Enhancing Object Clarity In Single Channel Night Vision Images Using Deep Reinforcement Learning

Mohammad Elham Robbani, Adil Hossain, Md. Riaz Ul Haque Sazid, Sk. Shahiduzzaman Siam,
Wasiu Abtahee, Amitabha Chakrabarty
Department of Computer Science and Engineering, Brac University, 66 Mohakhali, Dhaka-1212
{mohammad.elham.robbani,adil.hossain,md.riaz.ul.haque,sk.shahiduzzaman.siam,wasiu.abtahee}@g.bracu.ac.bd,
amitabha@bracu.ac.bd

2021 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE) | 978-1-6654-9552-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/CSDE53843.2021.9718444

***Abstract*—This paper implements a system of enhancing single channel night vision images using reinforcement learning approach and optimizing pixel prediction using q-table. We implemented some models to learn and process a small static images dataset using a reward bias q-table in a reinforcement learning architecture thus optimizing computational complexities and requirements of large dataset with the help of q-table. It also outperformed with respect to existing CNN models like SRCNN. Where SRCNN is observed to generate a PSNR of 24.813 on average at 256 batch size. Our system generated a PSNR of 24.1 on average with results in a 10.29% increase of relative efficiency at 3000 epoch. It has shown a 10.39% and 10.36% increase of efficiency with respect to VDSR (at 128 batch size) model and DRCN (at filter number 16) model respectively.**

***Index Terms*—Deep Q Learning, AI , Intelligent Agent , Reinforcement Learning, Deep Q Network, Single Channel Images, Night Footage, Data-set limitations.**

I. Introduction

In single channel images from night footage, things are sometimes hardly recognized so taking those single channel images of night camera footage to a recognizable level where people can easily identify those objects that they are desiring to see or investigate was our target. Many studies have been found in the literature on this similar topic, one of them is "Waterloo Exploration Database: New Challenges for Image Quality Assessment Models" [1]. In their paper to assess the generalization ability and aid the broad application of Image Quality Assessment (IQA), they made a large scale database called Waterloo Exploration database having 4744 clear natural images and 94880 distorted images.

Analyzing single channel night vision footage and enhancing the clarity of footage to objectify the target object is the main goal of this research. A detailed discussion is presented on which part of view we can relate to night vision camera footage to single channel image. How those footages are related to less clarity images and more noisy images. Moreover, our data-set which is hooked up to all real single channel images helps to outcome a more enhanced image to objectify the target. These particular features more or less answer those questionnaires.

Our main objective is to make single channel images more comprehensive where objects are complicated for bisecting in details by q learning approach. Keeping this in mind we started to collect data from different CCTV camera footage at night which are basically single channel images. We focused on getting those data where objects cannot be labeled properly due to a lack of details. We used relevant resources to test and train our model. We will ensure data relevancy-based peak signal to noise ratio (PSNR) for noise reductions and variance of laplacian variance to get rid of blurriness [2]. Finally, reinforcement learning with pixel-wise rewards takes us to get the results.

II. Background

Night footage is gray-scaled and in gray-scaled footage, the desired object identification also gets tougher for not having an RGB scale. So we decided to work on this particular sector to objectify target objects at night using a novel reinforcement learning approach that will utilize the Actor critic model in a fully convolutional network [3].

Convolutional Neural Network (CNN) has been used to separate highlights from pictures while decreasing the computational load with the algorithms in terms of crude pictures [4].This can further be taken to a higher level by using DQN or deep Q network that works on a reward biased platform that removes the requirement of a classed dataset. [5] [6].

To improve the agent's execution under boisterous conditions Adversary Robust A3C (AR-A3C) was used. The AR-A3C calculation beats A3C in both perfect and uproarious conditions [8]. Deep Reinforcement learning consists learning algorithms which demonstrates that actor-learners have a stabilizing effect prepares neural network regulators to optimize the whole basic and lightweight system. Research has been done where Convolutional Neural Network (CNN) to play ongoing exploration has demonstrated that map raw pixels from a solitary forward-looking camera straightforwardly to directing orders are shockingly incredible [7].

Here we propose to make a system demonstrate the features of pixel reinforcement learning on a Fully Convolutional Network (FCN). It would be a novel work as this would be the first time pixel reinforcement learning would be used on a networking system like FCN. FCN essentially decreases processing complexities compared to CNN or DQN and it also

Authorized licensed use limited to: BRAC UNIVERSITY. Downloaded on August 12,2026 at 09:46:11 UTC from IEEE Xplore. Restrictions apply.

optimizes computing time and so this creates a new horizon for future research and implementations.

