A NOVEL APPROACH OF DETECTION AND REMOVAL OF RAIN USING NEURAL NETWORKS

[1]Syed Sania, [2] Dr. K. Premalatha
[1]PG Scholar, MTech, Dept of ECE, Shadan Women's College of Engineering and Technology HYD, T.S.
[2]Professor, Dept of ECE, Shadan Women's College of Engineering and Technology HYD, T.S.

ABSTRACT

This image introduces a deep community architecture called DerainNet for putting off rain streaks from image. Based totally on the deep CNN, we immediately research the mapping dating among wet and clean picture layers from data. Because we will not possess the floor fact analogous to actual-world rainy pix, we combine pictures with rain for edification. In evaluation to other not surprising systems that blast power or expansiveness of the network, we use picture preparing territory learning to adjust the objective element and improve deraining with an unobtrusively measured CNN. In particular, we train our DerainNet on the detail layer rather than in the picture territory. In spite of the fact that DerainNet is taught on counterfeit insights, we find that the discovered network deciphers in all respects accurately to real worldwide depictions for giving it a shot. Moreover, we increment the CNN structure with picture upgrade to improve the obvious results. in examination with cutting edge single photo deraining strategies, our methodology has advanced downpour end and a lot faster calculation time after network instruction.

I. INTRODUCTION

The effects of rain can degrade the visual quality of images and severely affect the performance of outdoor vision systems. Under rainy conditions, rain streaks create not only a blurring effect in images, but also haziness due to light scattering. Effective methods for removing rain streaks are required for a wide range of practical applications, such as image enhancement and object tracking. We present the first deep convolutional neural network (CNN) tailored to this task and show how the CNN framework can obtain state-of-the-art results.

Figure 1 shows an example of a real-world testing image degraded by rain and our de-rained result. In the last few decades, many methods have been proposed for removing the effects of rain on image quality. These methods can be categorized into two groups: video-based methods and single-image based methods. We briefly review these approaches to rain removal, then discuss the contributions of our proposed DerainNet.

(a) Input rainy image (b) Our result
Fig. 1. An example real-world rainy image and our de-rained result.

Outdoor vision systems are used for various purposes such as tracking, recognition and navigation. Despite their widespread use, current systems do not account for common weather conditions such as rain, snow, fog and mist. In order to develop vision systems that perform under all weather conditions, it is essential to model the visual effects of the various weather conditions and develop algorithms to remove them. Weather conditions vary widely in their physical properties and in the visual effects they produce in images. Based on their differences, weather conditions can be broadly classified as steady (fog, mist and haze) or dynamic (rain, snow and hail). In the case of steady weather, individual droplets are too small ($1 - 10$ μm) to be visible to a camera, and the intensity produced at a pixel is due to the aggregate effect of a large number of droplets within the pixel's solid angle (see Figure 1(a)). Hence, volumetric scattering models such as attenuation and airlight [9] can be used to adequately describe the effects of steady weather. Algorithms [10] have been recently developed to remove the effects of steady weather from images. On the other hand, the constituent particles of dynamic weather conditions such as rain, snow and hail are larger (0.1−10mm) and individual particles are visible in the image. An example is shown in Figure 1(b), the streaks of rain are caused by individual drops. Here, aggregate scattering models previously used for steady conditions are not applicable. The analysis of dynamic weather conditions requires the development of stochastic models that capture the spatial and temporal effects of a large number of

particles moving at high speeds (as in rain) and with possibly complex trajectories (as in snow).

In this work, we focus on the problem of rain. Rain consists of a distribution of a large number of drops of various sizes, falling at high velocities. Each drop behaves like a transparent sphere, refracting and reflecting light from the environment towards the camera. An ensemble of such drops falling at high velocities results in time varying intensity fluctuations in images and videos. In addition, due to the finite exposure time of the camera, intensities due to rain are motion blurred and therefore depend on the background. Thus, the visual manifestations of rain are a combined effect of the dynamics of rain and the photometry of the environment. Rain has been studied extensively in the fields of atmospheric sciences, signal communication and remote sensing [8, 13, 6]. Most of these studies use active illumination sources (lasers) and specialized detectors (photo-cells) to examine the effects of rain on a transmitted signal. However, the effects of rain on a camera viewing a scene in a natural environment are very different and remain unexplored.

