IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 5, 2020, accepted April 24, 2020, date of publication May 4, 2020, date of current version May 20, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2991988

Image Denoising With Deep Convolutional Neural and Multi-Directional Long Short-Term Memory Networks Under Poisson Noise Environments

WUTTIPONG KUMWILAISAK[1], TEERAWAT PIRIYATHARAWET[1], PONGSAK LASANG[2], (Member, IEEE), AND NATTANUN THATPHITHAKKUL[3]

[1]Department of Electronics and Telecommunication Engineering, King Mongkut's University of Technology Thonburi, Bangkok 10140, Thailand
[2]Panasonic Research and Development Center Singapore (PRDCSG), Singapore 469332
[3]National Science and Technology Development Agency, Pathum Thani 12120, Thailand

Corresponding author: Wuttipong Kumwilaisak (wuttipong.kum@kmutt.ac.th)

This work was supported by the Engineering Faculty Research Fund of King Mongkut's University of Technology Thonburi, Bangkok, Thailand.

ABSTRACT Removal Poisson noise poses a very challenging technical issue because it is difficult to capture noise characteristics. This induces from the fact that Poisson noises from different sources affect each image pixel proportional to the pixel level. This paper addresses a new image denoising method for removing Poisson noise based on the Deep Convolutional Neural and Multi-directional Long-Short Term Memory Networks. The architecture of the proposed network contains some Convolutional Neural Network (CNN) layers and multi-directional Long-Short Term Memory (LSTM) layers. CNN layers are responsible to extract image features and to estimate some noise bases existed in images. The multi-directional LSTM layers are used to effectively capture and learn the statistics of residual noise components, which possess long-range correlations and appear sparse in the spatial domain. Moreover, designing deep learning models for image denoising involves several hyperparameters such as a number of layers. To select proper hyperparameters, it is beneficial to investigate what is the best image denoising performance we can achieve under different model complexities. Moreover knowing and realizing how far the employing image denoising algorithm can do to the optimal result makes us possible to design the efficient image denoising algorithm. We utilize the Blahut-Arimoto algorithm to derive numerically distortion-mutual information function of image denoising algorithm. The derived function serves as the distortion lower bound given the mutual information between the original image and the denoised image. Based on the knowledge of distortion-mutual information function, we can decide how deep the CNN layers should be deployed in our image denoising algorithm before applying the multi-directional LSTM layers. From our experiments, the proposed image denoising algorithm can outperform other algorithms in both subjective and objective qualities.

INDEX TERMS Poisson noise, deep learning, convolutional neural network, multi-directional LSTM network, distortion-mutual information function.

I. INTRODUCTION

Image denoising is one of the most classical problems in the field of computer vision and image processing whose objective is to remove noises while preserving the original image structures. Accurately modeling and capturing noise characteristics in image denoising algorithms lead to high quality restored images [1]–[3]. In general, there are two main classes of noise:1.) Signal-independent noise; and 2.) Signal-dependent noise. The Additive White Gaussian Noise (AWGN) is a widely used signal-independent noise model. The AWGN is generally is used to model noises induced by thermal vibrations of atoms, shot noise, and black body radiation from warm objects. Unfortunately, the AWGN can not effectively represent noise characteristics under domination of photon noise [4], [5], which is signal-dependent. The photon noise is caused by the random arrival of the photon onto an image sensor. Poisson distribution is deployed to model this photon noise [6].

The associate editor coordinating the review of this manuscript and approving it for publication was Shuihua Wang.

Removal of the AWGN can be done efficiently by several existed techniques such as the sparse 3D transform-domain collaborative filtering (BM3D) [7]. However, when we apply the BM3D technique to denoise Poisson noise especially in natural images, it can not provide good results as in the AWGN environments. To denoise Poisson noise, various methods have been proposed. Azzari and Foi [8] proposed the modified BM3D technique called Variance Stabilization for Noisy+Estimate Combination in Iterative Poisson Denoising (I+VST+BM3D). The $I + VST + BM3D$ method is relied on an iterative algorithm that progressively improves the effectiveness of the Variance Stabilizing Transformations (VST). The $I + VST + BM3D$ gives better denoising results than those of $BM3D$. However, it can not perform well on Poisson noise with low peak values. Sparsity-Based Poisson Denoising With Dictionary Learning (SPDA) [9] was proposed to denoise Poisson noise with a low peak value. The *SPDA* [9] can perform well in a very low peak value but cannot outperform the $I + VST + BM3D$ in Poisson noise environments with higher peak values. Feng *et al.* [10] proposed a method called Fast and Accurate Poisson Denoising With Trainable Nonlinear Diffusion (TRDPD). The *TRDPD* is an improved version of the Trainable Nonlinear Reaction Diffusion (TNRD) [11], which can perform well on denoising Gaussian noise. Unlike the *TNRD*, the *TRDPD* replaces the reaction term of the diffusion equation of the TNRD by a new function derived from the Poisson noise distribution. The *TRDPD* provides better denoising results in Poisson noise environments for all ranges of peak noise values but it leaves some artifacts on the denoised image.

With the recent advances of deep neural networks [12]–[16], the classical image denoising techniques have been outperformed by the deep learning-based techniques [17]–[21]. Zhang *et al.* [17] proposed the Residual Learning of Deep Convolutional Neural Network for Image Denoising (DCNN) technique that utilizes the deep Convolutional Neural Networks (CNNs) to eliminate the AWGN. The *DCNN* can significantly surpass previous denoising techiques in both qualitative and quantitative results. Remez *et al.* [20] proposed the Deep Convolutional Denoising of Low-Light Images (DenoiseNet) method that deploys the deep CNNs to eliminate Poisson noise. The *DenoiseNet* can perform better than the existing denoising algorithms in both objective and subjective qualities under weak Poisson noise environments. Unfortunately, under strong Poisson noise environments, some noticeable artifacts still remain sparsely all over the restored images.

This paper addresses a new image denoising method for removing Poisson noise based on the Deep Convolutional Neural and Multi-directional Long-Short Term Memory Networks. The architecture of the proposed network contains some Convolutional Neural Network (CNN) layers and multi-directional Long-Short Term Memory (LSTM) layers. CNN layers are responsible to extract image features and to estimate some noise bases existed in images. The multi-directional LSTM layers are used to effectively capture and learn the statistics of residual noise components, which possess long-range correlations and appear sparse in the spatial domain. Moreover, designing deep learning models for image denoising involves several hyperparameters such as a number of layers. To select proper hyperparameters, it is beneficial to investigate what is the best image denoising performance we can achieve under different model complexities. Moreover knowing and realizing how far the employing image denoising algorithm can do to the optimal result makes us possible to design the efficient image denoising algorithm. We utilize the Blahut-Arimoto algorithm to derive numerically distortion-mutual information function of image denoising algorithm. The derived function serves as the distortion lower bound given the mutual information between the original image and the denoised image. Based on the knowledge of distortion-mutual information function, we can decide how deep the CNN layers should be deployed in our image denoising algorithm before applying the multi-directional LSTM layers. The contributions of this paper can be summarized as

1) We propose the method to compute numerically distortion-mutual information of the image denoising problem. This function can serve as a guideline on determining the hyperparatemeters of deep learning networks for image denoising;
2) We propose the multi-directional LSTM networks to extract and learn sparse noise characteristics to reduce complexities from applying the LSTM network directly to two-dimensional signals;
3) We combine the DCNN and the multi-directional LSTM to denoise images corrupted Poisson noise and obtain better results in both subjective and objective image qualities compared to the existed methods.

This paper can be organized as follows. Section II formulates the framework of distortion-mutual information of image denoising algorithm. The algorithm to compute the distortion-mutual information function is also presented in this section. Section III discusses the utilization of DCNN on denoising Poisson noise. Its limitations are also discussed. Section IV describes the multi-directional LSTM networks in capturing and learning sparse noise characteristics. The combination between the DCNN and the multi-directional LSTM to form our proposed image denoising architecture is in Section V. Experimental results are in Section VI. Finally, concluding remarks are in Section VII.

II. DISTORTION-MUTUAL INFORMATION FUNCTION OF IMAGE DENOISING ALGORITHM

In this section, we try to derive numerically the lower bound of distortion from the considering image denoising algorithm. In other words, we want to know what is the best denoised image quality given the DCNN structure. Let us define $\mathbf{P}$ and $\mathbf{P}_N$ as the original image and the noisy image corrupted by the Poisson noise, respectively. Each pixel in $\mathbf{P}_N$ is an identically independent random variable with Poisson distribution. The value of noisy pixel is location-dependent. The conditional probability of the pixel value at position (x_p, y_p) can be

expressed as

$$P\{\mathbf{P}_N(u, x_p, y_p) = x|\mathbf{P}(x_p, y_p) = y\} = \begin{cases} \dfrac{y^x \cdot e^{-y}}{x!}, & y > 0, \\ \delta_{x,0}, & y = 0, \end{cases} \tag{1}$$

where $\mathbf{P}_N(u, x_p, y_p)$ is the random pixel value at position (x_p, y_p) and $\mathbf{P}(x_p, y_p)$ is the pixel value at position (x_p, y_p) of $\mathbf{P}_N$ and $\mathbf{P}$, respectively. $\delta_{x,0}$ is a Kronecker delta function defined as

$$\delta_{x,0} = \begin{cases} 0, & x \neq 0, \\ 1, & x = 0. \end{cases} \tag{2}$$

Let the denoised image of $\mathbf{P}_N$ be $\hat{\mathbf{P}}$, which can be obtained from

$$\hat{\mathbf{P}} = f(\mathbf{P}_N, \mathbf{w}), \tag{3}$$

where $f(\cdot)$ is an image denoising function and $\mathbf{w}$ is a set of denoising parameters.