III. Data and Feature

As we have collected the data from different sources like CCTV, night vision cameras and a few open sources. We will see how we processed and verified the Data Relevancy in this section. We will also analyze the features if they are fit for training and testing so that we can compare the efficiency of the model.

A. Data collection and Data Relevancy

After identifying the key features that our model will have and started collecting primary and secondary data. The validity of the data that we collected was checked using Peak Signal-to-Noise Ratio (PSNR) [8]. This works in an inversely proportional way that is, the higher the value the lower the noise [9]. In the data preprocessing part initially all the images were turned into single channel grayscale images. We used this type of images as fewer data will be collected for every pixel compared to other textured images. Indeed 'grey' texture is one in which the red, green, and blue segments all have the same power in RGB space, thus it is simply salient to indicate a sole force as an incentive for every pixel. [10]. Then we set a constraint on the size of each image, the width was set to 481 pixels and height 321 pixels. After getting all the datasets, we used 78% of data for training and 22% for testing. This concluded our data collection and data preprocessing part.

B. Feature Analysis

Here we will talk about the features we have used, Variance of Laplacian & Added Salt and Pepper Noise.

Variance of Laplacian: After mapping the response, the variance is recorded within a threshold, which utilizes the second degree subordinate of the picture. As a result of this the laplacian variance generated, can differentiate the quick intensity of the kernel changes similar to the Sobel and Scharr administrators. Similar to this administrator the Laplacian variance is repeatedly used for edge discovery. The presumption here is that if a picture contains high fluctuation at that point there's a widespread of reactions.As we know, the more a picture is obscured, the fewer edges there are [11].

Thus by setting up the laplacian variance architecture we run the whole pre-processed dataset and analyze the data further in data visualization to confirm the relevance of the dataset, reliability, and proving the ground truth of the work. In the analysis, we will take the combined dataset and run it into the open CV Laplacian operator to extract the data and sort all the blurred images to continue the further work.

Added Salt and Pepper Noise: Salt and pepper noise delivers a wide variety of processing and results in basic image distortion. The impact of the accumulation of dead pixels with a few noisy pixels is comparative to sprinkling white and dark dots—salt and pepper—on the image. One case where salt and pepper noise emerges is in transmitting pictures over noisy digital joins. Let each pixel be quantized to A bits within the normal mold. The value of the pixel can be composed as

$$Y = \sum_{i=0}^{A-1} b_i 2^i. \tag{1}$$

Accept the channel could be a twofold symmetric one with a hybrid likelihood of ϵ. At that point, each bit is flipped with likelihood ϵ. Call the gotten esteem, X. At that point, accepting the bit flips are autonomous,

$$\Pr\left[|Y - X| = 2^i\right] = \varepsilon(1-\varepsilon)^{A-1} \tag{2}$$

For $i = 0, 1.2, ..., m-1$ .The MSE due to the foremost critical bit is $\varepsilon 4^{A-1}$ comparing to $e(4^{m-1}-1)/3$ for all the other bits combined. To say in other sentences, the commitment to the MSE from the foremost noteworthy bit is around three times that of all the other bits. The pixels whose most critical bits are changed will likely show up as dark or white dots. Salt and pepper clamor is a case of (exceptionally) overwhelming followed noise. A simple model is: Let f(x,y) be the initial picture and q(x,y) be the picture after it is modified by salt and pepper noise [12].

$$\begin{gathered}\Pr[q = f] = 1 - \gamma; \Pr[\mathbf{q} = \mathbf{MAX}] = \gamma/2; \\ \Pr[\mathbf{q} = \mathbf{MIN}] = \gamma/2.\end{gathered} \tag{3}$$

Here, MAX and MIN are the greatest along with the least picture values, separately. For 8 bit pictures, MIN = 0 and MAX = 255. The modified pixels seem like dark and white specks sprinkled over the image. Fig 1 shows the impact of salt and pepper commotion. Around 15% of the pixels have been set to dark or white (85% are unaltered). Take note of the sprinkling of the dark and white dabs. Salt and pepper noise is effectively expelled along different arrange measurement channels, particularly the center-weighted middle and the LUM(luminescence) channel [13].