In computer graphics, rain has been rendered using particle systems [11] or heuristic models [12]. However, these rendering methods are not based on the physical properties of rain and fail to represent the complex visual effects of rain. One possible approach to analyzing the appearance of rain is to learn a model from real rain videos (for instance, dynamic textures [2]). However, as mentioned before, the appearance of rain depends on several factors including the physical properties of rain, the environment and the camera settings. Hence, learning a general model for the appearance of rain in an arbitrary scene and camera setting is hard.

II. LITERATURE REVIEW

Due to the redundant temporal information that exists in video, rain streaks can be more easily identified and removed in this domain [1]–[4]. For example, in [1] the authors first propose a rain streak detection algorithm based on a correlation model. After detecting the location of rain streaks, the method uses the average pixel value taken from the neighboring frames to remove streaks. In [2], the authors analyze the properties of rain and establish a model of visual effect of rain in frequency space. In [3], the histogram of streak orientation is used to detect rain and a Gaussian mixture model is used to extract the rain layer. In [4], based on the minimization of registration error between frames, phase congruency is used to detect and remove the rain streaks. Many of these methods work well, but are significantly aided by the temporal content of video. In this paper we instead focus on removing rain from a single image.

Compared with video-based methods, removing rain from individual images is much more challenging since much less information is available for detecting and removing rain streaks. Single-image based methods have been proposed to deal with this challenging problem, but success is less noticeable than in video-based algorithms, and there is still much room for improvement. To give three examples, in [5] rain streak detection and removal is achieved using kernel regression and a non-local mean filtering. In [6], a related work based on deep learning was introduced to remove static raindrops and dirt spots from pictures taken through windows. This method uses a different physical model from the one in this paper. As our later comparisons show, this physical model limits its ability to transfer to rain streak removal. In [7], a generalized lowrank model in which rain streaks are assumed to be low rank is proposed. Both single-image and video rain removal can be achieved by characterizing spatio-temporally correlations of rain streaks.

Recently, several methods based on dictionary learning have been proposed [8]–[12]. In [9], the input rainy image is first decomposed into its base layer and detail layer. Rain streaks and object details are isolated in the detail layer while the structure remains in the base layer. Then sparse coding dictionary learning is used to detect and remove rain streaks from the detail layer. The output is obtained by combining the de-rained detail layer and base layer. A similar decomposition strategy is also adopted in method [12]. In this method, both rain streaks removal and non-rain component restoration is achieved by using a hybrid feature set. In [10], a self-learning-based image decomposition method is introduced to automatically distinguish rain streaks from the detail layer. In [11], the authors use discriminative sparse coding to recover a clean image from a rainy image. A drawback of methods [9], [10] is that they tend to generate over-smoothed results when dealing with images containing complex structures that are similar to rain streaks, as shown in Figure 9(c), while method [11] usually leaves rain streaks in the de-rained result, as shown in Figure 9(d). Moreover, all four dictionary learning based frameworks [9]–[12] require significant computation time. More recently, patch-based

priors for both the clean and rain layers have been explored to remove rain streaks [13]. In this method, the multiple orientations and scales of rain streaks are addressed by pre-trained Gaussian mixture models.

Contributions of our DerainNet approach

As mentioned, compared to video-based methods, removing rain from a single image is significantly more difficult. This is because most existing methods [9]–[11], [13] only separate rain streaks from object details by using low level features, for example by learning a dictionary for object representation. When an object's structure and orientation are similar with that of rain streaks, these methods have difficulty simultaneously removing rain streaks and preserving structural information. Humans on the other hand can easily distinguish rain streaks within a single image using high-level features such as context information. We are therefore motivated to design a rain detection and removal algorithm based on the deep convolutional neural network (CNN) [14], [15]. CNN's have achieved success on several low-level vision tasks, such as image denoising [16], super-resolution [17], [18], image deconvolution [19], image inpainting [20] and image filtering [21]. We show that the CNN can also provide excellent performance for single-image rain removal.

