A Robust Faint Line Detection and Enhancement Algorithm for Mural Images

Mrinmoy Ghorai and Bhabatosh Chanda
Indian Statistical Institute

Abstract—**Mural images are noisy and consist of faint and broken lines. Here we propose a novel technique for straight and curve line detection and also an enhancement algorithm for deteriorated mural images. First we compute some statistics on gray image using oriented templates. The outcome of the process are taken as a strength of the line at each pixel. As a result some unwanted lines are also detected in the texture region. Based on Gestalt law of continuity we propose an anisotropic refinement to strengthen the true lines and to suppress the unwanted ones. A modified bilateral filter is employed to remove the noises. Experimental result shows that the approach is robust to restore the lines in the mural images.**

Index Terms—**line detection, mural image, correlation, anisotropic refinement, bilateral filter.**

I. Introduction

The motivation of this paper is to digitally restore heritage art. Mural is one type of artwork painted or applied on a wall, ceiling or any other large surface. These type of artworks mainly contain the history of ancient civilization. These murals are deteriorated due to prolonged environmental effect and also the quality of material used to make it. As a result various type of defects appear in the murals like cracks, faint lines, broken stroke, color distortion. The preservation and restoration of these mural is necessary for our civilization to propagate the history. Digital restoration is a demanding and challenging task to virtually restore the murals to their old glory. Several restoration techniques [1], [2], [3], [4], [5], [6] have been proposed to digitally enhance and restore the mural images. Bilateral filter is used to reduce the noise in gray and color images [7].

Here our aim is to detect clear as well as faint lines in the mural image (see figure [1]) so that after image enhancement and restoration the structure of the image remain unchanged and faint lines are enhanced. A novel edge detection technique is proposed in [8]. The visual perception affects to decrease the saliency of a contour in presence of surrounding texture [10]. Grigorescu et al. [9] proposed contour and boundary detection technique by anisotropic suppression of texture edges. But here our concern is about the lines instead of edges or contours and the existing techniques are not sufficient to detect the lines. For this, we have proposed a novel method based on correlation, convolution and anisotropic refinement to extract the lines in the image. We have used a modified bilateral filter to reduce the noise in the image. The basic steps of our algorithm is as follows.

Fig. 1. Mural image of Rangamandapam.

- Initial line detection using local correlation between the gray image and the proposed oriented templates. Non-maxima suppression for sharpening the resultant image.
- True Line detection using anisotropic refinement and line tracking based on Gestalt law of continuity.
- Noise reduction using modified bilateral filter.

II. Line Detection

We first find initial line strength at each pixel as a correlation between a patch centering the pixel and a set of templates of different orientations and of size same as the patch. We also apply convolution operation to reduce the line strength of some unwanted line pixels. Our templates can detect line segments in any orientation and have the flexibility for thickness of the lines. The steps involved in this line extraction are (i) computation of line strength at pixels by correlation and convolution, (ii) non-maxima suppression for sharpening, (iii) anisotropic refinement based on Gestalt law of continuity to enhance true lines, and (iv) finally line tracking for final line detection.

Maximum correlation coefficient among different orientations suggests the strength of the line pixel. These coefficients are taken as a magnitude of the lines, and we get an image containing all the clear and faint lines (see figure [4]). We find the weighted average (convolution) of the patch with the template for which the correlation coefficient is maximum. Then we multiply the correlation and convolution at each pixel to get the modified line image. In this method, the larger templates may connect the broken lines whereas the smaller templates usually extract the curved lines better but cannot link the broken lines. Therefore, the size of the template is decided based on the requirement.

A. Correlation-convolution Computation

First we generate template in different orientation. For zero orientation and size $r = 7$, we describe the template X_0 as

[1]This work is partially supported by Department of Science and Technology, Government of India (NRDMS/11/1586/09/Phase-I/Project No. 9).

follows and others are similar as well.

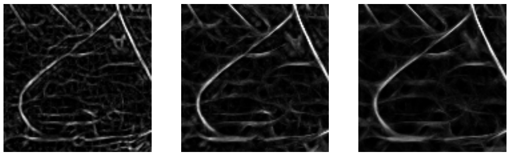

Fig. 2. Some cropped blocks from the convolution images for r=7, 11 and 15 respectively.

$$X_0 = \begin{bmatrix} 1 & 0 & 0 & -2 & 0 & 0 & 1 \\ 1 & 0 & 0 & -2 & 0 & 0 & 1 \\ 1 & 0 & 0 & -2 & 0 & 0 & 1 \\ 1 & 0 & 0 & -2 & 0 & 0 & 1 \\ 1 & 0 & 0 & -2 & 0 & 0 & 1 \\ 1 & 0 & 0 & -2 & 0 & 0 & 1 \\ 1 & 0 & 0 & -2 & 0 & 0 & 1 \end{bmatrix}$$

The values in the template are adjusted in such a way that their sum would be zero. The zero columns are taken for the flexibility of the thickness of the lines. Let n be the number of different templates for n number of different orientation. Now we apply correlation and convolution operators with these n templates for line detection.