Fig. 1. Example of a high PSNR count added with salt and pepper image representing noisy image in dataset

Keeping salt and pepper distortion as one of the major features of our training model, the dataset was trimmed and after preprocessing we need to ensure for any Gaussian noise present in the dataset so that we can prove the ground truth of removing the noise [14]. Just like we discussed the PSNR in the dataset section we will be running the PSNR again to ensure gain of PSNR over the post-stage of the dataset processing [15].

Authorized licensed use limited to: BRAC UNIVERSITY. Downloaded on August 12,2026 at 09:46:11 UTC from IEEE Xplore. Restrictions apply.

Feature Analysis: According to the previous discussion the result we generated by using laplacian variance and PNSR of salt pepper noise or any other noise present like Gaussian noise. Now we will visualize the data in chart scaling so that it is easier to have an idea about the whole dataset quality and further analysis to come to a conclusion on the features we are using in the relevance of the dataset. Firstly, let us take a look at the graph data of laplacian variance of the whole dataset Fig. 2. Here we can see the variance count of the laplacian over a gradual count of images. Keeping the note here that we can see a high peak of the variance that is confirming a very blurry image. So in the later part, of the work we will be training our Q agent and generate a model according to reward biases and again will have a look at the output data. And look for a decrease of the variance to approve our ground truth of decreasing blur in images. But for now, we can confirm the feature relevancy according to the laplacian variance of the dataset. Secondly, Let us have a look at the PSNR analysis of added noise based on salt pepper (First priority) or any other noise like Gaussian noise(if previously available)

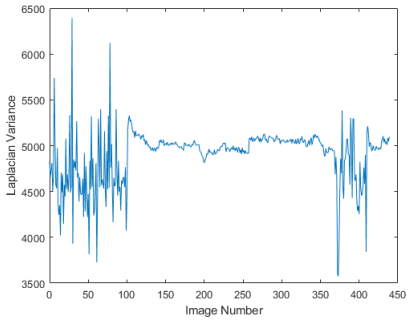

Fig. 2. Graph representation of Laplacian variance of the dataset

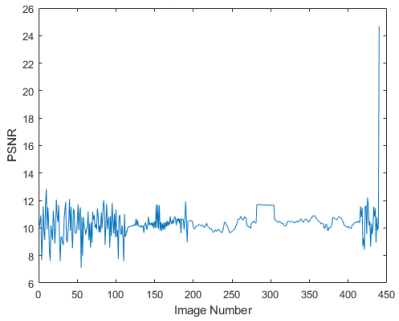

Fig. 3. Graph representation of PSNR count of the dataset

Image sequence of the dataset. The lowest value is approximately near 5 and the highest value is approximately near 15, (Values have been shown in Fig 3). Before the explanation let's know a bit more about how this ratio is working and how we should be expecting the future generation of results to be. As, PSNR represents a ratio that means $\Delta = \mathrm{x}/\mathrm{y}$ where x is a standard that represents a non-blurry image and y represents a blurry image from the dataset. So the resulting Δ or the PSNR value will be less in range of the axis it has been plotted. Or in other words more the noise is less the Δ value. So the visualized data is confirming the final dataset and the feature with the presence of enough noise that needs to be filled by prediction through our Q learning agent. Let's keep a note of this chart and move onto our gradual development of the trained model and the test result and see if this Δ expands its range and confirms the ground truth of denoising the image dataset. To conclude after the feature analysis let us overview the process we gradually developed our feature in our dataset. The features have been artificially added so that the ground truth can be obtained after these distorted images are run into the trained model. Firstly the laplacian variance was compiled and any image not reaching the threshold was excluded. And added noise based on salt and pepper / Gaussian noise so that the images can be developed by the intelligent agent. Thus a proper dataset containing suitable features for further training of the model has been generated.