III. PROPOSED SYSTEM

In this paper, we propose "DerainNet" for removing rain from single-images, which we base on the deep CNN. To our knowledge, this is the first approach based on deep learning to directly address this problem. Our main contributions are threefold:

1) DerainNet learns the nonlinear mapping function between clean and rainy detail (i.e., high resolution) layers directly and automatically from data. Both rain removal and image enhancement are performed to improve the visual effect. We show significant improvement over three recent state-of-the-art methods. Additionally, our method has significantly faster testing speed than the competitive approaches, making it more suitable for real-time applications.

2) Instead of using common strategies such as increasing neurons or stacking hidden layers to effectively and efficiently approximate the desired mapping function, we use image processing domain knowledge to modify the objective function and improve the de-rain quality. We show how better results can be obtained without introducing more complex network architecture or more computing resources.

3) Because we lack access to the ground truth for real world rainy images, we synthesize a dataset of rainy images using real-world clean images, which we can take as the ground truth. We show that, though we train on synthesized rainy images, the resulting network is very effective when testing on real-world rainy images. In this way, the model can be learned with easy access to an unlimited amount of training data.

IV. DERAINNET: DEEP LEARNING FOR RAIN REMOVAL

We illustrate the proposed DerainNet framework in Figure 2. As discussed in more detail below, we decompose each image into a low-frequency base layer and a high-frequency detail layer. The detail layer is the input to the CNN for rain removal. To further improve visual quality, we introduce an image enhancement step to sharpen the results of both layers since the effects of heavy rain naturally leads to a hazy effect.

A. Training on high-pass detail layers

We denote the input rainy image and corresponding clean image as I and J respectively. Initially, a goal may be to train a network architecture $h_P(.)$ that minimizes

$$L = \frac{1}{N}\sum_{n=1}^{N} \|f\mathbf{w}(\mathbf{I}^n) - \mathbf{J}^n\|_F^2,$$

where W are the network parameters and F is the Frobenius norm and n indexes the image. However, we found that the result obtained by directly training in the image domain is not satisfactory. In Figure 3(a), we show an example of a synthetic rainy image. Note that this image is used in the training process. In Figure 3(b) we see that even when this image is used as a training sample, the de-rained image still exhibits clear rain streaks when zoomed in. Figure 3(b) implies that the desired mapping function was not well learned when training on the image domain, i.e., the model under-fit the data. It is natural to ask whether it is necessary to train a more complex model to further improve the capacity of the network. As is well-known, there are two ways to improve a network's capacity in the deep learning domain.

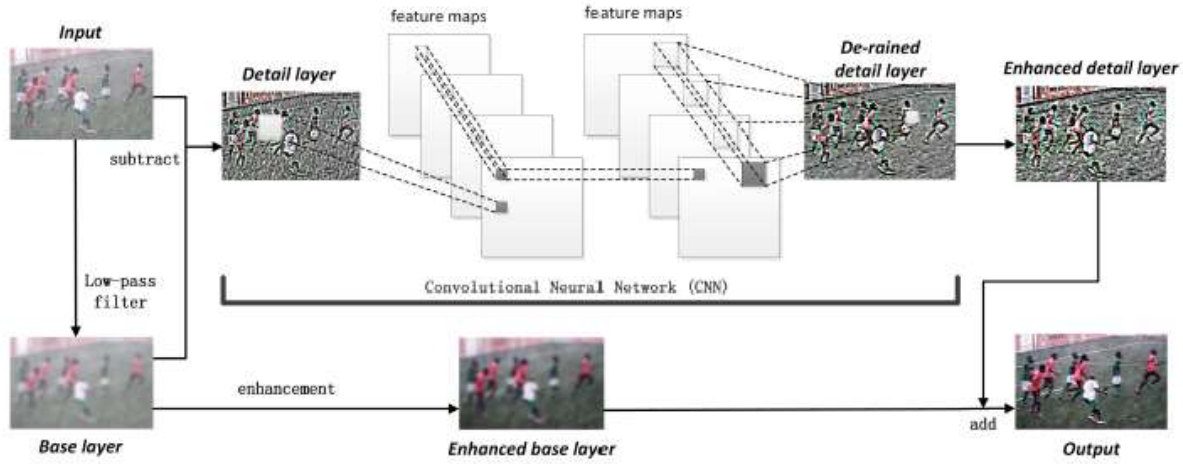


Fig. 2. The proposed DerainNet framework for single-image rain removal.