The objective of the image denoising problem is to find the optimal image denoising function with parameter $\mathbf{w}$ that minimizes distortion between the original image and the denoised image under the constraints on the effectiveness of function $f(\cdot)$. This can be translated to the complexity of function $f(\cdot)$ and a number of parameters in $\mathbf{w}$. Therefore, the image denoising problem can be formulated as

$$\min_{\mathbf{w}} D(\mathbf{P}, \hat{\mathbf{P}}), \tag{4}$$

subject to

$$I(\hat{\mathbf{P}}; \mathbf{P}_N, \mathbf{P}) \leq I_f(\hat{\mathbf{P}}; \mathbf{P}_N, \mathbf{P}), \tag{5}$$

where $D(\cdot)$ is the distortion function, $I(\hat{\mathbf{P}}; \mathbf{P}_N, \mathbf{P})$ is the mutual information of $\hat{\mathbf{P}}$ and $(\mathbf{P}_N, \mathbf{P})$, and $I_f(\hat{\mathbf{P}}; \mathbf{P}_N, \mathbf{P})$ is the best achievable mutual information of $\hat{\mathbf{P}}$ and $(\mathbf{P}_N, \mathbf{P})$ obtained from the denoising function $f(\cdot)$.

We may not be able to obtain the closed form solution of $D(\mathbf{P}, \hat{\mathbf{P}})$. In practical, the Blahut-Arimoto algorithm [24] can be utilized to compute numerically distortion-mutual information function of image denoising algorithm. First we need to compute joint probabilities among pixel values of $\hat{\mathbf{P}}$ and $(\mathbf{P}_N, \mathbf{P})$. Let $n(\mathbf{P}(x_p, y_p) = x, \mathbf{P}_N(x_p, y_p) = y, \hat{\mathbf{P}}(x_p, y_p) = \hat{y})$ be the total number of pixel, where its value in the original image is equal to x, the corresponding corrupted pixel value from the Poisson noise is equal to y, and the corresponding denoised pixel value is equal to $\hat{y}$ for all positions (x_p, y_p). Define N_p as the total number of pixels. In general, to obtain sufficient number of $n(\mathbf{P}(x_p, y_p) = x, \mathbf{P}_N(x_p, y_p) = y, \hat{\mathbf{P}}(x_p, y_p) = \hat{y})$ and N_p, we need to consider several images. The joint probability among pixels of $\hat{\mathbf{P}}$ and $(\mathbf{P}_N, \mathbf{P})$ is

$$P\{\mathbf{P}(x_p, y_p) = x, \mathbf{P}_N(x_p, y_p) = y, \hat{\mathbf{P}}(x_p, y_p) = \hat{y}\} = \frac{n(\mathbf{P} = x, \mathbf{P}_N = y, \hat{\mathbf{P}} = \hat{y})}{N_p}. \tag{6}$$

With the same consideration, the probabilities $P\{\hat{\mathbf{P}}(x_p, y_p) = \hat{y}\}$ and $P\{\mathbf{P}(x_p, y_p) = x, \mathbf{P}_N(x_p, y_p) = y\}$ can be computed from

$$P\{\hat{\mathbf{P}}(x_p, y_p) = \hat{y}\} = \frac{n(\hat{\mathbf{P}}(x_p, y_p) = \hat{y})}{N_p} \tag{7}$$

and

$$P\{\mathbf{P}(x_p, y_p) = x, \mathbf{P}_N(x_p, y_p) = y\} = \frac{n(\mathbf{P}(x_p, y_p) = x, \mathbf{P}_N(x_p, y_p) = y)}{N_p}, \tag{8}$$

where $n(\hat{\mathbf{P}}(x_p, y_p) = \hat{y})$ is the total number of denoised pixels having pixel values equal to $\hat{y}$ and $n(\mathbf{P}(x_p, y_p) = x, \mathbf{P}_N(x_p, y_p) = y)$ is the total number of pixels where their original pixel values are equal to x and its corresponding corrupted pixel values are equal to y. With these computed parameters, the distortion-mutual information function can be numerically calculated as follows.

Blahut-Arimoto Algorithm for Computing Distortion-Mutual Information of Image Denoising

- Step 1: Let y be the pixel value of the original image $\mathbf{P}$ at position (x_p, y_p) with probability $P\{\mathbf{P}(x_p, y_p) = y\} = p(y)$. Moreover let x be the pixel value of the noisy image at position (x_p, y_p). The conditional probability of pixel value x given the original pixel value y is equal to $P\{\hat{\mathbf{P}}(x_p, y_p) = x|\mathbf{P}(x_p, y_p) = y\} = p(x|y)$. Let $\hat{y}$ be the restored image pixel of denoised image $\hat{\mathbf{P}}$ at position (x_p, y_p). The conditional probability of pixel value $\hat{y}$ given a pair (x, y) is defined as $P\{\hat{\mathbf{P}}(x_p, y_p) = \hat{y}|(\mathbf{P}(x_p, y_p) = y, \mathbf{P}_N(x_p, y_p) = x)\} = p(\hat{y}|y, x)$.
- Step 2: Compute the expected distortion function $d(\hat{y}, y)$ over the joint probability of $\hat{y}$ and y of iteration $t+1$ from iteration t repeatedly until convergence via

$$p_{t+1}(\hat{y}) = \sum_x \sum_y p(x, y) p_t(\hat{y}|x, y) \tag{9}$$

$$p_{t+1}(\hat{y}|x, y) = \frac{p_t(\hat{y}) e^{-\beta d(\hat{y}, y)}}{\sum_{\hat{y}} p_t(\hat{y}) e^{-\beta d(\hat{y}, y)}}. \tag{10}$$

The Blahut-Arimoto algorithm numerically outputs $I(\hat{\mathbf{P}}; \mathbf{P}_N, \mathbf{P})$ corresponding to the distortion between $\hat{\mathbf{P}}$ and $\mathbf{P}$. The mutual information from the Blahut-Arimoto algorithm corresponds to the lowest distortion we can achieve. Note that the underlying mutual information is directly computed from the statistics obtained from the Blahut-Arimoto algorithm.

From the original image and the denoised image obtained from the DCNN, we can compute mutual information directly as

$$I(\hat{\mathbf{P}}; \mathbf{P}_N, \mathbf{P}) = \sum_{y \in \mathbf{P}_N} \sum_{x \in \mathbf{P}} \sum_{\hat{y} \in \hat{\mathbf{P}}} p(x, y, \hat{y}) \log\left(\frac{p(x, y, \hat{y})}{p(\hat{y}) p(x, y)}\right). \tag{11}$$

The difference between distortion of the original image and denoised image given the same mutual information can be used to measure the efficiency of image denoising algorithm.

Given the same mutual information, if the image denoising algorithm gives very close distortion to that from the Blahut-Arimoto algorithm, that image denoising algorithm has very high efficiency in noise removal.

III. NOISE FEATURE ESTIMATION WITH CONVOLUTIONAL NEURAL NETWORKS AND ITS LIMITATIONS

It is widely known that Deep Convolutional Neural Networks (DCNN) can learn to extract non-linear features far better than the human hand-crafted features. In image denoising problem, the DCNN is utilized to learn and extract noises from corrupted images. Then, it subtracts noises from the corrupted images to obtain the denoised images. The structure of DCNN for image denoising can be shown in Fig.8. Notice that each layer of the DCNN extracts noise features from the input by convolving the trained weights with the features extracted from the previous layer [25]. The output feature at layer i can be written as

$$F_{k,i}(x_p, y_p) = \sum_{m=1}^{M_{i-1}} ||W_{m,i-1} \circ Z_{m,i-1}(x_p, y_p)||_F, \quad (12)$$

where $F_{k,i}(x_p, y_p)$ be the feature value k at position (x_p, y_p) of the i^{th} layer of the DCNN. $||\cdot||_F$ is the Frobenius norm and $\circ$ is the Hadamard operation. M_{i-1} is a number of feature maps at the i^{th} layer. $W_{m,i-1}$ is the trainable weight matrix of feature map m at the $(i-1)^{th}$ layer. $Z_{m,i-1}(x_p, y_p)$ is an $N \times N$-patch of feature map m in the $(i-1)^{th}$ layer with the center at position (x_p, y_p). The output features at layer i are then weighted summed to obtain the noise component.