IV. Model Selection

In this research, we used some specialized models to increase efficiency and flexibility. Our 1st model is Q– learning which is a reinforcement learning model calculation to learn the nature of activities mentioning to an agent what move to make under what conditions [16]. It doesn't need a specific environment to work on and it can deal with issues with stochastic changes and rewards, without requiring transformations. We also used Q– mapping which is a procedural surface that guides painting prompts onto objects, and chips away at polygonal information, on volumes, and even on record. Q– mapping is mainly an advanced version of the Universal value function model which takes observation and the goal as an input and creates Q– values as an output for every action. All this output works as a Q – value which is mainly a goal-reaching reward function where the objective arriving at completing an episode will give the reward as 1 [17]. But if it fails to reach the goal i.e., if it does not complete the episode then it will get 0. There is a γ which is used as a discount value which creates dramatically decaying esteems $\gamma^{(\mathrm{k}-1)}$ which express the quantity of steps k to the objectives. This approach effectively exploits a single environmental transition to construct a collection of distinct one-step episodes, one for each goal, with partial-episode bootstrapping if the goal is achieved. All other models will work simultaneously as those are reward-based systems. As soon as the q value begins to generate rewards, all other processes will begin to operate on getting the reward 1. Q– learning algorithm mainly works on a formula,

$$Q\left(s_t, a_i\right) = Q\left(s_t, a_i\right)(1-\alpha) + \alpha\left[r_{i+1} + \gamma_a^{\max} Q\left(s_{t+1}, a\right)\right] \tag{4}$$

Also, we used Fully Convolutional Network(FCN) which is a specialized architecture used mainly for semantic segmentation. It employs only locally connected layers. It tries to avoid dense layers with many parameters so working under fewer parameters means faster to train. So the main function of FCN

Authorized licensed use limited to: BRAC UNIVERSITY. Downloaded on August 12,2026 at 09:46:11 UTC from IEEE Xplore. Restrictions apply.

is to create a local network where multiple agent models like A3C can share the parameters and work parallely on different environments so they explore and learn efficiently and render the training efficiently [17]. We used it instead of N network as N network is computationally impractical when it comes to handling a large number of pixels.

The 2nd model we used is A3C which is an Asynchronous Advantage Actor-Critic algorithm where numerous specific agents have their own arrangement of network parameters. At the same time distinct agents communicate with their own surroundings and cooperates with their own duplication in the environment [18]. The key reason we use this technique is that each agent's experience is independent of the experience of others, allowing them to advance to the next state faster. [5]. It uses forward–view and n – step updates which means the edge selects action using its exploration policy for up to specific t steps in the future. For boosting training stability, we use parallel actor learners and cumulative updates, just like with value-based approaches. The agents will then receive up to t rewards from the environment since its last update and all these activities take place in FCN. Agents' main function is to find the policy (π) maximizing the expected return. The policy is mainly a process to map the next move to interact with the next state.

The 3rd model is PixelRL which is a learning method where every pixel has its own agent and that agent changes the pixel esteem by taking an activity which includes a compelling learning mechanism that significantly enhances the presentation by considering not just the future conditions of one's own pixel, but also those of its neighbors [19]. Each pixel is backed by its own agents and the policy that is contained for them. The main objective to use PixelRL is to learn optimal policies that maximize the mean of the total expected rewards for all the pixels through output as Q– values or policies for all the actions. But it will be less effective if they are connected with many layers or the number of datasets is huge. So, the solution is to create a number of networks where an agent will find the optimal policy in that network. But in this process, it will be increasing the computation time because there is N number of networks which will create pressure on the GPU to process it [19]. So, we used Fully Convolutional Network to create only one shareable network which is easy to maintain and we also use it to connect it with A3C where multiple agents handle those N number of pixels and shared data in FCN created networks. By doing this we decrease the complexity and computation time of our system.

And the last method that is used in this research is the edge detection/finder method. An edge finder is a straightforward calculation to separate edges from a picture. Applying an edge identifier to a picture creates a bunch of associated curves which follow the limits of an object. Most edge finder methods utilize either first-order derivatives or second-order derivatives [20]. The first and second derivatives helps to find the local maxima of a point and gradient respectively. These maxima can be found from zero crossings of Laplacian or nonlinear differential expressions [21]. This method mainly works by detecting the change of intensity between two close points. The higher the intensity difference the more accurately we can detect the edge between those points and to detect that we used Sobel edge detection method [21]. The Sobel edge detector takes edges from the pictures utilizing edge esteem given by the reinforcement learning agents. The separated edges are evaluated by the assessment module and the outcome is used as compensation for the reinforcement learning agents, which takes the related action and gives another edge detector threshold. The learning measure proceeds until the ideal limit is found. Pratt Figure of Merit is a notable measure used to analyze edge detectors output. It tries to handle three types of errors that can create wrong edge mappings: missing legitimate edge points, disengaged edge points, and misclassification of commotion variances as edge points. Pratt Figure of Merit utilizes a known edge for correlation [2]. The main formula of Pratt Figure of Merit is -