One way is to increase the depth of network [22] by stacking more hidden layers. Usually, more hidden layers can help to obtain high-level features. However, the de-rain problem is a low-level image task and the deeper structure is not necessarily better for this image processing problems. Furthermore, training a feed-forward network with more layers suffers from gradient vanishing unless other training strategies or more complex network structures are introduced. When we add network depth to improve the modeling ability, the result actually becomes worse. The other approach is to increase the breadth of network [23] by using more neurons in each hidden layer. However, to avoid over-fitting, this strategy requires more training data and computation time that may be intolerable under normal computing condition.

To address these issues for the de-raining problem, we instead use a priori image processing knowledge to modify the objective function rather than increase the complexity of the problem. Conventional end-to-end procedures directly uses image patches to train the model by finding a mapping function f that transforms the input to output [6], [17]. Motivated by Figure 3, rather than directly train on the image, we first decompose the image into the sum of a "base" layer and a "detail" layer by using a low-pass filter,

$$\mathbf{J} = \mathbf{J}_{\text{base}} + \mathbf{J}_{\text{detail}}.$$

Using on image processing techniques, we found that after applying an appropriate low-pass filters such as [24]–[26], low-pass versions of both the rainy image I_{base} and the clean image J_{base} are smooth and are approximately equal, as shown in Figure 4. In other words, both the rain streaks and the object's details remain in the high-pass detail layer and $I_{base} \approx J_{base}$. This implies that the base layer portion can be removed from the training process, significantly simplifying the mapping needed to be learned by the CNN. Thus, we rewrite the objective function in (1) as

$$L = \frac{1}{N}\sum_{n=1}^{N} \|f_{\mathbf{W}}(\mathbf{I}^n_{\text{detail}}) - \mathbf{J}^n_{\text{detail}}\|_F^2$$

This directly lead us to train the CNN network on the detail layer instead of the image domain. Moreover, training on the detail layer has several advantages. First, after subtracting the base layer, the detail layer is sparser than the image since most regions in the detail layer are close to zero. As shown in Figure 5, the detail layer has many more pixels that are close to zero than the image itself. Taking advantage of the sparsity of the detail layer is a widely used technique in existing deraining methods [9]–[11]. In the context of a neural network, training a CNN on the detail layer also follows the procedure of mapping an input patch to an output patch, but since the mapping range has been significantly decreased, the regression problem is significantly easier to handle for a deep learning model. Thus, training on the detail layer instead of the image domain can improve learning the network weights and thus the de-raining result without a large increase in training data or computational resources.

A second advantage of training on sparse data is that it can improve the convergence of the CNN. As we show in our experiments (Figure 17), training on the detail layer converges much faster than training on the image domain. A third advantage is that decomposing an image into base and detail layers is widely used by the wider image enhancement community [27], [28]. These

enhancement procedures are tailored to this decomposition and can be easily embedded into our architecture to further improve image quality.

We therefore first decompose the image into a base layer by using a low-pass filter and a detail layer; the detail layer is equal to the difference between the image and the base layer. We use the guided filtering method of [24] as the low-pass filter because it is simple and fast to implement. In this paper, the guidance image is the input image itself. However, the choice of low-pass filter is not limited to guided filtering; other filtering approaches were also effective in our experiments, such as bilateral filtering [25] and rolling guidance filtering [26]. Results with these filters were nearly identical, so we choose [24] for its low computational complexity. After this decomposition we train the CNN on the detail layer image instead of raw image itself according to above Eq. This step represents the CNN portion of Figure 2.

B. Our convolutional neural network

We build our network structure same to [6]. Our network structure can be expressed as three operations:

$$f^l(\mathbf{I}_{\text{detail}}) = \sigma(\mathbf{W}^l * f^{l-1}(\mathbf{I}_{\text{detail}}) + \mathbf{b}^l), \quad l = 1, 2$$
$$f_{\mathbf{W}}(\mathbf{I}_{\text{detail}}) = \mathbf{W}^l * f^{l-1}(\mathbf{I}_{\text{detail}}) + \mathbf{b}^l, \qquad l = 3,$$

where l indexes layer number, * indicates the convolution operation and b^l is the bias. We define $\sigma(.)$ to be the nonlinear hyperbolic tangent function and $f^0(I_{detail}) = I_{detail}$. We use two hidden layers in our DerainNet architecture and Eq. (5) is the output of the cleaned detail layer.