For each pixel (x, y) in the original gray image I, we find the correlation coefficient as

$$\rho(x, y, \theta) = \frac{E[(X_\theta - \mu_{X_\theta})(Y - \mu_Y)]}{\sigma_{X_\theta}\sigma_Y}, \tag{1}$$

where E is the expectation, X_θ is the template oriented at angle θ and Y is the image patch at the pixel (x, y). μ_{X_θ} and μ_Y are the mean and σ_{X_θ} and σ_Y are the variance of X_θ and Y respectively.

We find $\rho(x, y, \theta)$ for n number of different orientations between 0°-180°. Then strength $I_c(x, y)$ of the line pixel given as

$$I_c(x, y) = \max\{\rho(x, y, \theta)\}. \tag{2}$$

Let $\hat{\theta}(x, y)$ be the orientation for maximum response, then

$$\hat{\theta}(x, y) = \arg\max_\theta\{\rho(x, y, \theta)\}. \tag{3}$$

This method is robust to detect even weak, broken and curve lines depending upon the size of the templates. It also recovers these lines perfectly by strengthening faint line and joining broken lines. The effect of choosing different scale is shown in the fig. 2 by selecting rectangular blocks from the line image obtained by correlation. But in this step we also obtain some unwanted extra lines due to noise in the image. To reduce these effect, we apply convolution on the original gray image with the template with orientation $\hat{\theta}(x, y)$ at every (x, y). We get the convolution matrix c as

$$c(x, y) = \sum_{u=-\lfloor r/2 \rfloor}^{\lfloor r/2 \rfloor} \sum_{v=-\lfloor r/2 \rfloor}^{\lfloor r/2 \rfloor} X_{\hat{\theta}}(u, v) Y(x - u, y - v). \tag{4}$$

The modified line image I_{cc} is obtained by

$$I_{cc}(x, y) = \lambda * I_c(x, y) * c(x, y), \tag{5}$$

where parameter λ controls the brightness.

B. Non-maxima suppression

The non-maxima suppression thins and sharpens the lines. Here our approach is similar to the method applied for sharpening the blur edges. Main goal of this step is to preserve all the local maxima in the image I_{cc} and remove everything else. The algorithm has three sub-steps. First step is to round the line direction θ to the nearest $45°$ for each pixel. Then compare the line strength I_{cc} of the current pixel with the line strength of the pixels along the normal direction. If the line strength of the current pixel is maximum, then keep the strength as it is, else assign zero to the strength.

C. Anisotropic Refinement

In this step, we refine the image obtained by non-maxima suppression by enhancing true lines. For this, gradient of surrounding pixels takes into account for local context measure. For each pixel, we assign a weight $w_\sigma(x, y)$ by difference of two Gaussian function $DOG_\sigma(x, y)$.

$$\begin{aligned} DOG_\sigma(x, y) = \frac{1}{2\pi(4\sigma)^2} \exp(-\frac{x^2 + y^2}{2(4\sigma)^2}) - \\ \frac{1}{2\pi\sigma^2} \exp(-\frac{x^2 + y^2}{2\sigma^2}) \end{aligned} \tag{6}$$

The weight $w_\sigma(x, y)$ is defined by

$$w_\sigma(x, y) = \frac{F(DOG_\sigma(x, y))}{||F(DOG_\sigma)||_1} \tag{7}$$

where

$$F(t) = \begin{cases} 0 & \text{if } t < 0 \\ t & \text{if } t \geq 0 \end{cases} \tag{8}$$

and $||.||_1$ is the L_1 norm.

We define anisotropic refinement term as a weighted sum of the the difference in the gradient orientations in the central point and the surround points. Gradient magnitude is also taken in the refinement term. The distance between the current point and it's surround point is taken into account by the weight function w_σ. For each image pixel (x, y), anisotropic refinement $T^m_\sigma(x, y, u_s, v_s)$ at m-th iteration for scale s is formularized by

$$T^m_\sigma(x, y, u_s, v_s) = w_\sigma(u_s, v_s) I_{cc}(x - u_s, y - v_s) \Gamma_{\theta,\sigma}(x - u_s, y - v_s) \tag{9}$$

where

$$\Gamma_{\theta,\sigma}(x - u_s, y - v_s) = |\cos(\hat{\theta}(x, y) - \hat{\theta}(x - u_s, y - v_s))| \tag{10}$$

If the gradient orientations $\hat{\theta}$ at point (x, y) and $(x - u, y - v)$ are identical, the weighting factor is maximum i.e., $\Gamma_{\theta,\sigma} = 1$. In case of orthogonal orientation of these two points, $\Gamma_{\theta,\sigma}$ takes minimum value 0. So the same orientation of surrounding points on a line contribute maximum inhibitory effect at the point.

