Detection and Reconstruction of Line Scratches in Old Motion Picture

[1] Mukkawar Vinayak V.,[2]Jondhale Kalpana C.
[1]Assistant Professor, [2]Professor,
[1]Department of Information Technology,
[1]Mahatma Gandhi Mission's College of Engineering, Nanded (MH), India

Abstract: It is very hard to detect line scratches in videos. Scratches are not always vertical and may present in multiple frames, so it requires spatial and temporal information to identify true scratches. The problems in scratch detection are structure, texture, noise and false detection. Abrasion of film material to mechanical part of projection mechanism is main reason for scratches. It is required to fill the information to reconstruct the image. Here it explains spatial algorithm, temporal filtering and scratch filling. Spatial algorithm uses the current frame information to detect scratch inside the frame. Temporal filtering results in true scratch detection by filtering false detection. The temporal filtering uses the concept of motion estimation. Scratch filling is done on the basis of interpolation. To get fine results pre-processing is essential. Results of temporal filtering show the detection of line scratches and scratch filling step provides betterment in frame by reconstructing the detected scratches.

I. Introduction

Incredible amounts of old film material present in film documents, the rebuilding of old movies is a matter of interest in this area. The automated tool must be designed for line scratch detection and reconstruction. Typical defects in the images are blotches, dust or dirt, flicker and line scratches. Scratches are caused due to friction of film to mechanical part of projection. The information of the physical inception of line scratches might be checked at [1]. Scratches in images may be represented by dark or bright lines. Possibly these scratches are considered as vertical and straight but it is not the case always. These shortcomings also represent the normal characteristic of temporal behavior, saying that line scratches stays in the same or a similar space position for few image sequences. The combination of spatial detection and temporal filtering can also be found in [16] and [17].

However, these attributes are truly factor, making line scratch identification and rebuilding it is an especially troublesome test. This paper presents three algorithms. 1) Presents the pixel-wise Line Scratch Detection (LSD), which depends on contrario methodology [2] to make it robust in nature to presence of noise. 2) Algorithm based on temporal filtering to remove false detection. This algorithm uses the motion coherence which is based on the consideration that scratch moves in same manner as in the next scene or frame are false scratches. Since such type of scratches are parts of video. 3) Algorithm for filling the scratches with true information to perform reconstruction of frames. The use of interpolation makes this tedious job of reconstruction easy.

Figure 1 shows the different steps of proposed work. The algorithm is applicable on still image so the first step is to convert video to frame. After converting video to frame the next step is to perform preprocessing. The preprocessing is required to remove noise if any in the frame and results in better detection of line scratches. To detect the scratch in the frame, firstly the pixel wise line scratch detection criteria is used, this criteria produces robust result against noise and texture. The use of contrario technique [2] makes the algorithm robust. The use of this method reduces the false detection. Still there are some scratches which are not true and are part of images; to remove such scratches temporal filtering technique is used.

The use of temporal filtering removes the false detection left over in pixel wise detection. The concept of motion coherence is used to removes false scratches. The scratch filling step includes the interpolation technique to find out missed information in images, ultimately results in scratch filling.

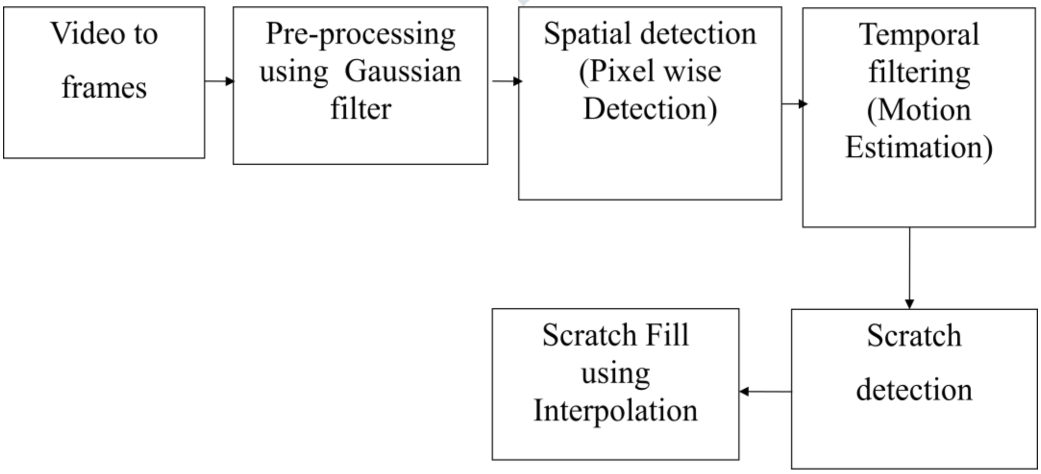


Fig. 1 Flow of proposed algorithm

II. PIXEL WISE LINE SCRATCH DETECTION (*LSD*)

Algorithm for LSD.