To better understand the effects of the network fW, we show the learned weights and intermediate results from the hidden layers in Figure 6. The first hidden layer performs feature extraction on the input detail layer, which is similar to the common strategy used for image restoration of extracting and representing image patches by a set of dictionary elements. Thus, W1 contains some filters that look like edge detectors that align with the direction of rain streaks and object edges. The second hidden layer performs the rain streaks removal and $f^2(I_{detail})$ looks smoother than $f^1(I_{detail})$. The third layer performs reconstruction and enhances the smoothed details with respect to image content. $f_W(I_{detail})$ contains clear details with most of the rain removed. The intermediate results show that the CNN is effective at feature extraction and helps to recognize and remove rain streaks.

C. Training

We use stochastic gradient descent (SGD) to minimize the objective function in Eq. (3). Since it is extremely difficult to obtain numerous clean/rainy image pairs from real-world data, we synthesize rain using Photoshop1 to create our training dataset. We randomly collected a total of 350 clean outdoor images from the UCID dataset [29], the BSD dataset [30] and Google image search which we used to synthesize rainy images. Each clean image was used to generate 14 rainy images of different streak orientations and intensity. An example is shown in Figure 7. Thus, we create a dataset containing 350 x 14 = 4900 rainy images, each having a corresponding ground truth clean image. We randomly selected one million 64 x 64 clean/rainy patch pairs from this synthesized data as training samples. A 56x56 output is generated to avoid border effects caused by convolution. In each iteration, t, the CNN weight and bias are updated using back-propagation.

D. Combining CNN with image enhancement

After training the network, the de-rained image can be obtained by directly adding the output detail layer to the base layer,

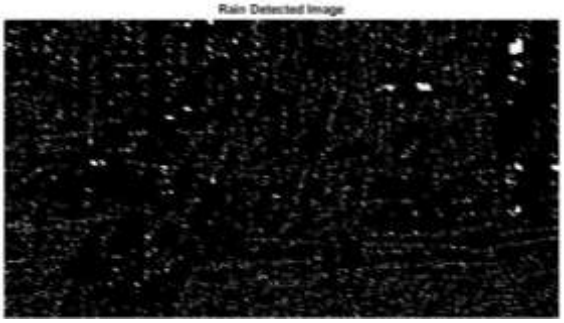
$$\mathbf{O} = \mathbf{I}_{\text{base}} + f_{\mathbf{W}}(\mathbf{I}_{\text{detail}}),$$

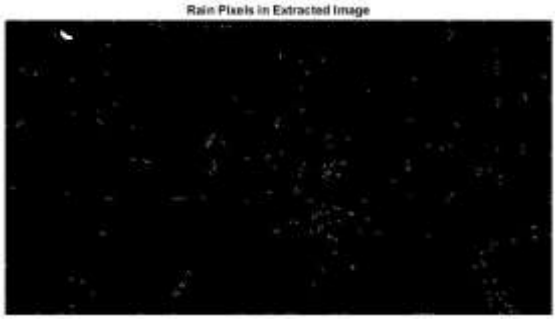
where O is the de-rained output. However, when dealing with heavy rain the result unsurprisingly looks hazy, as shown in Figure 8(b). Fortunately, we can easily embed image enhancement technology into our framework to create a better visual result. Different mature and advanced image enhancement algorithms can be directly adopted in this framework as postprocessing. In this paper, we use the non-linear function [31] to enhance the base layer, and boost the detail layer by simply multiplying the output of the CNN by two to magnify the details,

