TEXTURE CONSISTENT SHADOW REMOVAL

CSE 527 Introduction into Computer Vision

Arvind Chaudhary, Roman Pavlushchenko & Srikant Aggarwal

INTRODUCTION

Shadow removal is a major image processing technique used in various applications such as Adobe Photoshop and Adobe Illustrator. Various papers discuss about the technique.

One paper which discusses about the techniques of shadow removal is "Texture-Consistent Shadow Removal"[1]. It capitalizes on a simple idea that shadow is represented in its gradient field as a boundary. This boundary can then be removed and integrating the resultant gradient will produce a shadow free image.

However, the actual process is not simple and many factors and assumptions should be taken into account while performing shadow removal. For example, how does one deal with penumbra area where the intensity change is not constant, how to smoothness between neighboring lines. Also, if one has poor lighting in shadow area, it will weaken the texture and blur some details. How to restore it?

Before introducing our implementation it is worth taking a brief look onto the existing approaches and algorithms regarding creating shadow-free images.

LITERATURE SURVEY

Shadow is removed mostly in two stages: detecting shadow and image reconstruction.

The shadow region can be nullified by multiplying a suitable scalar value to the intensity. However, it will result in an over-saturated band in the boundary between lit area and the umbra. To solve it Baba et al. use several scalars depending on the shadow density [2].

In another approach, shadow is removed by setting gradient pixels to zero in the penumbra area and further reconstruction through two-dimensional integration. To solve such a problem, one should convert image to log scale. However, zeroing out gradient in penumbra clears the texture too, and as a workaround, in-painting methods are used [3, 4].

Basically, the methods above share the similar approaches: multiplying a constant and reintegrating the shadow-free image. In some cases just changing the illumination of umbra area to the appropriate level and fitting with lit area could work. However, the models do not take into account texture characteristics of the umbra area. While multiplying a constant may create noise, specific textures, such as specular and with strong illumination, would expose the details hidden in the umbra - both approaches lead us to texture inconsistency between umbra and lit area.

IMPLEMENTATION

The project implements the paper "Texture-Consistent shadow removal" [1]. Unlike the work before, in the paper a shadow removal method is presented which preserves texture consistency for the penumbra and the umbra area.

In our version of the code, instead of making users draw a boundary around the umbra, we use an umbra shadow mask for every image we are going to remove shadow from.

Starting from 3.1 we provide the main idea behind the project.

3.1 Estimation of Illumination Changes in Penumbra Area

The way a shadow affects an image is governed by the following equation:

$$I\_Final(x, y) = I(x, y).C(x, y)$$

where, I_Final is the final image, I is the initial image without the shadow and C captures illumination changes due to shadow (It is a fraction less than 1 in the umbra region and equal to 1 in the lit region).

The paper formulates process of shadow removal as a reconstruction of image from the gradient and for this purpose it initially translate the original picture into the log field and to return it back to original form in the end. The equation in log form becomes:

$$I\_Final(x, y) = I(x, y) + C(x, y)$$

In order to calculate illumination changes across the penumbra region, our implementation takes ideas from the paper "Editing soft shadows in a digital photograph" [10] by sampling horizontal and vertical lines across the shadow boundary. Along each line, the illumination changes can be modeled as in Fig. 1.

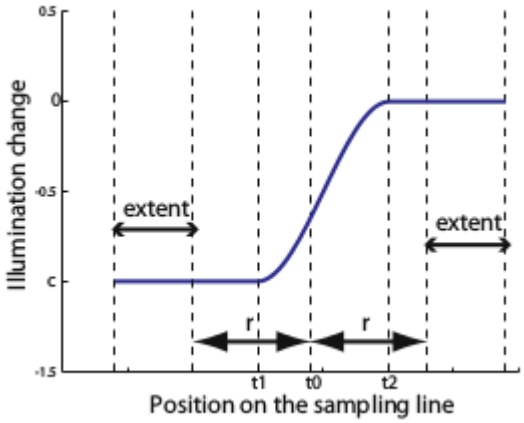


Figure 1. The figure shows changes in intensity due to shadow

Important parameters of this plot are c (the level of intensity in the umbra area), t1 and t2 (coordinate on the sampling line, for which the curve takes a constant value). r (which gives maximum possible extent of the penumbra about each side of the boundary) is not a part of the model, i.e. it is equal for all sampling lines and created models. The illumination changes, is then given by the following equations :

$$M(x, y) = c \ (c <= 0) \quad \text{Umbra region}$$
$$= f(t) \quad \text{Penumbra region}$$
$$= 0 \quad \text{Lit region}$$