Step1: Start
Step2: Input the video
Step3: Video to frames conversion
Step4: Preprocessing using Gaussian filter

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Step5: Spatial detection using pixel wise detection criteria.

$$I_{bin}(x, y) = \begin{cases} 1 \; if \; S_1(x, y) \; and \; S_2(x, y) \\ 0 \; otherwise \end{cases}$$

Step6: Temporal filtering using motion coherence.

$$e(p) = c_1 + a_1 x_{p+} a_2 y_p,$$
$$f(p) = c_2 + a_3 x_{p+} a_4 y_p$$

Step7: Scratch detection
Step9: Stop

In very first step video is converted to frames and pixel wise scratch detection is carried out per frame to identify the candidate pixels. Before performing the pixel wise scratch detection step, frames are undergone through pre-processing step. Pre-processing is done with Gaussian filter of standard deviation one. A pixel wise detection criterion is useful for the detection of interested pixel which acts as part of potential scratch. To detect the potential scratch in image various criteria's are present, one is discussed in [4] which uses the concept of one dimensional extrema detector to detect the line scratch candidates. One more approach can be seen in [5] relies on the neural network and morphological filters to detect the potential scratch points. Author in [5] used the texture classifier based on neural network that classifies input image in two different region (scratch region and non-scratch region) and shape filter based on morphology to confirm the classified scratch region. Here in this criterion it uses the method proposed by the Kokaram and found in [3]. This uses the threshold and is calculated by the difference of original grey scale image and its horizontal median filtered image. As it uses horizontally median filtered version of image, it detects outliers based on neighbor present horizontally. And then it uses the vertical sub sampling to accounts the vertical neighbor. Instead of all these calculation the proposed algorithm for candidate pixel detection uses the Gaussian filter to reduce the noise in the image that avoids the use of sub-sampling.

Figure 2 shows the profile for scratch point detection, which is used to calculate whether the pixel is candidate or not. Equation 1 is used to find out the binary detection map. This binary image indicates the detected scratch point.

$$I_{bin}(x, y) = \begin{cases} 1 \; if \; S_1(x, y) \; and \; S_2(x, y) \\ 0 \; otherwise \end{cases} \quad (1)$$

where $S_1(x, y)$ and $S_2(x, y)$are calculated by using equation 2.

$$\begin{aligned} S_1(x, y)&: \left|I_g(x, y) - I_{med}(x, y)\right| \geq T_{med}, \\ S_2(x, y)&: \left|I_{left}(x, y) - I_{right}(x, y)\right| \geq T_{avg}, \end{aligned} \quad (2)$$

Where $I_g(x, y)$ is Gaussian filtered image, $I_{med}(x, y)$ is median value of (x, y) based on horizontal neighborhood, $I_{left}(x, y)$ is left horizontal average and $I_{right}(x, y)$ is right horizontal average. The $S_1(x, y)$ and $S_2(x, y)$ are nothing but the two Boolean criteria used for generation of binary image $I_{bin}(x, y)$ as specified by equation 2, T_{med} and T_{avg} are thresholds and are having value 3 grey level and 20 grey level respectively. The width of the median filter is set as provided by Kokaram in [3] and is set to 5. As shown in figure 2 the left and right average are calculated over three pixels on both side of central 5 pixels.

The criteria used here prone to false detection and it may also not able to find some pixels that are actually the part of scratch. This step is responsible to find candidate pixel only, system is interested in line scratch, and hence grouping of these detected pixels is done to form the line. The Hough transform is used to detect the line in binary images [11], but it's having one problem regarding the threshold value. But this problem can be removed by using the cantrario line segment detection described in [2]. In contrario line segment detection the line can be detected by grouping the pixels on the basis of its gradient, the pixels whose gradient is perpendicular to the direction can be grouped together to form a line. Let's assume that the line made up of n pixels having the grouping of y_n pixels on the basis of gradient i.e. y_i is 1 if the pixel having perpendicular gradient otherwise it is set to 0. As we are dealing with scratches that are not always straight so it is recommended to give some threshold for angle л radians. So one can represent the segment using$Seg = x_1 + \cdots + x_n$, if the value of Seg is greater results into the good segment representation. Author in [3] uses the probability distribution of Seg as background model for line detection. It uses binomial distribution for the same indicating the segment having length len with x_0 pixels aligned is meaningful if the value of $B(p; x_0, len)$ is less. The value of $B(p; x_0, len)$ can be calculated using equation 3. On the basis of equation 3 it is easy to calculate number of false detection and can be given by equation 4. Where Num_{test} is number of segment to be tested. In all the videos used for working it sets the limit of scratch for its detection is one tenth of the height of the image [6].

