RAIN STREAKS ELIMINATION USING IMAGE PROCESSING ALGORITHMS

Dinesh Kadam[1] , Amol R. Madane[2] , Krishnan Kutty[2] and S. V. Bonde[1]

[1]Department of Electronics and Telecommunication, SGGSIET, Nanded, India
[2]Tata Consultancy Services Ltd., Pune, India

ABSTRACT

The paper addresses the problem of rain streak removal from videos. While, Rain streak removal from scene is important but a lot of research in this area, robust and real time algorithms is unavailable in the market. Difficulties in the rain streak removal algorithm arises due to less visibility, less illumination, and availability of moving camera and objects. The challenge that plagues rain streak recovery algorithm is detecting rain streaks and replacing them with original values to recover the scene. In this paper, we discuss the use of photometric and chromatic properties for rain detection. Updated Gaussian Mixture Model (Updated GMM) has detected moving objects. This rain streak removal algorithm is used to detect rain streaks from videos and replace it with estimated values, which is equivalent to original value. The spatial and temporal properties are used to replace rain streaks with its original values.

KEYWORDS

Dynamic Scene, Edge Filters, Gaussian Mixture Model (GMM), Rain Streaks Removal, Scene Recovery, Video Deraining

1. INTRODUCTION

Outdoor vision system, used for surveillance and monitoring systems is a concern these days. However, most algorithms used in outdoor vision systems assume clear weather and degrade their performance during bad weather such as rain, fog, snow etc. Bad weather degrades the image and video quality which degrade the performance of computer vision algorithms. Many computer vision algorithm used in outdoor vision system requires good quality of image for object detection, tracking, segmentation and recognition. So it is very difficult to implement such algorithm in various weather conditions [2][9].

Based on type of visibility, bad weather conditions are classified into two types such as steady weather and dynamic weather [9]. Steady weather consists of fog, mist and haze. These particles are smaller in size (1 – 10 µm). Individual detection of each particle is very difficult using camera. Each pixel is affected by aggregation effect of multiple particles. In dynamic weather consist of rain and snow. These particles are 1000 times larger than steady weather particles. Due to larger size, these particles are clearly visible to camera. Each particle affects multiple pixels of image.

Rain is major component in dynamic weather. Rain streaks are spatially and temporally distributed. Rain produces sharp intensity changes. Visibility of scene is reduced due to rain. Raindrop refract and reflect light from its surrounding. Due to that, rain streaks are properly visible making image quality low. Rain removal from a scene is tedious task. Rain streaks disturb the image analysis algorithm which depend on video captured using camera. A surveillance camera does not want its motion detection to trigger on different weather phenomena. The streaks may also cause failure in face recognition algorithms when raindrops occlude parts of the face [4]-[6].

Visual appearance during rainy season is low as compared with summer season. Rain produces sharp intensity changes in pixels of images and videos. The unique physical properties of rain are small size, high velocity, and spatial distribution [9]. Rain streaks are larger than steady weather which makes unclear visibility. These streaks are visible to video capturing sensors.

The paper is organized in various sections as- the literature survey of various papers from the same area have enlightened in next section. Various rain properties used for rain removal and Stauffer-Grimson method explained in detail. The explanation, comparison of proposed algorithm with existing algorithm have completed in algorithm, results and discussion section. Effectiveness of our algorithm is shown in detail using several examples.

2. LITERATURE SURVEY

Paper [1] has highlighted the algorithm to check the facial expressions of driver who is driving vehicle. The features are captured and matched with standard available featured stored in database. The method used for checking the driver drowsiness which helped to reduce the abnormal activities during driving. This drowsiness can be checked by checking eye blinking pattern and head pose movements. This algorithm is facing problem during rainy situations. Combination of this drowsiness algorithm with rain free road can provide best system in automotive domain.

Garg and Nayar analysed dynamics of rain [2]. They have presented visual effect of rain on image and blurring model that explains the photometric property of rain. They used photometric and physical property for detection of rain streaks. The temporal property of rain is used to removal rain by taking average of temporal pixels. Also, they have produced rain model which is used for rendering rain to video. They have used various properties for rendering rain also and same properties are used to detect rain pixels. However, their assumptions regarding uniform velocity of rain and direction of rain drop limit performance of algorithm. This algorithm does not handle steady effects of rain.

Kshitiz Garg and Shree K. Nayar presented unique properties of rain. They have experimentally proved that visibility of rain strongly depends on camera parameters such as exposer time and depth of field [3]. Using these parameters, rain appearance is removed without altering the scene. They have proposed algorithm, which remove effect of rain without post processing. However, this method cannot handle heavy rain or scene with object motion. Every time need to set camera parameters manually.