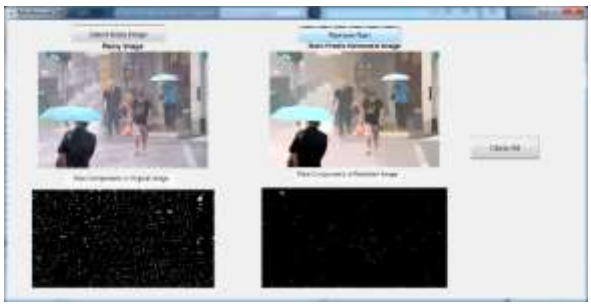
$$\mathbf{O}_{\text{enhanced}} = (\mathbf{I}_{\text{base}})_{\text{enhanced}} + 2f_{\mathbf{W}}(\mathbf{I}_{\text{detail}}),$$

where $O_{enhanced}$ is the de-rained output with enhancement and $(I_{base})_{enhanced}$ is the enhanced base layer. Virtually all of rain removal is being performed on the detail layer by the CNN, while the image enhancement on the base layer improves the global contrast and leads to a better visual result than without using enhancement.

V. RESULTS AND DISCUSSION







```
>> Rain_Removal_GUI
.....................
Processing CNN....
.....................
.....................
CNN Processed....
C A L C U L A T I O N S :
Rain Pixels Removed
   87.4937

MSE Calculated
  103.4762

PSNR Calculated
   56.0528
```

VI. CONCLUSION

We have presented a deep learning architecture called DerainNet for removing rain from individual images. Using a convolutional neural network on the high frequency detail content, our approach learns the mapping function between clean and rainy image detail layers. Since we do not possess the ground truth clean images corresponding to real-world rainy images, we synthesize clean/rainy image pairs for network learning, and showed how this network still transfers well to real-world images. We showed that deep learning with convolutional neural networks, a technology widely used for high-level vision task, can also be exploited to successfully deal with natural images under bad weather conditions. We also showed that DerainNet noticeably outperforms other state-of-the-art methods with respect to image quality and computational efficiency. In addition, by using image processing domain knowledge, we were able to show that we do not need a very deep (or wide) network to perform this task.

REFERENCES

[1] K. Garg and S. K. Nayar, "Detection and removal of rain from videos," in International Conference on Computer Vision and Pattern Recognition (CVPR), 2004.

[2] P. C. Barnum, S. Narasimhan, and T. Kanade, "Analysis of rain and snow in frequency space," International Journal on Computer Vision, vol. 86, no. 2-3, pp. 256–274, 2010.

[3] J. Bossu, N. Hautiere, and J.P. Tarel, "Rain or snow detection in image sequences through use of a histogram of orientation of streaks," International Journal on Computer Vision, vol. 93, no. 3, pp. 348–367, 2011.

[4] V. Santhaseelan and V. K. Asari, "Utilizing local phase information to remove rain from video," International Journal on Computer Vision, vol. 112, no. 1, pp. 71–89, 2015.

[5] J. H. Kim, C. Lee, J. Y. Sim, and C. S. Kim, "Single-image deraining using an adaptive nonlocal means filter," in IEEE International Conference on Image Processing (ICIP), 2013.

[6] D. Eigen, D. Krishnan, and R. Fergus, "Restoring an image taken through a window covered with dirt or rain," in International Conference on Computer Vision (ICCV), 2013.

[7] Y. L. Chen and C. T. Hsu, "A generalized low-rank appearance model for spatio-temporally correlated rain streaks," in International Conference on Computer Vision (ICCV), 2013.

[8] D. A. Huang, L. W. Kang, M. C. Yang, C. W. Lin, and Y. C. F. Wang, "Context-aware single image rain removal," in International Conference on Multimedia and Expo (ICME), 2012.

[9] L. W. Kang, C. W. Lin, and Y. H. Fu, "Automatic single image-based rain streaks removal via image decomposition," IEEE Transactions on Image Processing, vol. 21, no. 4, pp. 1742–1755, 2012.

[10] D. A. Huang, L. W. Kang, Y. C. F. Wang, and C. W. Lin, "Self-learning-based image decomposition with applications to single image denoising," IEEE Transactions on Multimedia, vol. 16, no. 1, pp. 83–93, 2014.

[11] Y. Luo, Y. Xu, and H. Ji, "Removing rain from a single image via discriminative sparse coding," in International Conference on Computer Vision (ICCV), 2015.

[12] D. Y. Chen, C. C. Chen, and L. W. Kang, "Visual depth guided color image rain streaks removal using sparse coding," IEEE Transactions on Circuits and Systems for Video Technology, vol. 24, no. 8, pp. 1430– 1455, 2014.