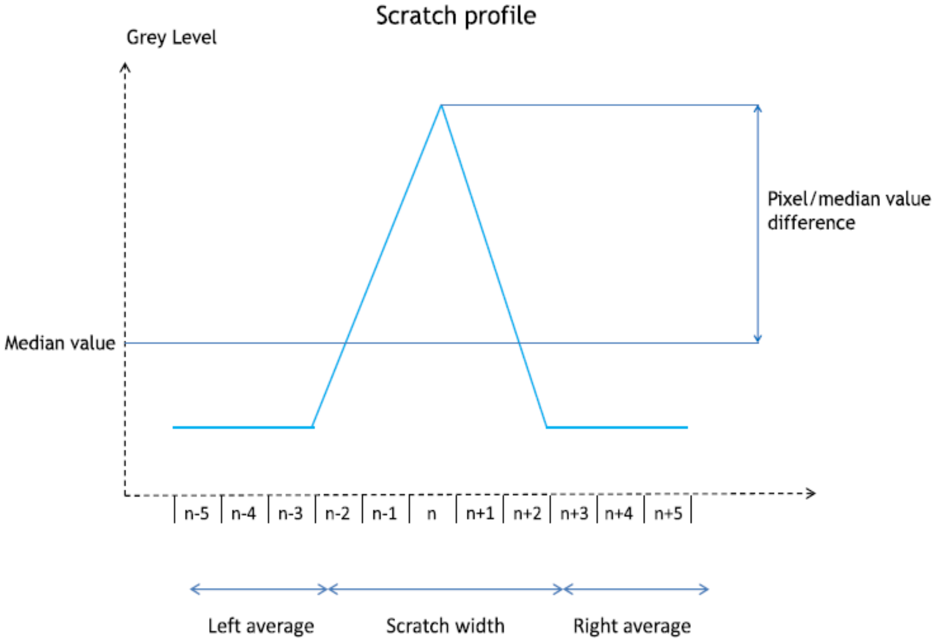


Fig. 2 Profiles for scratch point detection [6]

$$\Pr(s_{len} - x_0) = \sum_{x=x_0}^{len} \binom{len}{x} p^x (1-p)^{len-x} =: B(p; x_0, len) \quad (3)$$

$$NFA(len, x_0) = Num_{test} B(p; x_0, len) \quad (4)$$

The previous step detects the scratches in the images; the algorithm is robust in nature, but this algorithm does not deal with vertical object of particular frame which are thin in nature and detected as scratch. In per frame calculation the thin vertical object may be detected as line scratch but it is the part of image sequence and can concluded after seeing it in neighboring frame. Such kind of false detection can be handled by estimating the motion and the criteria used is motion coherence. Motion coherence is based on the trajectories. It is very difficult to calculate the trajectories of genuine scratches so in this process it calculates the trajectories of non genuine scratches. To find out trajectories of false detection one can use binary map detected in pixel-wise scratch detection and inverse it. Next step is to find out global motion of video to perform realignment. This motion can be calculated by using motion vector given in [7]. It suggests that at a pixel position $p = (x_p, y_p)$ the motion vector $(e(p), f(p))$ is given in equation 5.

$$e(p) = c_1 + a_1 x_{p+} a_2 y_p,$$
$$f(p) = c_2 + a_3 x_{p+} a_4 y_p \quad (5)$$

Where $c_1$and $c_2$are constant motion component and $a_1 \dots a_4$ are spatially varying components of motion. The motion estimation is calculated over all consecutive pair of images in video. To calculate the mapping between the positions of pixel in one image to position of pixel in next image, the relationship given in equation 6 is used. Once the line has been realigned using relation, the next step is to detect the line segment in its inverse. This ultimately results in finding the trajectories of false detection. A slope of 5 degree change is considerable or allowed for the detection of trajectories. One more evaluation criteria is used for filtering which indicates that entire trajectory set is rejected if it's starting and finishing image indices are within the temporal distance of 5 pixels and is implemented on 4 frames.

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_1 + 1 & a_2 & c_1 \\ a_3 & a_4 + 1 & c_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (6)$$

III. LINE SCRATCH RECONSTRUCTION USING INTERPOLATION

To fill the detected scratch large amount of literature were present and can be found in [12], [13], [14] and [15]. One can use simple filter which generate the smother effect and able to detect the value for missed pixel responsible for generation of scratch.

The example of such filter is median filter; but the disadvantage of this type of filtering is the resulting image gets blurred. Again it is difficult to select the size of filter because the type of video and its resolution. In order to fill the missing information in image [10] the use of interpolation technique is beneficial, since it is easy to implement. Interpolation is defined as technique used to estimate the unknown value between well-known discrete points. There are various techniques of interpolation, out of which this paper implements the bilinear interpolation. Bilinear interpolation takes into account 4 way connectivity to find out the 4 neighbors. The value of unknown pixel is calculated by using the value of four corner pixel. This will gives you fine result compared to Nearest Neighbor Interpolation. Here it proposes the algorithm to fill the line scratch with relevant information that uses region filling with interpolation and morphological dilation operation.

Algorithm for Reconstruction of scratch using interpolation.

Step1: Give the scratch detected image as an input image with the mask
Step2: Find mask perimeter i.e. boundary pixels
Step3: Create a look up table for dilation.
Step4: perform mask dilation using look up table using

$$A \oplus B = \bigcup_{b \in B} A_b$$

Step5: check out the neighboring pixels
Step6: perform the region fill using Bilinear Interpolation

$$CP \approx \frac{(M_2 - M)(N_2 - N)}{(M_2 - M_1)(N_2 - N_1)} Q_{11} + \frac{(M - M_1)(N_2 - N)}{(M_2 - M_1)(N_2 - N_1)} Q_{21} + \frac{(M_2 - M)(N - N_1)}{(M_2 - M_1)(N_2 - N_1)} Q_{12} + \frac{(M - M_1)(N - N_1)}{(M_2 - M_1)(N_2 - N_1)} Q_{22}$$

The first step to this algorithm is to provide two images, one is original image which consisting of line scratch and another is mask. The mask must be binary, so here it uses the binary image which was created at a time of pixel wise detection method. This image actually consists of possible pixel representing line scratch in an image. One can generate the mask on the basis of final output of previous step which give accurate scratch locations in an image. But it is time consuming job to create mask after finding the actual scratch in image since the value of recall is near to 80 percent showing only few of false pixel detection. If we consider these entire pixels in calculation of mask it does not affect the result and there is no wastage of time for calculation of mask.

