Vignetting

Nikolaos Laskaris
School of Informatics
University of Edinburgh

What is Image Vignetting ?

Image vignetting is a phenomenon observed in photography (digital and analog) which introduces some photon energy loss on the periphery of a captured image. This is translated into a lower color intensity of the pixels of the area where the phenomenon appears. Intuitively, this can be described as a gradual fade-out of the lightness as we move from the center of the image to the periphery. Sometimes, as it will be described later, the vignetting fade -out might be abrupt, causing a black (with no information) image periphery.

What causes image vignetting?

There are many causes of image vignetting. These are either due to blockage of the incoming light (photons) by certain parts of the camera (filters, lens diaphragm), or due to the physical properties of the lens. There are four different types of vignetting:

- The **Natural Vignetting**, which is dependent on the lens' geometry. Generally, it is a problem of energy loss of the photons which pass through the lens by an angle greater than 0^0 between the perpendicular to the lens axis and the direction of the incoming photon, as illustrated in Figure 1. The greater the angle is, the greater the distortion is (in terms of luminance loss). So, the more distant we move from the center of the image, the "darker" the image appears.

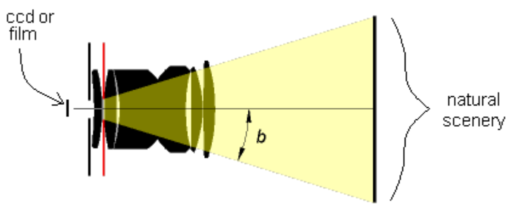
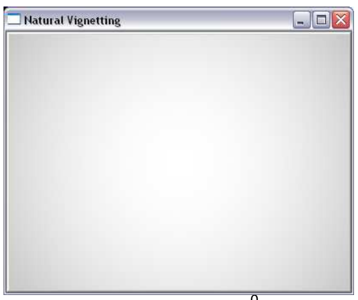


Figure 1.
Illustration of the luminance loss, which is dependent on the angle b. Note that the lens' opening (angle) equals twice the b. [1]

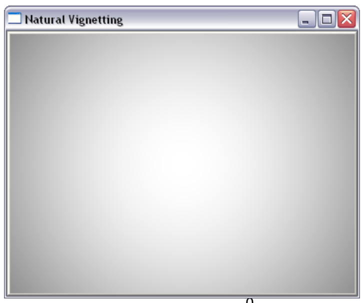
The falloff caused by the natural vignetting can be modeled as a $\cos^4(b)$ function, where b is the angle of figure 1, producing numbers in a range of [0,1], where 1 (white – no falloff) for $b = 0^0$ and 0 (absolutely no lightness) for $b = 90^0$ – however, 90^0 is impossible in real cases, but exists only mathematically.
A simple intensity image (mask) generated by using OpenCV library [3], demonstrating the radial intensity fallout of $\cos^4$ is presented in Figure 2. What can

be easily observed is that the intensity attenuation is dependent on the type of the lens (wide angle or telephoto).

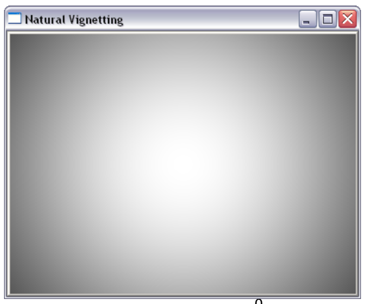
- **Optical Vignetting** is quite similar to the natural vignetting, in a sense that both are caused by the lens properties (in natural it is the lens' glass properties while in optical is the lens' diaphragm opening – called aperture) and produce similar radial by the center of the image, intensity attenuation. The radial attenuation is proportional to the aperture of the lens. Therefore, the smaller the aperture is, the smaller the optical vignetting is. This is because by using a smaller aperture, the different paths in which the light can pass through the diaphragm and reach each point on the film/ccd are getting less –and equally distributed in the whole ccd/film area, while using a big aperture the incident light on the central regions of the ccd/film is more than the one on the periphery (proportional to the different paths).