Triparty [4] proposed probabilistic spatial-temporal model where they have used various statistical features such as intensity fluctuation range and spread symmetry. This is to classify rainy pixels and remove the false rain detected pixels. This algorithm works only on intensity plane. It uses spatiotemporal properties to recover the scene. This algorithm able to handle dynamic scene. Time complexity is also less. But, some statistical features detects lot of false rain pixels.

Jie Chen and Lap Pui Chau [5] proposed algorithm for rain removal from video in which they have separated motion objects and rain pixels. Gaussian mixture model (GMM) along with optical flow and color cluster flow are used for motion detection. They have used photometric and chromatic property to detect the rain pixels. Then, they used rain removal filter for scene recovery by taking into consideration of motion pixels. This is to recover scene. Both spatial and temporal information is used for rain pixel recovery. Proposed algorithm works better in dynamic scene with large motion. However, time required for algorithm is too high.

A. K. Tripathi, S. Mukhopadhyay [7] proposed an algorithm that uses meteorological property of rain along with spatiotemporal property for detection and removal of rain from a scene. Various shape features such as area, aspect ratio etc are used for discrimination of rain pixels and moving objects. Proposed algorithm works only in intensity plane which reduces time complexity of algorithm and uses temporal frames for recovery of scene. Object edges are detected if con frames are not properly aligned.

3. PROBLEM STATEMENT

Outdoor video sequences are degraded due to various weather conditions such as rain, haze, fog etc. Visual appearance of rain produces sharp intensity changes in image. This makes it difficult to implement computer vision algorithms. During rainy season, rain streaks degrade the performance of applications, which rely on video input such as object tracking, video surveillance and robot navigation etc. Existing rain removal algorithms are used for offline post processing which takes time to process the data and blurs the final output. Some algorithms recover scene properly but algorithm complexity and processing time are too high. One of the important problem of existing rain removal algorithm is prediction of original value of rainy pixels. Pixel intensity in motion area is highly corrupted due to quick intensity change of motion pixels. This method replaces false intensities. Also, rain detection step produces high false and miss detection rate, which causes bad scene recovery in case of videos. We have defined algorithm to resolve above problems.

4. PROPOSED SOLUTION

4.1. Detection of Rain Streaks

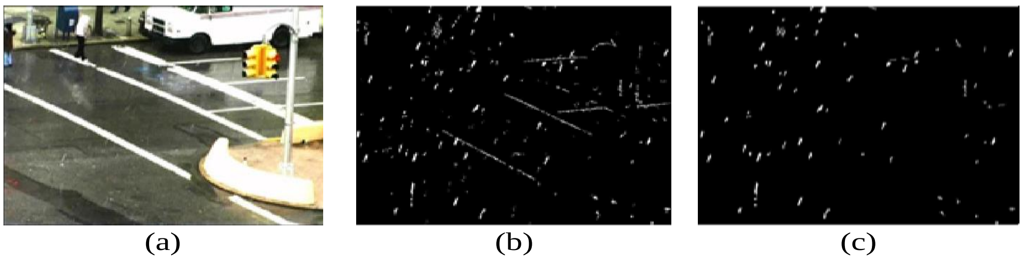

Figure 1: Rain detection (a) Rain input frame (Size- 480 X 720) (b) Results of rain detection using photometric and chromatic property (c) Results of rain detection after applying Prewitt edge detector.

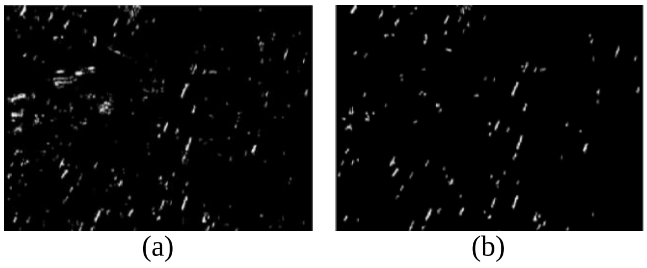

Figure 2: Final Rain Mask (a) Rain detection after applying photometric, chromatic, connected component and edge detector properties (b) Rain detection after applying streak width property on output of (a) step

Rain streaks are spatially distributed throughout the image. One of the unique physical property of rain is velocity [9]. Rain streaks falls at high velocity due to which same pixel is not covered by rain in successive frames [2]. Rain produces sharp intensity changes.