[13] Y. Li, R. T. Tan, X. Guo, J. Lu, and M. S. Brown, "Rain streak removal using layer priors," in International Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[14] A. Krizhevsky, I. Sutskever, and G.E. Hinton, "Imagenet classification with deep convolutional neural networks," in Advances in Neural Information Processing Systems (NIPS), 2012.

[15] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.

[16] J. Xie, L. Xu, E. Chen, J. Xie, and L. Xu, "Image denoising and inpainting with deep neural networks," in Advances in Neural Information Processing Systems (NIPS), 2012.

[17] C. Dong, C. L. Chen, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 38, no. 2, pp. 295–307, 2016.

[18] J. Kim, J. K. Lee, and K. M. Lee, "Accurate image super-resolution using very deep convolutional networks," in International Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[19] L. Xu, J. Ren, C. Liu, and J. Jia, "Deep convolutional neural network for image deconvolution," in Advances in Neural Information Processing Systems (NIPS), 2014.

[20] J. S. Ren, L. Xu, Q. Yan, and W. Sun, "Shepard convolutional neural networks," in Advances in Neural Information Processing Systems (NIPS), 2015.

[21] L. Xu, J. Ren, Q. Yan, R. Liao, and J. Jia, "Deep edge-aware filters," in International Conference on Machine Learning (ICML), 2015.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in International Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[23] J. Schmidhuber, "Deep learning in neural networks: An overview," Neural Networks, vol. 61, pp. 85–117, 2015.

[24] K. He, J. Sun, and X. Tang, "Guided image filtering," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 6, pp. 1397–1409, 2013.

[25] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in International Conference on Computer Vision (ICCV), 1998.

[26] Q. Zhang, X. Shen, L. Xu, and J. Jia, "Rolling guidance filter," in European Conference on Computer Vision (ECCV), 2014.

[27] B. Gu, W. Li, M. Zhu, and M. Wang, "Local edge-preserving multiscale decomposition for high dynamic range image tone mapping," IEEE Transactions on Image Processing, vol. 22, no. 1, pp. 70–79, 2013.

[28] T. Qiu, A. Wang, N. Yu, and A. Song, "LLSURE: local linear surebased edge-preserving image filtering," IEEE Transactions on Image Processing, vol. 22, no. 1, pp. 80–90, 2013.

[29] G. Schaefer and M. Stich, "UCID: an uncompressed color image database," in Storage and Retrieval Methods and Applications for Multimedia, 2003.

[30] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 3, no. 5, pp. 898–916, 2011.

[31] Y. Li, F. Guo, R. T. Tan, and M. S. Brown, "A contrast enhancement framework with JPEG artifacts suppression," in European Conference on Computer Vision (ECCV), 2014.

[32] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," IEEE Transactions on Image Processing, vol. 13, no. 4, pp. 600–612, 2004

[33] K. Garg and S. K. Nayar, "Photorealistic rendering of rain streaks," ACM Transactions on Graphics, vol. 25, no. 3, pp. 996–1002, 2006.

[34] A. K. Moorthy and A. C. Bovik, "A two-step framework for constructing blind image quality indices," IEEE Signal Processing Letters, vol. 17, no. 5, pp. 513–516, 2010.

AUTHOR PROFILE's

Ms. SYED SANIA has completed her B.Tech (ECE) from Shadan Women's college of engineering and technology, Khairtabad, JNTU University Hyderabad. Presently, she is pursuing her Masters in Digital System Computer Electronics from Shadan Women's college of Engineering and technology, Hyderabad, TS. India.

Dr. K. PREMLATHA is from Hyderabad, Telangana, completed her P.hd in LVLSI from sunrise university in 2018. She has completed M.tech in ECE with specialization (Digital Systems and Computer Electronics) from JNTUH in 2005. She has completed B.E in E.C.E from S.R.K.R Engineering College affiliated by Andhra University in 2000. Currently she is working as a Professor in ECE Department at Shadan Women's College of Engineering and Technology, Hyderabad from 2005. Her areas of Research interest include Low Power VLSI, Digital Systems Design, Design for Testability.