$$R = \frac{1}{I_N} \sum_{i}^{I_A} \frac{1}{1 + \alpha d_i^2} \tag{5}$$

V. Result Analysis

A. Reward Test Results

Here we can see the extracted data for each 100 episodes up to 3000 episodes which have been plotted into the following graphs Here, as we have shown the breakdown of the result of the 3000 episodes into gradual development based on the proposed model. We plotted the episodes into batch(100 in each) so this generated us 11 subsequent results and in this Fig 4 we have made a superposition of their performance based on reward exploration which confirms development ratio of the image. After we are done with training the model, the output trained model is then run with respect to the test dataset which contains noisy and blurred images.

That in other words means the reward bias is increasing fairly and the efficiency of the model is getting better with each batch of training episodes. We will have a look at their picture restoration performance in the later part of the work.

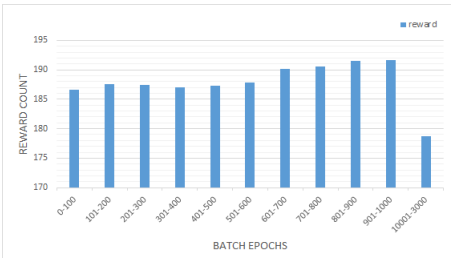

Fig. 4. Superposition of Reward Test results of 0-3000 episode

We plotted the performance again after 3000 episodes to observe any drastic changes as the batch training is stretched further, here in Fig 4 the 11th column represents up to 3000 batch episodes results, here the reward bias drops but the agent itself now can get a more satisfactory result because this has accumulated more accurate data thus the bias drops here and

Authorized licensed use limited to: BRAC UNIVERSITY. Downloaded on August 12,2026 at 09:46:11 UTC from IEEE Xplore. Restrictions apply.

on training further, it will gain the reward bias again. But performance has been recorded high at this stage.

B. PSNR Test Results

After carrying out the reward test analysis, just like we ran our PSNR test result in the data pre-processing and feature analysis chapter, we will be using the PSNR test again in the output images of the trained models based on batch episodic training. And observe the changes in PSNR data visualization to come to a conclusion if the model produced a satisfactory result. We have discussed before that the PSNR is a ratio(Peak Signal to Noise Ratio) that means the Δ represents the ratio of noise between a standard noise-free image and a noisy image. So we will be comparing the Input noisy image with the output noise-reduced image and visualize it in a graph to observe performance. We have to remember that, less noise more will be the result of the ratio. That means we had the highest threshold of 15 and we need to observe a high scale more than that in the test result. Let us have a look at the extracted data for test batch image on 1000 episodes and also up to 3000 episodes which have been plotted into the following graphs :

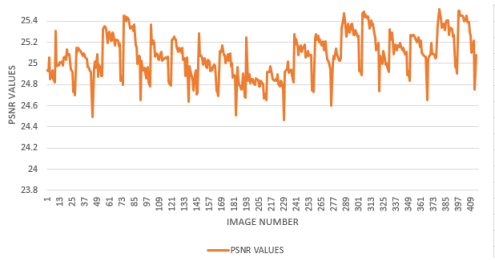

Fig. 5. ACCUMULATED PSNR VALUE 0-1000

Super positioned PSNR Test result of 0-1000 batches in Fig. 5 which shows the output images generated as a whole was within a higher PSNR threshold which clarifies that the output images were noise reduced and the gradual development of the exploration of the agent was stable. Thus it is also clear that for better output results the agent can be trained further and observed the performance to properly clarify our ground truth.

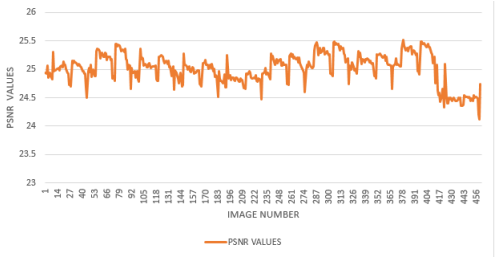

Fig. 6. ACCUMULATED PSNR VALUE 0-3000

After superpositioning the PSNR Test result of 0-3000 batches in Fig.6 we can observe consistency but the output result was drastically changed as we have mentioned before the pixel agents have exploited the dataset more than before and the Q maps have expanded drastically result in a better model that can generate efficient output restored images. We will be having looks and analysis on the output images in the following part of the chapter.