Fig.8 shows the extracted Poisson noise components from *Plane* image. Notice that the noise components obtained from Layer 1 and Layer 2 of the DCNN are very noisy. In contrast, the noise components from the deeper layers become more sparse (Layer 18 and Layer 19). This implies that the deeper the layer, the weaker the noise power. Note that even though in this paper, we utilize Poisson noise for our description, the concept of image denoising with DCNN can be applied to different kinds of noise such as Gaussian noise.

A number of layers and a number of neurons per layer in the DCNN involve directly with the neural network complexity. The larger the number of parameters, the longer the training period. Knowing and recognizing the limits of the using image denoising algorithm help us to optimize and select the proper DCNN structure(e.g., a number of layer and a number of parameters) for image denoising. Even though, the proper DCNN can remove noise quite efficiently, from the layers that noise components become sparse, increasing the number of CNN layers does not significantly improve performance of noise estimation. In other words, the image denoising performance does not improve much given higher complexities we give to the DCNN. This is because CNN cannot group noisy pixels to be a local patch for the convolution operation to learn the noise features. Therefore, the noise statistic cannot be effectively calculated. To capture the statistics of sparse noise, we need other measures to capture noise characteristics.

IV. SPARSE NOISE FEATURE EXTRACTION WITH MULTI-DIRECTIONAL LONG SHORT-TERM MEMORY NETWORKS (LSTM)

The LSTM network is widely known to be utilized to capture long-range dependencies of one dimensional data [23], [26], [27]. It is also possible to employ the LSTM network to capture some correlations in multi-dimensional signals with higher computational complexity [28]. This can be done by transforming multi-dimensional data to be long one-dimensional data. In this section, we deploy the LSTM network to extract sparse noise features. To obtain the optimal result, we need to transform the two-dimensional input feature map to one dimensional signal. This can be achieved by scanning feature maps using some scanning formats such as a raster scanning. However, training LSTM network to deal with very long one-dimensional input data may face several technical issues such as vanishing gradient and high computational complexity [22].

To solve these challenges, we propose the multi-directional LSTM network. The multi-directional LSTM network applies the LSTM network to inputs in four directions: 1.) from the left to the right (direction 1); 2.) from the right to the left (direction 2); 3.) from the top to the bottom (direction 3); and 4.) from the bottom to the top (direction 4). The proposed algorithm may not provide the optimal result on capturing noise features since we assume that we can obtain the sufficient sparse noise characteristics from applying the LSTM network only in four directions. Fig.2 illustrates the proposed multi-directional LSTM network as a combination of the operations of feature maps in four directions. The input feature maps will be first convoluted by $1 \times 1 \times C$ convolutional neural network before passing them to each direction of the directional LSTM module. The number of filters is equal to 32. To simplify the process, the left to the right, the bottom to the top, and the right to the left will be represented by 90°, 180°, and 270° rotations from the top to the bottom directional LSTM network, respectively. After rotations, we can apply the LSTM only from the top to the bottom. This will reduce our implementation difficulties greatly. Then, the output from all four directions will be concatenated and are fed to the $1 \times 1 \times 64$ convolutional neural network to get the output feature maps of the multi-directional LSTM network.

The procedure of the directional LSTM module from the top to the bottom can be described as follows. The feature maps with the size of $I \times J \times K$ are processed directly without transforming into one dimensional data. This possibly omits some correlated data that can be gathered from a long one-dimensional data of the LSTM, but it can largely mitigate the complexity of our proposed framework. In general, the complexity of the conventional LSTM is in the order $O(n^2)$. However, the proposed multi-directional LSTM has the complexity in the order of $O(n)$. The LSTM cells are connected sequentially as a straight line from the previous cell to the next LSTM cell in the processing direction. In the top to the bottom direction, the LSTM cells will be connected from

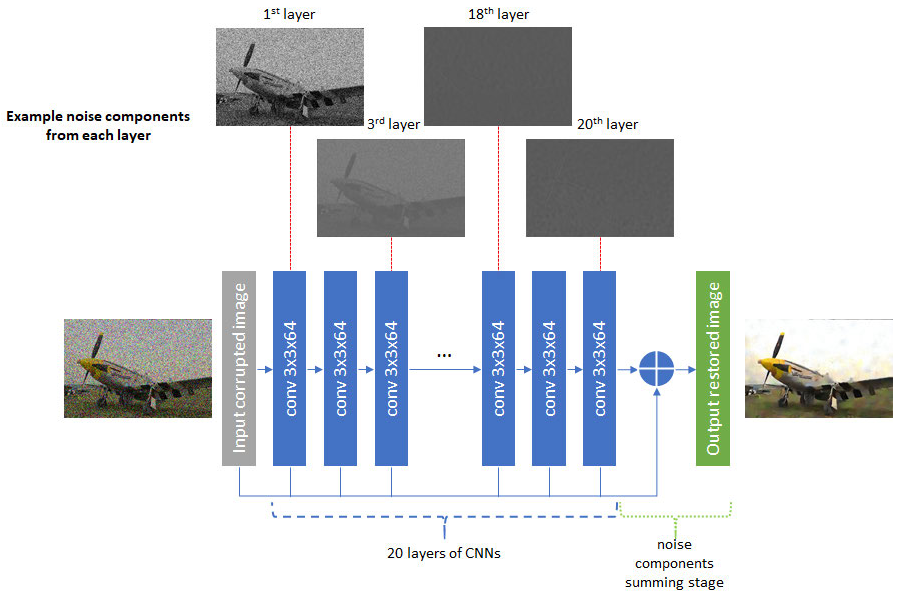
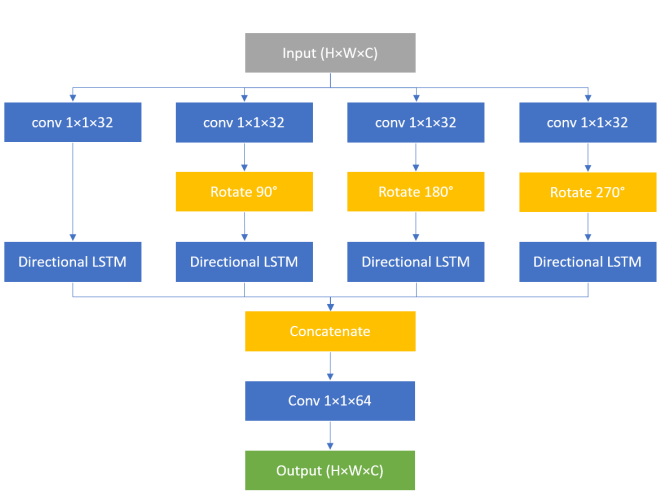
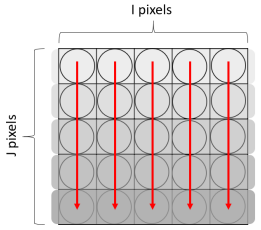


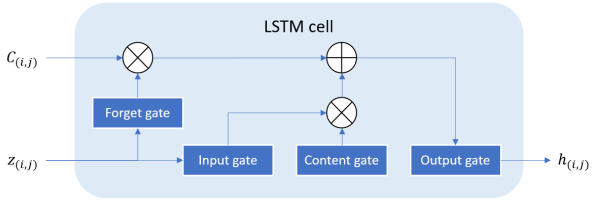
FIGURE 1. The visualized feature maps obtained from intermediate layers of DCNN [20] of *Plane* image.



(a) The architecture of the proposed multi-directional LSTM network.



(b) The connection of the LSTM cell in the top to the bottom direction



(c) The procedure of the LSTM cell in the multi-directional LSTM network

FIGURE 2. The architecture of the proposed multi-directional LSTM network and its LSTM cell.

the previous row to the next row. At the same time, the LSTM cells in each row are independent among others.

Fig.2b depicts the connection of the LSTM cells in the top to the bottom direction. To calculate the output of the LSTM cell at position (i, j), we feed the feature map value at position (i, j) and the outputs of the LSTM networks from the previous row. The linear transformation is applied to the feature map value at position (i, j) before passing through the activation function. The weights of linear transformation are learned during the training period. At the same time, the outputs of the LSTM cell from the previous row are convoluted with the training weights before passing them to the activation function. The summation of these two values are used as the input of the gates in the current LSTM cell. Fig.3 shows the example of the process from the input feature maps to the output feature maps in each row in the case of one feature map. The input of the gates at position (i, j) at channel k can be calculated as

$$z_{(i,j,k)} = ReLU(W^{f}_{(k)} \circ f_{(i,j)}) + ReLU(W^{h}_{(k)} \circ h_{(i-1,j)}), \quad (13)$$

where $ReLU(\ldots)$ is a rectifier linear unit function. $f_{(i,j)}$ is the one-dimensional input features with a size of K at position (i, j). $h_{(i-1,j)}$ is the output feature patch with the size of $3 \times K$ from the previous row output features centered at $(i - 1, j)$.

