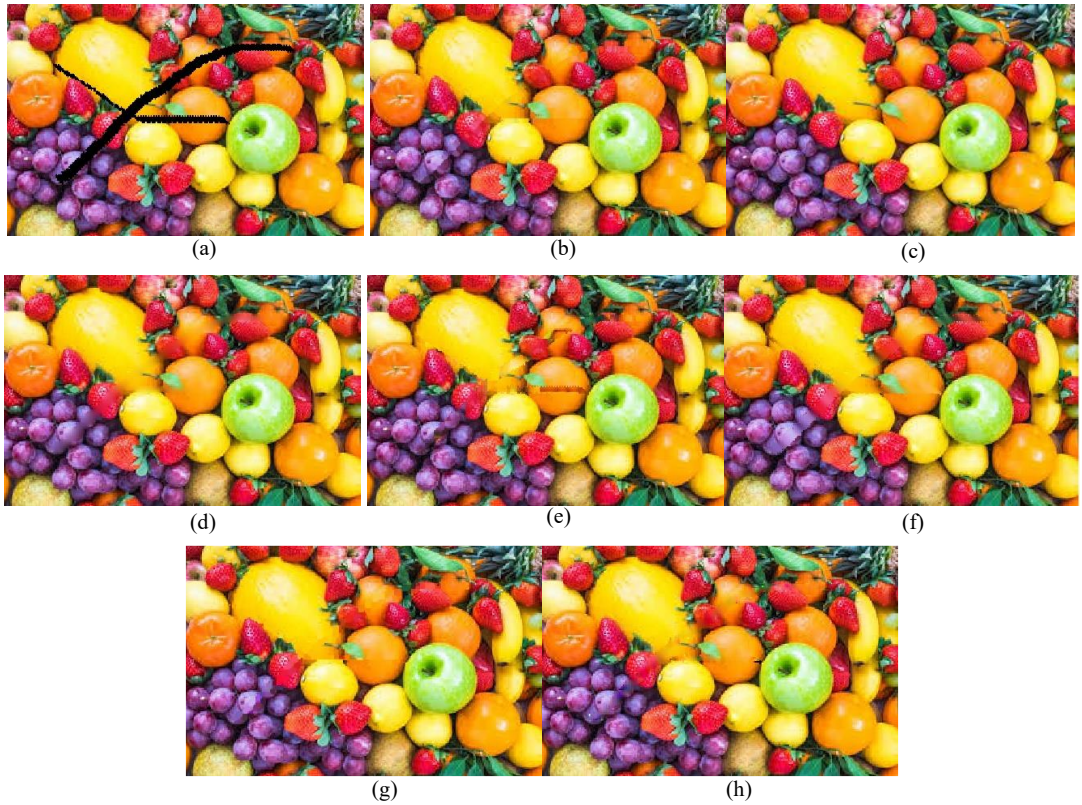


Fig. 2. (a) original image (b) degraded image, restored by (c) Bilateral [29], (d) Hadhood [28], (e) Anh [30], (f) Ding [20], (g) Li [9], (h) Proposed method