a. b = 20^0



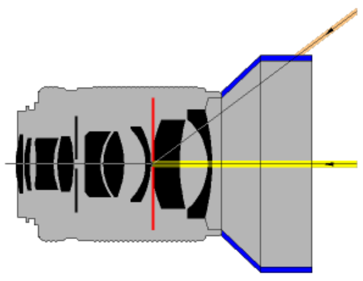
b. b = 30^0

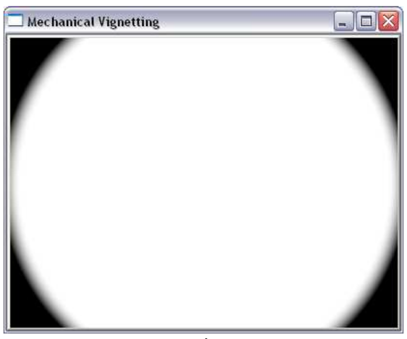


c. b = 40^0

Figure 2.
Different intensity falloff produced by different angle of views. In a, the angle of view (b angle in figure 1) is 2*b = 40^0 , in image b is 60^0 and in image c is 80^0 .

- Mechanical vignetting is another type of brightness attenuation, which in most cases is radial by the center of the image and usually produces a very steep brightness attenuation, where the brightness is attenuated to zero (visually a black color and complete information loss) within a few pixels distance. This is caused by attachments on the camera (filters, large exterior lenses, other hood), which entirely block a part of the incoming light to a specific region of the ccd/film – leading to a complete information loss. A depiction of how the camera attachments blocks the


a



b

Figure 3.
a) The attachment (additional lens, filter) blocks the incoming light which was to be captured by the regions on the periphery of the film/ccd.
b) A sample generation of mechanical vignetting (image intensity).

incoming light is presented in Figure 3.a. [2]. In Figure 3, picture b, by using the OpenCV processing library, a intensity image was created to demonstrate how the brightness attenuates in a typical mechanical vignetting case. It should be noticed that the falloff is quite abrupt (depends on the lens aperture), within a space of a few pixels and leads to complete darkness (brightness=0), which means that the information over the black region of the image has been lost, and thus it cannot be recovered. No algorithm can inverse the mechanical vignette effect.

- Last type, is the **Pixel Vignetting**, which occurs only in digital cameras. It is caused by the CCD nature to produce stronger signal from incident photons at a right angle, than other photons of same energy which reach the ccd by any other angle [2]. The effect of pixel vignetting on the captured image is similar to the natural and optical vignetting.

In most cases, the vignette effect on images is not desirable. However, there are some cases in which vignetting might be added in an image as an “artistic” effect, to make the observer emphasize on the central region of the image.

How can image vignetting be removed from an image?

Previously, all the different types of vignetting have been analyzed. There are some features which are common in all types:

a) Intensity falloff is symmetric and radial about the center of the image,
b) It is dependant only on the physical characteristics of the camera (lens, diaphragm, ccd and any light-blocking camera attachments), thus it can be measured. However, the measurement turns to be a very complicated process if the above mentioned physical characteristics are not all known.

There have been various approaches to find out the parameters of the vignetting, in order to remove the intensity attenuation by making the inverse computations.

A very simple way to compute the vignetting parameters is to use a “reference” photo of a uniformly illuminated white background and observe the vignetting effect on this image. A radius-to-intensity attenuation mapping function can then be easily derived and be used as a reference on next images in order to evaluate the “real” (not distorted) intensity level of each pixel. The only problem with this approach is that it should be done for various geometry transformations of the lens (zooming, aperture) and any unknown combinations of transformations should be interpolated from the closest pair of known. There is a lot of ongoing research in order to find methodologies of computing the vignetting without any prior calibration.