$W^f_{(k)}$ is the weight matrix of the linear transformation applied to input $f_{(i,j)}$ at channel k. $W^h_{(k)}$ is the weight to convolute with $h_{(i-1,j)}$ at channel k. Noted that the weights $W^f_{(k)}$ and $W^h_{(k)}$ of different gates in the LSTM cell are also independent from one another.

The proposed directional LSTM module contains cell state, forget gate, input gate, content gate, and output gate. The procedure of the multi-directional LSTM network is illustrated in Fig.2c. The cell state is the key of the LSTM cell. It connects the previous cell and has some minor process in the current cell to become an output cell state. The first operation in the cell state is at the forget gate. The forget gate controls how much of each component should be able to pass through by multiplying the value of the forget gate with the incoming cell state. A value of zero means nothing will be able to pass through and a value of one means everything is able to pass. The forget gate at position (i, j, k) can be calculated as

$$G^f_{(i,j,k)} = \sigma(z_{(i,j,k)}), \tag{14}$$

where $G^f_{(i,j,k)}$ is the forget gate at position (i, j) at channel k and $\sigma(\ldots)$ is a sigmoid function. Next, we need to process input features and decide how much of them will be stored in the cell state. There two parts of this. First, the input gate is used to decide how much of the input will be added into the cell state. Second, the content gate processes the input features before they pass through the input gate. The input gate and content gate at position (i, j, k) can be calculated by

$$G^i_{(i,j,k)} = \sigma(z_{(i,j,k)}), \tag{15}$$

where $G^i_{(i,j,k)}$ is the input gate at position (i, j) at channel k, and

$$G^c_{(i,j,k)} = \tanh(z_{(i,j,k)}), \tag{16}$$

where $G^c_{(i,j,k)}$ is the content gate at position (i, j) at channel k and $\tanh(\ldots)$ is a hyperbolic tangent function, respectively.

The old cell state obtained from the previous LSTM cell in the previous row will be used to compute the new cell state by combining all above computed values. The old cell state will be multiplied by the forget gate to filter some information it decides to omit earlier and then sum with the multiplication of content gate and input gate. This is the new candidate values scaled by how much we decide to update each state value. The current cell state at the position (i, j, k) after updating with the old cell state can be reckoned as

$$C_{(i,j,k)} = (G^f_{(i,j,k)} \times C_{(i-1,j,k)}) + (G^i_{(i,j,k)} \times G^c_{(i,j,k)}), \tag{17}$$

where $C_{(i,j,k)}$ is the current cell state at position (i, j) of channel k and $C_{(i-1,j,k)}$ is the old cell state at position (i, j) of channel k.

Finally, the output of the LSTM cell is based on the cell state and the value of the output gate. The output gate controls how much of the cell state will become an output of the LSTM cell. The cell state will be fed to the hyperbolic tangent function in order to control the output value to be between -1 and 1 and multiply it by the output gate. The output gate and the output features at the position (i, j) at channel k can be calculated by

$$G^o_{(i,j,k)} = \sigma(z_{(i,j,k)}), \tag{18}$$

$$h_{(i,j,k)} = G^o_{(i,j,k)} \times \tanh(C_{(i,j,k)}). \tag{19}$$

The output $h_{(i,j,k)}$ of the LSTM belongs to only one direction. Based on Fig.2(a), we need the outputs from four directions before concatenating them. C convolutional filters with the size of $1 \times 1 \times 64$ is applied to the concatenated output to obtain the estimated sparse noise features with the size of $H \times W \times C$.

V. IMAGE DENOISING WITH DCNN AND MULTIDIRECTIONAL LSTM NETWORKS

In this section, we combine both DCNN and Multi-directional LSTM networks to denoise Poisson noise. The DCNN works well in removing noise with reasonable complexities. However, as we can see from the previous section, after several layers of CNN, noise feature becomes sparse. Therefore, utilizing the CNN layers can not capture noise features well and the final denoised image qualities are not significantly improved. Here, the multi-directional LSTM networks can help to learn sparse noise features. Fig.4 shows the architecture of the combination between the DCNN and multi-directional LSTM networks. There are totally 18 layers in our proposed network. Each layer of the DCNN extracts noise features from the input by convolving the trained weights with the features extracted from the previous layer. We employ totally 15 layers of CNN. Then, three layers of multi-directional LSTM networks are cascaded to the DCNN. Output noise features obtained from every layers are weighted summed and are convolved with the trained weights to obtained the noise component. Thus, the restored image can be obtained from the summation of the noise component and the noisy image.

VI. EXPERIMENTAL RESULTS

A. IMAGE DENOISING PERFORMANCE

1) TRAINING METHODOLOGY

An image data set from the Microsoft COCO (2017) Dataset [29] is used to evaluate the proposed image denoising technique. We randomly select several image batches from the image data set. Poisson noises are applied to the selected image batches. The noisy images are used as the training inputs. There are two training stages. The first training stage is performed over each batch with the size of 256 image patches. Each patch is with the size of 32×32 pixels. In our experiment, we iterate over our data set for 35000 epochs. The learning rate α is set to 0.001. The training weights from the first stage will be used as the initial weights in the second training stage. In the second stage, each batch contains 32 image patches with the size of 128×128 pixels. In this stage, we iterate over the data set for 5000 epochs. The objective of the second training stage is to let our image

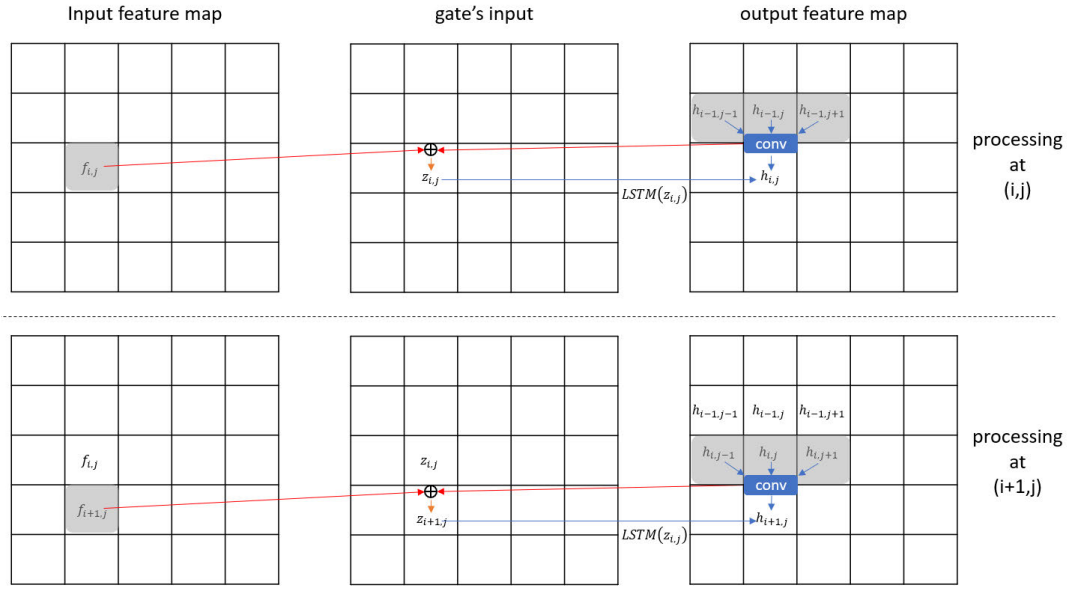


FIGURE 3. The example of the process from the input feature maps to the output feature maps in each row in the case of one channel.

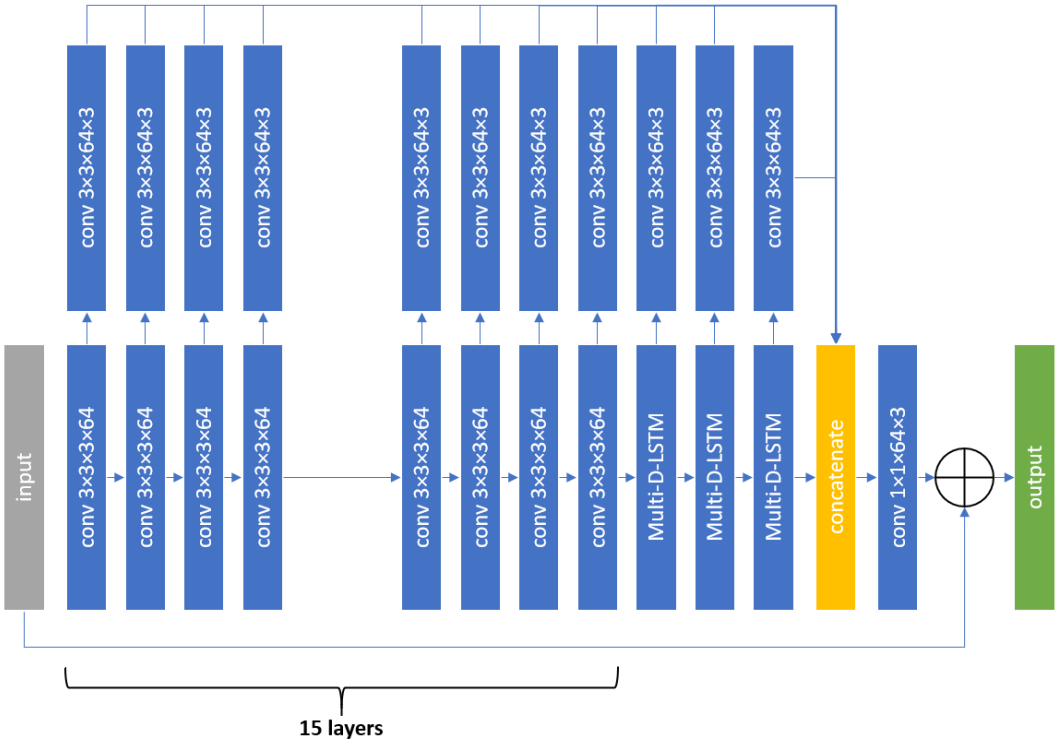


FIGURE 4. The architecture of the combination between DCNN and multi-directional LSTM networks.

denoising technique to learn the spatial information from natural images.