The next step in proposed algorithm for scratch filling is but finding the perimeter of the mask. This step has its own advantages, which avoids the index out of bound error for outer boundary pixel present in mask. In simple way padding will be done in this step. Create a look up table for dilation purpose. The dilation will be done by using the equation 7.

$$A \oplus B = \bigcup_{b \in B} A_b \qquad (7)$$

The above equation represents dilation of A by B, where A_b is translation of A by b. The look up table is responsible for returning the value 1 if the current pixel or at least one of its four neighbors is 1. Filter the input mask to generate a mask dilated by one pixel in North, South, West and East direction. Here it uses the Bilinear Interpolation for filling the region. The equation 4.2 gives the idea for calculating the Bilinear Interpolation. Before filling the region neighboring pixels are calculated. Special attention must be given to this step because interior pixels have 4 neighbors, border pixels have 3 neighbors and corner pixels have only two neighbors. After calculating the neighboring pixels next step is to perform region fill done by using interpolation. Consider the figure 4.2 which demonstrate the calculation of interpolated pixel using Bilinear Interpolation technique.

	M_1	M	M_2
N_1	Q_{11}		Q_{21}
N		CP	
N_2	Q_{12}		Q_{22}

Fig. 3 Illustration of Bilinear Interpolation [8]

In figure 3 CP is candidate pixel and Q_{11}, Q_{21}, Q_{12} and Q_{22} are neighboring pixels considered for calculation. M, M_1 and M_2 represents the x location of pixel and N, N_1 and N_2 represents the y location of pixel. By using all these values the Bilinear Interpolation can be calculated as:

$$CP \approx \frac{(M_2 - M)(N_2 - N)}{(M_2 - M_1)(N_2 - N_1)} Q_{11} + \frac{(M - M_1)(N_2 - N)}{(M_2 - M_1)(N_2 - N_1)} Q_{21} + \frac{(M_2 - M)(N - N_1)}{(M_2 - M_1)(N_2 - N_1)} Q_{12} + \frac{(M - M_1)(N - N_1)}{(M_2 - M_1)(N_2 - N_1)} Q_{22} \qquad (8)$$

IV. EXPERIMENTAL RESULTS

This chapter consists of experimental results of the algorithm. Three criteria are used for evaluation.

1. Recall
2. Precision

3. FinalScore

Definition of Recall can be given as the quantity of genuine discovery isolated by aggregate number of genuine scratches present in a picture. Fundamentally recall figures out what level of line scratch identified. Precision is characterized as number of genuine location separated by aggregate number of recognition. Fundamentally precision figures out what level of our location were right recognitions. The FinalScore is an impression of the two criteria and can be characterized as:

$$FinalScore = 2\left(\frac{Rec * Prec}{Rec + Prec}\right) \quad (9)$$

Where Rec: - Recall and Prec: - Precision

4.1 LSD Experimental results

The six film sequences are considered for experiments which are of different characteristics. The three videos namely “Knight”, “Sitdown” and “Star” are commonly used and are present in Kokaram’s book [10]. For the assessment of the spatial algorithm these three videos are more helpful than the other three, since the recordings are little or contains no false caution which might be rejected by utilizing the motion aspects, the temporal filtering having less work in these cases. The other three videos are having larger length than the above three and illustrate the improvement on precision obtained by using temporal filtering. Following figures shows application of both the algorithm on different videos. Before applying the algorithms some pre-processing were required to get fine results. Gaussian filter were applied on images which smooth an images and reduces noise if any present in image, its best result can be seen on frames of Sitdown and Star image sequences where the noise is present.

Figure 4 shows the application of Gaussian filter on the frame of videos that are used for work. In figure 4 images a and c are the original frame from the videos Afgrunden_1 and Knight while images b and d shows the result of Gaussian filter. Gaussian filter of size 3 x 3 were used with standard deviation of value 1.

A pixel-wise detection criterion is used to detect potential scratch point in an image. Figure 5 shows the result of pixel-wise detection criteria applied on Gaussian filtered images. In figure 5 a and c are Gaussian filtered images while images b and d shows binary detected images. The potential points are shown in white color and further these points are grouped together to determine the significant scratch segments. Significant scratch segment detection can be done by various ways; here Hough transform detects line scratches in an image. Figure 6 shows line detected, the line marked as red are the part of frame and are not a true scratch. To detect true scratch in an image compare the image with neighboring images and that can be done by temporal filtering using motion coherence. The result of temporal filtering is shown in figure 7. The marking in figures are done manually by using coordinates of line found by applying the algorithm.