Goldman and Chen [4], define an objective function

$$Q_d(a, \beta, t_i, L_x) = \sum_{x_i, d} d\left[P_{x,i}, R\left(t_i\, L_x M_a(r_{x,i})\right)\right], \quad (1)$$

which evaluates the summation of distance between the actual intensity value of a pixel $P_{x,i}$ - where x is its position in the image and i is the image number and the expected value of the response curve R() parameterized by the exposure value t_i , the radiance value of the pixel L_x and the vignetting function $M_a(r_{x,i})$, where r is the pixel’s distance from the image’s center. By keeping the lens’ geometry stable during the capture of the images and assuming that the incoming light from each specific point of the scenery does not change between consecutive shots, try to calibrate (find the parameters of) the vignetting function. They approximate the $M_a(r)$ vignetting function as a sixth order even polynomial $1 + a_2r^2 + a_4r^4 + a_6r^6$. After discovering (by approximation) all the parameters which cause the vignetting, each new pixel value (corrected, vignette-free) can be easily computed by

$$P_{x,i}^{new} = R_\beta\left(\frac{t_{new}R_\beta{}^{-1}(P_{x,i})}{M_a(r_{x,i})t_i}\right), \qquad (2)$$

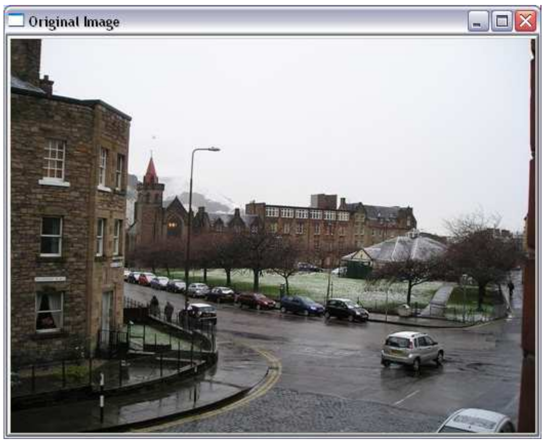
where $P_{x,i}^{new}$ is the new intensity value of the x pixel in the image, M_a is the vignetting function, R_β is the computed response curve and t_i , t_{new} are the exposure values.
It should be mentioned that the response curve is a function of radius. For every given value of radius, it maps an intensity value to it. Generally, it describes the brightness attenuation as the radius increases.

By a similar method, Yu, Chung and Soh [7] approximate the vignetting distortion with a hyper-cosine function of pixel distance from the image center for each scanline, having firstly the camera calibrated, by using a white uniformly illuminated paper. Other vignette distortion functions used by other researchers – quite earlier - were polynomial (Bastuscheck [8]) or exponential functions (Chen and Mudunuri [9]).

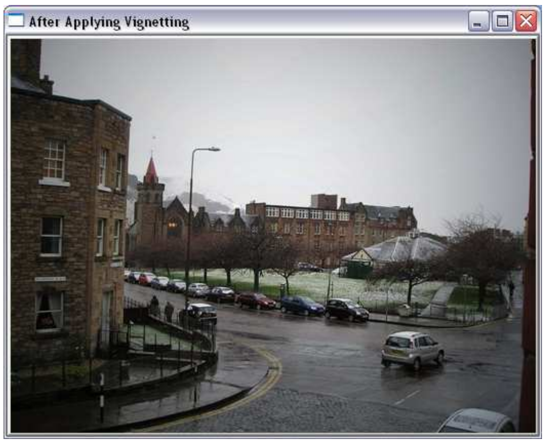
Finally, an interesting method for correcting the vignette effect on a single image, without any prior calibration or any other knowledge of the vignetting function, is the one proposed by Zheng, Lin and Kang [5], which tries to estimate the local tilt parameters of each region of the image, segmentate the image based on these parameters and by using the Kang-Weiss vignetting model to discover for every image segment the best fitting correction function.

Appendix.