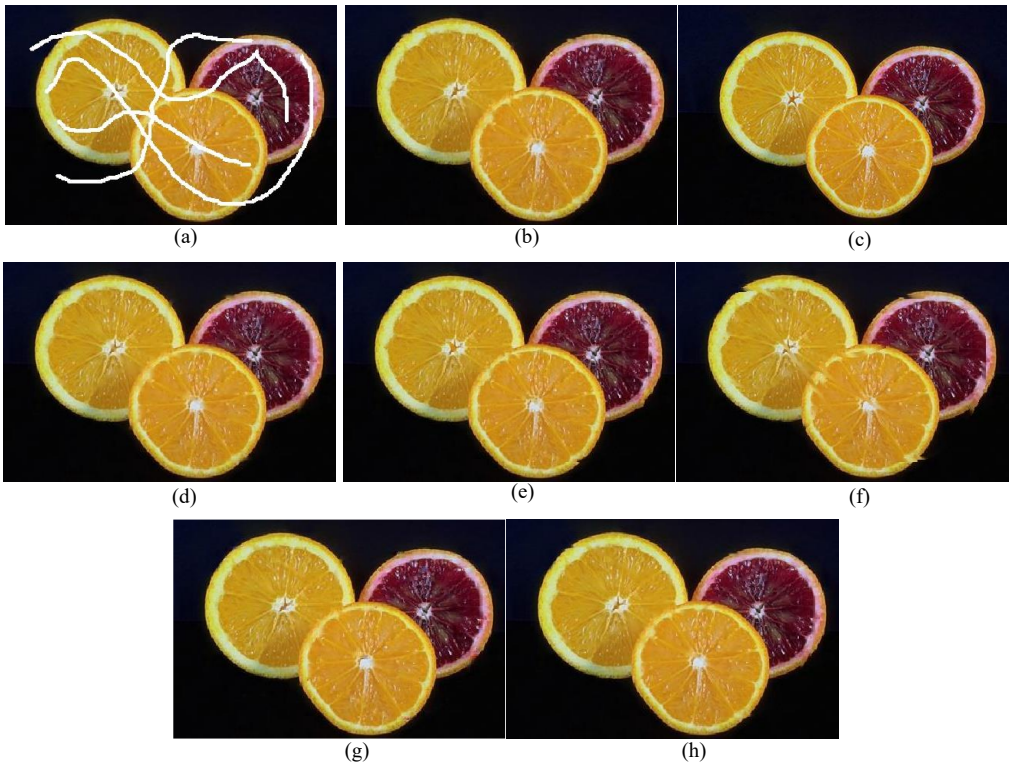


Fig. 3. (a) original image (b) degraded image, restored by (c) Bilateral [29], (d) Hadhood [28], (e) Anh [30], (f) Ding [20], (g) Li [9], (h) Proposed method

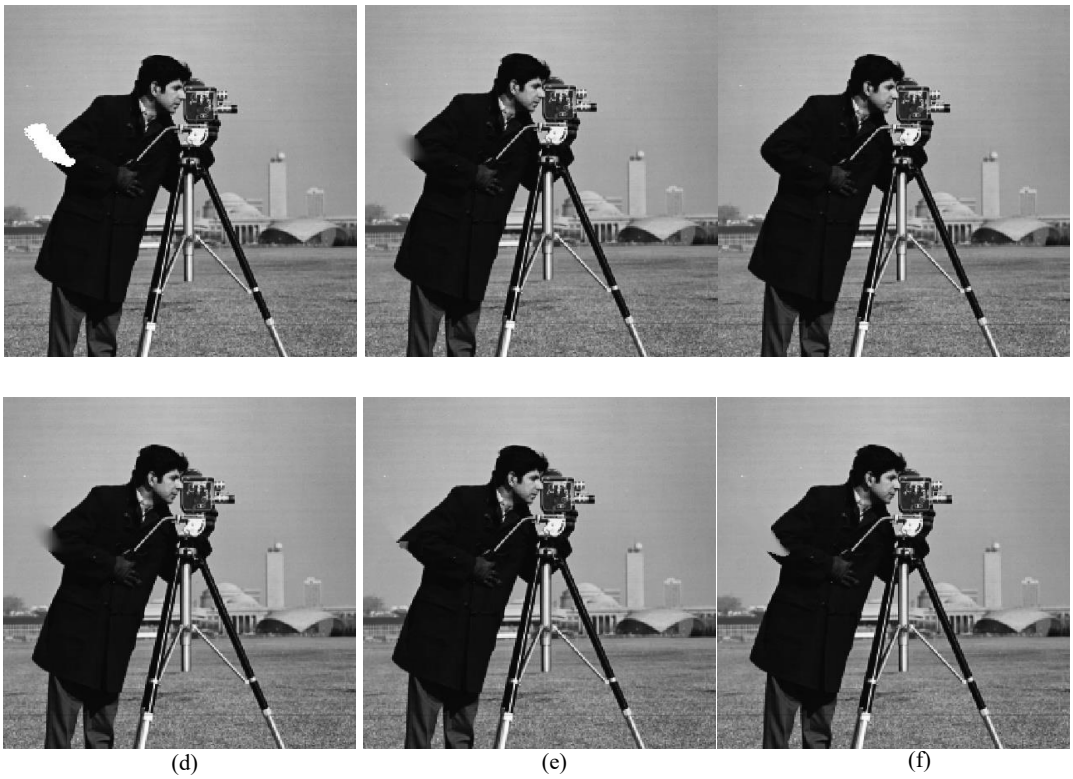


Fig. 4. (a) original image (b) degraded image, restored by (c) Bilateral [29], (d) Hadhood [28], (e) Anh [30], (f) Ding [20], (g) Li [9], (h) Proposed method