2) SUBJECTIVE AND OBJECTIVE QUALITIES

We evaluate our image denoising technique with those obtained from $I + VST + BM3D$ [8], DCNN [17], and DenoiseNet [20]. Test images are utilized from the Set14 image data set [30] and the LIVE1 image data set [31]. All image denoising techniques are trained with the image data set. However, a number of training epochs may be different to obtain the best denoised images. We employ both Peak Signal-to-Noise Ratio (PSNR) [32] and the Structural Similarity Index (SSIM) [33] to be our objective metrics. Table 1 and Table 2 compare the image objective qualities

(a) Original (b) Noisy (c) DCNN

(d) DenoiseNet (e) Our method

FIGURE 5. Subjective image qualities of the *Sailing* image obtained from different image denoising algorithms with a peak value of Poisson noise equaling 0.1.

TABLE 1. Objective image quality comparison obtained from different image denoising algorithms on the LIVE1 data set [31].

Peak	Noisy	I+VST+BM3D	DCNN	DenoiseNet	Our method
	PSNR/SSIM	PSNR/SSIM	PSNR/SSIM	PSNR/SSIM	PSNR/SSIM
0.1	6.26/0.02	7.02/0.06	18.90/0.60	19.15/0.60	**19.61/0.61**
1	6.58/0.06	16.47/0.59	22.68/0.73	22.59/0.72	**22.87/0.73**
10	14.44/0.27	22.03/0.70	27.77 /0.88	27.49/0.87	**27.77/0.88**
40	20.04/0.52	23.39/0.72	30.92/0.93	30.72/0.93	**31.01/0.93**
80	22.94/0.65	22.41/0.72	32.65/0.95	32.31/0.95	**32.71/0.95**

among different image denoising techniques. Our proposed image denoising technique outperforms other networks in terms of the objective quality metrics. Our proposed method provides up to 0.5 dB PSNR improvement by average. However, in SSIM, there is not much improvement gain from other works. This may imply that the SSIM may not be sensitive enough to measure the quality improvement in this comparison case. Fig.5, Fig.6, and Fig.7 show the subjective quality comparison under a peak value of Poisson noise equaling 0.1. Our method shows significantly improvement in subjective quality especially less color artifacts. Notice that our method has high impacts on image regions with low details as shown in Fig.7.

We also compare feature maps between our method and the DenoiseNet. Fig.8 shows features maps from both networks in the intermediate layers of *Plane* image in the LIVE1 data set [31]. Notice that feature maps of the first layer from both networks are very similar, which are very noisy. However, in deeper layers, feature maps of the DenoiseNet contain less structural information and some blurring artifacts. In contrast, our method provides feature maps with more edge information and less artifacts.

The reason that our algorithm can not outperform other image denoising algorithms objectively in high texture images because of the non-stationary pixel values of high textures. Since the Poisson noise applied to each pixel position is relied on its corresponding pixel value, in the multi-directional LSTM, noise characteristics in the operating patch are quite dynamic. Therefore, from past information, the multi-directional can not provide good predictions of the noise characteristics. However, as mentioned above, our method still gives superior subjective image qualities. To prove this claim, we explore our proposed image denoising on images with low details. To obtain a set of low detail images, we deploy the metric the two-dimensional High Frequency Component (HFC). The two-dimensional HFC of each image can be calculated via

$$HFC = \sum_{i=1}^{N-1} \sum_{j=1}^{N-1} i \times j \times |X(i,j)|, \tag{20}$$

where $X(i,j)$ is the $N-$point two-dimensional discrete Fourier transform of image data at frequency (i,j). Notice that the HFC is the weighted summation of all frequency components. The higher the frequency, the higher the weight. In general, low detail images tend to have low HFC values. To be more specific, we declare that the specific image has low details if its HFC is less than 10^8. Fig.9a, Fig.9b, and Fig.9c illustrate HFC histograms of the image data set in [30], the LIVE1 image data set [31], and the VOC2012 image data set [34], respectively. The HFC histograms imply that most images in the LIVE1 image data set contain high texture levels. In the the image data set in [30], there are some images having high texture levels, whereas some have less texture levels. We found that there are many low detail images in

TABLE 2. Objective image quality comparison obtained from different image denoising algorithms on the Set14 data set [30].

Images	Boats	Airplane	Baboon	Barbara	Flower	Goldhill	Lenna	Monarch	Pens	Pepper	Average
Peak = 0.1											
Noisy	5.6573	2.9786	6.7611	6.4746	6.4612	7.0502	5.4771	6.8083	7.2766	6.0534	6.0999
I+VST+BM3D	6.3674	3.533	7.5024	7.2434	7.1988	7.869	6.1842	7.5918	8.076	6.7599	6.8326
DCNN	19.6330	18.6004	17.3403	18.8485	19.5030	20.8979	19.3027	18.3962	19.9242	18.6673	19.1114
DenoiseNet	20.1019	19.3526	17.2945	18.9868	19.7949	21.1607	19.2652	18.3651	19.8891	19.2394	19.345
Our method	**20.5645**	**19.6629**	**17.5235**	**19.2097**	**20.2125**	**21.6332**	**20.4001**	**19.0322**	**20.178**	**19.6202**	**19.8037**
Peak = 1											
Noisy	6.5302	5.3705	6.9098	6.6788	6.9435	6.9498	6.2867	6.6955	7.1374	6.77	6.6272
I+VST+BM3D	16.7013	13.2919	15.8623	16.9993	17.1302	17.9205	16.3421	17.0936	17.799	16.6967	16.5837
DCNN	24.3669	22.9794	19.3033	22.0262	24.5221	24.4551	24.3333	23.7017	24.3002	23.5954	23.3583
DenoiseNet	24.2353	22.9393	19.3535	21.89	24.1778	24.4682	24.2808	23.5243	23.9927	23.4222	23.2284
Our method	**24.7395**	**23.5414**	**19.4414**	**22.2263**	**25.223**	**24.7897**	**24.7172**	**23.8159**	**24.4507**	**23.6172**	**23.6562**
Peak = 10											
Noisy	13.9563	13.2639	14.789	14.3827	14.7998	14.8409	14.0839	14.5378	15.1359	14.5685	14.4359
I+VST+BM3D	24.5675	23.8074	18.9924	22.3367	25.1047	24.5561	25.0231	22.4482	24.1913	24.6599	23.5687
DCNN	29.7620	28.3959	22.9458	27.2027	30.7386	28.6201	29.2749	29.9557	29.6415	28.3232	28.4860
DenoiseNet	29.442	28.1498	22.899	26.5699	30.3903	28.45	29.0287	29.6093	29.2105	28.1903	28.194
Our method	**29.8010**	**28.4292**	**23.0183**	**27.1465**	**30.7281**	**28.6764**	**29.3784**	**29.9399**	**29.5631**	**28.4951**	**28.5176**
Peak = 40											
Noisy	19.3457	18.1427	20.3387	19.9026	20.156	20.4955	19.525	20.1418	20.7366	19.8877	19.8672
I+VST+BM3D	24.2089	21.2771	19.1827	22.5585	25.4731	25.1188	23.8966	22.7513	24.8656	24.7024	23.4035
DCNN	32.7785	31.2303	25.3837	30.4266	33.6597	31.0060	31.2676	33.2338	32.4848	30.3480	31.1819
DenoiseNet	32.5776	31.1391	25.4009	29.9996	33.4791	30.8778	31.2492	32.9788	32.1783	30.6199	31.0501
Our method	**32.911**	**31.6445**	**25.5598**	**30.4926**	**33.8569**	**31.1139**	**31.578**	**33.3234**	**32.5603**	**30.8958**	**31.3936**
Peak = 80											
Noisy	22.2839	20.7491	23.2535	22.8094	23.0162	23.3831	22.3571	23.0337	23.611	22.7432	22.724
I+VST+BM3D	24.2188	20.1952	19.196	22.5624	25.1852	25.1517	23.6725	22.7984	24.9294	24.6304	23.254
DCNN	34.2891	32.5694	26.7802	31.9500	35.0697	32.3395	32.3285	34.8465	34.0059	31.4699	32.5649
DenoiseNet	33.8999	32.6085	26.8068	31.4988	34.5258	32.1625	32.1083	34.3492	33.4559	31.6258	32.3041
Our method	**34.369**	**33.1557**	**26.9808**	**32.0183**	**35.1603**	**32.3951**	**32.5002**	**34.9722**	**34.0352**	**31.9862**	**32.7573**

TABLE 3. The PSNR Comparison among Noisy images, I+VST+BM3D, DenoiseNet and our method on a set of low details images.