C. Result Analysis

In Fig. 7 we have represented an image from the test dataset on the left side named as the input image. And on the right side, we can see the relative output of the image going through the model trained up to 3000 episodes in batches. Specifically, there is a dramatic change in episode 3000 as the model was well trained due to the expansion of the Q map and more exploration of the past data set. The Pixel wise AI agent has shown remarkable output over time and episode. But keep in note that this model generated exponential memory complexity for which the result was carried out up to 3000 episodes. But within these 3000 episodes, the identity was clearly visible and the noise was reduced at a very considerable rate. Also after arranging the results, the blur edges of the image were restored and smoothed bringing the identity more clearly visible.

TABLE I
PERFORMANCE CHART WITH EXISTING APPROACHES AND PIXELRL TABLE

SRCNN		
Mectric	Batch Size	Average
PSNR	64, 128	24.623, 25.046
VDSR		
Metric	Batch Size	Average
PSNR	64, 128	25.267, 25.047
DRCN		
Metric	Filter No	Average
PSNR	16	24.983
Pixel RL		
Metric	Episode No	Average
PSNR	1000, 3000	25.268, 24.1

In the above TABLE I we can see the relative average PNSR Chart of our Pixel RL-based model compared to other CNN based-models like Super-Resolution Convolutional Neural Network (SRCNN). Very Deep Super Resolution (VDSR) and Deeply-Recursive Convolutional Network (DRCN). Here we can clearly see that the performance of noise reduction at 1000 epoch is not better than the other models but after further training, up to 3000 epoch, the noise reduction is 10% more than the other models.

Finally analyzing the facts based on the output image results and all the data represented in the previous sections of the chapter we can conclude that the ground truth of our work is enhancing the object clarity by removing the noise and blur edges using the proposed model which is deep reinforcement learning which also utilizes the Q-learning model has enhanced the clarity of images extracted from single channel image or video source.

VI. CONCLUSION

In our proposed system we implemented pixel wise reinforcement learning in single channel night vision images

Authorized licensed use limited to: BRAC UNIVERSITY. Downloaded on August 12,2026 at 09:46:11 UTC from IEEE Xplore. Restrictions apply.

Fig. 7. Generated Image Output By The Trained Model Showing The Enhancement And Noise Reduction Gradually

mostly from CCTV, to enhance the object clarity and quality. We implemented A3C Asynchronous Advantage Actor–Critic model and also implemented Q-learning model to optimize the system for handling an exponential number of increasing pixels. To verify the system efficiency we processed the data-set with added salt and pepper noise and verified the blurry image level in the Laplacian variance method. With a data size of 128 and having gamma=1 and sigma=10 with a learning rate of 0.05 PixelRL based model is being proposed but it showed issues related to complexities and hardware. To overcome these issues, the models is turned at 64 batch size having gamma=0.95 and sigma=15 with a learning rate of 0.001 and pruning some layers of the model we achieved better efficiency in training and output when we trained the model up to 3000 episodes. After training the agent with the data-wet where the average PSNR was 10.37 we tested another set of unexplored data set through our trained agent and observed the output PSNR to be 24.1 which clearly states the noise reduction with removal of blurred edges as we know in Peak Signal to Noise ratio, here more the Δ less is the noise. The model was also compared to efficiency result with other existing CNN approaches. These procedures were carried out in a still image based data-set. Although the image reinforcement based model worked decent with memory limitations in the future, we will investigate on performance optimization of this model and implement it in dynamic video channels to serve in real-time video data.

REFERENCES

[1] L. Wang, W. Shao, and Q. Ge, "A deep convolutional neural network-based low-light image enhancement using illumination map," in *Eleventh International Conference on Graphics and Image Processing (ICGIP 2019)*, Z. Pan and X. Wang, Eds., vol. 11373, International Society for Optics and Photonics. SPIE, 2020, pp. 276 – 283.

[2] R. Aparna and M. J. Josemartin, "Application of image intensity local variance measure for analysis of distorted images," in *2017 International Conference on Networks Advances in Computational Technologies (NetACT)*, 2017, pp. 382–386.

[3] R. Nakamura, Y. Mitsukura, and N. Hamada, "Blind restoration of single-channel image using iterative pca." IEEE Computer Society, Jan. 2013, pp. 84–87.