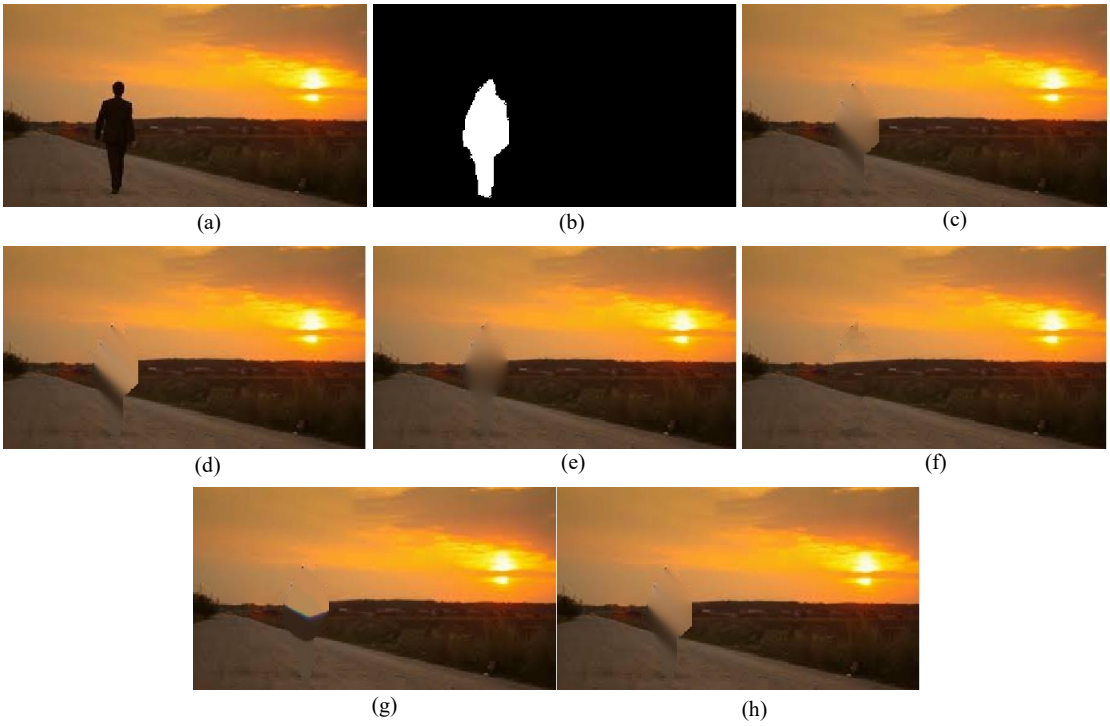


Fig. 5. (a) original image (b) degraded image, restored by (c) Bilateral [29], (d) Hadhood [28], (e) Anh [30], (f) Ding [20], (g) Li [9], (h) Proposed method