Peak	Noisy	I+VST+BM3D	DCNN	DenoiseNet	Our method
	PSNR/SSIM	PSNR/SSIM	PSNR/SSIM	PSNR/SSIM	PSNR/SSIM
0.1	6.78/0.04	7.62/0.09	21.41/0.75	21.90/0.76	**23.00/0.78**
1	6.95/0.05	18.03/0.74	26.44/0.85	26.38/0.84	**27.09/0.86**
10	14.83/0.22	26.69/0.85	32.13/0.93	31.62/0.92	**32.21/0.93**
40	20.37/0.44	25.75/0.86	34.73/0.95	34.40/0.95	**34.97/0.95**
80	23.27/0.57	25.49/0.86	36.13/0.96	35.60/0.96	**36.25/0.96**

the VOC2012 image data set. We can extract totally 528 low detail images for our evaluation.

Table 3 compares the objective qualities among different image denoising methods under Poisson noise environments. We use both PSNR and SSIM as the objective metrics. From the experimental results, our method can outperform other techniques under strong noise environments. To be specific, our method achieves 1.1 dB and 0.7 dB PSNR improvements under Poisson noises with peaks 0.1 and 1, respectively. The PSNR improvements of our method are not so significant when we have to deal with weak noise environments (lower peal noise value). This is because under weak noise environments, image textures are less affected by noise and our method can not give significant gains in such environments. However, in strong noise environments, the CNN module tends to smooth out the texture images causing texture loss. With the multi-directional LSTM modules in our method, image texture can be preserved and restored to obtain better denoised image qualities. In low detail images, the SSIM improvements of our method is also superior to other methods tally with those obtained from the PSNR improvements. The major improvements on our image denoising algorithm on low detail images are due to less dynamic on the noise characteristics in the operating patch. In low detail images, pixel values are not so dynamic. Hence, the Poisson noise characteristics are at different pixel positions are quite static. There are high correlations on the noise statistics on the operating patch. Therefore, the multi-directional LSTM can learn and predict noise characteristics better than those with highly varying textures.

B. NUMERICAL DISTORTION MUTUAL INFORMATION OF IMAGE DENOISING ALGORITHMS

To evaluate the numerical distortion-mutual information function of the image denoising algorithm, we utilize the VOC2012 image data set [34] since it contains a large number of images for training the DCNN. We randomly select several image batches from the image data set. The Poisson noise is applied to the selected image batches, where the peak value of the Poisson noise is equal to one. The noisy images are used as the training inputs. The training stage is performed over each batch with the size of 256 image patches. Each patch is with the size of 32 $\times$ 32 pixels. The learning rate α is initially set to 0.001 and Adam optimizer is utilized. Fig.10 shows the average mutual information between the

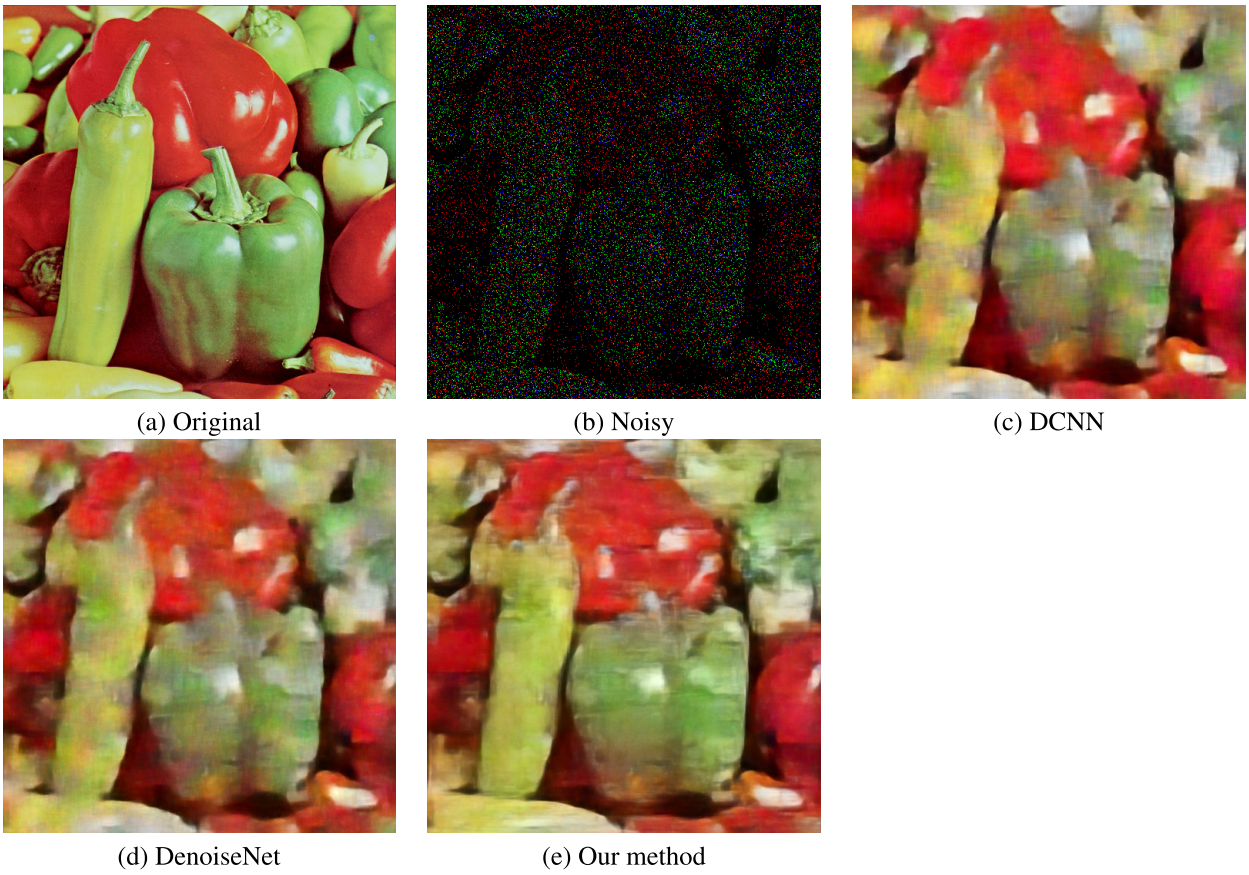


FIGURE 6. Subjective image qualities of the *Pepper* image obtained from different image denoising algorithms with a peak value of Poisson noise equaling 0.1.

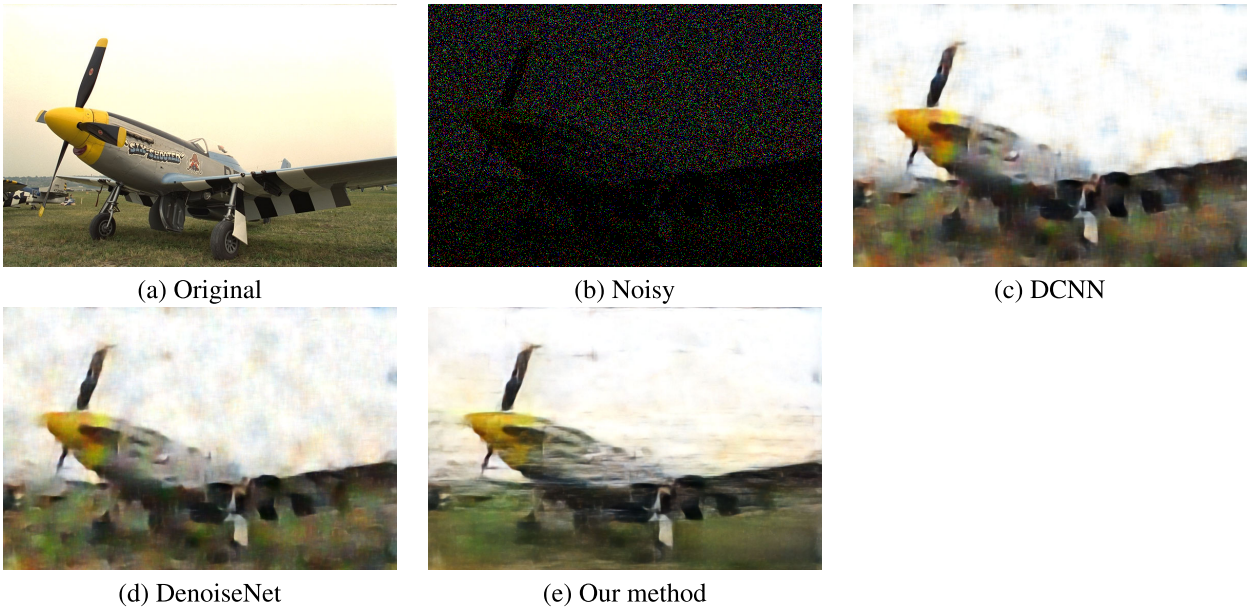


FIGURE 7. Subjective image qualities of the *Plane* image obtained from different image denoising algorithms with a peak value of Poisson noise equaling 0.1.