M(x, y) here is the Illumination Change Model. The curve f(t) governing changes of intensities due to shadow along with Penumbra region is a cubic function for which :

$$f(t1) = c$$
$$f(t2) = 0$$
$$f'(t1) = 0$$
$$f'(t2) = 0$$

The coefficients of the cubic function, f(t) can be calculated by solving the above system of linear equations.

3.2 Formulation of the goal

Having the set of sampling lines (horizontal and vertical) along the shadow boundary, the paper formulate the goal to achieve a shadow free image as a minimization problem. It strives to achieve two important conditions :

- The illumination change along a line in the penumbra region should be consistent with the umbra and lit region. This is reflected in the Fitness error (between a given spline and image data) which the paper tries to minimize.
- Smoothed illumination change surface for the all the lines by minimization of the similarity cost function.

3.3 Finding optimal illumination change model

An optimal illumination change model along a line, M_{li}^{o}, is one which ensures that illumination changes along the penumbra region confirms with the one in the lit and the shadow area. This is achieves by minimizing the fitness error function.

$$E_{fit}(M_{li}, \tilde{I}) = -\Pi_{t \in [t_{i0}-r_i, t_{i0}+r_i]} \varphi(\hat{G}_{li}(t), T_{li}^{tex})$$

$$\hat{G}_{li}(t) = \tilde{G}_{li}(t) - C'_{li}(t)$$

The φ function is governed by following equation :

$$\varphi(G_{li}(t), T_{li}^{tex}) = \frac{exp(-(G_{li}(t) - \mu_i)^2/2\sigma_i^2)}{\sqrt{2\pi\sigma_i^2}}$$

The equation denotes the fitness error in a context of current sample line for a current coordinate t. It is a Gaussian: the paper assume that texture is distributed normally along li.

The optimal model containing c, t1, t2 is then calculated using a brute-force search method by using the above two formulas. By putting values for t1, t2 (between t0 + r and t0-r), the algorithm tries to minimize the fitness error, the values where the result is minimum are taken as the model values (In our observation, this brute force method was the bottleneck in the implementation lowering the speed of algorithm by a factor of 4 for a unit increase in radius).

It should be noted that the illumination models calculated are not globally optimal. They just serve as an approximation instead of actual image data .

3.4. Taking illumination change models as a whole

Next, the implementation tries to find illumination profiles for each of the boundary points, such that the surface, which they make up, is smooth and each actual illumination model has a high degree of similarity with regards to corresponding optimal model. The equation governing the similarity between two neighboring lines is:

$$E_{sm}(M_{li}, M_{lj}) = \gamma(c_i - c_j)^2 + (1-\gamma)((t_{1i} - t_{1j})^2 + (t_{2i} - t_{2j})^2)$$

The similarity between two models is simple and takes into account two parts:

- Similarity between intensity in umbra. This c terms is given a higher weight because the texture inside of the shadow is highly uniform. Thus, adjacent splines with different texture characteristics will lose similarity points.
- Similarity between location of penumbra.

To ensure this, the paper solves a minimization function :

$$E = \sum_{li} E_{sm}(M_{li}, M_{li}^o) + \lambda \sum_{li} \sum_{lj \in N(li)} E_{sm}(M_{li}, M_{lj})$$

which is actually quadratic minimization problem. After calculations and taking into account above two formulas the problem reduces to optimization of a quadratic function. The function is then solved using a quadprog matlab function. Though the authors suggested pcg (Preconditioned Conjugate Gradient) method, we found quadprog to be more suitable in our case.