According to photometric property, rain pixels look brighter than background. While considering background and rainy frame, the pixel intensity value of rainy pixel is greater than intensity value of successive frame. In this case, the difference of successive frames followed by thresholding have taken. Value of threshold is selected in such a way that rain pixels should detect properly with less noise. If value of threshold is high then we may miss some rain pixels and if it is low, then noise is detected as rain pixels. Experimentally, value of threshold should be in between 5 to 12. Equation for photometric property is as shown below

The proceedings are the records of the conference. ACM hopes to give these conference by-products a single, high-quality appearance. To do this, we ask that authors follow some simple guidelines. In essence, we ask you to make your paper look exactly like this document. The easiest way to do this is simply to download a template from [2], and replace the content with your own material.

$$I_{thr} = \begin{cases} 1 & I_N - I_{N-1} \geq T \\ 0 & I_N - I_{N-1} \leq T \end{cases} \tag{1}$$

where IN is Nth frame and T is threshold used for rain separation.

Rain pixels get separated after applying photometric property. But, rain motion is also detected along with rain pixels. Recovery of such false detected rainy pixels may corrupt scene. This effect is present more at edges. So detection of rain requires some improvement. For improvement process, first we need to focus on moving object and edges. So, chromatic property of rain is used along with photometric property. Chromatic property states that absolute sum of colour channel differences (C) caused by rain should be within range C=7 to 20 [4] [9].

$$(\Delta R - \Delta G)^2 + (\Delta G - \Delta B)^2 + (\Delta B - \Delta R)^2 \leq C \tag{2}$$

If C value is low then few rain pixels will miss and if it is high then noise is detected as rain pixels. Experimentally, C value should be in between 15 to 20. Noise is reduced after applying chromatic property to rain detected in photometric property.

As rain pixels are smaller in size, area covered by rain streak is used for separation of rain and motion objects. Connected area of rain streak is limited to some pixels than motion object. Noise is getting reduced after applying connected component to detected rain streaks. Value of connected component chosen in such a way that all rain streak gets detected [5]. Experimentally, connected component value varies in between 8 to 600.

After applying photometric property, chromatic property and connected component, rain pixels are refined. There are some edges of object which are detected as rain pixels. Prewitt edge operator is used [4] to remove false rain pixels. Three successive frames are required for classification of edges. We need to classify edges into static and dynamic edges. If same edge is present in three successive frames then this edge belongs to static edge otherwise it is part of moving object. This removal of edge pixels produces better results. Edge information is obtained using below equation

$$PE_i(p) = \begin{cases} Noedge & PE_i(p) = 0 \\ Staticedge & PE_i(p) == PE_{i-1}(p) == PE_{i+1}(p) = 255 \\ Movingedge & otherwise \end{cases} \tag{3}$$

Where PEi (p) is edge information of Pth pixel of current frame and PE is Prewitt operator.

Falsely detected edges as rain streaks are removed so that these pixels not taken into consideration while inpainting which preserves edges and prevent from blurring output. Rain pixels get separated from frame after applying this step. Still, there are some pixels which satisfy all above properties but not part of rain. Those smaller rain streaks are removed by using rain streak width check. Rain streaks and motion objects is separated.

Next step is to measure rain streak width. Firstly we need to give labelling to each rain streak so that we can classify each rain streak. Then we just move in horizontal direction to find out rain pixel count in current row only i.e. scan entire streak and find out maximum count of same label in only particular row. This is width of rain streak. Experimentally, threshold is applied whose value is in between 15 to 20. The result is stored in Irain and shown in Figure 2.

4.2. Segmentation of Motion Pixels

Sharp intensity changes are due to rain and motion object. We need to separate the motion pixels as motion pixel intensity changes quickly [5]. If the rain streak is on moving object then it is difficult to replace the rain streak with current approach. This will corrupt the scene because motion pixels change quickly as compare to background pixel. Gaussian Mixture Model (GMM) is used to detect motion.

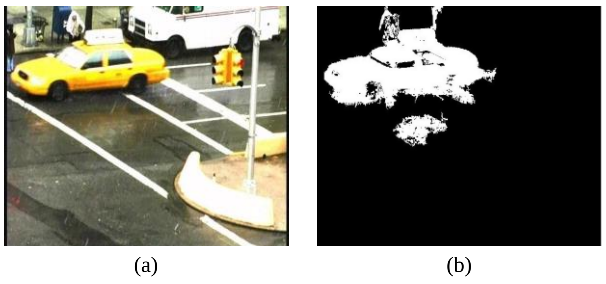

Figure 3: Simulation result of motion field segmentation using GMM (a) Input frame (480 X 720) (b) Result of foreground detection using GMM