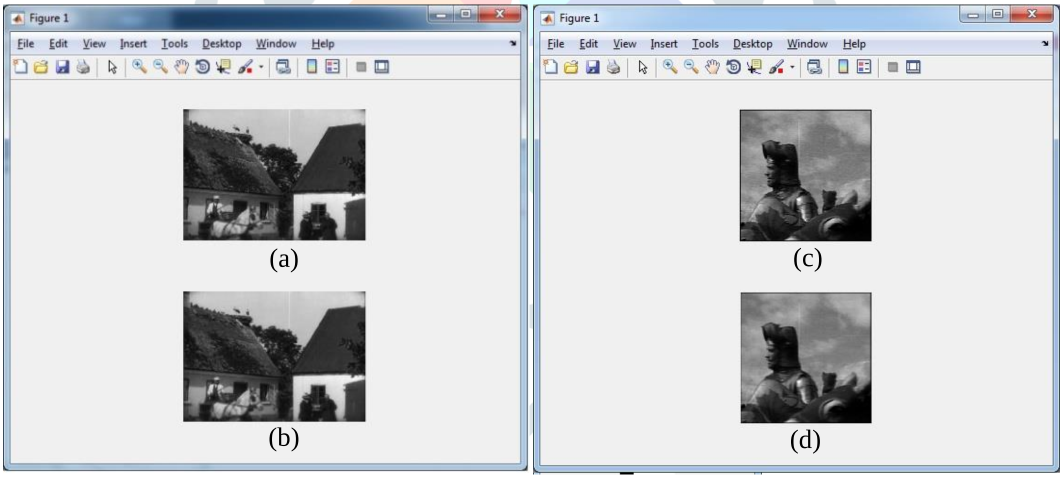


Fig. 4: (a) and (c) original frame in videos Afgrunden_1 and Knight. (b) and (d) shows Gaussian filtered image of (a) and (c).

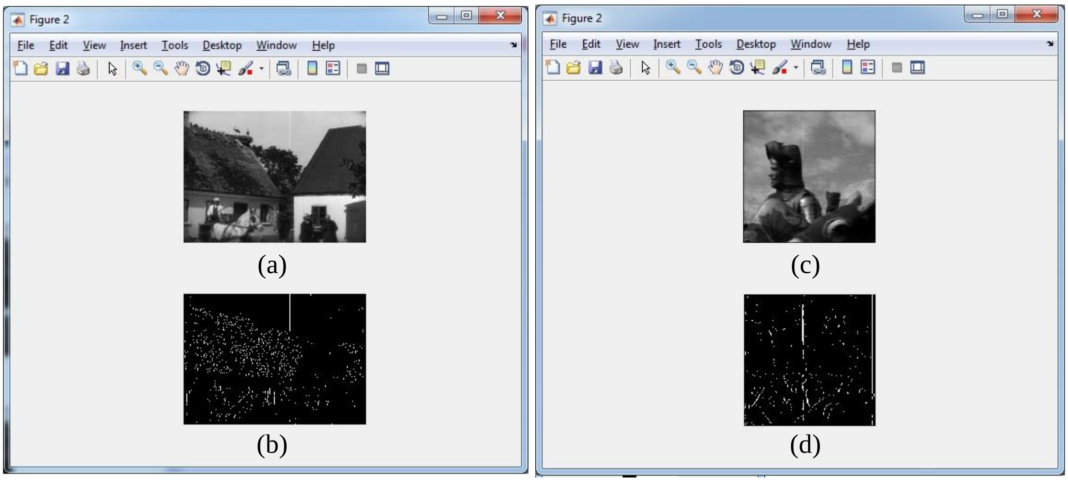


Fig. 5: (b) and (d) shows pixel wise detection on the images (a) and (c).

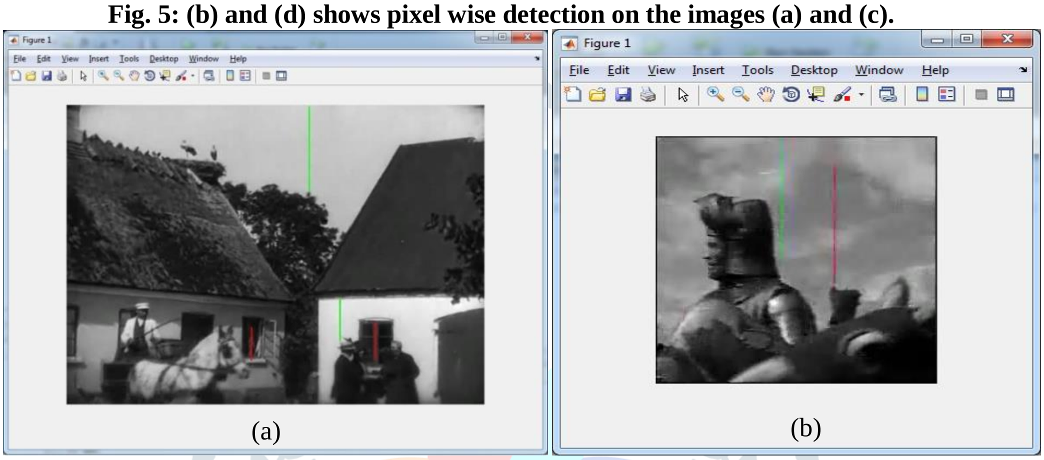


Fig. 6: (a) and (b) shows line scratch detection using spatial algorithm.