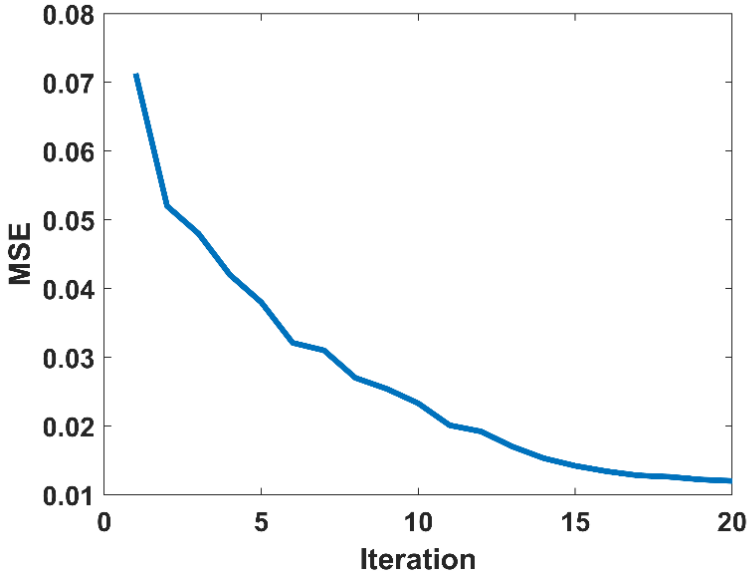


Fig. 6. (a) original image (b) degraded image, restored by (c) Bilateral [29], (d) Hadhood [28], (e) Anh [30], (f) Ding [20], (g) Li [9], (h) Proposed method

Additionally, the proposed method is applied to Places2 dataset, and the results are compared with the algorithm in [31]. The Fréchet Inception Distance (FID) score is obtained as 32.1 for the proposed algorithm, while it is obtained as 11.7 for [31]. Therefore, the FID scores confirm that [31] results in much better reconstruction than the proposed method. Additionally, the implementation time of the algorithm in [31] is three times faster than the proposed method, but it is necessary to mention that in this comparison, we consider only prediction time for [31] and the learning time isn't considered.

4- Conclusion and Future Work

In this paper, a new convolution-based method for image inpainting is proposed. the proposed method utilizes the image gradient in the exponent of Gaussian function when calculating the weights. In this way, the size of Gaussian kernel can be controlled in different regions of the image. The experimental results confirm the effectiveness of the algorithm. However, as we see in the formulation, the gradient of the image is calculated only in two vertical and horizontal orientations. In the future work we will try to improve this to get better results.

References

[1] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image Inpainting," SIGGRAPH: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, pp. 417 -424, 2000.

[2] T. F. Chan, J. Shen, S. H. Kang, "Euler's elastica and curvature based inpainting," SIAM Journal of Applied Mathematics, vol. 63, I. 2, pp. 564-592, 2002.

[3] M. Bertalmio, A. L. Bertozzi and G. Sapiro, "Navier-stokes fluid dynamics, and image and video inpainting", Proceedings of IEEE Computer Vision and Pattern Recognition (CVRP), 2001.

[4] H. Grossauer and O. Scherzer, "Using complex ginzburg-landau equation for digital inpainting in 2d and 3d," Scale Space Method in Computer Vision, Lecture Notes in Computer Science, 2695, 2003.

[5] P. Chen and Y. Wang, "Fourth-order partial differential equations for image inpainting", International Conference on Audio, Language and Image Processing, Shanghai, pp. 1713-1717, 2008.

[6] L. li and H. Yu, "Nonlocal curvature-driven diffusion model for image inpainting," Fifth International Conference on Information Assurance and Security, Xi'an, pp. 513-516, 2009.

[7] X. C. Tai, S. Osher, and R. Holm, "Image inpainting using a TV-stokes equation," Mathematics and Visualization, Springer, Berlin, Heidelberg, pp 3-22, 2007.

[8] A. Telea, "An image inpainting technique based on the fast marching method," Journal of Graphics Tools, vol. 9, no. 1, 2004.

[9] P. Li, S. Li, Z. Yao, and Z. Zhang, "Two anisotropic fourth-order partial differential equations for image inpainting," IET Image Processing, vol. 7, no. 3, pp. 260–269, Jun. 2013.

[10] M. Ghorai, S. Samanta, S. Mandal and B. Chanda, "Multiple pyramids-based image inpainting using local patch statistics and steering kernel feature," IEEE Transactions on Image Processing, vol. 28, no. 11, pp. 5495-5509, Nov. 2019.

[11] K. 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, 1 Dec. 2014.

[12] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Generative image inpaining with contextual attention," IEEE Computer Vision and Pattern Recognition, pp. 5505–5514, 2018.

[13] H. Liu, X. Bi, G. Lu and W. Wang, "Exemplar-based image inpainting with multi-resolution information and the graph cut technique," IEEE Access, vol. 7, pp. 101641-101657, 2019.