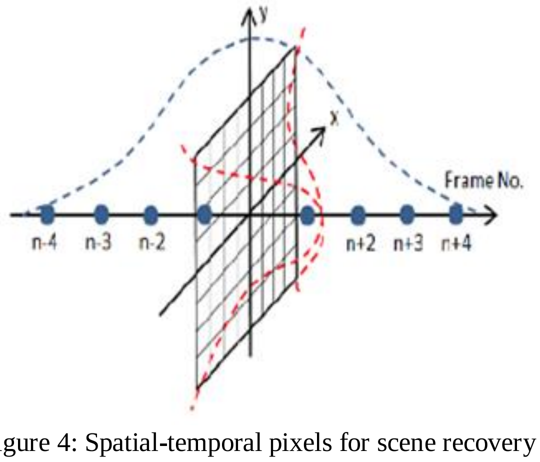

Figure 4: Spatial-temporal pixels for scene recovery

In GMM, each pixel can be represented as mixture of Gaussian distribution [12]. Initially, each Gaussian has given equal weightages. The addition of all the weightages should be 1. Weight parameter shows the time period for which the specific intensity presents in scene. Next step is to classify the Gaussian distribution of pixel into background and foreground [12] [13] [14]. At any time t, what is known about a particular pixel { x_0, y_0 } is its history.

$$\left\{X_1,.....,X_t\right\} = \left\{I\left(x_0, y_0, i\right) : 1 \le i \le t\right\} \tag{4}$$

where I is image sequence and i is frame number. When background is static then the value of pixel is same with respect to time. Each pixel from scene is modeled by number of Gaussian (K) distributions [11]-[12]. The K value is estimated by using two parameter available memory and computational power. Value of K indicates how many background need to be modelled. Pixel is part of multiple surface in real time scene. GMM used to identify changes when foreground object comes into scene. Multiple Gaussian distributions are used to identify illumination changes. Initially, background is modelled using first Gaussian. If there is sudden change in image then second Gaussian distribution track those changes and first Gaussian distribution preserve the background. Value of K decides the number of such backgrounds to be modelled. But increase in value of K, increases computational complexity. So need to set value of K accordingly. Experimentally, its value lies in between 3 to 5. Probability distribution of pixel is given by the equation as shown below,

$$P(X_t) = \sum_{i=1}^{K} w_{i,t} * n(X_t, \mu_{i,t}, \Sigma_{i,t}) \tag{5}$$

where $w_{i,t}$ is weightage given to each Gaussian and n is the normal Gaussian distribution of Kth component which is represented by the equation as shown below:

$$n(X_t, \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} \left|\Sigma\right|^{\frac{1}{2}}} e^{-\frac{1}{2}(X_t - \mu_t)^T \Sigma^{-1} (X_t - \mu_t)} \tag{6}$$

where μ is mean of each Gaussian and Σ is the covariance of each Gaussian. To reduce the time required for algorithm to run, covariance matrix is assumed in the form which is as shown below:

$$\Sigma_{k,t} = \sigma_k^2 I \tag{7}$$

Eq. (6) assumes that Red (R), Green (G), and Blue (B) pixel values are independent and have the same variances. But, this is not true in real time. They have considered this to reduce the number of calculations, which degrades the accuracy of algorithm. Each pixel is modelled using Gaussian distribution. Accordingly, all the Gaussian distributions have updated. Initially, mean covariance and weightages have to define.

Online K-means algorithm is used for pixel classification and updating. GMM is applied on colour image. Each plane has its particular mean and variance. First frame is assumed as mean of each distribution. All Gaussian distribution mean value is first frame pixel values. Initially, high variance and equal weightages are given to each distribution. RGB values of each pixel are accessed and compared with Gaussian distribution using Mahalanobis distance criteria. In this, Gaussian mean and variance is used to find out distance. This distance is compared with matching threshold (value lies in between 0 to 2.5 of standard deviation). Value of matching threshold is selected in way that if any object comes into picture then that particular area variance is increased. We need to find out that increased variance. This value might be change with little effect of accuracy. If match is found in one of the K distribution then just need to update the model but if match is not found in any of the Gaussian distribution then the least probable Gaussian distribution is replaced with the new Gaussian distribution which has current intensity value as mean of that Gaussian and starting initial high variance and low weight.

If match found in one of the distribution then the weights of all the distribution are updated. More weightage is given to matched distribution than other distribution. If one pixel is part of background continuously then its weightage is keep on increasing continuously and variance decreases and vice versa. So, weightage parameter plays a significant role in the classification of Gaussian distribution also value of learning constant controls how quickly model adapts to changing conditions. If its value is set to low then model takes high time to adapt itself and vice versa to adapt quickly. (1/Learning Constant) defines speed at which distribution parameter changes.