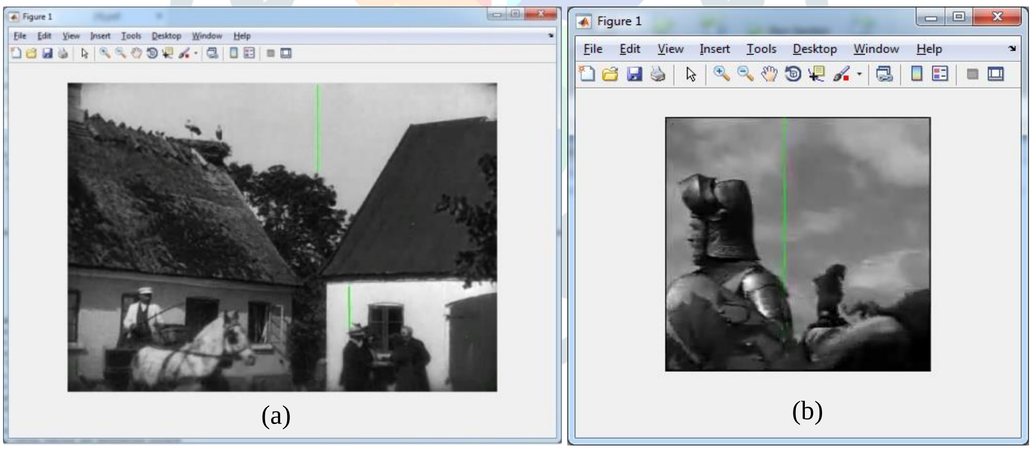


Fig. 7: (a) and (b) shows line scratch detection using temporal algorithm

The table I shows the results for calculation of Recall, Precision and FinalScore after application of proposed algorithms. In most of the cases it is found that recall is better except the video Knight, indicating accurate detection of interested pixel responsible for representing scratch in the given images. While precision calculated are also better and denotes the true detection rate. Only for the video Afgrunden_gate the precision value is less because of number of vertical objects present as a part of scene, which results in less FinalScore. The graphical representation of FinalScore is given in figure 8.

Table I: Experimental results showing Recall, Precision and FinalScore

Sr. No.	Name	Recall	Precision	FinalScore
1	Afgrunden_1	85.92	62.28	72.21
2	Afgrunden_2	87.10	45.58	59.84
3	Afgrunden_gate	80.04	11.19	19.63
4	Knight	66.81	81.48	73.41
5	Sitdown	71.44	86.76	78.35
6	Star	84.49	63.38	72.48

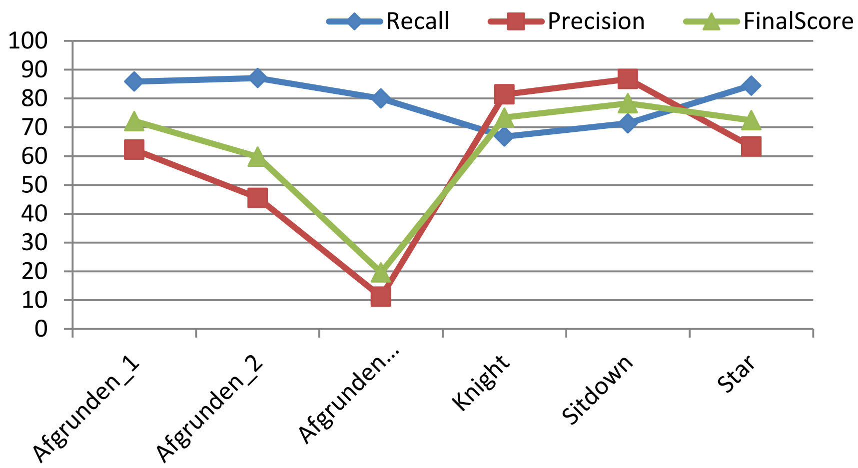


Fig.8: Graphical representation of Recall, Precision and FinalScore.

The graph in figure 8 shows the values for Recall, Precision and FinalScore. To calculate the FinalScore equation 9 is used which is simply the harmonic mean of the two values calculated previously. This score gives the clear idea about the performance than the individual use of recall or precision. Higher the value of FinalScore better is the accuracy of algorithm. It found that if the value of Recall and value of Precision are higher results in higher value of FinalScore else the FinalScore become less, demonstrating that FinalScore depends upon both the value, it does not give the significance to single value i.e. neither the Recall nor the Precision. It is shown in the table IV that FinalScore for afgrunden_gate video is too small, the reason behind this is the less value of precision. The result of precision is less due to the number of false detection.