[14] A. Efros and T. Leung, "Texture synthesis by non-parametric sampling," Proceedings of the Seventh IEEE International Conference on Computer Vision, pp. 1033 - 1038, vol. 2, Greece, 1999.

[15] A. Criminisi, P. P´erez, and K. Toyama, "Object removal by exemplar-based inpainting," IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Madison, WI, USA, pp1200-1212, 2003.

[16] [16] D. J. Tuptewar and A. Pinjarkar, "Robust exemplar-based image and video inpainting for object removal and region filling," International Conference on Intelligent Computing and Control (I2C2), Coimbatore, pp. 1-4, 2017.

[17] A. Wong and J. Orchard, "A nonlocal-means approach to exemplar-based inpainting," 15th IEEE International Conference on Image Processing, San Diego, CA, pp. 2600-2603, 2008.

[18] J. Huang, S. Kang, N. Ahuja, and J. Kopf, "Image completion using planar structure guidance," ACM Transaction on Graphics, vol. 33, no. 4, pp. 129:1–129:10, Jul. 2014. [Online]. Available: https://github.com /jbhuang0604/Struct Completion.

[19] L. Deng, T. Huang, and X. Zhao, "Exemplar-based image inpainting using a modified priority definition," Plos One, vol. 10, no. 10, pp. 1–18, Oct. 2015. [Online]. Available: http://www.escience.cn/people/dengliangjian/codes.html

[20] D. Ding, S. Ram and J. J. Rodr´ıguez, "Image inpainting using nonlocal texture matching and nonlinear filtering," IEEE Transactions on Image Processing, vol. 28, no. 4, pp. 1705-1719, April 2019.

[21] M. Bertalmio, L. Vese, G. Sapiro and S. Osher, "Simultaneous Structure and Texture Image Inpainting," IEEE Transactions on Image Processing, vol. 12, no. 8, pp. 882-889, Aug. 2003.

[22] 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, Dec. 2018.

[23] F. Altinel, M. Ozay and T. Okatani, "Deep structured energy-based image inpainting," 24th International Conference on Pattern Recognition (ICPR), Beijing, pp. 423-428, 2018.

[24] D. Kim, S. Woo, J. Lee and I. S. Kweon, "Recurrent temporal aggregation framework for deep video inpainting," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 42, no. 5, pp. 1038-1052, 1 May 2020.

[25] H. Xiong, C. Wang, X. Wang and D. Tao, "Deep representation calibrated bayesian neural network for semantically explainable face inpainting and editing," IEEE Access, vol. 8, pp. 13457-13466, 2020.

[26] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Free-form image inpainting with gated convolution," arXiv preprint arXiv:1806.03589, 2018.

[27] M. Oliveira, B. Bown, R. Mckenna, and Y. S. Chang, "Fast digital image inpainting," International Conference on Visualization, Imaging and Image Processing (VIIP 2001), Marbella, Spain, September 3-5, pp. 261-266, 2001.

[28] M. M. Hadhoud, K. A. Moustafa and S. Z. Shenoda , "Digital images inpainting using modified convolution based method," Proceedings of SPIE - The International Society for Optical Engineering, 2008.

[29] H. Noori, S. Saryazdi, and H. Nezamabadipoor, "A bilateral image inpainting," Iranian Journal of Science and Technology (IJST), Transactions of Electrical Engineering, vol. 35, no. E2, pp 95-108, 2011.

[30] D. N. Anh, "An adaptive bilateral filter for inpainting," Fourth International Conference of Emerging Applications of Information Technology, Kolkata, pp. 237-242, 2014.

[31] C. Saharia, W. Chan, H. Chang, Ch. Lee, J. Ho, T. Salimans, D. Fleet, M. Norouzi, "Palette: Image-to-Image Diffusion Models," arXiv:2111.05826v2 [cs.CV] ,3 May 2022.

HOW TO CITE THIS ARTICLE

H. Noori, Gradient-Controlled Gaussian Kernel for Image Inpainting, AUT J. Elec. Eng., 55(1) (2023) 11-20.

DOI: 10.22060/eej.2022.20857.5444