The μ and ρ value of unmatched Gaussian distribution remains same. For matched distribution the value of μ and ρ are updated in every iteration. Important point in GMM is, if pixel is considered as a part of background then the previous background model is not altered. It remains in the field until the Kth probable new background color is comes into picture. Therefore, if car came into picture and remains steady for some time then it will adapt quickly to background and new model is created with that background in one of Gaussian distribution. Then, if car starts moving then the previous background which is present in Gaussian distribution comes into the picture with lower weight. That is why it will adapt quickly to the new scene.

As model is updating each new pixel continuously, there is need to classify the Gaussian distribution under background distribution and foreground distribution. Eventually, the incidences are more important in the distributions which have most supporting evidence and least variance. For example, tree is present into picture. As this is part of static background so its intensity value is remains same with respect to time so it generally falls into particular Gaussian. So its weightage is increased high and accordingly the variance is low. Suppose new object comes into picture then particular area the intensity gets changed and variance of that area is increased. However, according to algorithm, if pixel is not part of any of the Gaussian then they create new distribution with low weight and high variance. From above scene, it is clear that for static and persistent object, the weightage is high and corresponding variance is low. In addition, for moving object, the weightage is low and corresponding variance is high.

Gaussians are ordered by using value of ω / ρ such that its value increase when weight is increasing and variance is decreasing and vice versa. After ordering these distributions, it goes from foreground distribution to background distribution. This ordering is arranged in such a way that most background distribution comes first and the other foreground distribution which replace the lower distribution comes later.

It is good to create background frame using Gaussian model directly rather than modeling each pixel and classifying the pixel in background and foreground. So that, the ghost object can be removed by subtracting background frame from current frame. Most important part in background creation is updating background w.r.t. time. If the pixel is updated on continuous basis then noise may forward to each frame which create the false background.

Those pixels which are part of foreground, going to update continuously and remaining pixels which are part of background more than specific amount of time need to update it's Gaussian distribution. So that, it can adapt to any illumination changes properly. That is also taken care by our algorithm.

4.3. Pixel Inpainting

Our algorithm uses temporal pixels for scene recovery. If rain is present in current pixel then its value replaced with nearest temporal pixel intensity which is not part of rain [1]. But using above approach, there might be possibility of scene corruption. If motion is present in scene then pixels are falsely replaced [4]. Rather than replacing the pixel directly, the pixel intensity can be calculated by applying various weightages nearby pixels [4]. The motion pixel can be taken care in this step. Rain pixels within motion object and background needs to treat separately. As motion pixel changes quickly, we have to use neighbour motion pixels without rain for replacement of that corrupted pixel.

Three buffers are used for scene recovery such as input frame buffer $B_i(l,\ w,\ s)$, rain buffer $B_r(l,\ w,\ s)$ and motion buffer $B_m(l,\ w,\ s)$ where I,w,s is length, width and stack respectively. Here, length and width are input video size and stack is how many frames used for rain recovery. The stack value is set to 9. This is for better recovery with less time complexity. Each cell of frame buffer contains one input video frame. When new frame is comes into picture then it is pushed onto top side and all frames moved to bottom side. Last frame is moved out of buffer. For every new frame buffer is updated accordingly and it is same for all the three buffers. In rain buffer $B_r(l,\ w,\ s)$, each frame rain streaks are stored and in motion buffer $B_m(l,\ w,\ s)$, each frame motion object is stored. Both buffers are updated in synchronous with input video buffer $B_i(l,\ w,\ s)$.
Scene recovery algorithm recovers central frame $n = stack + 1/2$ and it uses past $n = 1 : stack - 1/2$ and future frame $stack + 3/2 : stack$ for better recovery of scene by using rain and motion pixels of corresponding frames. Using all above information scene is recovered properly.
Total 88 spatiotemporal pixels are used for scene recovery. 9*9 mask used for pixel intensity of current spatial pixels. The chosen mask size is 9*9 as pixel length is 3 to 9 connected components. Along with these 80 pixels, past and future temporal pixels are used for scene recovery. Suppose rain pixel is $B_i(x, y, n)$ then P contains 8 temporal pixels $B_i(x, y, (n-4) : (n+4))$ and 80 pixels from current frame $B_i((x-4) : (x+4), (y-4) : (y+4), n)$. We have chosen value of stack 9 for better recovery of scene in motion and time required for algorithm should be less.