4.2 Results of Reconstruction

Figure 9 shows the experimental results of scratch filling using the concept of interpolation. Bilinear interpolation is used to fill the scratch, which considers the neighboring four pixels to calculate the missing pixel value. The special precaution is taken at the time of calculation of neighboring pixels for corner pixel and boundary pixel. This results into avoidance of array index out of bound exception at the time of execution of algorithm. To perform the scratch filling it takes the two input images, one is the image consists of line scratch and other is mask (logical image) consist of the information for where to apply the algorithm for filling purpose. It provides the single output image in which the missing information of image is replaced by using interpolation technique. In figure 9 (a) is demonstrating the application of algorithm on one of the frame from the Afgrunden_1 video, likewise figure 9 (b) shows the application of proposed algorithm for scratch filling on the videos Knight. The proposed algorithm shows the improvement in the quality of frame.

Filling of information loss due to the scratch can be done by using the proposed algorithm. In order to show the enhancement in the image the Peak Signal to Noise Ratio is used. The PSNR is nothing but the ratio of maximum possible power of signal to power of noise distorting it. It is expressed using logarithmic decibel and calculated easily by using the Mean Squared Error. The formula used to calculate the Mean Squared Error and PSNR is given in equation 10 and 11 respectively.

$$MSE = \frac{1}{mn}\sum_{0}^{m-1}\sum_{0}^{n-1}\|f(i,j) - g(i,j)\|^2 \tag{10}$$

$$PSNR = 10log_{10}\left(\frac{MAX_f}{\sqrt{MSE}}\right) \tag{11}$$

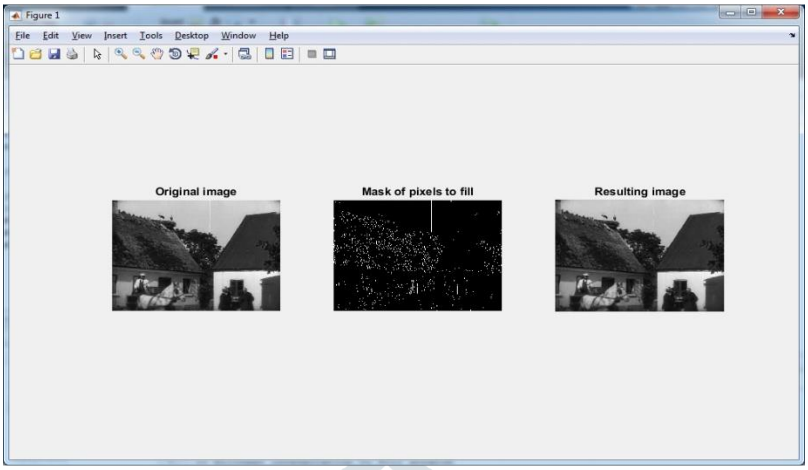
Where m and n are the numbers of rows and column in image and i and j are index of m and n respectively. $f(i,j)$ represents the original image and $g(i,j)$ represents the reconstructed image, and with MAX_f is maximum signal value that present in original image. The PSNR is the quality metrics used in image processing. The table II represents the PSNR values after the reconstruction of images using proposed algorithm.

Table II: PSNR values after comparing original image with reconstructed image

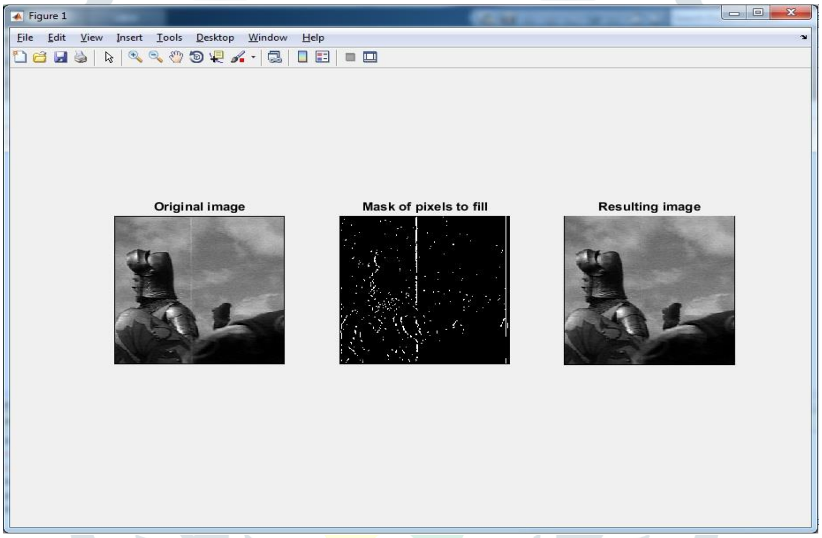
Sr. No	Frame from the video	PSNR
1	Afgrunden_1	41.9023
2	Afgrunden_2	38.6516
3	Afgrunden_gate	36.8617
4	Knight	39.7901
5	Sitdown	39.5430
6	Star	30.3161

It is known that PSNR is commonly used to measure the quality of reconstruction. In general the PSNR in the image are in range of 30 to 50 dB. Higher value of PSNR indicates better result of reconstruction. The table II shows the PSNR for one frame from each video. These values are calculated by using standard Matlab method. The range for calculated values is in between 30 to

50 indicating the better reconstruction of images. Here it is observed that the better reconstruction is done for the frame from Afgrunden_1 video having PSNR 41.9023 and least is for the frame from video Star i.e. 30.3161.



(a)



(b)

Fig. 9 (a) and (b) shows the result of application of proposed algorithm for filling the line scratch.