[4] M. Garg and G. Dhiman, "Deep convolution neural network approach for defect inspection of textured surfaces," *Journal of the Institute of Electronics and Computer*, vol. 2, no. 1, pp. 28–38, 2020.

[5] R. Furuta, N. Inoue, and T. Yamasaki, "Pixelrl: Fully convolutional network with reinforcement learning for image processing," *IEEE Transactions on Multimedia*, vol. PP, pp. 1–1, Dec. 2019.

[6] W. Li, X. Feng, H. An, X. Y. Ng, and Y.-J. Zhang, "Mri reconstruction with interpretable pixel-wise operations using reinforcement learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, pp. 792–799, Apr. 2020.

[7] T. Liu, Z. Tan, C. Xu, H. Chen, and Z. Li, "Study on deep reinforcement learning techniques for building energy consumption forecasting," *Energy and Buildings*, vol. 208, p. 109675, 2020.

[8] M. Kazemi, M. Ghanbari, and S. Shirmohammadi, "The performance of quality metrics in assessing error-concealed video quality," *IEEE Transactions on Image Processing*, vol. 29, pp. 5937–5952, 2020.

[9] S. Wan, Y. Xia, L. Qi, Y.-H. Yang, and M. Atiquzzaman, "Automated colorization of a grayscale image with seed points propagation," *IEEE Transactions on Multimedia*, vol. 22, no. 7, pp. 1756–1768, 2020.

[10] M. Paramasivam, R. Sabeenian, and P. Dinesh, "Perceptually weighted color-to-grayscale conversion for images with non-uniform chromatic distribution using multiple regression," *ICTACT Journal On Image And Video Processing*, Nov. 2020.

[11] R. Bansal, G. Raj, and T. Choudhury, "Blur image detection using laplacian operator and open-cv," in *2016 International Conference System Modeling Advancement in Research Trends (SMART)*, 2016, pp. 63–67.

[12] W. Preedanan, T. Kondo, P. Bunnun, and I. Kumazawa, "A comparative study of image quality assessment," in *2018 International Workshop on Advanced Image Technology (IWAIT)*, 2018, pp. 1–4.

[13] M. A. Baig, A. A. Moinuddin, and E. Khan, "Psnr of highest distortion region: An effective image quality assessment method," in *2019 International Conference on Electrical, Electronics and Computer Engineering (UPCON)*, 2019, pp. 1–4.

[14] A. Sharadqh, B. Ayyoub, Z. Alqadi, and J. Al-Azzeh, "Experimental investigation of method used to remove salt and pepper noise from digital color image," pp. 10–12, 01 2018.

[15] C. R. Helmrich, M. Siekmann, S. Becker, S. Bosse, D. Marpe, and T. Wiegand, "Xpsnr: A low-complexity extension of the perceptually weighted peak signal-to-noise ratio for high-resolution video quality assessment," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 2727–2731.

[16] B. Jang, M. Kim, G. Harerimana, and J. W. Kim, "Q-learning algorithms: A comprehensive classification and applications," *IEEE Access*, vol. 7, pp. 133 653–133 667, 2019.

[17] F. Pardo, V. Levdik, and P. Kormushev, "Q-map: a convolutional approach for goal-oriented reinforcement learning," 10 2018.

[18] K. Chen, R. Li, Z. Zhao, and H. Zhang, "The implementation of asynchronous advantage actor-critic with stigmergy in network-assisted multi-agent system," in *2020 International Conference on Wireless Communications and Signal Processing (WCSP)*, 2020, pp. 1082–1087.

[19] C. Zhong, Z. Lu, M. C. Gursoy, and S. Velipasalar, "Actor-critic deep reinforcement learning for dynamic multichannel access," in *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2018, pp. 599–603.

[20] B. Jang, M. Kim, G. Harerimana, and J. Kim, "Q-learning algorithms: A comprehensive classification and applications," *IEEE Access*, vol. PP, pp. 1–1, 09 2019.

[21] Y. Dar, M. Elad, and A. Bruckstein, "Restoration by compression," *IEEE Transactions on Signal Processing*, vol. PP, 11 2017.

Authorized licensed use limited to: BRAC UNIVERSITY. Downloaded on August 12,2026 at 09:46:11 UTC from IEEE Xplore. Restrictions apply.