Pixel values of centre pixel is used to predict the rain corrupted intensity value by weighted sum value. The weights have assigned to each pixel by considering its location from centre pixel. If neighbouring pixel is near to centre pixel, more weightage is given to that pixel and vice versa. Formula of weight is as shown in Eq. (7). Weightage will assign to each pixel at the time of scene recovery.

$$\begin{aligned} w_{x,y,n}(i, j, t) &= \overline{B}_r(x, y, n) \times \exp(-\alpha(v(x, y, n) - v(x, y, t)))^2 \\ &\quad -\beta(v(x, y, n) - v(x, y, t))^2) \end{aligned} \tag{8}$$

Here $\overline{B}_r$ is binary compliment of B_r. It is used to remove pixels in neighbourhood which are part of rain. $v(i, j, t)$ Is current spatiotemporal pixel location in P (88 neighbourhoods). In above equation, first exponential term adjusts the weight along the time axis and second equation adjusts value of spatial pixels. Value α and β are set according to different dynamic property of pixels.

<u>Rain Covered Pixels in Static Scene</u>

If rain covered pixel is part of static scene then set the values of filter coefficient such as α=2/stack and β=0. As value of β is zero, no spatial values are used for static pixel scene recovery. All 8 temporal values with assigned weightage is used for scene recovery. The spatial-temporal for scene recovery as show in Fig 4. According to weights, temporal neighbor pixels

($B_I(x,y,n-1)$ and $B_I(x,y,n+1)$) give more importance. Considering fact of motion and illumination changes, adjacent pixels are more close to real values than farther pixels.

The rain removal filter kernel for static scene is as given by,

$$B_I(x,y,n) = \frac{\sum_{v(i,j,t)\subset v} w_{x,y,n}(i,j,t)\overline{B_M}(i,j,t)B_I(i,j,t)}{\sum_{v(i,j,t)\in v} w_{x,y,n}(i,j,t)\overline{B_M}(i,j,t)} \tag{9}$$

Where $\overline{B_M}$ is binary compliment of motion pixel. It is going to separate the pixels that are part of motion in time axis. Motion buffer tells the information about which temporal pixel is useful for pixel recovery.

After applying above filter, rain streaks are properly removed. The 8 (4 past and 4 future) temporal values are used for scene recovery but sometimes it is possible that all pixels may be part of motion. This is due to false motion segmentation. It can also be possible that motion is very fast. In all above cases, no temporal pixel intensity is used for pixel recovery. The spatial information is used for scene recovery to avoid such situations. In such specific case value of β is set to be $1/4\sqrt{2}$.

Rain Recovered Pixels in Motion Objects

If rain covered pixel is part of dynamic scene then set the values of filter coefficient as α=1 and $\beta = 1/4\sqrt{2}$. As value of β is $\beta = 1/4\sqrt{2}$, more importance is given to spatial values as scene is changing quickly. In temporal pixels, more importance is given to only 2 nearest pixels for scene recovery. Also in spatial case, pixels which are near to centre pixel have given more weightage than farther pixels in spatial axis. This makes filter which is shown in Fig. 1. The rain removal filter kernel for dynamic scene is as shown in equation below:

$$B_I(x,y,n) = \frac{\sum_{v(i,j,t)\subset v} w_{x,y,n}(i,j,t)B_M(i,j,t)B_I(i,j,t)}{\sum_{v(i,j,t)\in v} w_{x,y,n}(i,j,t)B_M(i,j,t)} \tag{10}$$

Pixels Uncovered by Rain

Finally, the pixels, which are not part of rain, are kept as it is.

4. SIMULATION RESULTS

All simulation have done in MATLAB 2015 environment. Proposed algorithm works on intensity plane. We have done experiment on street video in which moving car is present. Frame size of street video is 480*720. This is dynamic scene outdoor video. Our algorithm is successfully able to separate rain pixels and motion pixels. Also, our algorithm is successfully able to recover the scene. We have used different videos for testing our algorithm such as lamp video (480*504), matrix movie video (480*720) and traffic light video (480*720) as shown in Fig 5. Rain takes traffic light colour in traffic light video. The frequency of rain fall is also high. Video zooming effect is present in input video. Our algorithm is able to detect rain pixels and recover the scene without corrupting scene. In lamp light video, our algorithms work well. Results of intermediate step for our algorithm are as shown in Fig 6. Intermediate results show the suppression of false rain detection is possible after applying rain streak width and Prewitt operator. Scene is getting recovered properly due to reduction in false detection.

We have analysed execution time of our algorithm and highlighted results in table 1. We have used PC with 4 GB ram, 1.70 GHz CPU and Intel i3 processor. We have compared our algorithm with Kim et al algorithm. Our algorithm is faster than there algorithm. First 2 rows shows the result of Kim algorithm and 3rd to 7th rows shows the result of our algorithm. Frame size is not same still if we compare using total number of pixels then our algorithm works faster than Kim algorithm.