V. CONCLUSION AND FUTURE WORK

Three algorithms are implemented; the spatial technique is used in first algorithm which uses a contrario checking step to determine the true scratches, second algorithm is temporal algorithm which uses motion coherence to eliminate false detection and third algorithm is used to fill the scratches found by using spatial algorithm and temporal algorithm.

The approaches used here are robust in nature and are responsible for elimination of the false detection results in true scratch detection. The robustness of line scratch detection is shown by recall and precision. The accuracy can be calculated by FinalScore which again depends on Recall and Precision. The FinalScore is greater than 70 percent except the two videos, this indicate better accuracy as well as refinement to get better result on other two videos. Algorithm used for scratch filling measures accuracy in term of PSNR. By observing the PSNR it is clear that quality of images is increased.

The current work has several aspects which could be developed further. The pixel-wise detection criteria can be further used to identify the different shape rather than line only. Enhancement in different algorithm can be done to reduce the time required for execution. The motion model used here estimate only one affine motion; one can use more complex motion estimation techniques to find out several motions. For the restoration of scratches the simple Bilinear Interpolation is used. No dough that algorithm used here for reconstruction produces the fine result, in future other techniques and can be used, that can provide better result for reconstruction.

VI. ACKNOWLEDGMENT

Firstly, I would like to thank my guide Dr. Ms. K. C. Jondhale for introducing me the area of Image Processing. I would like to thank Dr. Mrs. G. S. Lathkar, Director of MGM's College of Engineering, Nanded for her moral boosting and encouragement. I would again like to thank Dr. Mrs. A. M. Rajurkar, Head of Computer Science & Engineering Department for the college facilities she provided. I would also like to thanks to all my department staff for helping me whenever I needed it. I would like to thank my Parents for their love and support.

REFERENCES

[1] Physical Damage, National Film and Sound Archive [Online] available, www.nfsa.gov.au/preservation/handbook/damage-films/physicaldamage/

[2] A. Desolneux, L. Moisan, and J. M. Morel, “Meaningful alignments,” *International Journal on Computer Vision*, vol. 40, pp. 7–23, Oct. 2000.

[3] A. Kokaram, “Detection and removal of line scratches in degraded motion picture sequences,” *Signal Processing*, vol. 1, no. 9, pp. 5–8, Sep. 1996.

[4] L. Joyeux, S. Boukir, and B. Besserer, “Tracking and MAP reconstruction of line scratches in degraded motion pictures,” *Machine Vision Application*, vol. 13, no. 3, pp. 119–128, 2002.

[5] K.-T. Kim and E. Y. Kim, “Film line scratch detection using neural network and morphological filter,” in *proceeding of IEEE Conference on Cybern Intelligence System*, pp. 1007–1011, Sep. 2008.

[6] Alasdair Newson, Andres Almansa, Yann Gousseau, and Patrick Perez “Robust automatic line scratch detection in films.” *IEEE Transaction on Image Processing*, volume 23, number 3, March 2014.

[7] J. M. Odobez and P. Bouthemy, “Robust multiresolution estimation of parametric motion models,” in *Journal of Visual Communication and Image Representation*, vol. 6, no. 4, pp. 348–365, 1995.

[8] https://www.ajdesigner.com/phpinterpolation/bilinear_interpolation_equation.php.

[9] A. Kokaram, F. Pitie, D. Corrigan, D. Vitulano, V. Bruni, and A. Crawford. *Advances in Automated Restoration of Archived Material*. CRC Press, 2011.

[10] M. Bertalmio, G. Sapiro, V. Caselles and C. Ballester, “Image Inpainting”, in *proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, July 2000.

[11] Duda, R. O. and P. E. Hart, "Use of the Hough Transformation to Detect Lines and Curves in Pictures," *Communication ACM, Vol. 15*, pp. 11–15, January, 1972.

[12] V. Bruni, D. Vitulano, and A. Kokaram, “Line scratches detection and restoration via light diffraction,” in *proceeding of 3rd International Symposium on Image Signal Processing Analysis*, vol. 1, pp. 5–10, 2003.

[13] K. Chishima and K. Arakawa, “A method of scratch removal from old movie film using variant window by Hough transform,” in *proceeding of 9th International Symposium on Communication Information Technology*, pp. 1559–1563, 2009.

[14] L. Joyeux, S. Boukir, and B. Besserer, “Film line scratch removal using Kalman filtering and Bayesian restoration,” in *proceeding of 15th IEEE Workshop on Application of Computer Vision*, pp. 8–13, Dec. 2000.

[15] Z. Qingyue and D. Youdong, "Scratch Line Detection and Restoration Based on Canny Operator," *Asia-Pacific Conference on Information Processing*, Shenzhen, pp. 148-151, 2009.

[16] R. Kumar, S. Gupta and K. S. Venkatesh, "Line scratch detection using Spatio-Temporal regularity flow vectors," *19th International Conference on Digital Signal Processing*, Hong Kong, pp. 36-41, 2014.

[17] R. Kumar, S. Gupta and K. S. Venkatesh, "Detection of line scratch using energy regularity of spatio-temporal video cube," *International Conference on Image Processing Theory, Tools and Applications*, 2015.