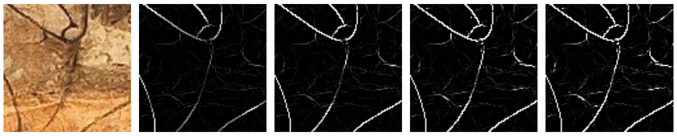

Fig. 3. Image blocks from original image and output of anisotropic refinement at iterations 1, 3, 5 and 6.

To better distinguish between the pixels in the texture region and on the lines, we find a term $L(x, y)$ as

$$L(x,y) = \sum_s [(L(x,y,s) - \frac{1}{n_s} \sum_s L(x,y,s)) \times I_{cc}(x,y)]^2 \tag{11}$$

where

$$L(x,y,s) = \max\{F(T)\} - \min\{F(T)\}. \tag{12}$$

The response of the term L is high in case of points on the clear as well as faint lines. The points in the texture region and on the strong lines, have similar value of L in different scale but for the points on the faint line surrounding strong line, these values are different. The value of L is large in case of points on the true lines. So the points at which the value of L is smaller than a given threshold, do not refine otherwise refined by an amount of maximum of L in different scale. We take a threshold T_r and the mathematical formulation is given by

$$R^{m+1}(x,y) = \begin{cases} 0 & \text{if } L(x,y) < T_r \\ \max_s L(x,y,s) & \text{elsewhere} \end{cases} \tag{13}$$

So the new line strength at each pixel (x, y) for $(m+1)$-th iteration is computed by

$$I_{cc}^{m+1}(x,y) = I_{cc}^{m}(x,y) + \alpha R^{m+1}(x,y), \tag{14}$$

where α is the refinement factor which controls the strength of refinement in each iteration. By this process texture lines are eliminated from the true lines, strong lines are preserved and faint lines are enhanced. Figure [3] shows the effect of anisotropic refinement for faint line detection in different iterations for an image block.

D. Line tracking

Here a binary map is computed from the line pixels by thresholding and final lines are obtained by line tracking algorithm similar to the edge tracking in Canny edge detection technique. The line image obtained after anisotropic refinement have two different type of lines, one for strong and faint lines due to true lines and another for weak lines due to noise or texture variations. We remove all the texture lines by the threshold value T_r and get a binary line image. After thresholding there exist some noisy line pixels and we remove those line pixels by blob analysis. If the blob with greater than some fixed size T_b contain some strong line within some neighborhood keep it as strong line pixel, otherwise remove those unwanted line pixels.

III. Noise Reduction And Line Enhancement

Main aim of line detection is to extract the clear, faint and broken lines from the original image so that after denoising the image, we can restore the lines with proper clarity. For this, we first extract the lines from the original color image, then denoise the image using a modified bilateral filter. Traditional Bilateral filter smooths the image as Gaussian filter but it tries to avoid to smoothing on the edge pixel. Smoothing coefficients combine both spatial distance and intensity similarity of neighboring pixels as follows

$$BF[I]_p = \frac{1}{W_p} \sum_{q \in S} G_{\sigma_a}(\|p-q\|) G_{\sigma_b}(|I_p - I_q|) I_q \tag{15}$$

where the first term within the sum measure the geometric distance between the points p and q, the second term measure the photometric similarity between I_p and I_q. G_{σ_a} and G_{σ_b} are the Gaussian functions with standard deviation σ_a and σ_b respectively, W_p is the normalization factor and S is the window surrounding the pixel p. One major problem of this filter is that it either retains or enforces large grain (composed of few pixels) noise which are very common in damaged murals. To overcome this problem we propose probabilistic bilateral filter as

$$PBF[I]_p = \frac{1}{W_p} \sum_{q \in S} P(|I_p - I_q|) G_{\sigma_a}(\|p-q\|) G_{\sigma_b}(|I_p - I_q|) I_q \tag{16}$$

where $P(|I_p - I_q|)$ is the probability of $|I_p - I_q|$ within S.

Our next step is to restore the enhanced lines obtained in section II. The broken lines are joined in the line image and it is preserved in the final restored image. But faint lines are enhanced by the pixels of strong lines which is taken as large component greater than some threshold. Since strong line pixels are near to black pixels we take minimum of all strong line pixels and assign the value to each of the line pixels. The weak lines which are smaller than the threshold, keep as it is in the original image. Note that in the line detection strong lines include all the faint lines and weak lines are those which are not true lines.