We have rendered rain in scene and created video with low rain, medium rain and high rain. We have created video into three category using average number of rain pixels affected by rain per frame. Frame size of created video is 400x744 i.e. (297.6X103) number of pixels. If average area of rain pixels is in between (120X103) to (160X103) then we classify that video as low rain video. If average area of rain pixels is in between (160X103) to (190 X103) then we classify video into medium rain video and if average area is in between (190X103) to (230X103) then video is high rain video. Then, we applied our algorithm to rendered video and created rain free video. We calculated three parameter using this video such as Structural Similarity (SS), Peak Signal to Noise Ratio (PSNR) and Mean Square Error (MSE). The equations are as shown below:

Table 1: Average Execution Time (per frame) of Proposed Algorithm

Video Name	**Frame Size**	**Time (Sec)**	**Time(µs per pixel)**
CIF [5]	352 x 288	31.41	309
nHD [5]	640 x 360	41.16	178.6
Street	480 x 720	11.0938	32
Traffic Light	408 x 514	7.4000	35
Low Rain	400 x 744	2.0800	60
Mid Rain	400 x 744	5.2000	17

Table 2 : Performance parameter of created video

Type of Rain	**Performance Parameter**			**Miss Detection (MD)**	**False Detection (FD)**
	SS	**PSNR**	**MSE**		
Low rain	0.9468	36.4907	14.5715	675	657
Medium rain	0.9344	36.2505	15.4227	953	2546
High rain	0.9210	35.8307	17.0820	1610	4568

Table 3: Average Miss and False Detection Rate

(PM: Proposed Method)

Algorithms	**Frame size**	**MD**	**FD**	**Total Error**	**% Error**
Garg and Nayar [1]	240x320	274	1987	2261	2.94
Zhang et al [9]	240x320	247	2236	2483	3.23
Triparthi [3]	240x320	63	1822	1885	2.45
Jie Chen [4]	240x320	686	786	1472	1.91
PM (Low Rain)	400x744	675	657	1332	0.44
PM (Medium Rain)	400x744	953	2546	3499	1.17
PM (High Rain)	400x744	1610	4568	6178	2.07

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x{}^2 + \mu_y{}^2 + c_1)(\sigma_x{}^2 + \sigma_y{}^2 + c_2)} \tag{11}$$

$$PSNR = 10\log_{10}(peakval^2 / MSE) \tag{12}$$

$$MSE = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}[i(j,k) - r(j,k)]^2 \tag{13}$$

Quantitative analysis of our method is carried out. We have rendered rain in black car video (frame size 400 X 744) and rain streak ground truth is calculated by taking difference between video taken before and after rain rendering. Measuring metrics used are rain Miss Detection rate (MD) and rain False Detection rate (FD). We have taken the results of other algorithm from the paper [4]. We can see our method shows less total error in case of large frame rate than other methods. For quantitative analysis, they used blue car with frame size of 240x320. We have used black car with frame size of 400x744. We have calculated the percentage error of each method. As shown in table 3, our method has less percentage error than other method in case of heavy rain. Our algorithm is able to detect rain and recover scene in heavy rain. Our method shows better results than other methods in case of large frame rate.

6. CONCLUSION

We have presented the algorithm, which is dependent on dynamic scene motion segmentation. Object is moving along with rain in dynamic scene. Pixels in motion sensitive area are replaced with corrupted value due to non-consideration of motion occlusion. Our algorithm separated motion and rain pixels, recover scene according to pixel is part of motion or static scene. Both spatial and temporal information is used. If pixel is part of motion then spatial property is used and if pixel is part of background then temporal property is used. Also, blurring at edges is preserved by using edge filter for rain detection. Experimental result shows that our algorithm works better in a highly dynamic scene. This algorithm can be used in various applications such as video surveillances, video editing and navigation. Currently ore algorithm works poor in handling large camera motion. In future, we will work on rain removal from dynamic scene.

7. ACKNOWLEDGMENT

I express our deep sense of gratitude and indebtedness to Dr. Amol R. Madane (TCS) who was very kind to provide us with an opportunity to work under his immense expertise. His prompt inspiration, suggestions with kindness and dynamism enabled us to shape the present work as it shows. We would like to express our sincere gratitude towards our project guide Dr. S. V. Bonde for his constant encouragement and valuable guidance.

REFERENCES

[1] Astha Modak, Samruddhi Paradkar, Shruti Manwatkar, Amol R. Madane, Ashwini M. Deshpande, “Human Head Pose and Eye State Based Driver Distraction Monitoring System”, 3rd Computer Vision and Image Processing (CVIP) 2018, Indian Institute of Information Technology, Design and Manufacturing, Jabalpur (IIITDMJ), India.