original and the denoised images from several trained DCNN models with different training epochs. The average mutual information of each DCNN model is compute from averaging mutual information between the original images and denoised images obtained from the considering DCNN model. As we can see, the DCNN models obtained from small training

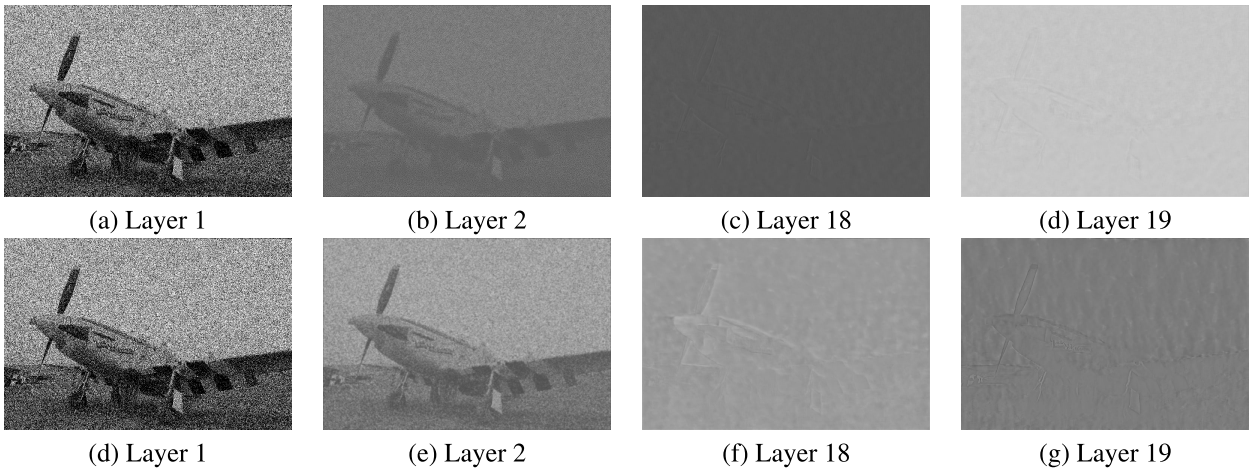


FIGURE 8. The visualized feature maps obtained from intermediate layers of *Denoisenet* (upper row) and our method (lower row) of *Plane* image in the LIVE1 image data set.

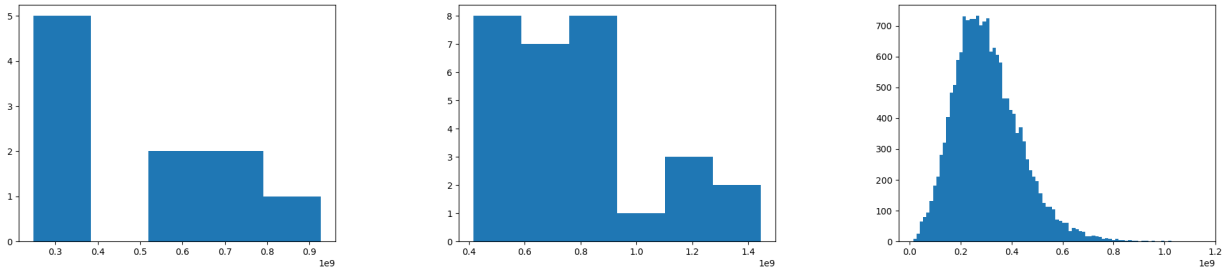


FIGURE 9. The *HFC* histograms from various image data sets.

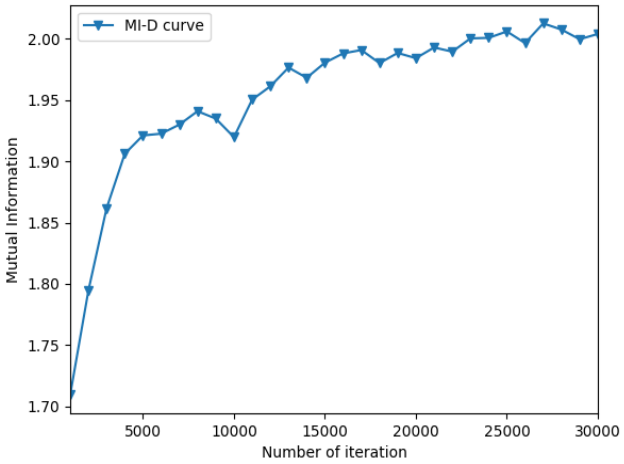


FIGURE 10. Average mutual information from the DCNN models in different training iterations, when the peak value of Poison noise equal to one.

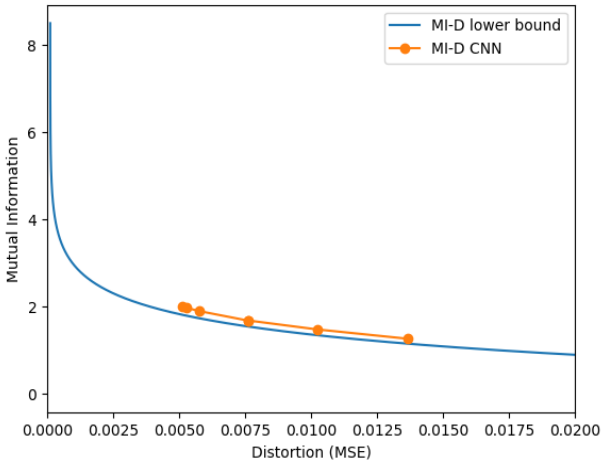


FIGURE 11. Distortion-mutual information curves obtained from the Blahut-Arimoto algorithm and the DCNN image denoising models under Poisson noise with peak equal to one.

epochs possess less average mutual information than those with higher training epochs. Notice that the average mutual information of different DCNN models is not much different after 15000 iterations.

We employ the Blahut-Arimoto algorithm to compute the distortion mutual information function of image denoising algorithm numerically. The results from the Blahut-Arimoto algorithm serve as the best distortion we can achieve given

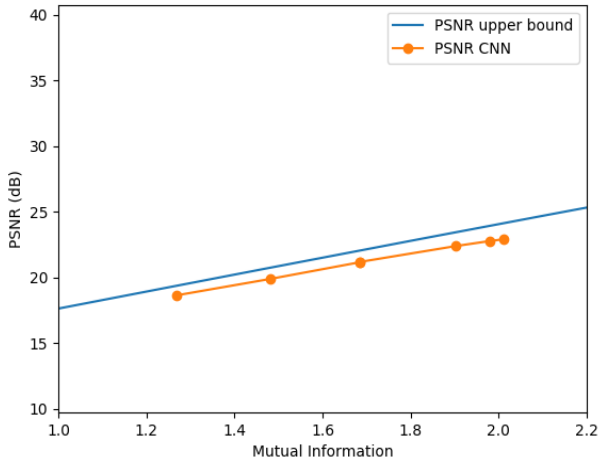


FIGURE 12. Average PSNR comparison between the DCNN and the Blahut-Arimoto algorithm.

the mutual information. We compare the image denosing performance of different DCNN models with those obtain from the Blahut-Arimoto algorithm. We vary the number of CNN layers in the DCNN with one, two, three, four, five, ten, and 15 layers. All DCNN models are trained with 35000 epochs. The results are shown in Fig.11. As we can see, the image denoising algorithm based on the DCNN gives very close performance to the best performance we can obtain given the mutual information between the original image and the denoised image. Fig. 12 shows the comparison results in the PSNR domain between the denoised image obtained from the DCNN and those from the Blahut-Arimoto algorithm.

VII. CONCLUSION

We propose a new architecture of deep convolutional and multi-directional LSTM networks to eliminate Poisson noise. Poisson noise is challenging to remove since the noise level is relied on its corresponding pixel intensity. The proposed network is designed to have two stages. The deep convolutional networks for extracting the noise bases with different variances are contained in the first stages. The deeper the layer is, the lower the noise variance is and the more sparse the noise is. Then, the multi-directional LSTM networks are in the second stage of the proposed network. The sparse noise components are grouped by the second stage of the network so that the remaining noise information still can be effectively removed. The proposed network is trained with several natural images before is applied on the test sets of images. The experimental results show that our proposed network provides better qualities of denoised images and fewer artifacts in both subjective and objective quality measures than those of the existing algorithms. We also derive the numerical distortion-mutual information function of image denoising algorithm. It provides the bound on the image denoising performance given the mutual information between the original image and the denoised image. The denoising results under the Poisson noise environment from the DCNN give near optimal qualities under different hyperparameter settings such as a number of CNN layers. This agrees with the fact that most noises are removed during the first stage. Only sparse noises still remain. However, sparse noises still affects the overall subjective qualities of denoised images. The insights given this framework can lead to the proper selection of a number of CNN layers and the design of image denoising algorithm.