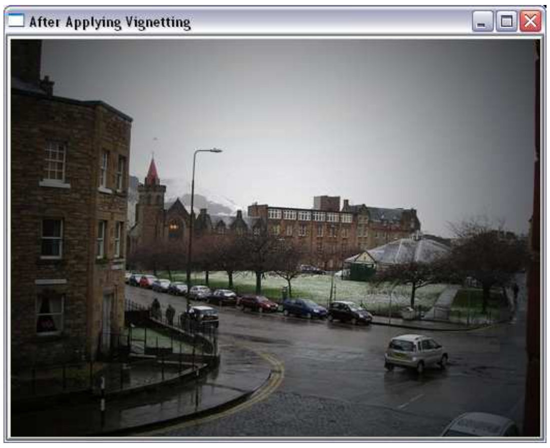
On the next figure, we apply the artificial vignetting mask which was created with the use of OpenCV library, as previously described on figure 2. The RGB image is transformed into the Lab color space, where the intensity mask works as a weight factor for the 'L' (for luminance) dimension of the Lab. After modifying the L, the image is converted back into the RGB color space. A few results are presented in figure 4.



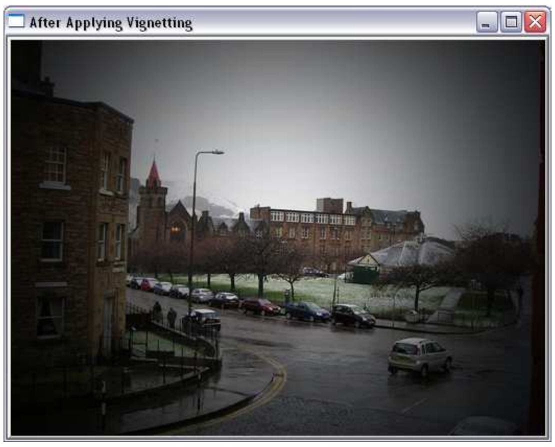
a



b



c



d

Figure 4.

A depiction of how vignetting affects the image quality.

a) Original image of Edinburgh.