[2] A. K. Tripathi, S. Mukhopadhyay, “Video Post Processing: Low latency Spatiotemporal Approach for Detection and Removal of Rain”, IET Image Processing Journal, Vol. 6, no. 2, pp. 181-196, March 2012.

[3] Kshitiz Garg and Shree K. Nayar, “Detection and Removal of Rain From Videos”, IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), vol. 1, pp. 528-535, 19 july 2004.

[4] Kshitiz Garg and Shree K. Nayar, “When Does a Camera See Rain ?”, Tenth IEEE International Conference on Computer Vision, ICCV 2005, 17-21 Oct. 2005

[5] A. K. Tripathi, S. Mukhopadhyay, “Video Post Processing: Low latency Spatiotemporal Approach for Detection and Removal of Rain”, IET Image Processing Journal, Vol. 6, no. 2, pp. 181-196, March 2012.

[6] Jie Chen, Lap-Pui Chau, “A Rain Pixel Recovery Algorithm for Videos with Highly Dynamic Scene”, IEEE Transactions on Image Processing, vol. 23, no. 3, pp. 1097-1104, March 2014.

[7] Jin Hwan Kim, Jae Young Sim, Chang Su Kim, “Video Deraining and Denoising Using Temporal Correlation and Low Rank Matrix Completion”, IEEE Transactions on Image Processing, vol 24, no. 9, September 2015

[8] A. K. Tripathi, S. Mukhopadhyay, “Meteorological Approach for Detection and Removal of Rain From Video”, IET computer vision 2013 Vol 7, no. 1, pp. 36-47, 23 may 2013

[9] Jing Xu, Wei Zhao, Peng Liu, Xianglong Tang, “an improved guidance image based method to remove rain and snow in a single image”, Computer and Information Science, vol. 5, no. 3, May 2012

[10] Kshitiz Garg and S. K. Nayar, “Vision and rain”, International Journal on Computer Vision, vol. 75, no. 1, pp. 3–27, 2007.

[11] X. Zhang, H. Li, Y. Qi, “Rain Removal in Video by Combining Temporal and Chromatic Properties”, in Proc. IEEE International Conference Multimedia and Expo., 2006, pp. 461-464.

[12] Duan Yu Chen, Chien Cheng Chen and Li Wei, ‘”Visual Depth Guided Color Image Rain Streaks Removal Using Sparce Coding”, IEEE Transaction on circuits and system for video technology, vol. 24 , no. 8, august 2014.

[13] C. Stauffer, W.E.L. Grimson, “Adaptive background mixture models for real-time tracking”, Computer Vision and Pattern Recognition IEEE Computer Society Conference , vol. 2, pp. 252-259, 23-25 June 1999

[14] KaewTraKulPong P, Bowden R., “An improved adaptive background mixture model for real-time tracking with shadow detection,” Proceedings 2nd European Workshop on Advanced Video Based Surveillance Systems (AVBS 2001) , Kingston, UK, September 2001.

[15] Zivkovic Z., “Improved adaptive Gaussian mixture model for background subtraction,” Int Conf Pattern Recognition (ICPR 2004), 2004, 2: 28-31.

[16] Zang Q, Klette R., “Evaluation of an adaptive composite gaussian model in video surveillance,” CITR Technical Report 114, Auckland University, August 2002.

[17] White B, Shah M., “Automatically tuning background subtraction parameters using particle swarm optimization,” IEEE Int Conf on Multimedia & Expo (ICME 2007), Beijing, China, 2007; 1826-1829

[18] Grimson Wel, Stauffer C. Romano R. Lee L., “Using adaptive tracking to classify and monitor activities in a site,” 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No.98CB36231). IEEE Comput. Soc. 1998. 1998.

[19] Stauffer C, Grimson W. E. L., “Learning patterns of activity using real-time tracking,” IEEE Transactions on Pattern Analysis & Machine Intelligence, 2000. 22(8): p. 747-57.

[20] Pushkar Gorur, Bharadwaj Amrutur, “Speeded up Gaussian Mixture Model Algorithm for Background Subtraction,” 8th IEEE Int Conf on Advanced Video and Signal-Based Surveillance, 2011.

[21] Thierry Bouwmans, Fida El Baf, Bertrand Vachon, “Background Modeling using Mixture of Gaussians for Foreground Detection - A Survey,” Recent Patents on Computer Science, Bentham Science Publishers, 2008, 1 (3), pp.219-237. <hal-00338206>

[22] L. Li, W. Huang, Q. Tian, “Statestical Modelling of Complex Background for Foreground Object Detection,” IEEE Transaction on Image Processing. 13(11):1459-1472, 2004.