IV. Experiment

In this section, we set the parameters used in our method and show the result of different steps for the mural images. The proposed algorithm is applied on the image shown in figure [1]. In the line detection step, we have taken the size of the template $r = 15$, the number of orientation $n = 36$ with interval length 5 in between $0°$-$180°$, the brightness parameter $\lambda = 2.5$. Note that, we have not taken $180°$ for template synthesis since it is same with $0°$. In anisotropic refinement step, the value of $\sigma = 1.6$, the number of iteration $m = 6$, for $s = 3$ different scale patch sizes are 5, 7 and 9, the threshold value $T_r = .4$ and the refinement factor $\alpha = 1.0$. The value of T_b in line tracking step is equal to 10. In the step of noise reduction, we set the window size is 11, $\sigma_a = 3$ and $\sigma_b = .1$.

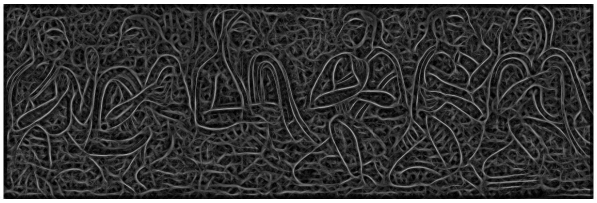

Fig. 4. Line image after correlation.

Fig. 5. Line image after convolution.

Fig. 6. Line image after non-maxima suppression.

Fig. 7. Line image after anisotropic refinement.

Fig. 8. Line detection after line tracking.

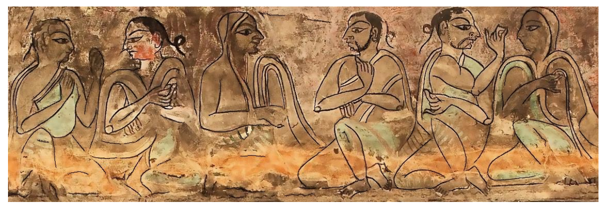

Fig. 9. Denoising and line enhancement.

Fig. 10. Part of images before and after restoration

Figure [4-8] shows the output of correlation, convolution, non-maxima suppression, anisotropic refinement and thresholding with line tracking. The output of denoising and line enhancement is shown in figure [9]. The result shows that our proposed approach is robust to detect clear as well as faint and broken lines. The result of noise reduction and line enhancement is also good enough. For better visualization, we show some blocks from the images before and after restoration in figure [10].

V. Conclusion

In this paper we try to restore mural images by detecting and enhancing clear as well as faint and broken lines. Also we remove noise from the image to produce better appearance by applying bilateral filter. Our results show that the method is robust to restore the deteriorated mural images. But in some cases, it fails to remove the unwanted texture lines and to correctly detect the lines at high curvature. In the future, we will try to address the problem of texture line removal for the detection of true lines in a better way.

References

[1] M. Pappas and I. Pitas, *Digital Color Restoration of Old Paintings*, IEEE Transaction on Image Processing Vol. 9, pp. 291294, 2000.

[2] S. C. Pei, Y. C. Zeng and C. H. Chang, *Virtual Restoration of Ancient Chinese Paintings Using Color Contrast Enhancement and Lacuna Texture Synthesis*, IEEE transactions on image processing, Vol. 13, pp.416429, 2004.

[3] S. Awate and R. Whitaker, *Unsupervised, information-theoretic, adaptive image filtering for image restoration*, IEEE Transaction on Pattern Analysis and Machine Intelligence, vol.-28, NO.-3, March 2006.

[4] Giakoumis, Nikolaidis, Pitas, *Digital image processing techniques for the detection and removal of cracks in digitized paintings*, IEEE transactions on image processing, volume 15, issue 1; January 2006.

[5] B. Chanda, Dhruv Ratra and B.L.S. Mounica, *Virtual Restoration of Old Mural Paintings using Patch Matching Technique*, Third International Conference on Emerging Applications of Information Technology (EAIT) 2012.

[6] B. Chanda and Pulak Purkait, *Digital Restoration of Damaged Mural images*, The 8th Indian Conference on Vision, Graphics and Image Processing, 2012.

[7] C. Tomasi and R Manduchi, *Bilateral Filtering for Gray and Color Images.* Sixth International Conference on Computer Vision, 1998.

[8] J. Canny. *A computational approach to edge detection*. Pattern Analysis and Machine Intelligence, IEEE Transactions on, PAMI-8(6):679698, Nov. 1986

[9] C. Grigorescu, N. Petkov and MA. Westenberg. *Contour and boundary detection improved by surround suppression of texture edges*. Image and Vision Computing 22 (2004) 609622.

[10] G. Kanizsa, *Organization in Vision: Essays on Gestalt Perception*, Praeger, New York, 1979.