b) Vignetting the image using the intensity image of figure 2.a

c) Vignetting the image using the intensity image of figure 2.b

d) Vignetting the image using the intensity image of figure 2.c

OpenCV C/C++ Code demonstrating the Vignetting

```
#include<iostream>
using namespace std;
#include<math.h>

#include<cv.h>
#include<cxcore.h>
#include<highgui.h>

#define P 3.14159

#define MAX_ANGLE P/9  //from center of image to one corner (whichever one)
// P/9 means  180degrees/9 = 20 degrees , which is half the angle of the lens.
// The lens angle is 20*2 = 40 degrees in this case.

double dist (CvPoint a,CvPoint b){
        return sqrt(pow((double)(a.x-b.x),2)+pow((double)(a.y-b.y),2));
}


void mechanicalVignetting(int pixels_falloff){
        int image_width = 400,image_height = 300;
        double radius = (double)(image_width/2)*95/100;
        CvPoint image_center = cvPoint(image_width/2,image_height/2);
        double max_image_rad = sqrt(pow((double)image_width/2,2)+pow((double)image_height/2,2));
        CvSize image_size = cvSize(image_width,image_height);
        IplImage * eikona = cvCreateImage(image_size,IPL_DEPTH_64F,1);
        cvSet(eikona,cvScalar(1));
        double distance;
        for(int i=0;i<eikona->height;i++){
                for(int j=0;j<eikona->width;j++){
                        distance = dist(image_center,cvPoint(j,i));
                        if(distance>radius){

                                if(distance>radius+pixels_falloff)
                                        cvSet2D(eikona,i,j,cvScalar(0));
                                else
                                        cvSet2D(eikona,i,j,cvScalar(1-pow((double)((distance-
                                                                radius)/pixels_falloff),2)));
                        }
                }
        }
        cvNamedWindow("Mechanical Vignetting");
        cvShowImage("Mechanical Vignetting",eikona);
        //cvWaitKey(0);
        //cvDestroyWindow("Mechanical Vignetting");
        cvReleaseImage(&eikona);

}
```

```
IplImage * naturalVignetting(int image_width = 400, int image_height = 300){
	CvPoint image_center = cvPoint(image_width/2,image_height/2);
	double max_image_rad = sqrt(pow((double)image_width/2,2)+pow((double)image_height/2,2));
	CvSize image_size = cvSize(image_width,image_height);
	IplImage * eikona = cvCreateImage(image_size,IPL_DEPTH_64F,1);
	cvSet(eikona,cvScalar(1));
	for(int i=0;i<eikona->height;i++)
		for(int j=0;j<eikona->width;j++)

	cvSet2D(eikona,i,j,cvScalar(pow(cos((dist(image_center,cvPoint(j,i))/max_image_rad)*MAX_ANGLE),4))
);
	cvNamedWindow("Natural Vignetting (intensity)");
	cvShowImage("Natural Vignetting (intensity)",eikona);
	//cvWaitKey(0);
	//cvDestroyWindow("Natural Vignetting");
	return eikona;
}

void artificial_vignetting(char * filename){
	IplImage * bgr = cvLoadImage(filename);
	cvNamedWindow("Original Image");
	cvShowImage("Original Image",bgr);
	//cvWaitKey(0);
	CvScalar value;
	IplImage * Lab = cvCreateImage(cvGetSize(bgr),IPL_DEPTH_8U,3);
	IplImage * reference = naturalVignetting(Lab->width,Lab->height);
	cvCvtColor(bgr,Lab,CV_BGR2Lab);
	for(int i=0;i<Lab->height;i++){
		for(int j=0;j<Lab->width;j++){
			value = cvGet2D(Lab,i,j);
			value.val[0] *= cvGet2D(reference,i,j).val[0];
			cvSet2D(Lab,i,j,value);
		}
	}
	cvReleaseImage(&reference);
	cvCvtColor(Lab,bgr,CV_Lab2BGR);
	cvReleaseImage(&Lab);
	cvNamedWindow("Original Image with Natural Vignetting");
	cvShowImage("Original Image with Natural Vignetting",bgr);
	cvWaitKey(0);
	cvDestroyAllWindows();
}



void main(){
	//naturalVignetting();
	mechanicalVignetting(20);
	artificial_vignetting((char*)("c://P1010033.JPG"));

}
```

Bibliography

[1] http://www.vanwalree.com/optics/vignetting.html

[2] http://en.wikipedia.org/wiki/Vignetting

[3] http://opencv.willowgarage.com/wiki/
Intel's open – source c++ library for computer vision.

[4] Vignette and Exposure Calibration and Compensation. Goldman D.B., Jiun-Hung Chen. Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference.
Volume 1, 17-21 Oct. 2005 Page(s):899 - 906 Vol. 1

[5] Single-Image Vignetting Correction. Yuanjie Zheng, Lin. S., Sing Bing Kang.
Computer Vision and Pattern Recognition, 2006 IEEE Computer Society. Volume 1, 17-22 June 2006 Page(s):461 – 468.

[6] Single-image vignetting correction using radial gradient symmetry. Yuanjie Zheng, Jingyi Yu, Sing Bing Kang, Lin. S, Kambhamettu. C. IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008. 23-28 June 2008 Page(s):1 - 8

[7] Vignetting Distortion Correction Method for High Quality Digital Imaging . W.Yu, Y.Chung, J. Soh. Procceedings of the 17th IEEE International Conference on Pattern Recognition. Pp.666-669, Aug 2004.

[8] Correction of Video Camera Response Using Digital Techniques. C.M. Bastuscheck.
J. Optical Engineering , vol 26, no 12, pp. 1257-1262, 1987

[9] An Anti-Vignetting Technique for Superwide Field of View Mosaicked Images.
J. Imaging Technology, vol 12, no. 5, pp. 293-295, 1986