The output of the step is an array of the "smooth" illumination models, parameters of which ensure smooth surface created by all profiles of these models, and also fit them to optimal models.

3.5 Removing shadow effect

In this step, for each horizontal/vertical line and making use of the actual illumination change models calculated above, the gradient is reduced appropriately to remove the effect of shadow on gradient while not altering the gradient due to texture.

$$\hat{G}_{li}(t) = \tilde{G}_{li}(t) - C'_{li}(t)$$

3.6 Texture reconstruction

A post image processing step tries to restore texture (which gets affected due to the shadow) after shadow removal. This it does by parameterizing the global texture characteristics by using the sampling mean and deviation of the gradient field.

So if the target mean and deviation are μ^t and σ^t, it transforms the gradient field in the shadow area using the equation :

$$G_f(x, y) = \mu^t + (G_i(x, y) - \mu^s) * \sigma^t / \sigma^s$$

Where $G_f(x, y)$ is the final gradint in the shadow region, $G_i(x, y)$ is the initial gradient in the shadow region, μ^s , σ^s is the mean and deviation of G_i.

If the texture distribution is globally homogeneous, the target characteristics parameters can be calculated by estimating the shadow effect on the gradient distribution and cancelling this effect from the original gradient field. Firstly, the shadow effect parameters are calculated as :

$$\mu_{se} = \mu_b^s - \mu_b^l$$

$$\sigma_{se} = \sigma_b^s - \sigma_b^l$$

where μ_{se} and σ_{se} are the mean and deviation of the shadow effect on gradients in the shadow area. μ_b^s and σ_b^s are the mean and deviation of the gradients in the umbra side along the shadow boundary (the extent parts as illustrated in Figure 1) , and μ_b^l and σ_b^l are those in the lit area side. Accordingly, the target mean and deviation can be calculated by canceling the shadow effect as follows:

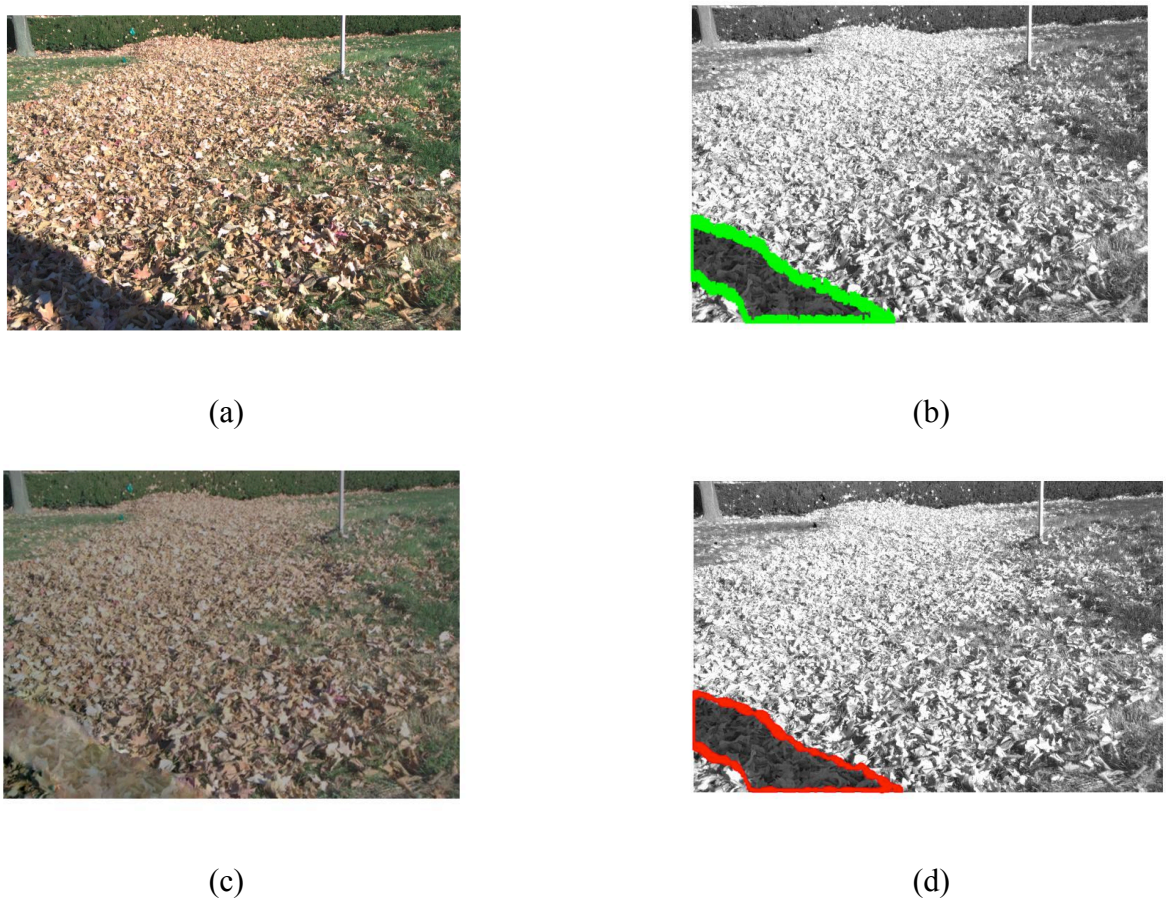
$$\mu^t = \mu^s - \mu_{se}$$
$$\sigma^t = \sigma^s - \sigma_{se}$$