REFERENCES

[1] A. K. Boyat and B. K. Joshi, "A review paper: Noise models in digital image processing," *Signal Image Process. Int. J.*, vol. 6, no. 2, pp. 63–75, Apr. 2015.
[2] A. Foi, M. Trimeche, V. Katkovnik, and K. Egiazarian, "Practical Poissonian-Gaussian noise modeling and fitting for single-image raw-data," *IEEE Trans. Image Process.*, vol. 17, no. 10, pp. 1737–1754, Oct. 2008.
[3] M. Lebrun, M. Colom, A. Buades, and J. M. Morel, "Secrets of image denoising cuisine," *Acta Numerica*, vol. 21, pp. 475–576, May 2012.
[4] C. Liu, R. Szeliski, S. Bing Kang, C. L. Zitnick, and W. T. Freeman, "Automatic estimation and removal of noise from a single image," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 2, pp. 299–314, Feb. 2008.
[5] S. Lee, M. Lee, and M. Kang, "Poisson–Gaussian noise analysis and estimation for low-dose X-ray images in the NSCT domain," *Sensors*, vol. 18, no. 4, p. 1019, Mar. 2018, doi: 10.3390/s18041019.
[6] S. W. Hasinoff, "Photon, Poisson noise," in *Computer Vision*. Boston, MA, USA: Springer, 2014, pp. 608–610.
[7] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-D transform-domain collaborative filtering," *IEEE Trans. Image Process.*, vol. 16, no. 8, pp. 2080–2095, Aug. 2007, doi: 10.1109/TIP.2007.901238.
[8] L. Azzari and A. Foi, "Variance stabilization for Noisy+Estimate combination in iterative Poisson denoising," *IEEE Signal Process. Lett.*, vol. 23, no. 8, pp. 1086–1090, Aug. 2016, doi: 10.1109/LSP.2016.2580600.
[9] R. Giryes and M. Elad, "Sparsity-based Poisson denoising with dictionary learning," *IEEE Trans. Image Process.*, vol. 23, no. 12, pp. 5057–5069, Dec. 2014, doi: 10.1109/TIP.2014.2362057.
[10] W. Feng, P. Qiao, and Y. Chen, "Fast and accurate Poisson denoising with trainable nonlinear diffusion," *IEEE Trans. Cybern.*, vol. 48, no. 6, pp. 1708–1719, Jun. 2018, doi: 10.1109/TCYB.2017.2713421.
[11] Y. Chen and T. Pock, "Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1256–1272, Jun. 2017, doi: 10.1109/TPAMI.2016.2596743.
[12] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.
[13] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," 2014, *arXiv:1408.5093*. [Online]. Available: http://arxiv.org/abs/1408.5093
[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[15] X. Mao, C. Shen, and Y.-B. Yang, "Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2802–2810.
[16] R. Jaroensri, C. Biscarrat, M. Aittala, and F. Durand, "Generating training data for denoising real RGB images via camera pipeline simulation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 1–15.
[17] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising," *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3142–3155, Jul. 2017.
[18] J. Chen, J. Chen, H. Chao, and M. Yang, "Image blind denoising with generative adversarial network based noise modeling," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3155–3164.
[19] P. Liu, H. Zhang, W. Lian, and W. Zuo, "Multi-level wavelet convolutional neural networks," *IEEE Access*, vol. 7, pp. 74973–74985, 2019.
[20] T. Remez, O. Litany, R. Giryes, and A. M. Bronstein, "Deep convolutional denoising of low-light images," *CoRR*, vol. abs/1701.01687, pp. 1–11, Feb. 2017.

[21] J. Zeng, J. Pang, W. Sun, and G. Cheung, "Deep graph Laplacian regularization for robust denoising of real images," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2019, pp. 1–10.
[22] L. Theis and M. Bethge, "Generative image modeling using spatial lstms," in *Proc. 28th Int. Conf. Neural, Inf. Process. Syst.*, vol. 2, Cambridge, MA, USA, 2015, pp. 1927–1935.
[23] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[24] R. E. Blahut, "Computation of channel capacity and rate-distortion functions," *IEEE Trans. Inf. Theory*, vol. IT-18, no. 4, pp. 460–473, Jul. 1972.
[25] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2014, pp. 813–833.
[26] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 6645–6649.
[27] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Proc. Ann. Conf. Int. Speech Commun. Assoc.*, 2014, pp. 1–5.
[28] W. Byeon, "Image analysis with long short-term memory recurrent neural networks," Ph.D. dissertation, Dept. Comput. Sci., Univ. of Kaiserslautern, Kaiserslautern, Germany, 2016.
[29] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2014, pp. 740–755.
[30] R. Timofte, V. De, and L. V. Gool, "Anchored neighborhood regression for fast example-based super-resolution," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 1920–1927.
[31] H. R. Sheikh, M. F. Sabir, and A. C. Bovik, "A statistical evaluation of recent full reference image quality assessment algorithms," *IEEE Trans. Image Process.*, vol. 15, no. 11, pp. 3440–3451, Nov. 2006.
[32] A. Hore and D. Ziou, "Image quality metrics: PSNR vs. SSIM," in *Proc. IEEE Int. Conf. Pattern Recognit.*, Aug. 2010, pp. 23–26
[33] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.
[34] M. Everingham and J. Winn, "The PASCAL visual object classes challenge 2012 (VOC2012) development kit," in *Proc. Pattern Anal., Stat. Model. Comput. Learn.*, Oct. 2011, pp. 1–32.

WUTTIPONG KUMWILAISAK received the B.E. degree from Chulalongkorn University, in 1995, the M.S. and Ph.D. degrees from the University of Southern California, in 2003, all in electrical engineering, with the support from Thai Government Scholarship.

From May–August of 2001 and 2002, he was a Research Intern with the Ericsson Eurolab, Aachen, Germany, and with the Microsoft Research Asia, Beijing, China, respectively. From April 2003 to August 2004, he was a Senior Engineer and a Project Leader of the mobile platform solution team and multimedia laboratory at Samsung Electronics, Suwon, South Korea. He was a Postdoctoral Fellow with the Thomson Research Laboratory, Princeton, USA, from March 2006 to November 2006. He has been an Associate Professor with the Electronics and Telecommunication Department, King Mongkut's University of Technology Thonburi, Bangkok, Thailand. His research interests are in the optimization and algorithmic design for wireless communications and multimedia communication systems. His current focused researches include multimedia communication, multimedia compression and processing, and 3-D image processing.

TEERAWAT PIRIYATHARAWET received the B.E. degree in electronics and telecommunication engineering from the King Mongkut's University of Technology Thonburi (KMUTT), Bangkok, Thailand, in 2016, where he is currently pursuing the M.E. degree in electrical engineering.

His research interests include image processing, computer vision, deep learning, object detection, and learning based depth estimation.

Mr. Piriyatharawet was a co-recipient of the 18th International Symposium on Communications and Information Technologies (ISCIT 2018) Best Paper Award, in 2018.

PONGSAK LASANG (Member, IEEE) received the B.E. degree (Hons.) in electronics and telecommunication engineering, the M.E. degree in electrical engineering, and the Ph.D. degree in electrical and computer engineering from the King Mongkut's University of Technology Thonburi (KMUTT), Bangkok, Thailand, in 2005, 2006, and 2016, respectively.

From 2005 to 2006, he was a Research Assistant with the Thailand's National Electronics and Computer Technology Center (NECTEC). Since December 2006, he has been with the Panasonic Research and Development Center Singapore (PRDCSG), Singapore, and he is currently a Senior Research and Development Manager. Since then, he has been working on camera processing and 3D related algorithms design. He is the author of more than 60 inventions and holds ten patents. His research interests include multiview image/video processing, depth map estimation and 3D reconstruction, SLAM, 3D point cloud compression, digital camera image processing pipeline, computational photography, and light-weight deep learning for edge devices.

Dr. Lasang is a member of the ACM. He was a co-recipient of the IEEE Consumer Electronics Society Best Paper Award in ICCE 2010 and the 18th International Symposium on Communications and Information Technologies (ISCIT 2018) Best Paper Award, in 2018.

NATTANUN THATPHITHAKKUL received the B.E. and M.E. degrees from Suranaree University, Thailand, in 2000 and 2002, respectively, and the Ph.D. degree in computer engineering from the King Mongkut's Institute of Technology Ladkrabang, in 2008. He is currently the Chief of the Accessibility and Assistive Technology Research Team, National Science and Technology Development Agency, Thailand. His research interests include speech and speaker recognition, speech synthesis, natural language processing, human machine interaction, and assistive technology.