3.7 Image Reconstruction from Gradients

After calculation of the final gradients along x and y direction, the original image is obtained by integrating the gradients using Poisson Solver.

In the code implementation we have used Matlab. Apart from this we have used open source code available for Poission Solver [6] .

Major Steps in a nutshell



(a) (b)

(c) (d)

Figure 2. (a) show the original image with shadow, (b) show the boundary vertical lines in gradient image after Mlio calculation, (c) shows the boundary vertical lines after Mli calculation and (d) shows the final texture restored image.

RESULTS

The implementation was run on both grayscale and RGB images.

Sample results on Grayscale Images:

Around 10 grayscale images (kept in ‘Grayscale’ directory) were used.

Initial Image Final Image

Figure 3 : Image 1

Initial Image Final Image

Figure 3 : Image 2

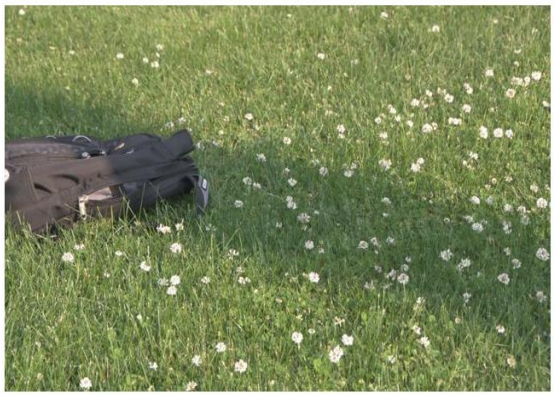
Sample Colored Images

We split the colored images in three channels, and then at the end we recombine the images again. We have attached around 10 images with the report in a directory named as ‘Colored’

Initial Image

Final Image

Figure 4: RGB Image 1



Initial Image

Final Image

Figure 4: RGB Image 2

We tried two different gradient calculation methods (Sobel and CentralDifference) and two Poisson Equation Solvers (Provided by MIT [5], Wavelet based Image Reconstruction from Gradient Data [6]).

Comparison of both gradient methods and Poisson Equation Solvers:

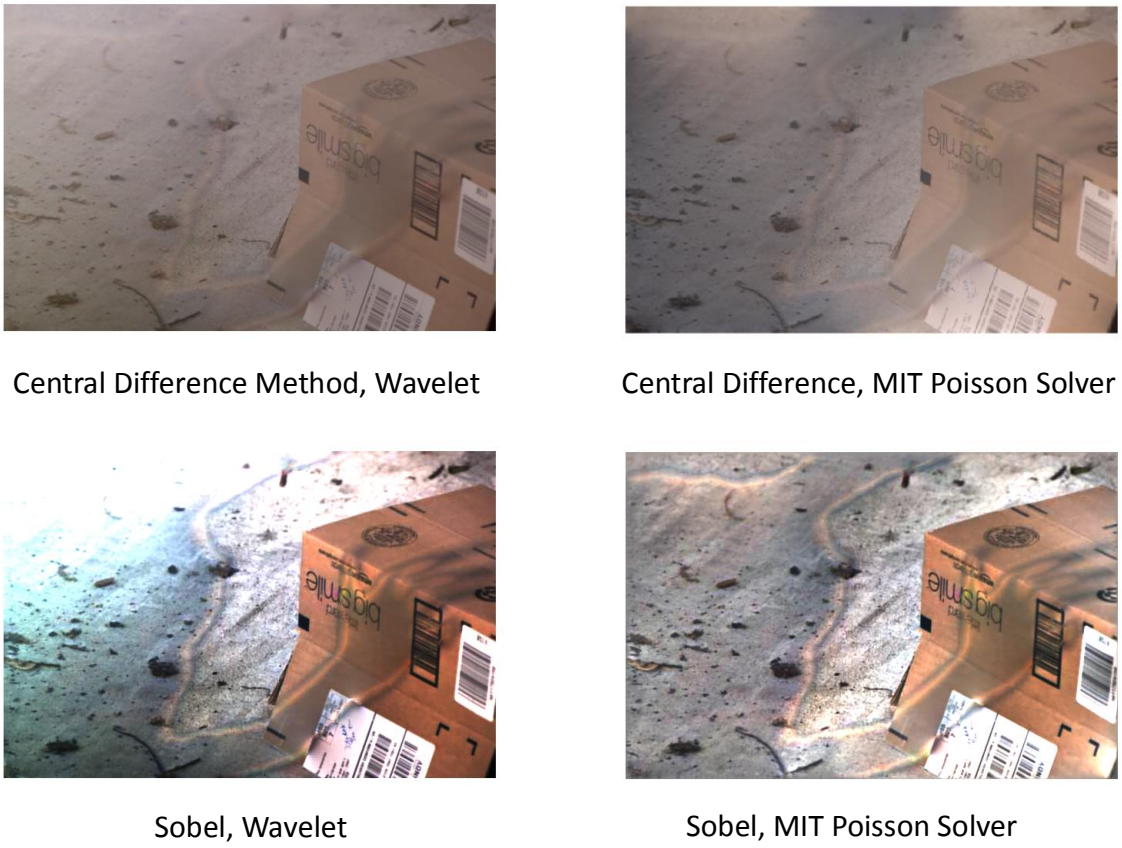


Figure 5: Comparison of different Gradient Calculation Methods and Poisson Solvers

Issues Faced :

1. In few images, as you can see that shadow are became brighter than the lit area.
2. We could use a better Poisson Image Reconstruction tool, the currently available tool slightly overlays the whole image in terms of color intensity.

REFERENCES

1. Liu, F., Gleicher, M.: Texture-Consistent Shadow Removal. In: ECCV 2008, Part IV, LNCS 5305, pp. 437-450, (2008)
2. Weiss, Y.: Deriving intrinsic images from image sequences. In: IEEE ICCV, pp. 68–75 (2001)

3. Arbel, E., Hel-Or, H.: Texture-preserving shadow removal in color images containing curved surfaces. In: IEEE CVPR (2007)
4. Finlayson, G.D., Hordley, S.D., Lu, C., Drew, M.S.: On the removal of shadows from images. IEEE Trans. Pattern Anal. Mach. Intell. 28(1), 59–68 (2006)
5. Poisson Image Solver, http://web.media.mit.edu/~raskar/photo/code.pdf
6. "Wavelet based Image Reconstruction from Gradient Data“, Copyright (c) 2014, Ioana Sevcenco http://www.mathworks.com/matlabcentral/fileexchange/48066-waveletbased-image-reconstruction-from-gradient-data/content/ImageRecH_V01/ImageRecH.m
7. Mohan, A., Tumblin, J., Choudhury, P.: Editing soft shadows in a digital photograph. IEEE Comput. Graph. Appl. 27(2), 23–31 (2007)